Object Tracking via Tensor Kernel Space Projection

Jiashu Dai
College of Computer Science and Technology, Harbin Engineering University, Harbin, China
Email: daijiashu2008@163.com

Tingquan Deng[a] Tianzhen Dong[a] Kejia Yi[b]
a. Laboratory of Fuzzy Information Analysis and Intelligent Recognition, Harbin Engineering University, Harbin, China
b. Science and Technology on Underwater Acoustic Antagonizing Laboratory, Systems Engineering Research Institute of CSSC, Beijing, China
Email: {tq_deng, tianzhen.d, yikejia2007}@163.com

Abstract—**Although there has been significant progress in the past decade, object tracking under complex environment is still a very challenging task, due to the irregular changes in object appearance. To alleviate these problems, this research developed an object tracking algorithm via tensor kernel space projection. In the initial stage of tracking, a template matching algorithm was used to obtain a priori images of the appearance of the object. The steps taken were as follows: define the tensor kernel function based on a multi-linear singular value decomposition, view the object appearance color image as tensor data, calculate the kernel matrix for the priori appearance image samples, use KPCA to obtain the projection matrix of the image samples in kernel space, and finally, obtain an optimal estimate of the object state through Bayesian sequence interference. Meanwhile, the projection matrix in kernel space was updated on-line. Experiments on two real video surveillance sequences were conducted to evaluate the proposed algorithm against two classical tracking algorithms both qualitatively and quantitatively. Experimental results demonstrate that the proposed algorithm is robust in handing occlusion and object scale changes.**

Index Terms—**object tracking, tensor kernel, SVD, kernel pace, projection**

I. INTRODUCTION

Object tracking as an important research area in computer vision and pattern recognition, has many applications, such as: object recognition, video surveillance, traffic monitoring, human-computer interaction, video compression, weapon automatic tracking, *etc*. [1-3].

The chief challenge in object tracking is how to adapt to appearance variability in the object. The appearance variability includes the effects of: camera motion, camera viewpoint changes, illumination change, pose variation, shape deformation and occlusions [4]. To solve this problem, there are two types of methods based on different appearance models: generative and discriminative methods.

Generative methods model appearance of targets use gray value or other features, and predict the target state by find the image patch most similar to the target. The classical mean-shift [5-6] tracking method using the kernel function forms a weighted model of the color histogram, and defining the similarity measure by its Bhattacharyya coefficient, seeks the most similar region to the reference template by a first-order gradient descent algorithm. The CBWH tracker [7] transforms only the object model but not the object candidate model. Ross [4] uses an incremental sub-space model combined with a particle filter to adapt to appearance changes. Gao [8] uses a Mexican hat wavelet to change the mean-shift tracking kernel and embedded a discrete Kalman filter to achieve satisfactory tracking. The ℓ_1-tracker [9] uses a visual tracking problem as a sparse approximation, obtains sparse representation by solving ℓ_1 regularisation, and finally finds the minimum reconstruction error region. Gao [10] fused multi-feature weights through DS evidence theory and combined with a particle filter achieves better results when considering the particle degeneration phenomenon.

Discriminative methods formulate object tracking as a binary classification problem for separates the target from the background. The Support Vector Tracking [11] integrates SVM into optical flow classifier for object tracking. Avidan [12] combined weak classifiers into a strong classifier to distinguish the object with background. The online boosting tracker [13] selects features using boosting classifier. The MIL tracker [14] using Multiple Instance Learning to overcome slight inaccuracies in labeled training examples can cause drift, the samples are considered within positive and negative bags.

The authors proposed an algorithm that uses tensor kernel space projection for object tracking to estimate the object state in real-world scenarios. For the purpose of highlighting the real differences between color images, a tensor kernel based on multi-linear singular value decomposition was defined. The Bayesian sequence interference framework was combined therewith to project the observe image samples in kernel space using a kernel matrix. Experimental results showed that the

Supported by the Natural Science Foundation of Inner-Mongolia Autonomous Region China under grant 2012M0931.
Corresponding author: Jiashu DAI
E-mail:daijiashu2008@163.com

doi:10.4304/jcp.9.6.1503-1510

proposed algorithm could achieve object tracking in real-world and occlusion environments, it was also adaptive to object scale changes.

II. Tensor Kernel Based on Multi-linear Singular Value Decomposition

A. *Tensor Algebra*

A tensor can be seen as a multi-order array which exists in multiple vector space, meanwhile, tensor algebra forms the mathematical basis of the multi-linear analysis [15]. A scalar is a zero-order tensor, a vector is a first-order tensor and a matrix is a second-order tensor. It is obviously that a color image is a third-order tensor. An N-order tensor can be denoted by $\tilde{\boldsymbol{X}} \in \mathrm{R}^{I_1\times I_2\times\cdots\times I_N}$, the element of $\tilde{\boldsymbol{X}}$ is $x_{i_1,\cdots,i_n,\cdots,i_N}$, where $1 \le i_n \le I_n$.

Definition 1: The n-mode unfolding matrix of a tensor data $\tilde{\boldsymbol{X}}$ is denoted as

$$\tilde{\boldsymbol{X}}_{(n)} \in \mathbb{R}^{I_n\times(I_1\times\cdots\times I_{n-1}\times I_{n+1}\times\cdots\times I_N)} \tag{1}$$

The element $x_{i_1,\cdots,i_n,\cdots,i_N}$ of $\tilde{\boldsymbol{X}}$ in $\tilde{\boldsymbol{X}}_{(n)}$ at i_n*-th* row and at $[(i_{n+1}-1)I_{n+2}\cdots I_N I_1\cdots I_{n-1}+\cdots+(i_N-1)I_1\cdots I_{n-1}+(i_1-1)I_2\cdots I_{n-1}+\cdots i_{n-1}]$*-th* column.

