Regular Paper

Review on Kernel based Target Tracking for Autonomous Driving

Yanming Wang[1] Jiguang Yue[1] Yanchao Dong[1] Zhencheng Hu[2,a)]

Received: April 12, 2015, Accepted: October 2, 2015

Abstract: Significant progress has been made in the field of autonomous driving during the past decades. However, fully autonomous driving in urban traffic is still extremely difficult in the near future. Visual tracking of vehicles or pedestrians is an essential part of autonomous driving. Among these tracking methods, kernel-based object tracking is an effective means of tracking in video sequences. This paper reviews the kernel theory adopted in target tracking of autonomous driving and makes a qualitative and quantitative comparison among several well-known kernel based methods. The theoretical and experimental analysis allow us to conclude that the kernel based online subspace learning algorithm achieves a good trade-off between the stability and real-time processing for target tracking in the practical application environments of autonomous driving. This paper reports on the result of evaluating the performances of five algorithms by using seven video sequences.

Keywords: visual tracking, kernel-based tracking, kernel-based learning

1. Introduction

Driver assistance systems and autonomous driving have gained rapid development in the last decades for the purpose of reducing traffic accidents and enhancing driving comfort [1]. The earliest concept of autonomous driving has been proposed since General Motors presented a mock-up of an automated vehicle highway system at the 1939 World's Fair [2]. During the 1990s, many autonomous driving projects have been supported by government, such as the National Automated Highway Systems Consortium (NAHSC) project in U.S., the Assist Highway Systems Research Association (AHSRA) project in Japan, the PROMETHEUS project in Germany and the La Route Automatisée (LARA) project in France [3]. The early autonomous driving was designed for low-speed freeway scenarios and has the function of lane changes, obstacle avoidance and close-headway platooning [4]. These applications usually need the help of supported infrastructures (such as magnetic markers or dedicated lanes). As the rapid development of information processing systems and sensor technology autonomous driving has been extended to the urban traffic environment in recent years.

Autonomous driving will perform the lateral or longitudinal control automatically according to the perception result of driving environment. For example, with the lateral control on steering actuator, the vehicle can change driving lane to realize the overtaking or obstacle avoidance; and with the longitudinal control on brake and throttle actuator the vehicle can realize the automated platooning. Many sensors have been employed to provide the surrounding information for the lateral and longitudinal controls, such as infrared image sensor, sonar, laser range finder, radar, and so on. Compared to these sensors, visual sensors can provide much rich information such as the shape and texture with high resolution. The perception module will use this visual information to detect, recognize and track interesting targets like lane, vehicle, pedestrian, traffic sign and traffic light. Therefore, visual perception has been a prime technology for autonomous driving [5], [6]. Target detection and tracking are the two import part of visual perception, recent comprehensive reviews of computer vision based vehicle and pedestrian detection can be found in Refs. [7] and [8]. However target tracking for autonomous driving has not been driven enough attention.

One of essential task in autonomous driving is to track surrounding objects, which includes the position and dynamic information of the certain moving objects (nearby vehicles and pedestrians) or stable objects (traffic sign and traffic lights) encountered in the environment. In the semi-autonomous driving the driver face has to be detected and tracked to analysis the fatigue driving. Many tracking algorithms in computer vision field can be utilized for the target tracking in autonomous driving and promising results were reported in recent years [9], [10]. This paper reviews some recent tracking techniques which can be utilized for vision based autonomous driving system, especially the tracking methods based in kernel theory.

Target tracking is the process of finding the most matched candidates with the target template in the sequence of videos. Typically, a visual tracker contains the components of target representation, target localization, filtering and data association, these processes can be combined to improve the robustness and efficiency of a tracker. However, the loss of information caused by projection of the 3D world onto a 2D image, noises in im-

[1] Tongji University, School of Electronics and Information, Shanghai, 201804 China
[2] Kumamoto University, Department of Computer Science, Chuo, Kumamoto 860–8555, Japan
[a)] hu@cs.kumamoto-u.ac.jp

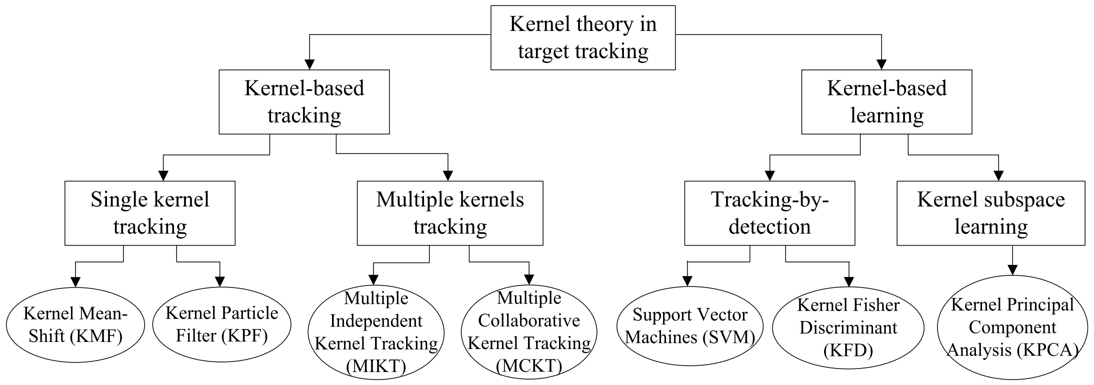

Kernel theory in target tracking
Kernel-based tracking
Kernel-based learning
Single kernel tracking
Multiple kernels tracking
Tracking-by-detection
Kernel subspace learning
Kernel Mean-Shift (KMF)
Kernel Particle Filter (KPF)
Multiple Independent Kernel Tracking (MIKT)
Multiple Collaborative Kernel Tracking (MCKT)
Support Vector Machines (SVM)
Kernel Fisher Discriminant (KFD)
Kernel Principal Component Analysis (KPCA)

Fig. 1 An overview of the methodology of this paper.

ages, complex and non-rigid object motion, occlusions between object-to-object and scene-to-object, illumination changes as well as the requirement of real-time processing pose great challenges on target tracking [11]. Many tracking algorithms have been proposed in the past decades and significant progress has been made. Generally, the properties of a tracking algorithm are evaluated by the qualitative comparisons of robustness, accuracy, reliability, adaptability as well as working in real time. Among these tracking methods utilized in autonomous driving, kernel-based target tracking has gained approval due to its succinct description, nonlinear representation and computational efficiency.

The applications of kernel theory in target tracking can be classified into two aspects: One is acting as the description of target in the spatially-weighted intensity histogram, combines with mean shift algorithm for tracking, named as *kernel-based tracking*; the other is corresponding to a scalar product in the high dimensional nonlinear feature space introduced by some machine learning algorithms such as Kernel Support Vector Machine (kernel-SVM), Kernel Fisher Discriminant (KFD) analysis and Kernel Principal Component Analysis (KPCA), known as *kernel-based learning*. In this paper, we briefly review the recent progresses in these two aspects, the methodology of this paper can be found in **Fig. 1**. We do not attempt a full treatment of all available literatures; rather, we introduce some basic concepts of kernel theory and exhibit its applications or variants in target tracking. A qualitative evaluation is conducted at the end of each part, and a quantitative comparison of several well-known kernel based trackers is made by using publicly available test sequences database of vehicles and pedestrians. The performance of these trackers is discussed and their strengths and weaknesses are highlighted. The aim of this paper is to provide a feasible tracking method in autonomous driving for readers and propose some promising future directions in target tracking.

The arrangement of this paper is as follows: In Section 2, some basic concepts of kernel density estimation and kernel trick are briefly presented. In Section 3, the kernel-based target tracking is introduced, where both simple kernel and multiple kernel methods are to be discussed and a qualitative comparison is given. In Section 4, the kernel-based learning algorithm for tracking is summarized. We also introduce the basic idea of kernel-SVM, KFD and KPCA and analysis the kernel trick applied in these algorithms. A quantitative comparison of five tracking approaches for seven video sequences is proved and discussed in Section 5. The conclusion and discussion on future directions are presented in Section 6.

2. Kernel Description

This section briefly introduce some basic concepts of kernel density estimation and kernel trick, which will be helpful for readers to understand the algorithms to be mentioned in the following sections.

2.1 Kernel Density Estimation

Kernel density estimation (KDE) or Parzen estimation is a kind of nonparametric density estimation method. Given N observations $\{\boldsymbol{X}_i\}_{i=1,\dots,n}$ drawn form an unknown probability density $p(\boldsymbol{X})$ in D-dimensional space $\boldsymbol{R}^{\mathrm{D}}$. The density at $\boldsymbol{X}$ can be estimated as Ref. [12]:

$$p(\boldsymbol{X}) = \frac{1}{N \cdot h^D} \sum_{n=1}^{N} K\left(\frac{\boldsymbol{X} - \boldsymbol{X}_i}{h}\right) \tag{1}$$

where $K(\cdot)$ is the kernel function having the following definition:

$$K(\boldsymbol{u}) = \begin{cases} 1 & |u_i| < \frac{1}{2},\ i = 1, \dots, D \\ 0 & \text{else} \end{cases} \tag{2}$$

With the definition of (2), $K\left(\frac{X-X_i}{h}\right)$ is 1 if the observation $\boldsymbol{X}_i$ lies inside a cube with side h centered on $\boldsymbol{X}$, and 0 otherwise; $\sum_{n=1}^{N} K\left(\frac{X-X_i}{h}\right)$ is the total number of observations that lies beside $\boldsymbol{X}$. If h is small enough, the probability associated with the adjacent region of $\boldsymbol{X}$ can be calculated as $P = \frac{1}{N} \sum_{n=1}^{N} K\left(\frac{X-X_i}{h}\right)$. Furthermore, if $p(\boldsymbol{X})$ is roughly constant over this region, we have $p(\boldsymbol{X}) = P/h^D$, h^D is the volume of a hypercube of side h in D dimensions, as (1) is explained.

Equation (2) is named as uniform kernel or Parzen window, h is the bandwidth of the kernel. The profile of a kernel K is defined as a function $[0, \infty) \to \boldsymbol{R}$, such that $K(\boldsymbol{X}) = k(\|\boldsymbol{X}\|^2)$, other commonly used kernels and their profiles can be found in **Table 1**, all these kernels should satisfy the following conditions [13]:

$$\begin{aligned} &\int_{R^D} K(\boldsymbol{X}) d\boldsymbol{X} = 1 && \lim_{\|\boldsymbol{X}\| \to \infty} \|\boldsymbol{X}\|^D K(\boldsymbol{X}) = 0 \\ &\int_{R^D} \boldsymbol{X} K(\boldsymbol{X}) d\boldsymbol{X} = 0 && \int_{R^D} \boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}} K(\boldsymbol{X}) d\boldsymbol{X} = \boldsymbol{C}_K \boldsymbol{I} \end{aligned} \tag{3}$$

Table 1 The commonly used kernels in kde and their profiles.

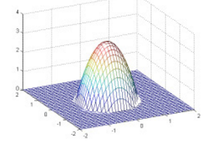
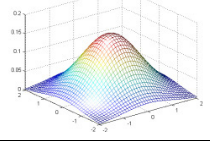
Kernel name	Kernel function	Profile of kernel	2D kernel space
Epanechnikov kernel	$K_E(\boldsymbol{X}) = \begin{cases} \frac{1}{2} C_D^{-1}(D+2)(1-\Vert\boldsymbol{X}\Vert^2) & \Vert\boldsymbol{X}\Vert \le 1 \\ 0 & else \end{cases}$ where, C_D is the volume of the unit D-dimensional sphere.	$k_E(x) = \begin{cases} 1-x & 0 \le x \le 1 \\ 0 & x \ge 1 \end{cases}$	
Gaussian kernel	$K_N(\boldsymbol{X}) = (2\pi)^{-D/2} \cdot exp(-\frac{1}{2}\Vert\boldsymbol{X}\Vert^2)$	$k_N(x) = exp(-\frac{1}{2}x)$	

Table 2 Some widely used kernels in kernel trick.