Definition 2: The n-mode product of a tensor data $\tilde{\boldsymbol{X}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ and a matrix $\boldsymbol{U} \in \mathbb{R}^{J_n\times I_n}$ denoted as $\tilde{\boldsymbol{X}}\times_n \boldsymbol{U} \in \mathbb{R}^{I_1\times\cdots\times I_{n-1}\times J_n\times I_{n+1}\cdots\times I_N}$ .The element of $\tilde{\boldsymbol{X}}\times_n \boldsymbol{U}$ is

$$\left(\tilde{\boldsymbol{X}}\times_n \boldsymbol{U}\right)_{i_1\cdots i_{n-1}j_n i_{n+1}\cdots i_N} = \sum_{i_n} x_{i_1 i_2\cdots i_N} u_{j_n i_n} \tag{2}$$

Definition 3: The inner product of two tensors $\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ denoted as

$$\left\langle \tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}}\right\rangle = \sum_{i_1}\sum_{i_2}\cdots\sum_{i_N} x_{i_1\cdots i_n\cdots i_N}\, y_{i_1\cdots i_n\cdots i_N} \tag{3}$$

B. *Tensor Kernel Based on Multi-linear Singular Value Decomposition*

Tensor data can retain the original spatial structure of an object. Viewing the object color image as tensor data can help us exploit the intrinsic geometric relationships of object classes. The object image data belong to the same class reside in a low-dimensional manifold embedded in high-dimensional vector space.

The purpose of the kernel method is to identify and learn the relationship in a data set.

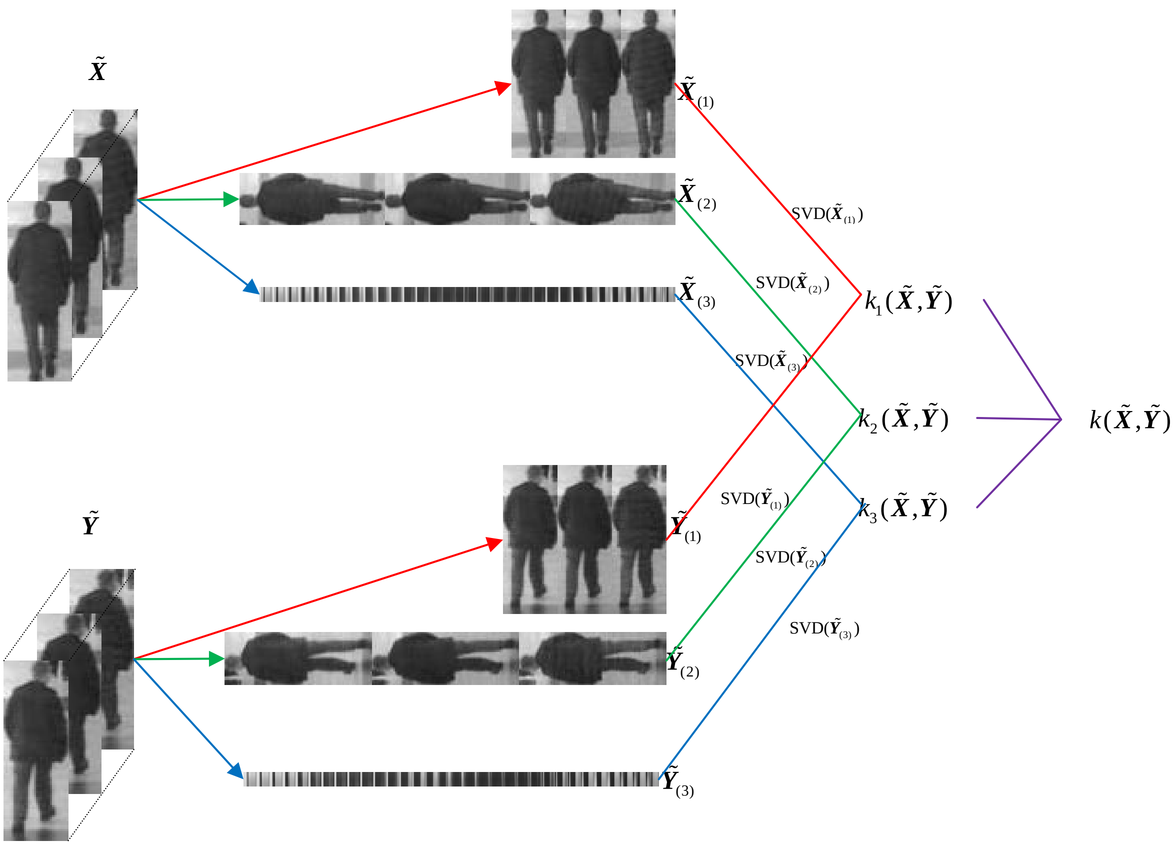


Fig.1 The schematic diagram of tensor kernel based on multi-linear SVD.

The original data can be embedded in high-dimensional feature space through a nonlinear mapping φ . The traditional kernel function $k(x,y)$ for vector data is known as: $\mathrm{R}^N\times\mathrm{R}^N\to\mathrm{R}$ and satisfied $k(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}})=\left\langle \varphi(\tilde{\boldsymbol{X}}),\varphi(\tilde{\boldsymbol{Y}})\right\rangle$, so the kernel function $k(x,y)$ for tensor data must obey the rule: $\mathrm{R}^{I_1\times I_2\times\cdots\times I_N}\times\mathrm{R}^{I_1\times I_2\times\cdots\times I_N}\to\mathrm{R}$.

Kernel methods for non-linear models have proven successful in many computer vision fields[16]. So it is feasible to improve the discriminatory power of supervised tensor-based models using a tensor kernel. The most recent tensor-based techniques are based on matrix unfolding. The multi-linear singular value decomposition of tensor data $\tilde{\boldsymbol{X}}$ can be reduced to the SVD of $\tilde{\boldsymbol{X}}_{(n)}$, where $\tilde{\boldsymbol{X}}_{(n)}$ is the matrix unfolding of $\tilde{\boldsymbol{X}}$, $n\le I_N$,

$$\tilde{\boldsymbol{X}}_{(n)}=U_{\tilde{\boldsymbol{X}}_{(n)}}\Sigma_{\tilde{\boldsymbol{X}}_{(n)}}V_{\tilde{\boldsymbol{X}}_{(n)}}{}^{T} \tag{4}$$

where $U_{\tilde{\boldsymbol{X}}_{(n)}}$, $\Sigma_{\tilde{\boldsymbol{X}}_{(n)}}$, $V_{\tilde{\boldsymbol{X}}_{(n)}}{}^{T}$ are the three components of this SVD, and can be sated in block-partitioned form,

$$U_{\tilde{\boldsymbol{X}}_{(n)}}=\begin{pmatrix}U^1_{\tilde{\boldsymbol{X}}_{(n)}} & U^2_{\tilde{\boldsymbol{X}}_{(n)}}\end{pmatrix} \tag{5}$$

$$\Sigma_{\tilde{\boldsymbol{X}}_{(n)}}=\begin{pmatrix}\Sigma^1_{\tilde{\boldsymbol{X}}_{(n)}} & 0\\ 0 & 0\end{pmatrix} \tag{6}$$

$$V_{\tilde{\boldsymbol{X}}_{(n)}}{}^{T}=\begin{pmatrix}V^1_{\tilde{\boldsymbol{X}}_{(n)}}{}^{T}\\ V^2_{\tilde{\boldsymbol{X}}_{(n)}}{}^{T}\end{pmatrix} \tag{7}$$

Definition 4: Given a dataset $\boldsymbol{X}:\tilde{\boldsymbol{X}}_1,\tilde{\boldsymbol{X}}_2,\cdots,\tilde{\boldsymbol{X}}_m\in\boldsymbol{X}$, the kernel matrix of this set is an $N\times N$ matrix K, and the element k_{ij} is

$$k_{ij}=k(\tilde{\boldsymbol{X}}_i,\tilde{\boldsymbol{X}}_j) \tag{8}$$

A simple template matching object tracking method used in the first m frame images, then we can get the prior acquisition object sample tensor dataset $\{\tilde{\boldsymbol{X}}_1,\tilde{\boldsymbol{X}}_2,\cdots,\tilde{\boldsymbol{X}}_m\}$.

Because $k_{ij}\ge 0$, the kernel matrix K is non-negative, and positive semi-definite.

$k_i(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}})$ correspond to the kernel function in mode i of the two tensors, and can be calculated by:

$$k_i(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}})=\exp\left(-\frac{\left\|V^1_{\tilde{\boldsymbol{X}}_{(i)}}V^1_{\tilde{\boldsymbol{X}}_{(i)}}{}^{T}-V^1_{\tilde{\boldsymbol{Y}}_{(i)}}V^1_{\tilde{\boldsymbol{Y}}_{(i)}}{}^{T}\right\|}{2\sigma^2}\right) \tag{9}$$

Where σ indicates an appropriate bandwidth.

Then the kernel function between tensors is defined as:

$$k(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}})=k_1(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}})k_2(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}})k_3(\tilde{\boldsymbol{X}},\tilde{\boldsymbol{Y}}) \tag{10}$$

The kernel function of colour image tensor data is showed as Figure1.

III. Tensor Kernel Space Projection

Principle component analysis can only handle the linear information in a dataset, a kernel trick may be used in kernel space to distinguish object and background [17].

Given a training dataset $\alpha_1,\alpha_2,\cdots,\alpha_m$, using kernel mapping φ projecting them into kernel space, we get $\varphi(\alpha_1),\varphi(\alpha_2),\cdots,\varphi(\alpha_m)$, Let $\aleph=[\alpha_1,\alpha_2,\cdots,\alpha_m]$ and $\varphi(\aleph)=[\varphi(\alpha_1),\varphi(\alpha_2),\cdots,\varphi(\alpha_m)]$. Projecting $\varphi(\alpha_i)$ in kernel space along projection direction p,

$$\beta_i=p^T\varphi(\alpha_i) \tag{11}$$

The total dispersion of the projected training dataset is

$$\begin{aligned}D&=\frac{1}{m}\sum_{i=1}^{m}(\beta_i-\bar{\beta})(\beta_i-\bar{\beta})^T\\ &=p^T\left[\frac{1}{m}\sum_{i=1}^{m}(\varphi(\alpha_i)-\bar{\alpha})(\varphi(\alpha_i)-\bar{\alpha})^T\right]p\\ &=p^TSp\end{aligned} \tag{12}$$

Where $\bar{\varphi}=\left[\sum_{i=1}^{m}\varphi(\alpha_i)\right]\Big/m$ is the mean value of the dataset in kernel space.