Kernel name	Kernel function expression
Gaussian kernel:	$k(\boldsymbol{X}, \boldsymbol{X}') = \exp(-\frac{\Vert\boldsymbol{X}-\boldsymbol{X}'\Vert^2}{2\sigma^2})$
Polynomial kernel:	$k(\boldsymbol{X}, \boldsymbol{X}') = (\boldsymbol{X}^T \cdot \boldsymbol{X}' + c)^p$ where $p \in \mathbb{N}$ and $c \ge 0$
Fisher kernel [18]:	$k(\boldsymbol{X}, \boldsymbol{X}') = g(\theta, \boldsymbol{X})^T \boldsymbol{F}^{-1} g(\theta, \boldsymbol{X}')$ where $g(\theta, \boldsymbol{X})$ is the Fisher score, $\boldsymbol{F}$ is the fisher information matrix.
Kullback–Leibler kernel [19]:	$K\left(p(\boldsymbol{X}\vert\theta_i), p(\boldsymbol{X}\vert\theta_j)\right) = exp(-A \cdot D\left(p(\boldsymbol{X}\vert\theta_i), p(\boldsymbol{X}\vert\theta_j)\right) + B)$ where $D\left(p(\boldsymbol{X}\vert\theta_i), p(\boldsymbol{X}\vert\theta_j)\right)$ is the Kullback-Leibler (KL) divergence, A and B are the scale and shift factors.
Probability product kernels (PPK) [20]:	$k(p, p') = \int_x p(x)^\rho \cdot p'(x)^\rho \, dx$

2.2 Kernel Trick

Kernel concept was first introduced by Aizerman et al. [14], but it did not draw considerable interest until Boster et al. [15] re-introduced it in SVM. Kernel function is defined to represent the scalar or inner product in some high dimensional feature space:

$$k(\boldsymbol{X}, \boldsymbol{X}') = \emptyset(\boldsymbol{X})^T \cdot \emptyset(\boldsymbol{X}') \quad (4)$$

where $\emptyset(\boldsymbol{X})$ is a nonlinear feature space mapping. In fact, the kernel functions given in Section 2.1 are the special kind of this definition as $\Vert\boldsymbol{X}\Vert^2 = \boldsymbol{X}^T \cdot \boldsymbol{X}$ is still a scalar product. If an algorithm is formulated in such a way that the input vector $\boldsymbol{X}$ enters only in the form of scalar products, then we can replace this scalar product with a suitable choice of kernel, this kernel substitution is known as *kernel trick*.

A valid kernel is defined as $\emptyset(\boldsymbol{X})^T \cdot \emptyset(\boldsymbol{X})$, if the feature mapping is known, the corresponding kernel can be found. However, in most cases, $\emptyset(\boldsymbol{X})$ is unknown explicitly. Shawe-Taylor and Cristianini [16] gave a necessary and sufficient condition for $k(\boldsymbol{X}, \boldsymbol{X}')$ to be a valid kernel, that is the Gram matrix $\boldsymbol{K}$ (whose elements are given by $\boldsymbol{K}_{nm} = k(\boldsymbol{X}_n, \boldsymbol{X}_m)$) should be positive semi-definite for all possible choices of the set $\{\boldsymbol{X_n}\}$.

Table 2 lists some widely used kernels and it is referred to [12] and [17] for detailed descriptions.

3. Kernel-based Tracking

Template matching is a common approach for target tracking in video sequences. Image intensity, color features or image gradients are often used to form templates, and then the similarity measure is utilized to calculate the position of the target by matching the templates and candidates. However, template matching is a brute force method, which makes it a high computation cost. By limiting the candidates' position at the target neighboring locations in the previous frame the computation cost can be reduced [21]. Kernel-based object tracking (KBOT) is a kind of this strategy. It was first proposed by Comaniciu et al. [22], and then many researchers' works mainly focuses on the following questions:

- How to estimate the scale and orientation changes of the target?
- Which kind of kernel profile is suitable for tracking?
- How to adapt the bandwidth of the kernel?
- How many kernels are appropriate for target tracking?

The first three questions should be solved in both single and multiple kernel tracking, so we divide this section into single kernel and multiple kernel tracking in KBOT.

3.1 Single Kernel Tracking

In most common cases, one kernel function is enough to represent the weighted-histogram of the target. This representation is simple and needs less computation. The kernel-based mean-shift is introduced first and then the kernel particle filter is discussed.

3.1.1 Kernel-based Mean-shift

Mean-shift algorithm is a nonparametric method based on KDE for mode seeking [23]. Typically, in mean-shift tracking the target is represented by a circular region and the color histogram is used as features. However, this representation only contains the intensity information of the target and this is not enough in applications. Comaniciu et al. [22] utilized an isotropic kernel to regularize the histogram-based target feature representations, which considered the descriptions of both intensity values and spatial positions of the target region together. The target model is represented by its *pdf* in the feature space as $\boldsymbol{q_u} = C \cdot \sum_{i=1}^{n} k(\Vert\boldsymbol{X_i^*}\Vert^2) \cdot \delta[b(\boldsymbol{X_i^*}) - u]$, where $k(\cdot)$ is the kernel function as defined in Section 2.1, $\{\boldsymbol{X_i^*}\}_{i=1,\dots,n}$ be the normalized pixel locations in the region defined as the target model, δ is the Kronecker delta function, $u = 1 \dots m$ is the bins of histogram,

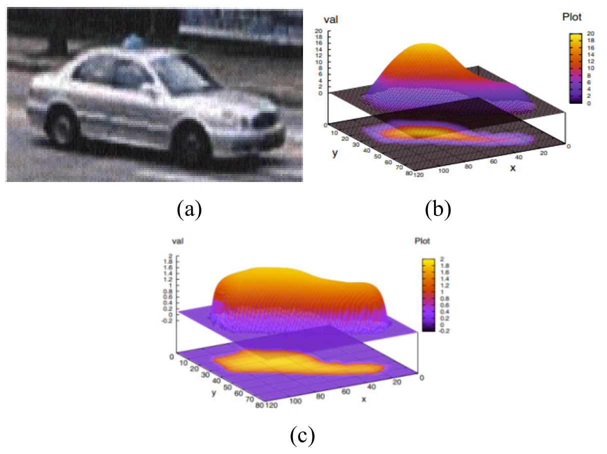


Fig. 2 (a) The original image, (b) the kernels generated by target shape, (c) the gradient kernel generated from (b), (Figure from Ref. [28]).

C is the normalization constant such that $\sum_{u=1}^{m} \boldsymbol{q_u} = 1$, the function b: $\boldsymbol{R}^2 \to \{1, .., m\}$ associates to the pixel at location $\boldsymbol{X}_i^*$ the index $b(\boldsymbol{X}_i^*)$ of its bin in the quantized feature space. The target candidates $\{\boldsymbol{X_i}\}_{i=1,...,n_h}$ centered at $\boldsymbol{y}$ can also be written in the same way as $\boldsymbol{p_u}(\boldsymbol{y}) = C_h \cdot \sum_{i=1}^{n_h} k(\|\frac{y-X_i}{h}\|^2) \cdot \delta[b(\boldsymbol{X_i}) - u]$. The Bhattacharyya coefficient is employed as the similarity measure, and then mean-shift is performed to obtain the target position. The kernel profile used here is Epanechnikov kernel, which acts as the weights of the feature histogram. The Epanechnikov kernel is a convex and monotonic decreasing kernel function, which assigns smaller weights to pixels farther from the target center. This kernel application increases the robustness of the density estimation since the peripheral pixels are the least reliable, being often affected by occlusions (clutter) or interferences from the background. This tracking approach based on kernel function is termed as KBOT.

In KBOT, the bandwidth of kernel profile corresponds to the search region of the target and it should be adapted automatically when the scale of the target changes. Many algorithms have been proposed to adapt the scale and orientation changes of target. Object affine model is employed to describe scaling problem in Ref. [24], and the object corner correspondences between two frames are used to estimate the parameters of this model. With the registration of object centroid in consecutive frames by backward tracking, scaling magnitude in the affine model can be estimated. However, this method needs more memory space and high computation cost. Parameswaran et al. [25] used tunable kernels to track humans walking, in which the target is divided into several blocks and each block corresponding to a kernel with different bandwidths. However, these bandwidths are specified in the training of sample images from motion capture and the dependence of the optimal bandwidth parameters on the pose of the person is not considered. An asymmetric target kernel is proposed to track the target with scale and orientation changes [26], [27], [28], [29]. This asymmetric kernel represents the target's shape (as shown in **Fig. 2**), reducing the estimation-bias introduced by non-target regions residing inside the kernel. The affine kernel transformation was proposed by Leichter et al. [30], this tracker incorporated the target boundary cue into the tracking process and the kernels were affinely transformed then the scale of the target changed. Additionally, Li et al. [31] applied the basic set analysis to estimate the scale of the target. Zhang et al. [32] used the canny edge detection to determine the change tendency of the target. Some deformable contours tracking methods [33], [34] were utilized to handle the scale change problem. These scale adaption algorithms are mostly depend on the additional information such as the corner, edge or contour of the target, these additional information is not so stable to the illumination change and non-rigid deformation. What's more, the affine model or transform cannot handle the out of plane motions.

Occlusion is another intractability issue in KBOT, approaches that based on fragments or local information have been proposed to handle this problem [35], [36], [37], [38]. In this method target is divided into several fragments, the color histogram of each target and candidate-patch is represented as $\{\boldsymbol{q}^{(k)}\}_{k=1,..,K}$ and $\{\boldsymbol{p}^{(k)}(y)\}_{k=1,...,K}$, where, K is the number of fragments. Then the KBOT is used to calculate the new candidate locations for each fragment as $\{\hat{\boldsymbol{y}}^{(k)}\}_{k=1,...,K}$. The new target position $\hat{\boldsymbol{y}}$ is computed as the weighted sum of the K candidate locations, which is formulated as $\hat{\boldsymbol{y}} = \sum_{k=1}^{K} \lambda_k \cdot \hat{\boldsymbol{y}}^{(k)}$ where λ_k is the weight of the k-th fragment determined by the proportion of target and background distributions. However, this method doesn't consider the relations in these fragments and is not suitable for non-rigid target tracking. The relationship of local and global information should also be considered, the local location of each fragment may be affected by the partial occlusion and drift away from the ground truth, and how to detect this drifting and use the suitable local information to obtain the global is still an open issue.

Another limitation of KBOT is the inability to track fast moving objects. Ali et al. [39] calculated the Gaussian pyramids of the image, and the KBOT is first applied to the most coarsest image level l, then the estimated location at the level l is transferred to the next higher resolution level $(l-1)$ as an initial guess and the same procedure is performed form level $(l-1)$ to $(l-2)$ and so on till the highest resolution image level comes. This method could easily track the fast moving targets having larger motions as compared to their sizes. However, the kernel bandwidth in the different pyramid levels keeps constant. In the coarsest image, kernel includes large part of non-target regions which bias the motion estimation and results in loss of the tracked target. A novel multi-bandwidth mean shift procedure was proposed by Shen et al. [40], which combines simulated annealing algorithm with the kernel-based mean-shift tracking process, termed as annealed mean-shift. The bandwidth of the procedure plays the same role as the temperature in conventional annealing. The proposed algorithm can offer considerable promise in finding the true target location even when it was initialized from a distant point. The fast moving target tracking can be solved by increasing the search bandwidth of KBOT and this can be realized by using pyramids or multi bandwidth search strategy. However, as the bandwidth increase more background information is included and that made the algorithm unstable especially when the background has the similar features with the target or other similar object exists within the background.