The purpose of this kernel space projection is to find the optimal projection direction p by maximizing the total dispersion of the projected training dataset D. Thus,

$$J(p)=\arg\max_{p}\, p^TSp \tag{13}$$

To maximize the objective function, we need to find the Eigen-value decomposition to S .

$$\begin{aligned}\lambda p=Sp&=\frac{1}{m}\sum_{i=1}^{n}(\varphi(\alpha_i)-\bar{\varphi})^T(\varphi(\alpha_i)-\bar{\varphi})p\\ &=\sum_{i=1}^{m}\left[\frac{1}{m}(\varphi(\alpha_i)-\bar{\varphi})^Tp\right](\varphi(\alpha_i)-\bar{\varphi})\end{aligned} \tag{14}$$

Here p can be regarded as the feature vector of S . It can also be seen that the projection direction p is the linear composition of $(\varphi(\alpha_i)-\bar{\varphi})$, Namely that

$$\psi(\aleph)=\left[(\varphi(\alpha_1)-\bar{\varphi}),(\varphi(\alpha_2)-\bar{\varphi}),\cdots,(\varphi(\alpha_m)-\bar{\varphi})\right] \tag{15}$$

That is

$$p=\sum_{i=1}^{m}c_i(\varphi(\alpha_i)-\bar{\varphi})=\psi(\aleph)C \tag{16}$$

Where $C=\left[c_1,c_2,\cdots,c_l\right]^T$ is the matrix which comprising the linear combination coefficients. Now, solving for the optimal projection direction was transformed into a problem whereby solving the combination coefficient matrix was required, the objective function became:

$$J(C)=\arg\max_{C}\frac{1}{m}C^T\left[\psi(\aleph)^T\psi(\aleph)\right]^2C \tag{17}$$

Let A_m is a $m\times m$ matrix with all entries equal to $1/m$, $\hat{K}=\psi(\aleph)^T\psi(\aleph)$. Then,

$$\widehat{K} = K - A_m K - K A_m + A_m K A_m \tag{18}$$

Eigen-value decomposition to S ,

$$Sp = (\lambda/m)\, p \tag{19}$$

Then

$$\psi(\aleph)^T S\psi(\aleph)C = (\lambda/m)\widehat{K}C \tag{20}$$

$$\widehat{K}C = (\lambda/m)C \tag{21}$$

It is can be seen that C is the matrix composed of the Eigen-vectors corresponding to the first l maximum Eigen-values of $\widehat{K}$.

The tensor kernel space projection algorithm works as shown in Table1:

TABLE 1.
TKSP ALGORITHM

1) Define tensor kernel function $k(\alpha,\beta)$.
2) Compute kernel matrix K and matrix $\widehat{K}$.
3) Eigen-value decomposition to $\widehat{K}$ to obtain Eigen-values and Eigen-vectors, and sort Eigen-values in descending order $\lambda_1 \ge \cdots \ge \lambda_l \ge \cdots \ge \lambda_m$, v_i is the eigenvector corresponding to λ_i . Take the first l Eigen-values, let $c_i = v_i / \sqrt{\lambda_i}\ (i = 1, 2, \cdots, l)$. Then, the projection matrix $C = [c_1, c_2, \cdots, c_l]$.
4)The projected data in kernel space are $\beta = C^T k(\aleph, \alpha)$

IV. Bayesian Sequence Interference Framework For Object Tracking

A. Bayesian Sequence Interference

In the problem of object tracking, the object state in the current frame only related to the object state in the prior frame. Define o_t as the state variables of the object at time *t*. Let $o_t = (x_t, y_t, w_t, h_t)$, where x_t, y_t, w_t, h_t denotes the center coordinate, the width and height of the tracking rectangle.

Given a set of targets $Z_t = \{z_1, z_2, \cdots, z_t\}$, the objective of object tracking is to obtain the optimal estimate value of the hidden state variables o_t . According to Bayesian theorem, it can be obtained in a similar fashion as that of the object state:

$$P(o_t | Z_t) \propto P(z_t | o_t) \int P(o_t | o_{t-1}) P(o_{t-1} | Z_{t-1}) do_{t-1} \tag{22}$$

Where $P(o_t | o_{t-1})$ refers to the state transition model and $P(z_t | o_t)$ refers to the observation model. It can be seen that the observation model $P(z_t | o_t)$ determine the tracking results.

State transition model: This was used to model the object motion between consecutive frames. Because of the irregular movement of object, the state o_t is modeled by independent Gaussian distribution around its counterpart in state o_{t-1} . Described as

$$P(o_t | o_{t-1}) = N(o_t; o_{t-1}, \Delta) \tag{23}$$

Where Δ means the diagonal covariance matrix of x_t, y_t, w_t, h_t , and the element is $\sigma_x^2, \sigma_y^2, \sigma_w^2, \sigma_h^2$. Point to Gaussian distribution, N particles can be randomly generated. According to the particle can obtain multiple states $\{o_t^i, i = 1, 2, \cdots, N\}$. During the computing process, with the increase in the number of particles, the *a posteriori* probability estimate was more accurate, but at the same time, the computational efficiency was low, so a balance was sought between these factors.

Observation model: this was used to measure the similarity between the appearance observation and the object appearance model. Given a drawn particle state o_t^i and a cropped version of the corresponding image patch z_t^i from the frame image I_t . The probability of an image patch being generated from the kernel space is inversely proportional to the difference between image patch and the appearance model, and could be calculated between the negative exponential distance of the projected data and the center of dataset

$$p(z_t^i | o_t) = \exp\left\{ -\left\| s_t^i - \overline{s} \right\|_F / \sigma \right\} \tag{24}$$

Where σ indicates an adjustment factor, $\|\bullet\|_F$ is the Frobenius norm, $s_t^i = C^T k(X, z_t^i)$, $\overline{s} = C^T k(X, \overline{x})$, C is the projection matrix in kernel space, and $\overline{x}$ is the center of dataset in kernel space.

The state o_t^i corresponding to the maximum $p(z_t^i | o_t)$ is the optimal object state at time *t*.

B. Projection Matrix Online Update

Along with the movement of the object in scenarios, the object appearance also changed. The projection matrix in kernel space must be updated on-line so that at time *t*, we have obtained the former *t - 1* time object states. When *t - 1* is the times used as initial training dataset numbers, we recalculated the projection matrix in kernel space using the newly acquired *t - 1* object tensor data points.

The flow chart of whole proposed object tracking algorithm is shown as figure 2.

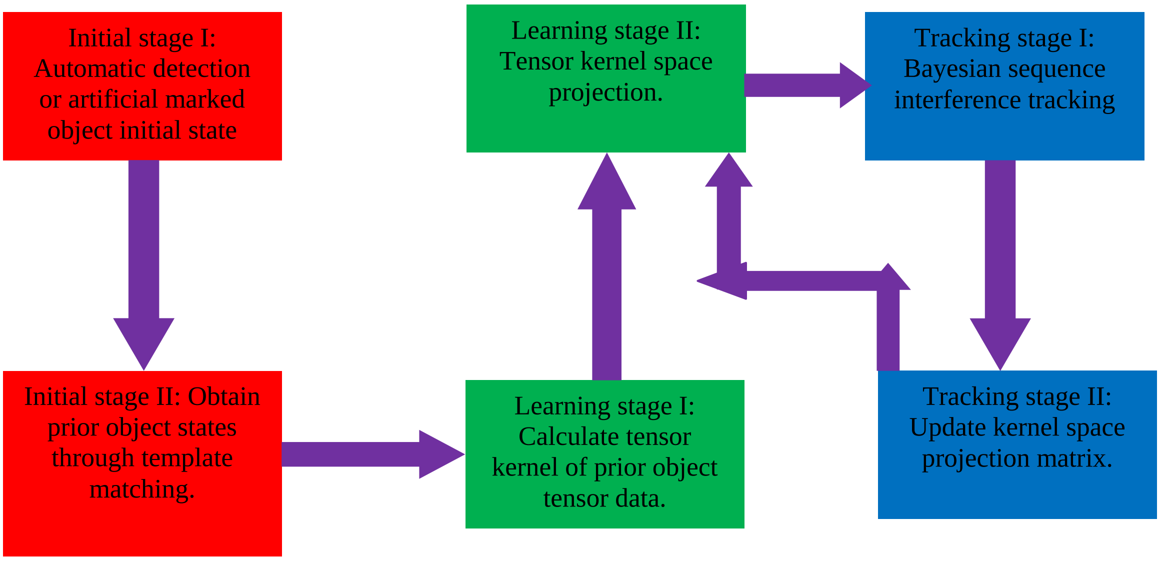


Fig.2 The flow chart of the proposed visual tracking algorithm.

V. Comparative Experiments And Analysis

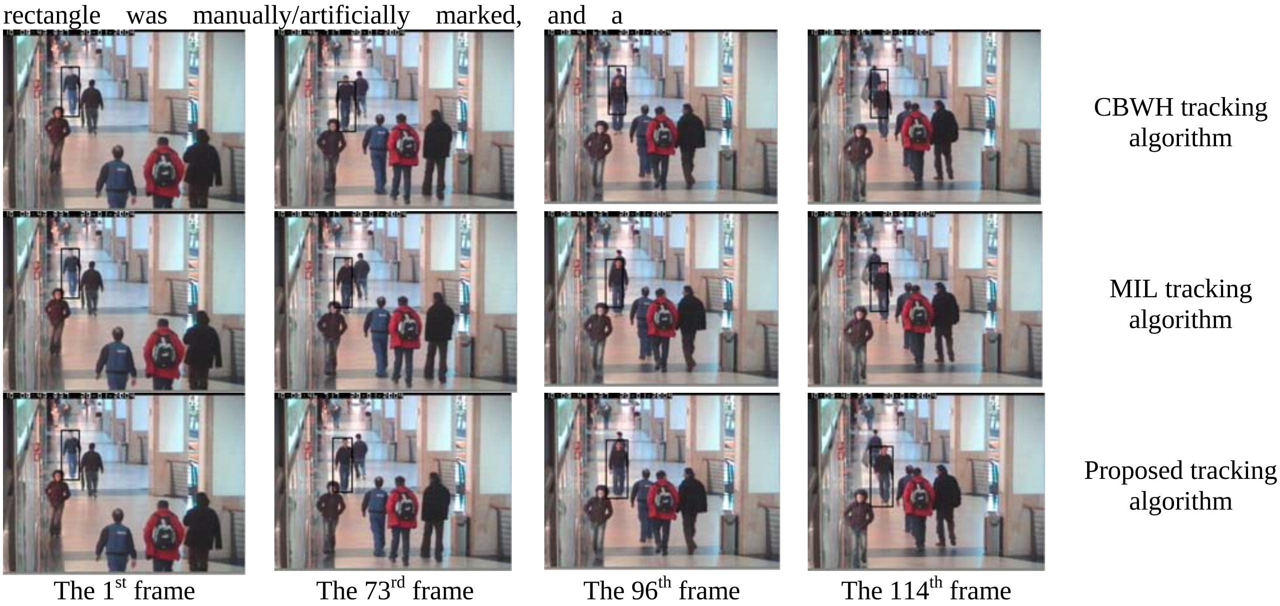
All the experiments were carried out in the MATLAB® 2010a environment running on a Pentium® Dual-Core E5200 processor with a clock-speed of 2.50 GHz.