More flexible algorithms of KBOT have been proposed in the past decades. Tyagi et al. [41] extended KBOT to 3D by combining evidence from multiple calibrated cameras. This algorithm uses a feature level fusion framework to track the target directly in 3D space, and could be bootstrapped with an automatic re-initialization technique based on clustering 3D point clouds of the foreground targets. This tracker works in 3D space, since the real size of the object does not change over time, the bandwidth of the kernel function remains constant in the tracking process. Birchfield and Rangarajan [42] proposed the spatiogram feature in kernel-based trackers. Spatiogram is identical to a histogram of its features, except that it contains additional spatial mean and covariance for each histogram bin. Spatiogram captures a richer description of the target to increase robustness in tracking. Pouladzadeh et al. [43] combined the mean-shift method with LBP based tracking algorithm to achieve more accurate results. Ju et al. [44] proposed a fuzzy color histogram generated by a self-constructing fuzzy cluster to reduce the inference from lighting changes in the mean-shift tracking algorithm. Mazinan and Amir-Latifi [45] carried out the motion information by a binary mask and a new kernel function was obtained by multiply this mask. This kernel can overcomes the clutter and tracks the target having the similar color with the background. Liu [46] constructed the multiple feature pseudo-color images (MFPCIs) including the Gabor and entropy features, KBOT is directly used in MFPCIs to realize the tracking in infrared images. Wang et al. [47] incorporated shape knowledge into the appearance model of kernel based trackers, etc. Some other similarity measure criteria are utilized in kernel tracking; Yang et al. [48] measured the similarity in average separation criterion in cluster analysis and Leichter [49] employ the Cross-bin metrics in mean-shift tracking. Minwoo et al. [50] combined the mean-shift and Belief Propagation (BP) for multi-target tracking and the adaptive binning color model was utilized in Ref. [51] for mean-shift tracking.

KBOT is a simple and effective algorithm for target tracking due to its low computational complexity and robustness of tracking both rigid and non-rigid targets. So it can be used in autonomous driving to satisfy the real-time processing, however, in the complex environment this method may be less robustness due to the simple target representation. Some cautions would be helpful when adopt this algorithm in real-time video sequence.

- In most cases, KBOT is a color feature based tracking. It should be combined with other features to handle the complex environments, i.e., background with the similar color as target, sudden light changes and occlusions.
- The target representation weighted-histogram should be auto-update according to the tracking result, especially for non-rigid target tracking.
- It is necessary for applying multiple template images in target tracking.

It should also be noticed that an appropriate tracking method should consider the specific of tracking target and the surroundings.

3.1.2 Kernel Particle Filter

Particle Filter (PF) has been widely used to deal with nonlinear and non-Gaussian systems in multi-modal visual tracking problems, it's a hypothesis tracker based on recursive Bayesian filter with Monte Carlo sampling. However, PF usually demands large numbers of particles to estimate the posterior probability density function of the state variables, which leads to heavy computation cost and thus limits its applications in real-time tracking.

Kernel Particle Filter (KPF) was proposed by Chang et al. [52], [53] to handle this problem, KPF introduced KDE and mean-shift in the traditional frameworks of PF to reduce the number of particles that demanded for tracking. Given the particles and the associated weights at time t, KPF can be generalized as follows:

- KDE is used to estimate the posterior density $\hat{p}(\boldsymbol{x}_t \mid \boldsymbol{Y}_t)$, where, $\boldsymbol{x}_t$ is the target state and $\boldsymbol{Y}_t$ is the observation at time t.
- Mean-shift is applied to move particles along the gradient direction toward the modes of the posterior estimation and the new particles are re-weighted.
- The posterior density $p(\boldsymbol{x}_t \mid \boldsymbol{Y}_t)$ is calculated based on these new particles and weights.

Some applications and modifications have been made in recent years. Schmidt et al. [54] presented the KPF for 3D body tracking in the video stream acquired from the un-calibrated camera. Zhang and Wong [55] applied KPF for tracking person's location on an indoor floor map by wireless local area networks (WLANs). Chia et al. [56] adopted KPF and edge orientation histogram (EOH) for tracking target that having the similar color features with backgrounds. Yao et al. [57] pointed that not all particles in KPF were suitable by applying mean-shift to refine their positions. The incremental Bhattacharyya dissimilarity (IBD) and matrix condition number are proposed to determine the suitable particles for running mean-shift.

In kernel particle filter, mean-shift is used to move particles with negligible weights in gradient ascent direction, which made these particles converge to the neighboring local maxima. These new particles have high likelihood and most of them are preserved after re-sampling, which reduces the number of particles required for tracking and solves the impoverishment problem in regular PF as well. However, another problem that the KPF introduced is the sub-optimal solution; the variety of the particles after the mean-shift process is reduced and may converged to the local optimal. As all known, PF is the numerically implement of the Bayesian framework, this kernel-based mean-shift can be directly applied to modify the model in Bayesian framework as well [58], [59], [60].

3.2 Multiple Kernels Tracking

With the wider applications of KBOT in visual tracking, it is found that single kernel function doesn't perform well in tracking target with complex motion or under complex surroundings as expected. Gregory et al. have pointed that [61]: a single kernel, no matter what its structure, was ultimately limited by two factors:

- Dimensionality of the histogram (which in turn may be a function of available image structure).
- The interaction between its derivative structure and the spatial structure of the image as it was exposed by the histogram.

Therefore, Multiple Kernel Tracking (MKT) was proposed to enhance the effectiveness of KBOT. Multiple kernels can have the properties of different characterizations or locations. Collins [62] employed spatial kernel and scale kernel deal with

Table 3 A qualitative comparison of some algorithms in KBOT.

Algorithm	Main Contributions	Merit	Demerit	Assumption
KBOT [22]	RGB histogram representation + Isotropic kernel	Low-computational cost, about 150 fps on a 1GHz PC	Sensitive to occlusions; unstable to large scale and appearance changes.	Suitable updates of the target models; no drastically change between consecutive frames.
Kernel-bandwidth adaptation[24]	Affine model + Object corner to estimate the change of scale	Robustness to large scale changes	Need both the information of consecutive frames and more memory space; Just rigid object tracking.	The motion of target satisfies affine model in consecutive frames
Tunable kernels[25]	Feature distribution + spatial configurations; multi-blocks target representation with different bandwidth kernels	Robustness to small or intermittent occlusions. Allows for local scale change.	No model update; bandwidth for each blocks is not optimized when the pose of target changed	Having known a priori model of the target (spatial positions and motion features of each block)
Asymmetric kernel [26-29]	Asymmetric kernel according to the contour of the target is used for target representation	Robustness to scale and orientation changes; background information is removed; by combined the Gaussian mixture models of segment can deal with the out-of-plane rotations	Sensitive to intense blurring and occlusions; non-rigid target motion can cause ambiguities; better used for the target tracking in the aerial imagery ;in-plane rotations only[19-21]	The contour of the target is already known; Objects that move with rigid motion and do not change shape
Fragments-based [35-38]	Template object is represented by multiple image fragments or patches	Can handle partial occlusions or pose change	Not suitable to non rigid object; partial versus full explanation dilemma	At least a quarter of the patches will be visible; rigid movement between fragments
Gaussian-KBOT [39]	KBOT + Gaussian Pyramids	Can track the target that having larger motion as compared to its size.	Easy to drift in the coarsest image; Sensitive to occlusions	rigid movement; Suitable updates of the target models;
Annealed mean-shift [40]	A monotonically decreasing sequence of bandwidths is used for target search.	Track fast moving object.	No model update; Sensitive to scale change	The optimal bandwidth of the target is known; The over-smoothed density function with a sufficiently large bandwidth is unimodal
Different Features [42-46]	Spatiograms feature[42] LBP histogram[43] fuzzy color histogram[44] Gabor feature and entropy feature[46]	A richer description of target; reduce noisy interference and illumination changes; reduce the effect of similar background	High computation cost;	No drastically change between consecutive frames.
KPF [52-57]	Particle Filter + Mean Shift	Particles converge faster to the high probability areas of the posterior	The result maybe suboptimal; model update should be considered for particle filter; extra computation cost	Assume to know the motion model of the target; suitable parameters of system noise and observation noise
MIKT [61]	Placing multiple independent kernels on the target + Matusita metric	More sensitive to the directions of motion.	The illumination change, target appearance change and occlusion were not considered; The relations between kernels should also be considered.	No drastically change between consecutive frames; tracking rigid object, the movement has no rotation out of the image plane
MCKT [64,65]	Metric constraints and Euclidean restrictions were considered in the multi-kernel tracking	Can be used for articulated motions and this complex motion can be represented by a set of inter-correlated simpler motions	Sensitive to occlusions; unstable to large scale and appearance changes	Target with rigid motion and certain spatial structure
PAKT [66]	Constraints in piecewise affine kernel was considered	Can track rigid targets with out of plane motions and deformable targets. Partial occlusions	Cannot run in real time and the computation cost increase as the number of kernels increasing	No drastically appearance and illumination change between consecutive frames

the problem of selecting kernel scale for mean-shift blob tracking. Zhang et al. [63] adopted a dual-kernel for visual tracking, one for the similarities between candidates and target model, another for the contrasts between candidates and their neighboring background, this dual-kernel was measured by Bhattacharyya coefficient and Jensen-Shannon divergence respectively.

The MKT framework was proposed by Hager [61], in this framework, the sum of squared differences (SSD) is employed as the similarity measure instead of Bhattacharyya coefficient, and the Newton-style iterative procedure is derived to solve the optimization process. The displacement of target region is obtained by using several kernels placed at different locations of the target. The representation histogram of the target and candidate region $\boldsymbol{q}$ and $\boldsymbol{p}(\boldsymbol{c})$ are obtained by a histogram concatenation approach. This multiple kernel SSD tracking performs well in tracking and converges with less iteration. The kernels utilized in tracking are independent with each other. Therefore, this approach is also known as multiple independent kernel tracking (MIKT). One advantage of using independent kernel is that the incorrect estimation of one kernel may not affect the proper estimation of others. However, once a kernel suffers a bad iteration the tracker is unable to recover.

Multiple Collaborative Kernel Tracking (MCKT) was proposed by Fan et al. [64], which consider the constraints between kernels and view the kernel tracking problem as a linear system with $\begin{cases} \sqrt{\boldsymbol{q}} - \sqrt{\boldsymbol{p}(y)} = \boldsymbol{M} \cdot \triangle \boldsymbol{y} \\ \boldsymbol{\Omega}(\boldsymbol{y} + \triangle \boldsymbol{y}) = 0 \end{cases}$, where, $\boldsymbol{y}$ is the kernel center vector, $\boldsymbol{M}$ is the measurement matrix and $\boldsymbol{\Omega}$ are the constraints matrix between kernels. By exploiting the inherent relationship in multiple kernels, not only the "kernel-observability" has been improved, but also the applicability of KBOT is naturally extended to cope with articulated targets and complex motions. However, the constraints considered here are geometric constraints, which can restore the in-plane rotation and translation, but invalid with view and scale changes. Therefore, the cross ratio invariant constraint [65] was proposed.

Based on MIKT and MCKT, the Piecewise Affine Kernel Tracking (PAKT) was proposed by Martinez and Binefa [66] for tracking non-planar target. In this approach, triangular mesh described by the centers of each three kernels is considered. For each mesh triangle affine transform is estimated, subject to the constraints that each affine transform of a triangle must be compatible with the affine transforms coming from contiguous triangles, these constraints are explained in **Fig. 3**. This method can track rigid targets with out-of-plane motions and deformable targets. Unfortunately, it cannot run in real time.

Generally, MKT approach divides the tracking region into several pieces, for each piece a single kernel is implied for tracking. Intuitively speaking, not all of the kernels will be affected by the complex clutter, partial occlusion or luminance change conditions, so the MKT performs well than single kernel tracking. Applied in autonomous driving, MKT is more suitable for tracking rigid objects (vehicle or traffic sign and lights) because the kernels have stable spatial relationship and made the tracking result robustness. However, if tracking non-rigid target (pedestrian or driver face) the constraints between kernels will be much more

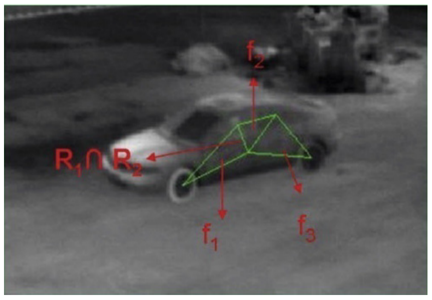


Fig. 3 Example of a piecewise defined function. On each of the three regions (pieces) the piecewise affine transformation is defined by a different affine transformation f_i. Each two consecutive pieces share one edge, for example pieces R_1 and R_2 intersect on the signaled edge ($f_1(R_1 \cap R_2) = f_2(R_1 \cap R_2)$). The objective will be to find those transformations f_i, respecting the constraint given over the intersection (Figure from Ref. [66]).

complicated and cannot processed in real-time. MKT can work on mean-shift framework [62], [63], [67], [68] or SSD framework [61], [64], [65], [66]. In most of mean-shift framework the kernels are independent, while in SSD tracking the relations between kernels are considered. These constraints improve the robustness of MKT at the cost of computation complex.

Table 3 gives some of the qualitative comparison of the algorithms mentioned in this section.

4. Kernel-based Learning

Kernel learning machines have been widely used in machine learning and pattern recognition [69], [70], [71], [72], it has been proven that the kernel machines have a stronger mathematical slant than earlier machine learning method (e.g., neural networks) and also attract significant interest from the statistics and mathematics community [17]. Meanwhile, these kernel machines are becoming particularly popular in target tracking; exceptionally in complex environment that kernel based tracking cannot give good performance. According to the learning mechanism of kernel method, this section is divided into two parts: tracking-by-detection and increasing kernel subspace learning. Some detailed descriptions of SVM have to be given in order to make readers comprehend of the *kernel-trick* that utilized in these learning algorithms.

4.1 Tracking-by-detection

Tracking-by-detection treats the tracking problem as a classification task, which applies an offline or online learning classifier to distinguish the tracking target from the background. Support Vector Machines (SVM) and Kernel Fisher Discriminant (KFD) are these classification algorithms, which have shown practical relevance not only in classification and regression problems but also, in supervised and unsupervised learning [73].

4.1.1 Support Vector Machines

For a two-class linear classification problem in supervised learning, the training data set comprises of N input vectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$, respectively corresponds to target values $y_1, \ldots, y_N$ where $y_N \in \{-1, 1\}$. If these training samples can be separated by a hyper-plane, the linear classify models has a decision function of the form

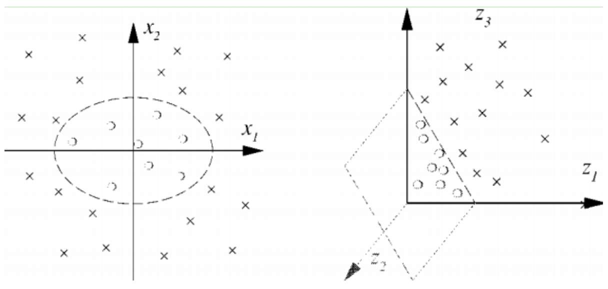


Fig. 4 Two-dimensional classification example. (a) In input space this construction corresponds to a nonlinear ellipsoidal decision boundary. (b) By mapped into three-dimensional feature space by $(x_1, x_2) \rightarrow (z_1, z_2, z_3) = (x_1{}^2, \sqrt{2}\,x_1x_2, x_2{}^2)$, there exists a linear hyper-plane that can separate the features (Figure from Ref. [74]).

$$y(\boldsymbol{X}) = \boldsymbol{w}^T \cdot \boldsymbol{X} + b \tag{5}$$

where $\boldsymbol{w}$ is the hyper-plane's normal vector, b is an offset.

Equation (5) satisfies $y(\boldsymbol{X_n}) > 0$ for training sample having $y_n = 1$ and $y(\boldsymbol{X_n}) < 0$ for $y_n = -1$. The decision boundary is defined as $\{\boldsymbol{X} \,|\, \boldsymbol{w}^T \cdot \boldsymbol{X} + b = 0\}$. However, there exist many solutions of $\boldsymbol{w}$ and b that satisfy the above conditions. SVM applies a concept of margin to confirm an optimal hyper-plane having the maximum margin. The margin is defined to be the smallest distance between decision boundary and any of the samples. If $\boldsymbol{w}$ and b are rescaled such that all the training samples satisfies $y_n \cdot (\boldsymbol{w}^T \cdot \boldsymbol{X} + b) \geq 1$, the margin can be measured as $2/||\boldsymbol{w}||$, which equivalent to minimizing $||\boldsymbol{w}||^2$. The training samples that satisfy $y_n \cdot (\boldsymbol{w}^T \cdot \boldsymbol{X} + b) = 1$ are termed as support vectors (SVs). By this means, the classification problem is a kind of a quadratic programming problem defined by:

$$\min_{\boldsymbol{w},\boldsymbol{b}} \frac{1}{2}||\boldsymbol{w}||^2$$
$$\text{Subject to } \; y_t \cdot (\boldsymbol{w}^T \cdot \boldsymbol{X}_t + b) \geq 1 \;\text{ for }\; t = 1, \dots, N \tag{6}$$

This constrained optimization problem can be transformed into a dual representation by introducing Lagrange multipliers

$$\min_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{X}_i \cdot \boldsymbol{X}_j)$$
$$\text{Subject to} \quad \begin{array}{c} \alpha_i \geq 0 \quad i = 1, \dots, N \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{array} \tag{7}$$

where $\boldsymbol{\alpha} = (\alpha_i, \dots, \alpha_N)^T$ is the Lagrange multiplier. By solving the above equations, the coefficients $\boldsymbol{\alpha}$ can be obtained, the decision function in Eq. (5) can be expressed as

$$y(\boldsymbol{X}) = \mathrm{sgn}\left(\sum_{i=1}^{N} y_i \alpha_i (\boldsymbol{X}_i \cdot \boldsymbol{X}) + b \right) \tag{8}$$

However, in most applications, linear separable is rather a strict condition (as shown in **Fig. 4** (a), there is no linear hyper-plane that can separate these two classes), in this case, a nonlinear feature space mapping function $\emptyset(\boldsymbol{X})$ is applied for mapping the training data into a potentially much higher dimensional feature space (as shown in Fig. 4 (b)). In this high-dimensional feature space, the mapped training data is linear separable. Notice that the calculation process in classifier enters only in the form of scalar products, the *kernel trick* can be used and Eq. (7) becomes $\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \cdot k(X_i \cdot X_j)$. By assigning a specific form of kernel function $k(\boldsymbol{X}_i, \boldsymbol{X}_j)$, $\emptyset(\boldsymbol{X}_i)^T \cdot \emptyset(\boldsymbol{X}_j)$ can be directly calculated avoiding the explicit introduction of the feature mapping function $\emptyset(\boldsymbol{X})$, which allows a implicit way to use feature spaces of high, even infinite, dimensionality. This nonlinear classification problem based on kernel trick is termed as kernel support vector machines (kernel SVM).

The application of SVM for tracking was applied in Ref. [75], in which SVM is integrated with the optic-flow-based tracker and termed as support vector tracking (SVT). The process of SVT can be generalized as follows: The SVM classifier can detect possible candidates in the current frame and hand them over to SVT. The SVT can refine their position so that a local maximum of SVM score is achieved. If the score is positive, the candidate will be declared as a tracking target and the optic-flow-based tracker start. The refined position in the current frame can serve as the initial guess in the next frame, etc. SVT combines the computational efficiency of optic-flow-based tracking with the power of a general classifier SVM, which extends the power of both the tracker and the classifier. However, this approach cannot handle partial occlusions, momentary disappearance and reappearance. Moreover, in Ref. [75] the classifier and tracker worked independently, Shen et al. [76] made them a cooperated work. Shen et al. generalized the standard kernel-based mean-shift tracker by maximizing a sophisticated cost function defined by SVM. In SVM a probability product kernels (PPK) is used to form the decision function, the optimal position in the next frame is determined by maximizing this cost function. In this framework, multiple temples are considered and the temple update is realized by SVM. Chen et al. [77] proposed a description-discrimination collaborative for target tracking, the Support Vector Data Description (SVDD) acted as the descriptive component to describe the global properties of the target and Structured Output SVM (SSVM) worked as the discrimination collaborative to distinguish the tracking target from the background. A self-paced learning algorithm in Ref. [78] was proposed to realize the long-term tracking, in which the appearance-based templates were learned and updated by SVM.

In most tracking-by-detection methods, classifier update needs to convert the estimated target position into a set of labeled training samples. However, it is not clear how best to perform this intermediate step. Hare et al. [79] extended the online structured output SVM learning method [80], [81] to handle this problem. The structured output prediction can directly predict the change in target location between frames and avoid the need for intermediate classification step, which reduces the errors in sampler and labeler process. A budgeting mechanism is applied to prevent the unbounded growth in the number of SVs. A semi-supervised SVM for tracking was also proposed [82].

4.1.2 Kernel Fisher Discriminant

Fisher linear discriminant (FLD) is a linear classifier that terms the classification problem as a dimensionality reduction problem. As for the two classes classification problem in Section 4.1.1, Fisher maps the high dimensional training samples $\boldsymbol{X}$ into one dimension by $y = \boldsymbol{w}^T \cdot \boldsymbol{X}$, thus for a new arrive sample $\boldsymbol{X_t}$ if

y_t greater than a specified threshold, $y_t = 1$, otherwise $y_t = -1$. FLD minimize the class overlap by giving a large separation between the projected class means while also giving a small variance within each class. Therefore, the Rayleigh coefficient is defined by $J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \boldsymbol{S_B} \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w}}$, where $\boldsymbol{S_B}$ is the between-class covariance matrix; $\boldsymbol{S_W}$ is the within-class covariance matrix. Similar as in Section 4.1.1, kernel method can also be applied in FLD to handle the nonlinear classification problem by the nonlinear mapping $\emptyset(\boldsymbol{X})$. This kernel-base FLD was termed as kernel Fisher discriminant (KFD), the detailed descriptions can be found in Refs. [83], [84] and [85].

Incremental KFD was also improved to solve the online update issues in target tracking. Lin et al. [86] formulated the target/background problem as a multiclass problem. The sample distribution of the target class was modeled by a single Gaussian and the non-target background class was modeled by a mixture of Gaussians. FLD was applied at each frame to solve this classification problem and Sequential Karhunen-Loeve algorithm (SKL) was used to incrementally update FLD based on the previous results. Shen et al. [87] introduced a computationally efficient nonlinear kernel learning strategy to find a discriminative model which distinguishes the tracked target from the background. This discriminative model was formed in KFD model and QR decomposition made it possible to update the optimal nonlinear subspace in real-time.