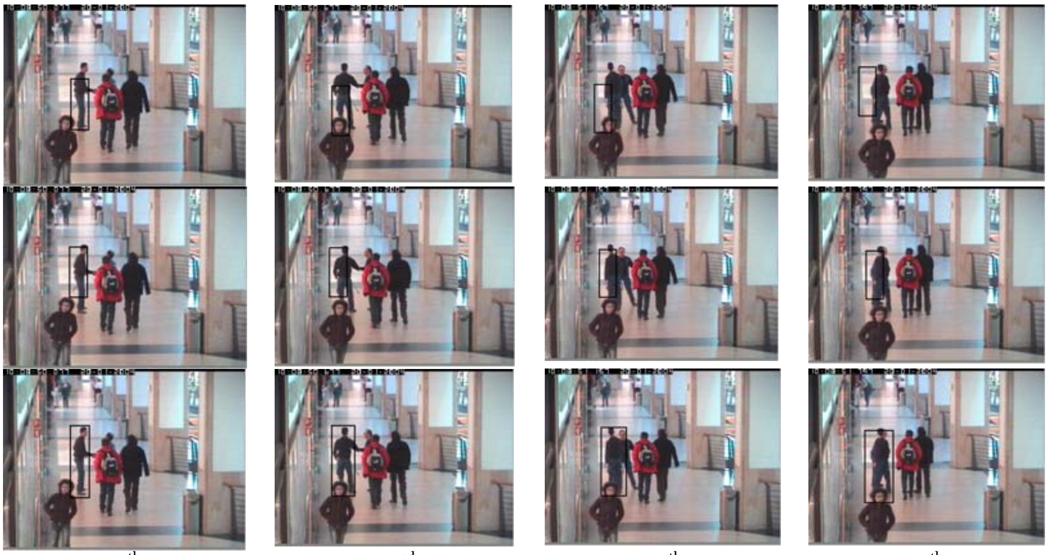
In these experiments, we use Portuguese shopping center video monitoring data to test. The video frame image size was $388 \times 284 \times 3$, and the experimental data were compared with classical CBWH, and MIL, tracking algorithms. The CBWH tracking algorithm select 16×16×16 bin histogram. In MIL tracking algorithm, the feature pool is 250, weak classifiers number is 50 and the learning rate is 0.85. The variance of Gaussian distribution is [2, 2, 0.5, 0.5] in the proposed tracking algorithm. In the initial frame, the object tracking rectangle was manually/artificially marked, and a template matching tracking algorithm was used to get the previous twenty frame object states.

A. *Expeiment A*

We first selected scale changes in an environment of complex surveillance scenario, meanwhile the movement of object was irregular. The object was walking toward the camera. The length of this video was 200 frames.

The tracking results of experiment A are shown as Figure 3. It can be seen from the pictures that when a similar object appeared around the target, the CBWH and MIL tracking algorithms cannot achieve well tracking; our tracking algorithm could adapt thereto. In the tracking progress, the size of the object changed from 28 × 79 to 44 × 111.