The purpose of tracking-by-detection approach is to train a classifier to distinguish the target form the surrounding backgrounds, FLD and SVM are two typically linear classifiers used in pattern recognition and machine learning. With the application of kernel trick, kernel classifier (KFD and kernel-based SVM) can deal with the nonlinear classification problem by mapping the training data into a linear separable high dimensional feature space. By assigning a specific form of kernel function, the direct calculation in high (even infinite) dimensional feature space can be avoided. However, this method needs large number of positive and negative samples for training the classifier and it will be time consuming to detect the target in an image without the prior information about the region and scale of target. Therefore, in autonomous driving this tracking-by-detection method has to be combined with some tracking algorithms such as optic-flow-based tracker [75] or kernel-based mean shift [76] to satisfy the real-time processing conditions. Additionally, in real applications it will be more important to choose a suitable training dataset for classifier learning than choose the learning algorithm. Generally speaking, SVM performs well than KFD. However, Chakrabarti et al. [88] and Cooke [89] have proved the equivalence between SVM and FLD on support vectors classification. Typically, two parts are necessary in tracking-by-detection approach: a) the generation of samples and labeled; b) update the classifier. This increasing learning method ensures the template update of target and suitable for tracking deformable target with appearance changes due to the unstable illumination and pose variation. It should be noticed that boosting [90] is another widely used classification approach that is not outlined here. The connections between boosting and kernel method can be found in Ref. [73]. It has been pointed in Ref. [79] that the SVM was more successful than boosting-based classifier due to its good generalization ability, robustness to label noise, and flexibility in target representation by the application of kernels.

4.2 Kernel Subspace Learning

Subspace features have been widely used in tracking for many years [89], [90], [91], [92], [93] due to its low-dimensional representation on the appearance of tracking target. Under the requirements of online and real-time applications, incremental subspace learning has been proposed recently [94]. In most of these subspace methods, Principal Component Analysis (PCA) is the prime dimension data feature extraction algorithm. Essentially, PCA is a linear subspace method, which can only produce the linear subspace feature extraction. It is unsuitable for highly complex and non-linear data distributions. In contrast, kernel subspace extraction method has shown its advantages in non-linear feature extraction applications such as face recognition [95], single frame super-resolution [96], image denoising [97], acquisition of multiple view feature descriptors [98] and target tracking [99].

Kernel Principal Component Analysis (KPCA) [100] is the main process of kernel subspace extraction, which applies the non-linear mapping function $\emptyset(\boldsymbol{X})$ to form a high dimensional feature space (Hilbert space) for linear feature extraction. The covariance matrix in KPCA to be eigen-decomposed is $\boldsymbol{C} = \frac{1}{N}\sum_{i=1}^{N} \emptyset(\boldsymbol{X}_i)\emptyset(\boldsymbol{X}_i)^T$, with the kernel trick it can be calculated implicitly. Chin and Suter [99] have pointed that there were high costs for storage resources and computational load during runtime applications of KPCA and the incremental KPCA was proposed to maintain constant update speed and memory usage under target's appearance and surrounding illumination changes. Typically, a positive definite kernel (e.g., polynomial or Gaussian RBF) is demanded in KPCA. Liwicki et al. [101] proposed an exact framework for online learning with a family of indefinite (not positive) kernels for tracking. This algorithm is applied in Krein space instead of Hilbert space, which does not require the calculation of pre-images and therefore is both efficient and accurate. The indefinite kernels are gradient-based, which directly calculate the scalar product in $4 * D$-dimensional feature space (D is the dimension of input vector) instead of using the kernel trick, thus it cannot run in real time. Zhang et al. [102] approximated the calculation of standard PCA and KPCA algorithms. These approximations discard data points that are close to the mean center and apply the rest data points to approximate the standard PCA and KPCA, by applying this few partial data points the proposed algorithm costs less memory and time consuming.

In short, KPCA is the extension of standard PCA by applying it in kernel feature space. KPCA can effectively extract the non-linear features of target and is often combined with particle filter for tracking. This robustness performance made it a good choice for tracking the target in autonomous driving especially in complicated environment or non-rigid target. However, in each iteration of KPCA or increasing KPCA, it calculates the singular value decomposition of the feature covariance matrix (often of high-dimensioned), which leads to high computation cost and unsuitable for real-time tracking (4 frames/s in Ref. [94], 3 frames/s in Ref. [101], 0.7-1 frames/s in Ref. [99]). Another

Table 4 A qualitative comparison of the algorithms in kernel-based-learning.

Algorithm	Main Contributions	Merit	Demerit	Assumption
SVT [75]	SVM + Optic-flow-based tracker	Large motions between successive frames	Cannot handle partial occlusions, momentary disappearance and reappearance	Suitable training database of target and background.
Generalized KBOT[76]	SVM+ Kernel-based mean shift	Online template adaptation	The result is unstable when the target and the background are similar.	Suitable training database ; local maximum of the SVM score corresponds to the target location
KPCA [99-101]	Kernel subspace features for target representation	Stable to illumination and target appearance change; template update with increasing or online learning	High computation cost	Suitable initial target location; Combined with particle filter for tacking

challenge in online learning is that the support set may grow to arbitrarily large size over time, some strategies as truncation and shrinking can be applied [103]. Some other dimensionality reductions as Fisher criteria [104], [105] and kernel entropy component analysis (kernel ECA) [106], [107] are also proposed.

Table 4 lists some qualitative comparison of the algorithms mentioned in this section.

5. Algorithm Implementation and Discussion

In this section a quantitative comparison of a few representative algorithms that mentioned above are tested for tracking vehicle, pedestrian and human face in autonomous driving or driving assistance system situations. We first present the trackers and the datasets for experimental comparison, and then the results are discussed.

5.1 Trackers and Dataset

Five tracking approaches with different methods are implemented to evaluate their performances on target tracking. GKT (Generalized Kernel-based tracking) [76] was proposed with the KBOT and SVM framework, Struck [79] was realized with SVM theory, On-line Boosting [90] utilized the boosting method for tracking, IPCA [94] performed the tracking by increasing PCA and DIKT [101] by incremental KPCA framework.

Seven video sequences $V_{1\text{-}7}$ have been used to evaluate the algorithm performances ($V_{1\text{-}3}$ are from pedestrian detection sequences in Ref. [8] and $V_{4\text{-}7}$ have been used in Refs. [79], [90], [94] and [101] for target tracking). These video sequences nearly contain the practical application environment of illumination change, scale variation, large pose change, local deformation and temporary occlusion. The tracking performance of vehicle and pedestrian in autonomous driving and driver face in driving assistance system can be evaluated with these videos. **Table 5** summarizes the main characteristics of these test sequences.

5.2 Tracking Result and Discussion

The representative tracking results of these sequences are illustrated in **Fig. 5**. All the trackers are initialized with the same target template in the initial frame, the tracking results are plotted as rectangle with different colors and line types for each tracker.

As in Refs. [94] and [101], the root mean square (RMS) error between the estimated location and the ground truth of feature points is adopted as the criterion of quantitative performance evaluation. In $V_{1\text{-}3}$ sequences the vertexes of target location are selected as the feature points and the ground truth of feature points in $V_{4\text{-}7}$ sequences can be found in the dataset of DIKT. This evaluation criterion is a pixel error and affected by the size of target or video sequences. Therefore, for each sequence the range of the RMS is different. However, this criterion can be used to compare the results of different trackers in the same sequence. The RMS errors are plotted in **Fig. 6** and the color of each RMS error corresponds to the color of the tracking results in Fig. 5. As can be observed in Fig. 5, GKT lost the target in the sequences of V_4 and V_5, so its RMS error is not plotted for comparison in Fig. 6.

Table 5 The characteristics of the test sequences.

NO.	Frames Length	Frame Size	Initial Target Size
V_1 (pedestrian 1)	365	640×480	26×66
V_2(pedestrian 2)	132	640×480	11×27
V_3(pedestrian 3)	203	640×480	21×46
V_4(Car4)	659	720×480	200×150
V_5(Car11)	393	320×240	25×20
V_6(David_indoor)	462	320×240	75×97
V_7(Trellis)	501	320×240	45×50

All the algorithms lost the target in V_1 when the target is completely occluded after frames 329. DIKT exhibits the best performance in all the rest of six video sequences. IPCA and Struck can track the target in the whole sequences of $V_{2\text{-}6}$. However, they lost the target in V_7 when the target suffers from large pose change and illumination change in simultaneous. On-line Boosting is less stable than other trackers and lost the target when it encounters illumination change. GKT tracks the target in most frames of V_1, V_6 and V_7. However, it lost the target in other videos due to the quality of sequences (grayscale-only and the target color feature is similar with the background). What's more, Struck, On-line Boosting and GKT cannot deal with the scale variation and orientation change of target, which made these algorithms have poor performances in V_2 and V_7 when the scale of target changes.

Real-time processing is another important requirement for target tracking in autonomous driving. The computational efficiency of each tracker is listed in **Table 6**. All these results are obtained on a desktop with Intel Core E2160 at 1.8 GHz with 2 GB of RAM.

Considering the computational efficiency of each tracker, we can conclude that online subspace learning algorithms (IPCA and

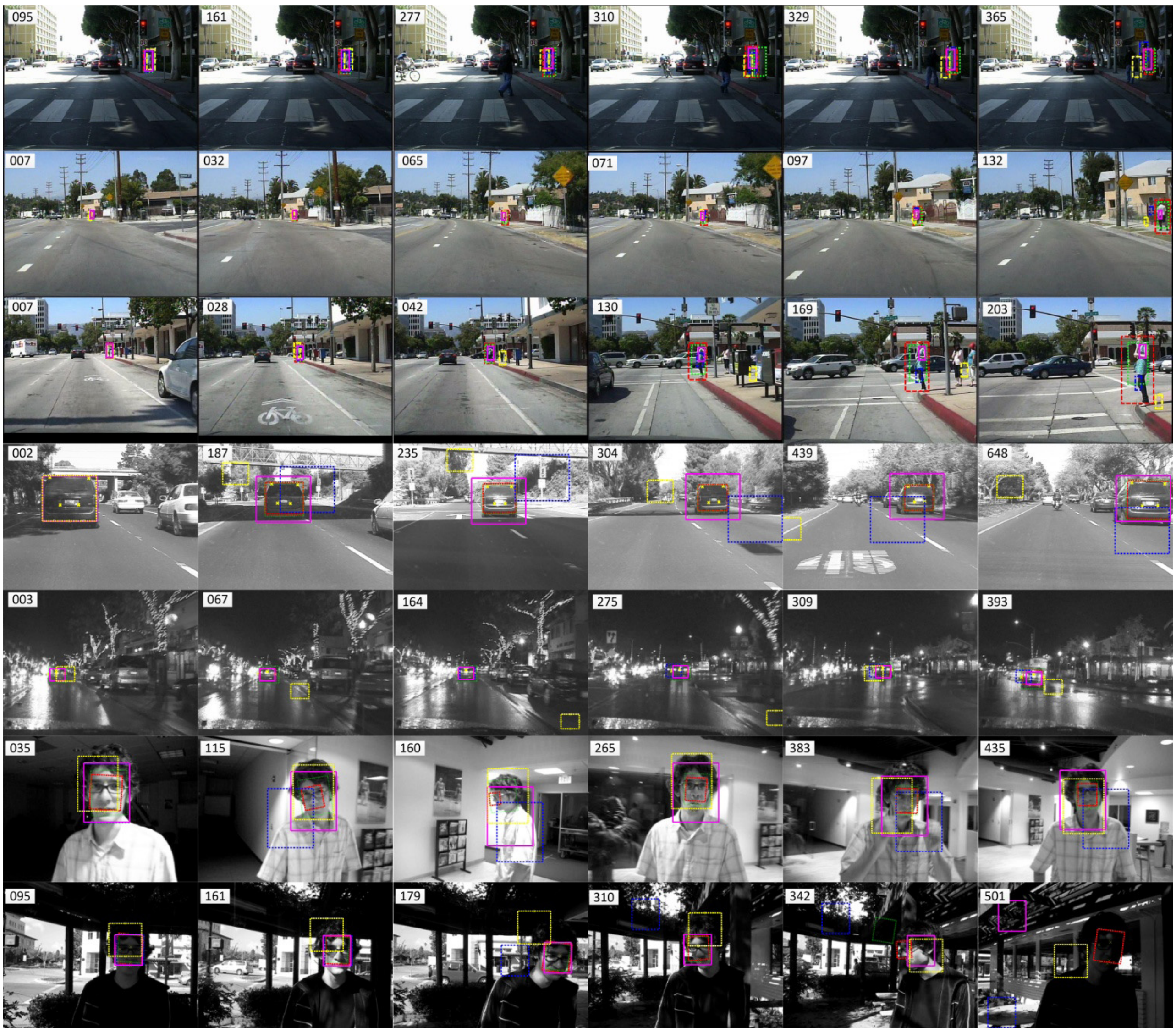


Fig. 5 The tracking result of video sequences V_1-V_7 (up to bottom) by applied On-line Boosting (▬·▬·▬), IPCA (┈┈┈), GKT (- - -✱- - -), Struck (▬▬▬) and DIKT (▬ ▬ ▬). The ground truth is indicated by yellow crosses. V_1-V_3 are sequences of pedestrian tracking; V_4 and V_5 are of moving vehicle tracking in day-time and night-time; V_6 and V_7 are sequences of facial tracking indoor and outdoor.

Table 6 The computational efficiency (frames/second) of each algorithm for video sequences V_1-V_7.

Algorithm	V_1	V_2	V_3	V_4	V_5	V_6	V_7
On-line Boosting	22.1	31.5	24.3	15.2	35.3	25.2	26.9
IPCA	11.8	12.8	11.9	11.6	12.5	12.2	11.6
GKT	40	56.3	43.7	20.37	60	46.5	43.4
Struck	18.3	16.5	13.3	12.3	17.1	18.2	15.1
DIKT	1.7	1.8	1.7	1.4	1.3	1.9	1.1

DIKT) achieve a good trade-off between the stability and real-time working. The subspace tracking based on PCA and KPCA can handle the illumination change, large pose change and non-rigid movement by subspace update, the scale variation of the target can be solved by considering the noise disturbance in the variables of particle filter. The additional information of subspace can improve the stability of trackers. IPCA fails after frames 310 in the sequence of V_7 because of the drastic pose and illumination changes. DIKT utilizes the additional information of gradient-based kernel for tracking, which increases the robustness of the tracker with the cost of reduce computational efficiency. Moreover, the multiple subspace temples in IPCA and DIKT also strengthen the stability of trackers. GKT is a combination of kernel mean-shift and SVM, which is a color feature based tracking so it mostly fails in sequences V_4 (grayscale-only) and $V_{2,3,5}$ (the target color feature is similar with the background). This color feature is easily affected by illumination change but more stability for facial expressions in face tracking. Struck and On-line Boosting are the kind of tracking by detection method respectively based on SVM and boosting. Struck performs well than boosting in sequences V_1-V_7 for illumination change, however, it costs the highest computational efficiency in all these five algorithms due to the structured output prediction process. Moreover, the computational efficiency of On-line Boosting and GKT are affected by the size of target region, with the highest computational efficiency in V_6 for the smallest target region and the

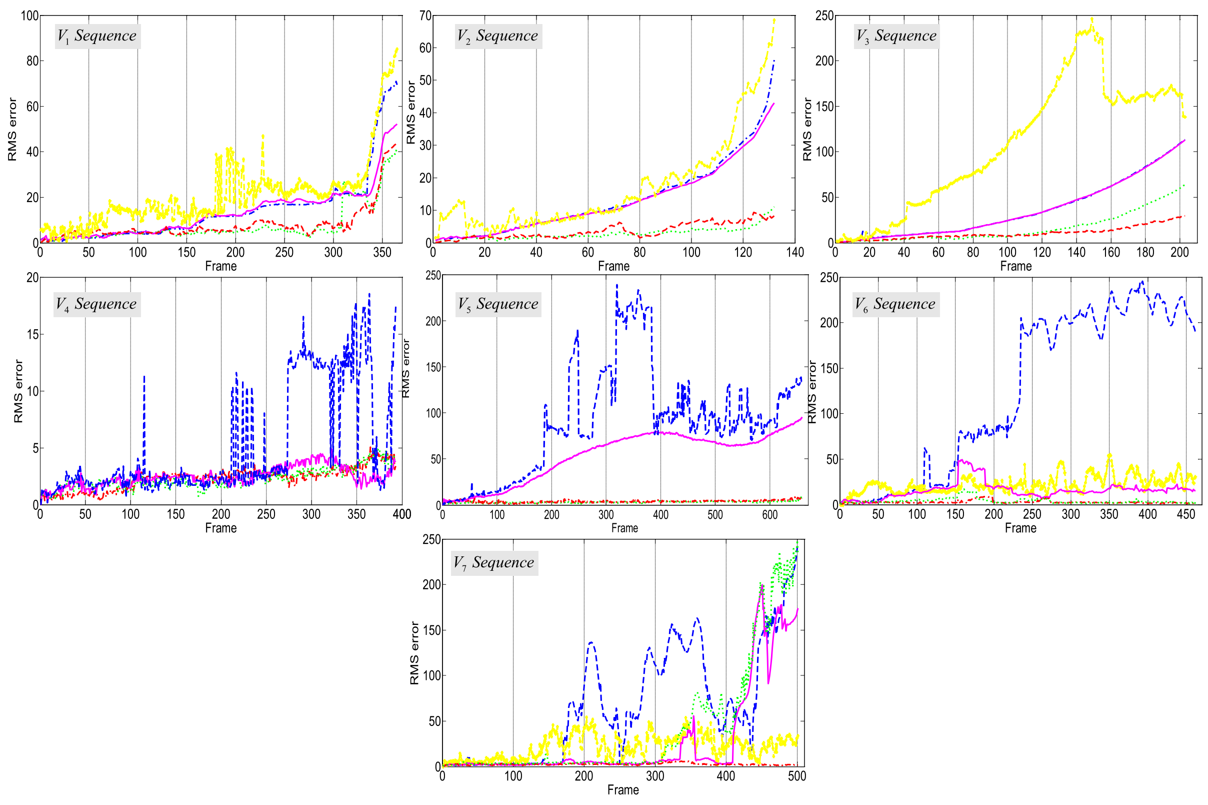


Fig. 6 The RMS errors between the estimated location and the ground truth of feature points for each frame in video sequence V_1-V_7. On-line Boosting (-·-·-·-), IPCA (··········), GKT (---✱---), Struck (——) and DIKT (------), GKT lost the target in sequence V_4 and V_5 so we doesn't plot its RMS error these two sequences.

lowest computational efficiency in V_4 for the largest target region.

6. Conclusions and Feature Directions

Fully autonomous driving in urban traffic is still an extremely difficult problem in the near future**.** The most difficult challenge of autonomous driving in urban traffic encounters is the target detection and tracking in visual image. These paper reviews the kernel based target tracking strategies that can be utilized in autonomous driving with visual sensors. Kernel-based target tracking has become a hot topic in visual tracking and gained approval due to its succinct description and computational efficiency. In this paper, we introduce the basic concepts of kernel density estimation (KDE) and kernel trick and survey the kernel method that applied in target tracking of autonomous driving. The applications of kernel-based techniques are divided into two aspects, namely, kernel-based target tracking and kernel online learning. In kernel-based tracking, kernel function is used for adding the spatial position information into the weighted-histogram representation of the target. Mean-shift approach works on this representation for tracking the target in continuous frames. Single and multiple kernel tracking approaches are reviewed. In kernel online learning, kernel function implicitly calculates the scalar product in some high dimensional feature space and solves the non-linear problem. Some kernel learning machines (e.g., SVM, KFD and KPCA) are briefly introduced for tracking. A qualitative and quantitative comparison is given. Five trackers are evaluated with seven test sequences, the theoretical and experimental analysis allow us to conclude that the online subspace learning algorithm achieves a good trade-off between the stability and real-time working for target tracking in autonomous driving environments. From the survey, a few promising directions are recommended as follows, hoping to improve the flexibility of tracking approach in autonomous driving.

Error estimation is an important problem that has been neglected in the development of tracking algorithms. In tracking process, errors may occur in the projection of 3D world onto a 2D image, feature points extracted from the blurred image cased by the complex surroundings, state parameters of the transition model and observation model in tracking framework, etc. In classical approaches, Gaussian noise model is suggested for error estimation and propagation, this Gaussian model is a probabilistic model, which guarantees the most probabilistic value that the result is the truth-value. However, sometimes, a guaranteed result that certainly contains the truth-value is preferred but not with the largest probability. Interval analysis (IA) is a kind of this strategy. Telle and Ramdani [108] have utilized IA for camera calibration and 3D reconstruction, in which pixel coordinates were seen as two unknown but bounded variables (interval number), interval constraint propagation method was applied to propagate this uncertainties. IA can also be applied in auto-calibration [109], deformable image registration [110] and design of 2D image filter [111]. Considering the application of IA in the tracking framework (e.g., interval kernel function) can guarantee the truth-value of the tracking results and improve the robustness of tracking algorithm in autonomous driving.

In the practical application of tracking system, auto-registration and re-initialization are two important issues should be considered. Auto-registration needs the online learning of the target, as mentioned above, many online learning algorithm have been proposed. However, re-initialization is seldom considered. In some special conditions, target may be occluded by the background scene structure for a second and reappears in any directions. So far, none tracking methods can handle this problem. Though there has been methods for re-initialization based on multiple cameras [41], [112], the re-initialization criterion for single camera is still an open challenge.

In general, nearly all the tracking approach only utilized the image information from 2D image plane; this 2D-based tracking is computational efficiency but less robustness due to the loss of information caused by projection of the 3D world on a 2D image. Multiple cameras have been used to obtain the 3D information of the tracking target [113], [114], however, the calculation of the relationship between multiple cameras and the depth estimation is a lot of computational cost. With the development of hardware devices in camera manufacturers, many depth cameras (e.g., kinect or Creative Senz3D provide by Microsoft) are available. We can directly obtain the depth information from these cameras, avoiding the complex calculation by ourselves. The advantages of using depth information for tracking arise for two reasons. One is the depth information increases the feature representation of tracking target. The other is that the coordinates of 3D points which we interested can be directly obtained; with these coordinates in word coordinate system the dynamical model and motion constraints can be estimated.

Acknowledgments The work was supported by the National Natural Science Foundation of China (No.61305023).

References

[1] Bengler, K. et al.: Three Decades of Driver Assistance Systems.

[2] Bishop, R.: *Intelligent vehicle technology and trends*, Artech House (2005).

[3] Shladover, S.E.: Literature Review on Recent International Activity in Cooperative Vehicle–Highway Automation Systems, No. FHWA-HRT-13-025 (2012).