CBWH tracking algorithm

MIL tracking algorithm

Proposed tracking algorithm

The 157th frame The 172nd frame The 184th frame The 200th frame

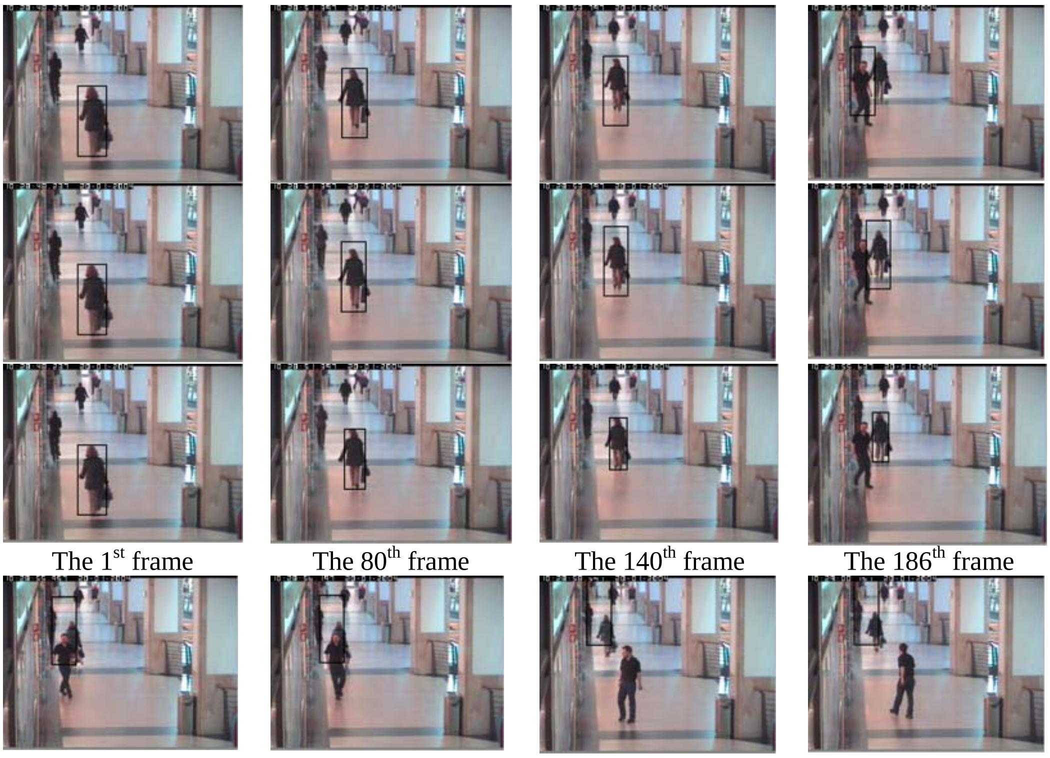
Fig.3 The tracking results of experiment A.

B. *Expeiment B*

The second video had scale changes and occlusion in the surveillance scenarios studied. The object was walking away from the camera. The object was walking away from the camera. The length of this video was 300 frames.

The tracking results from experiment B are shown in Figure 4. It can be seen from the pictures that when the object was almost occluded by other moving objects, the CBWH and MIL tracking algorithms suffered interference from the other objects, and could not have tracked the target. Our tracking algorithm overcame the problem of occlusion and tracked the target.

In the tracking progress, the size of object changed from 39× 112 to 21× 64.



CBWH tracking algorithm

MIL tracking algorithm

Proposed tracking algorithm

The 1st frame The 80th frame The 140th frame The 186th frame

CBWH tracking algorithm

Fig.4 The tracking results of experiment B.

C. *The Evaluation of Tracking*

To quantitatively compare the experimental results of the tracking algorithms, we used two metrics to describe it: *Tracking errors* and *Tracking correct ratio*. We initially hand-labelled the object state in each experimental scenario.

The error in X-axis e_x is

$$e_x = \left| x_e - x_g \right| \tag{25}$$

Where x_e, x_g are the X-axis coordinates of the center of the experiment tracking rectangle and the ground truth rectangle.

The error in Y-axis e_y is

$$e_y = \left| y_e - y_g \right| \tag{26}$$

Where y_e, y_g are the Y-axis coordinates of the center of the experiment tracking rectangle and the ground truth rectangle.

The error in center coordinate e_c is

$$e_c = \sqrt{(x_e - x_g)^2 + (y_e - y_g)^2} \tag{27}$$

The error in experiment A and B are shown as Table 2 and 3. The data in bold refer to optimal results.

TABLE 2
THE ERRORS OF EXPERIMENT A

Items(pixels)	x-errors	y-errors	c-errors
Proposed Method	**1.7145**	**2.3905**	**3.2788**
CBWH	4.0725	15.6325	16.9349
MIL	4.2025	5.0400	6.8776

TABLE 3
THE ERRORS OF EXPERIMENT B

Items(pixels)	x-errors	y-errors	c-errors
Proposed Method	**1.8276**	**2.4318**	**3.2209**
CBWH	7.0183	19.1450	21.0142
MIL	12.7800	19.4817	23.5559

The tracking correct ratio r is

$$r = \frac{area(R_e \cap R_g)}{area(R_e \cup R_g)} \tag{28}$$

Where R_e is the experiment tracking rectangle, R_g is the ground truth rectangle, $area()$ means the area of the region. It was thought that if the tracking correctly ratio was greater than 0.5, the tracking in this frame was successful. The tracking correctly ratios in two scenarios are shown as Figures 5 and 6. The red line indicates the results of the proposed tracking algorithm, the green line indicates the CBWH tracking algorithm and the blue line indicates the MIL tracking algorithm. It can be seen that our proposed tracking algorithm had better results than the other two tracking algorithms. Also, the tracking correctly ratio of our algorithm was almost always greater than 0.5, implying that the algorithm was successful.

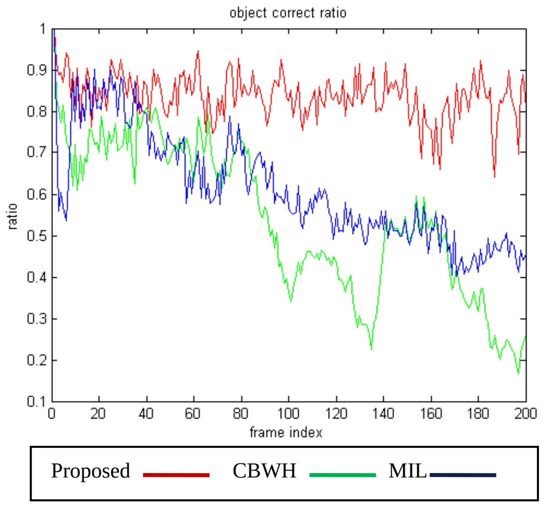


Fig.5 The tracking correct ratio in experiment A.

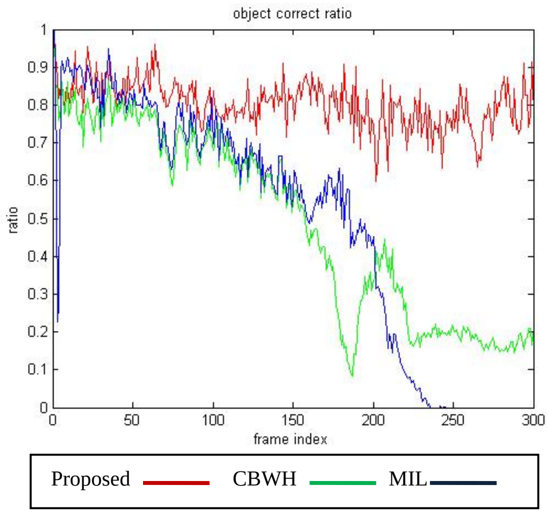


Fig.6 The tracking correct ratio in experiment B.

VI. Conclusion

In this work, we proposed the use of a tensor kernel space projection for object tracking in video surveillance scenarios. Considering the spatial structure of the object image, we defined a tensor kernel function based on multi-linear singular value decomposition. We projected the object image tensor data into kernel space, and combined this with a Bayesian sequence interference tracking framework to achieve object tracking. To adapt to the object appearance changes, we updated the projection matrix on-line and also analyzed the performance of our proposed tracking algorithm in an assessment against challenging real-world video surveillance scenarios and compared the output with two classical tracking algorithms. The experimental results demonstrated the accuracy and robustness of the proposed tracking algorithm.

References

[1] A. Yilmaz, O. Javed and M. Shah, "Object tracking: a survey," *ACM Computing Surveys,* vol. 38, pp. 1-45, 2006.

[2] H. Yang, L. Shao, F.Zheng, L. Wang and Z. Song, "Recent advances and trends in visual tracking: a review," *Neurocomputing,* vol. 74, pp. 3823-3831, 2011

[3] Z. Feng, B. Yang, Y. Zheng, H. Tang and Y. Li, "Initialization of 3D human hand model and its applications in human hand tracking," *Journal of Computers,* vol. 7, pp. 419-426, February 2012.

[4] D.A. Ross, J. Lim, R. Lin and M. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision,* vol. 77, pp. 125-141, 2008.

[5] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, pp. 564-577, 2003.

[6] H. Zhou, Y. Yuan and C. shi, "Object tracking using sift features and mean shift," *Computer Vision and Image Understanding,* vol. 113, pp. 345-352, 2009.

[7] J. Ning, L. Zhang, D. Zhang and C. Wu, "Robust mean shift tracking with corrected background-weighted histogram," *Iet Computer Vision,* vol. 6, pp. 62-69, 2012.

[8] T. Gao, Z. Yao, P. Wang, C. Wang and J. Yang, "Automatic Stable Scene based Moving multi-target detection and tracking," *Journal of Computers,* vol. 16, pp. 2647-2655, Decemble 2011.

[9] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 33, pp. 2259-2272, 2011.

[10] J. Gao, W. Li, D. wu, "Multi-feature fusion tracking based on a new particle filter," *Journal of Computers,* vol. 7, pp. 2939-2947, December 2012.

[11] S. Avidan, "Support vector tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 26, pp. 1064-1072, 2006.

[12] S. Avidan, "Ensemble tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, pp. 261-271, 2007.

[13] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via online boosting," *British Machine Vision Conference,* Edinburgh, British, vol. 1, pp. 47-56, 2006.

[14] B. Babenko, M. Yang and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 33, pp. 1619-1632, 2011.

[15] H. Lu, K.N. Plataniotis and A. N. Venetsanopoulos, "A survey of multilinear subspace learning for tensor data," Pattern Recognition, vol. 44, pp. 1540-1551, 2011.

[16] M. Signoretto, L. De. Lathauwer and J. A. K. Suykens, "A kernel-based framework to tensorial data analysis," *Neural Networks,* vol. 24, pp. 861-874, 2011.

[17] B. Scholkopf, A. J. Smola and K. R. Muller, "Nonlinear component analysis as a kernel eigen-value problem," *Neural Computing,* vol. 10, pp. 1299-1319, 1998.

Jiashu Dai received his B.S. degree from Mudanjiang Normal University, Mudanjiang, China, in 2008. He is currently a candidate for Ph.D. degree in College of Computer Science and Technology, Harbin Engineering University. His research interests include computer vision, pattern recognition as well as image processing.

Tingquan Deng received his B.S. degree in Mathematics, M.S. degree in Applied Mathematics, and Ph.D degree in Fundamental Mathematics from Harbin Institute of Technology, Harbin, China, in 1987, 1990 and 2002, respectively. He was a visiting scholar from Center for Mathematics and Computer Science, Amsterdam, the Netherland from 1999 to 2000 for one year and a postdoctoral research fellow in Department of Automation, Tsinghua University, Beijing, China from 2003 to 2005. Currently, he is a professor in College of Science as well as in College of Computer Science and Technology, Harbin Engineering University, Harbin, China. His research interests include uncertainty theory, image processing and pattern recognition, and data mining and machine learning.