[4] Gusikhin, O., Filev, D. and Rychtyckyj, N.: Intelligent vehicle systems: Applications and new trends, Informatics in Control Automation and Robotics, pp.3–14, Springer Berlin Heidelberg (2008).

[5] Bertozzi, M., Broggi, A. and Fascioli, A.: Vision-based intelligent vehicles: State of the art and perspectives, *Robotics and Autonomous systems*, Vol.32, No.1, pp.1–16 (2000).

[6] Geiger, A., Lenz, P. and Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite, *2012 IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), pp.3354–3361, IEEE (2012).

[7] Sivaraman, S. and Trivedi, M.M.: Active learning for on-road vehicle detection: A comparative study, *Machine Vision and Applications*, Vol.25, No.3, pp.599–611 (2014).

[8] Dollar, P., Wojek, C., Schiele, B., et al.: Pedestrian detection: An evaluation of the state of the art, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.34, No.4, pp.743–761 (2012).

[9] Petrovskaya, A. and Thrun, S.: Model based vehicle tracking for autonomous driving in urban environments, *Proc. Robotics: Science and Systems IV*, Zurich, Switzerland (2008).

[10] Ess, A., Schindler, K., Leibe, B., et al.: Object detection and tracking for autonomous navigation in dynamic environments, *The International Journal of Robotics Research*, Vol.29, No.14, pp.1707–1725 (2010).

[11] Yilmaz, A., Javed, O. and Shah, M.: Object tracking: A survey, *ACM Computing Surveys* (*CSUR*), Vol.38, No.4, p.13 (2006).

[12] Bishop, C.M. and Nasrabadi, N.M.: *Pattern recognition and machine learning*, Springer, New York (2006).

[13] Comaniciu, D. and Meer, P.: Mean shift: A robust approach toward feature space analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.24, No.5, pp.603–619 (2002).

[14] Aizerman, A., Braverman, E.M. and Rozoner, L.I.: Theoretical foundations of the potential function method in pattern recognition learning, *Automation and Remote Control*, Vol.25, pp.821–837 (1964).

[15] Boser, B.E., Guyon, I.M. and Vapnik, V.N.: A training algorithm for optimal margin classifiers, *Proc. 5th Annual Workshop on Computational Learning Theory*, pp.144–152, ACM (1992).

[16] Shawe-Taylor, J. and Cristianini, N.: *Kernel methods for pattern analysis*, Cambridge university press (2004).

[17] Hofmann, T., Schölkopf, B. and Smola, A.J.: A review of kernel methods in machine learning, Mac-Planck-Institut für biologische, Kybernetik, Technical Report (2006).

[18] Jaakkola, T. and Haussler, D.: Exploiting generative models in discriminative classifiers, *Advances in neural information processing systems*, pp.487–493 (1999).

[19] Moreno, P.J., Ho, P.P. and Vasconcelos, N.: A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications, *Advances in Neural Information Processing Systems* (2003).

[20] Jebara, T., Kondor, R. and Howard, A.: Probability product kernels, *The Journal of Machine Learning Research*, Vol.5, pp.819–844 (2004).

[21] Fieguth, P. and Terzopoulos, D.: Color-based tracking of heads and other mobile objects at video frame rates, *Proc. 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.21–27, IEEE (1997).

[22] Dorin, C., Ramesh, V. and Meer, P.: Kernel-based object tracking, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.25, No.5, pp.564–577 (2003).

[23] Cheng, Y.: Mean Shift, mode seeking and clustering, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.17, No.8, pp.790–799 (1995).

[24] Ning-Song, P., Yang, J. and Chen, J.X.: Kernel-bandwidth adaptation for tracking object changing in size, *Image Analysis and Recognition*, pp.581–588, Springer Berlin Heidelberg (2004).

[25] Parameswaran, V., Ramesh, V. and Zoghlami, I.: Tunable kernels for tracking, *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol.2, pp.2179–2186, IEEE (2006).

[26] Yilmaz, A.: Object tracking by asymmetric kernel mean shift with automatic scale and orientation selection, *IEEE Conference on Computer Vision and Pattern Recognition, CVPR'07*, pp.1–6, IEEE (2007).

[27] Yi, K.M., Ahn, H.S. and Choi, J.Y.: Orientation and scale invariant mean shift using object mask-based kernel, *19th International Conference on Pattern Recognition, ICPR 2008*, pp.1–4, IEEE (2008).

[28] Yilmaz, A.: Kernel-based object tracking using asymmetric kernels with adaptive scale and orientation selection, *Machine Vision and Applications*, Vol.22, No.2, pp.255–268 (2011).

[29] Quast, K. and Kaup, A.: Shape adaptive mean shift object tracking using gaussian mixture models, *Analysis, Retrieval and Delivery of Multimedia Content*, pp.107–122, Springer New York (2013).

[30] Leichter, I., Lindenbaum, M. and Rivlin, E.: Tracking by affine kernel transformations using color and boundary cues, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.31, No.1, pp.164–171 (2009).

[31] Li, P., Cai, Z., Wang, H., et al.: Scale invariant kernel-based object tracking, *International Conference on Computer Vision in Remote Sensing* (*CVRS*), pp.252–255, IEEE (2012).

[32] Zhang, L., Zhang, D., Su, Y., et al.: Adaptive kernel-bandwidth object tracking based on Mean-shift algorithm, *4th International Conference on Intelligent Control and Information Processing* (*ICICIP*), pp.413–416, IEEE (2013).

[33] Li, P.: Tensor-sift based earth mover's distance for contour tracking, *Journal of Mathematical Imaging and Vision*, Vol.46, No.1, pp.44–65 (2013).

[34] Freedman, D. and Zhang, T.: Active contours for tracking distributions, *IEEE Trans. Image Processing*, Vol.13, No.4, pp.518–526 (2004).

[35] Adam, A., Rivlin, E. and Shimshoni, I.: Robust fragments-based tracking using the integral histogram, *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp.798–805 (2006).

[36] Babu, R.V., Perez, P. and Bouthemy, P.: Robust tracking with motion estimation and local kernel-based color modeling, *Image and Vision Computing*, Vol.25, No.8, pp.1205–1216 (2007).

[37] Wang, F., Yu, S. and Yang, J.: A novel fragments-based tracking algorithm using mean shift, *Proc. International Conference on Control, Automation, Robotics and Vision*, pp.694–698 (2008).

[38] Li, G. and Wu, H.: Robust object tracking using kernel-based weighted fragments, *2011 International Conference on Multimedia Technology* (*ICMT*), pp.3643–3646, IEEE (2011).

[39] Ali, Z., Hussain, S. and Taj, I.A.: Kernel based robust object tracking

using model updates and Gaussian pyramids, *Proc. IEEE Symposium on Emerging Technologies, 2005*, pp.144–150, IEEE (2005).

[40] Shen, C., Brooks, M. and Van Den Hengel, A.: Fast global kernel density mode seeking: Applications to localization and tracking, *IEEE Trans. Image Processing*, Vol.16, No.5, pp.1457–1469 (2007).

[41] Tyagi, A., Keck, M., Davis, J.W., et al.: Kernel-based 3d tracking, *IEEE Conference on Computer Vision and Pattern Recognition, CVPR'07*, pp.1–8, IEEE (2007).

[42] Birchfield, S.T. and Rangarajan, S.: Spatiograms versus histograms for region-based tracking, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, Vol.2, pp.1158–1163, IEEE (2005).

[43] Pouladzadeh, P., Semsarzadeh, M., Hariri, B., et al.: An enhanced Mean-Shift and LBP-based face tracking method, *2011 IEEE International Conference on Virtual Environments Human-Computer Interfaces and Measurement Systems* (*VECIMS*), pp.1–4, IEEE (2011).

[44] Ju, M.Y., Ouyang, C.S. and Chang, H.S.: Mean shift tracking using fuzzy color histogram, *2010 International Conference on Machine Learning and Cybernetics* (*ICMLC*), Vol.6, pp.2904–2908, IEEE (2010).

[45] Mazinan, A.H. and Amir-Latifi, A.: Improvement of mean shift tracking performance using a convex kernel function and extracting motion information, *Computers & Electrical Engineering* (2012).

[46] Liu, R. and Lu, Y.: Infrared target tracking in multiple feature pseudo-color image with kernel density estimation, *Infrared Physics & Technology* (2012).

[47] Wang, Z., Salah, M.B., Zhang, H., et al.: Shape based appearance model for kernel tracking, *Image and Vision Computing*, Vol.30, No.4, pp.332–344 (2012).

[48] Yang, C., Duraiswami, R. and Davis, L.: Efficient spatial-feature tracking via the mean-shift and a new similarity measure, *IEEE Conference on Computer Vision and Pattern Recognition*, Vol.1, pp.176–183 (2005).

[49] Leichter, I.: Mean shift trackers with cross-bin metrics, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.34, No.4, pp.695–706 (2012).

[50] Minwoo, P., Liu, Y. and Collins, R.T.: Efficient mean shift belief propagation for vision tracking, *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2008*, IEEE (2008).

[51] Li, P.: An adaptive binning color model for mean shift tracking, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.18, No.9, pp.1293–1299 (2008).

[52] Chang, C. and Ansari, R.: Kernel particle filter: Iterative sampling for efficient visual tracking, *ICIP 2003, Proc. 2003 International Conference on Image Processing*, Vol.2, No.3, p.III-977-80, IEEE (2003).

[53] Chang, C. and Ansari, R.: Kernel particle filter for visual tracking, *Signal processing letters*, Vol.12, No.3, pp.242–245, IEEE (2005).

[54] Schmidt, J., Fritsch, J. and Kwolek, B.: Kernel particle filter for real-time 3D body tracking in monocular color images, *7th International Conference on Automatic Face and Gesture Recognition, FGR 2006*, pp.567–572, IEEE (2006).

[55] Zhang, V.Y. and Wong, A.K.: Kernel Based Particle Filtering for Indoor Tracking in WLANs, *Journal of Network and Computer Applications* (2012).

[56] Chia, Y.S., Kow, W.Y., Khong, W.L., et al.: Kernel-based object tracking via particle filter and mean shift algorithm, *2011 11th International Conference on Hybrid Intelligent Systems* (*HIS*), pp.522–527, IEEE (2011).

[57] Yao, A., Lin, X., Wang, G., et al.: A compact association of particle filtering and kernel based object tracking, *Pattern Recognition*, Vol.45, No.7, pp.2584–2597 (2012).

[58] Han, B., Zhu, Y., Comaniciu, D., et al.: Kernel-based bayesian filtering for object tracking, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, Vol.1, pp.227–234, IEEE (2005).

[59] Zhang, X., Hu, W., Luo, G., et al.: Kernel-Bayesian framework for object tracking, *Computer Vision, ACCV 2007*, pp.821–831, Springer, Berlin Heidelberg (2007).

[60] Zhang, X., Hu, W., Bao, H., et al.: Robust Head Tracking Based on Multiple Cues Fusion in the Kernel-Bayesian Framework (2013).

[61] Hager, G.D., Dewan, M. and Stewart, C.V.: Multiple kernel tracking with SSD, *Proc. 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2004*, Vol.1, pp.790–797, IEEE (2004).

[62] Collins, R.T.: Mean-shift blob tracking through scale space, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol.2, p.II-234-40, IEEE (2003).

[63] Zhang, C.L., Jing, Z.L., Jin, B., et al.: A dual-kernel-based tracking approach for visual target, *Science China Information Sciences*, Vol.55, No.3, pp.566–576 (2012).

[64] Fan, Z., Wu, Y. and Yang, M.: Multiple collaborative kernel tracking, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, Vol.2, pp.502–509, IEEE (2005).

[65] Ma, L., Cheng, J. and Lu, H.: Multi-cue collaborative kernel tracking with cross ratio invariant constraint, *19th International Conference on Pattern Recognition, ICPR*, pp.1–4, IEEE (2008).

[66] Martinez, B. and Binefa, X.: Piecewise affine kernel tracking for non-planar targets, *Pattern Recognition*, Vol.41, No.12, pp.3682–3691 (2008).

[67] Chu, C.T., Hwang, J.N., Pai, H.I., et al.: Robust video object tracking based on multiple kernels with projected gradients, *2011 IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp.1421–1424, IEEE (2011).

[68] Chu, C.T., Hwang, J.N., Wang, S.Z., et al.: Human tracking by adaptive Kalman filtering and multiple kernels tracking with projected gradients, *2011 5th ACM/IEEE International Conference on Distributed Smart Cameras* (*ICDSC*), pp.1–6, IEEE (2011).

[69] Varma, M. and Babu, B.R.: More generality in efficient multiple kernel learning, *Proc. 26th Annual International Conference on Machine Learning*, pp.1065–1072, ACM (2009).

[70] Szafranski, M., Grandvalet, Y. and Rakotomamonjy, A.: Composite kernel learning, *Machine learning*, Vol.79, No.1-2, pp.73–103 (2010).

[71] Mendelson, S. and Neeman, J.: Regularization in kernel learning, *The Annals of Statistics*, Vol.38, No.1, pp.526–565 (2010).

[72] Gönen, M. and Alpaydın, E.: Multiple kernel learning algorithms, *The Journal of Machine Learning Research*, Vol.12, pp.2211–2268 (2011).

[73] Muller, K.R., Mika, S., Ratsch, G., et al.: An introduction to kernel-based learning algorithms, *IEEE Trans. Neural Networks*, Vol.12, No.2, pp.181–201 (2001).

[74] Schölkopf, B. and Smola, A.J.: *Learning with Kernels*, Cambridge, MA: MIT Press (2001).

[75] Maldonado-Bascon, S., Lafuente-Arroyo, S., Gil-Jimenez, P., et al.: Road-sign detection and recognition based on support vector machines, *IEEE Trans. Intelligent Transportation Systems*, Vol.8, No.2, pp.264–278 (2007).

[76] Shen, C., Kim, J. and Wang, H.: Generalized kernel-based visual tracking, *IEEE Trans. Circuits and Systems for Video Technology*, Vol.20, No.1, pp.119–130 (2010).

[77] Chen, D. et al.: Description-Discrimination Collaborative Tracking, *Computer Vision, ECCV 2014*, pp.345–360, Springer, International Publishing (2014).

[78] Supancic, J.S. and Ramanan, D.: Self-paced learning for long-term tracking, *2013 IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), IEEE (2013).

[79] Hare, S., Saffari, A. and Torr, P.H.S.: Struck: Structured output tracking with kernels, *2011 IEEE International Conference on Computer Vision* (*ICCV*), pp.263–270, IEEE (2011).

[80] Bordes, A., Bottou, L., Gallinari, P., et al.: Solving multiclass support vector machines with LaRank, *Proc. 24th International Conference on Machine Learning*, pp.89–96, ACM (2007).

[81] Bordes, A., Usunier, N. and Bottou, L.: Sequence labelling SVMs trained in one pass, *Machine Learning and Knowledge Discovery in Databases*, pp.146–161, Springer, Berlin Heidelberg (2008).

[82] Tang, F., Brennan, S., Zhao, Q., et al.: Co-tracking using semi-supervised support vector machines, *IEEE 11th International Conference on Computer Vision, ICCV 2007*, pp.1–8, IEEE (2007).

[83] Mika, S., Ratsch, G., Weston, J., et al.: Fisher discriminant analysis with kernels, *Neural Networks for Signal Processing IX* (1999). *Proc. 1999 IEEE Signal Processing Society Workshop*, pp.41–48, IEEE (1999).

[84] Mika, S.: *Kernel fisher discriminants*, Universitätsbibliothek (2002).

[85] Baudat, G. and Anouar, F.: Generalized discriminant analysis using a kernel approach, *Neural Computation*, Vol.12, No.10, pp.2385–2404 (2000).

[86] Lin, R.S., Yang, M.H. and Levinson, S.E.: Object tracking using incremental fisher discriminant analysis, *Proc. 17th International Conference on Pattern Recognition, ICPR 2004*, Vol.2, pp.757–760, IEEE (2004).

[87] Shen, C., Van Den Hengel, A. and Brooks, M.J.: Visual tracking via efficient kernel discriminant subspace learning, *ICIP 2005, IEEE International Conference on Image Processing*, Vol.2, p.II-590-3, IEEE (2005).

[88] Chakrabarti, S., Roy, S. and Soundalgekar, M.V.: Fast and accurate text classification via multiple linear discriminant projections, *The VLDB Journal*, Vol.12, No.2, pp.170–185 (2003).

[89] Cooke, T.: Two variations on Fisher's linear discriminant for pattern recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.24, No.2, pp.268–273 (2002).

[90] Grabner, H., Grabner, M. and Bischof, H.: Real-Time Tracking via

On-line Boosting, *BMVC*, Vol.1, No.5, p.6 (2006).
[91] Attallah, S. and Abed-Meraim, K.: Fast algorithms for subspace tracking, *Signal Processing Letters*, Vol.8, No.7, pp.203–206, IEEE (2001).
[92] De La Torre, F. and Black, M.J.: A framework for robust subspace learning, *International Journal of Computer Vision*, Vol.54, No.1-3, pp.117–142 (2003).
[93] Tzimiropoulos, G., Zafeiriou, S. and Pantic, M.: Subspace learning from image gradient orientations (2012).
[94] Ross, D.A., Lim, J., Lin, R.S., et al.: Incremental learning for robust visual tracking, *International Journal of Computer Vision*, Vol.77, No.1-3, pp.125–141 (2008).
[95] Yang, M.H.: Kernel Eigenfaces vs. Kernel Fisherfaces: Face Recognition Using Kernel Methods, *FGR*, Vol.2, p.215 (2002).
[96] Kim, K.I., Franz, M.O. and Scholkopf, B.: Iterative kernel principal component analysis for image modeling, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.27, No.9, pp.1351–1366 (2005).
[97] Jorgensen, K.W. and Hansen, L.K.: Model selection for Gaussian kernel PCA denoising, *IEEE Trans. Neural Networks and Learning Systems*, Vol.23, No.1, pp.163–168 (2012).
[98] Meltzer, J., Yang, M.H., Gupta, R., et al.: Multiple view feature descriptors from image sequences via kernel principal component analysis, *Computer Vision-ECCV 2004*, pp.215–227, Springer, Berlin Heidelberg (2004).
[99] Chin, T.J. and Suter, D.: Incremental kernel principal component analysis, *IEEE Trans. Image Processing*, Vol.16, No.6, pp.1662–1674 (2007).
[100] Schölkopf, B., Smola, A. and Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem, *Neural Computation*, Vol.10, No.5, pp.1299–1319 (1998).
[101] Liwicki, S., Zafeiriou, S., Tzimiropoulos, G., et al.: Efficient online subspace learning with an indefinite kernel for visual tracking and recognition (2012).
[102] Zhang, R., Wang, W. and Ma, Y.: Approximations of the standard principal components analysis and kernel PCA, *Expert Systems with Applications*, Vol.37, No.9, pp.6531–6537 (2010).
[103] Kivinen, J., Smola, A.J. and Williamson, R.C.: Online learning with kernels, *IEEE Trans. Signal Processing*, Vol.52, No.8, pp.2165–2176 (2004).
[104] Sugiyama, M.: Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis, *The Journal of Machine Learning Research*, Vol.8, pp.1027–1061 (2007).
[105] Sugiyama, M., Idé, T., Nakajima, S., et al.: Semi-supervised local Fisher discriminant analysis for dimensionality reduction, *Machine Learning*, Vol.78, No.1-2, pp.35–61 (2010).
[106] Jenssen, R.: Kernel entropy component analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.32, No.5, pp.847–860 (2010).
[107] Shekar, B.H., Sharmila Kumari, M., Mestetskiy, L.M., et al.: Face recognition using kernel entropy component analysis, *Neurocomputing*, Vol.74, No.6, pp.1053–1057 (2011).
[108] Telle, B., Aldon, M.J. and Ramdani, N.: Camera calibration and 3d reconstruction using interval analysis, *Proc. 12th International Conference on Image Analysis and Processing*, pp.374–379, IEEE (2003).
[109] Fusiello, A., Benedetti, A., Farenzena, M., et al.: Globally convergent autocalibration using interval analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.26, No.12, pp.1633–1638 (2004).
[110] Noblet, V., Heinrich, C., Heitz, F., et al.: 3-D deformable image registration: A topology preservation scheme based on hierarchical deformation models and interval analysis optimization, *IEEE Trans. Image Processing*, Vol.14, No.5, pp.553–566 (2005).
[111] Wang, Y., Yue, J., Su, Y., et al.: Design of Two-Dimensional Zero-Phase FIR Digital Filter by McClellan Transformation and Interval Global Optimization, *IEEE Trans. Circuits and Systems II: Express Briefs*, Vol.60, No.3, pp.167–171 (2013).
[112] Zhang, Z., Potamianos, G., Senior, A., et al.: A joint system for person tracking and face detection, *Computer Vision in Human-Computer Interaction*, pp.47–59, Springer, Berlin Heidelberg (2005).
[113] Eshel, R. and Moses, Y.: Tracking in a dense crowd using multiple cameras, *International journal of computer vision*, Vol.88, No.1, pp.129–143 (2010).
[114] Srivastava, S., Ng, K.K. and Delp, E.J.: Color correction for object tracking across multiple cameras, *2011 IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp.1821–1824, IEEE (2011).

Yanming Wang received his B.Eng. degree from the School of Electronic and Information, Tongji University, China, in 2011. Currently he is pursuing his Dual Degree (M.S. and Ph.D.) on Control Theory and Control Engineering, Tongji University. His major research interests include pattern recognition and machine learning, image and face analysis.

Jiguang Yue received his M.Eng. degree from Harbin University of Science and Technology in 1986 and Ph.D. degree from Harbin Institute of Technology in 1997, China. Currently, he is a professor in the Department of Control Theory and Control Engineering, Tongji University, China. Before he joined Tongji University, he was an associate professor in the Department of Automation, Harbin University of Science and Technology from 1986 to 1993. He has published more than 100 papers. His major research areas include digital signal processing, reliability analysis and image processing.

Yanchao Dong received his B.Eng. and M.Eng. degrees from Tongji University, China in 2005 and 2008. He was selected as one of Chinese government's overseas scholars to be a Ph.D. candidate at Kumamoto University, Japan in 2008 and he received his Ph.D. degree in Computer Science from Kumamoto University, Japan, in 2012. He has been a lecturer at Tongji University (China) from 2013. His research interests include Human Computer Interface, Pattern Recognition and Machine Learning, Facial Animation Tracking.

Zhencheng Hu received his B.Eng. degree from Shanghai JiaoTong University, China in 1992, and his M.Eng. degree from Kumamoto University, Japan, in 1998. He received his Ph.D. degree in System Science from Kumamoto University, Japan, in 2001. Dr. Hu has held various positions in computer science and machine vision industry. He is currently an associate professor with the Department of Computer Science, Kumamoto University, Japan. His research interests include camera motion analysis, augmented reality, machine vision applications in industry and ITS. Dr. Hu is a member of IEEE and IEICE.