J. Vis. Commun. Image R. 23 (2012) 254–263

Contents lists available at SciVerse ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

# Online selection of the best $k$-feature subset for object tracking

Guorong Li [a,*], Qingming Huang [a,b], Junbiao Pang [b], Shuqiang Jiang [b], Lei Qin [b]

[a] Graduate University of Chinese Academy of Sciences (CAS), Beijing 100190, China
[b] Key Lab of Intell. Info. Process., Inst. of Comput. Tech., CAS, Beijing 100080, China

ARTICLE INFO

*Article history:*
Received 7 September 2010
Accepted 9 November 2011
Available online 20 November 2011

*Keywords:*
Object tracking
Feature subset selection
Feature selection
Feature subset tree
Conditional entropy
Greedy search algorithm
Particle filter
Online selection

ABSTRACT

In this paper, we propose a new feature subset evaluation method for feature selection in object tracking. According to the fact that a feature which is useless by itself could become a good one when it is used together with some other features, we propose to evaluate feature subsets as a whole for object tracking instead of scoring each feature individually and find out the most distinguishable subset for tracking. In the paper, we use a special tree to formalize the feature subset space. Then conditional entropy is used to evaluating feature subset and a simple but efficient greedy search algorithm is developed to search this tree to obtain the optimal $k$-feature subset quickly. Furthermore, our online $k$-feature subset selection method is integrated into particle filter for robust tracking. Extensive experiments demonstrate that $k$-feature subset selected by our method is more discriminative and thus can improve tracking performance considerably.



## 1. Introduction

Object tracking has been a hot research topic for decades, for its wide applications in computer vision fields such as traffic surveillance, human computer interaction and 3D scene reconstruction. As analyzed in [1], feature selection has become an important approach to model the observation model for visual tracking. This is because, intuitively, not all the available features are useful to distinguish objects with background, and some may generate negative effects and even lead to "drift".

Another problem for object tracking is that target moves continually, while background often changes. As a result, the most representative features may change over time and online feature selection method is needed for updating features. However, most existing methods (e.g. [1–5]) usually evaluate features separately through a feature evaluation function, such as log likelihood ratio in [1] and Bayes error in [5]. This provides every feature a score or builds each feature into a weak classifier ([2–4]) in boosting, and then selects better features or classifies separately. However, these approaches may neglect two interesting facts pointed out in [6] and [7]. One is that a feature that is bad when used by itself may become a very good one when it is used together with some other features. The other is that those presumably redundant features could help each other providing stronger classification ability. In the context of tracking, diverse features, such as different channels of color, texture, shape, describe the target from different aspects. Intuitively, compared to selecting feature separately, selecting subset of features together generally could obtain feature subsets that have stronger predictive and discriminative ability for visual tracking. As an example shown in Fig. 1, image (b) displays the positions and weights of particles obtained using features selected by our feature subset selection method, while (c) displays particles' weights generated with features selected by Collins et al. [1]. We can see that in (d), in the selected feature space, object and background could be separated. So using features selected with our method, the weights of those particles distributing on the target man are relative higher. Therefore tracking result marked by the red rectangle in (b) is accurate. However, as shown in (e), with each single feature, it is not easy to distinguish the distributions. But they generate good result when they are used together as shown in (d). Moreover, as shown in (c), the used features could not distinguish the target from his surroundings and those particles located on the man in the red coat have higher weights. So the tracker generates wrong tracking result (green rectangle in (c)).

Therefore, considering the target and background as two classes in tracking, we evaluate the ability of the feature subset as a whole and select the most effective feature subset for discriminating the two classes. Firstly, we use a subset tree whose leaves represent all of the candidate subsets. Secondly, we propose to evaluate the node of this tree with conditional entropy [6], which reflects the uncertainty of class label when feature subset is known. Thirdly, we use greedy algorithm to search the feature subset tree in order to find out the global optimal feature subset quickly. Finally, we

* Corresponding author.
*E-mail address:* grli@jdl.ac.cn (G. Li).

1047-3203/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2011.11.001

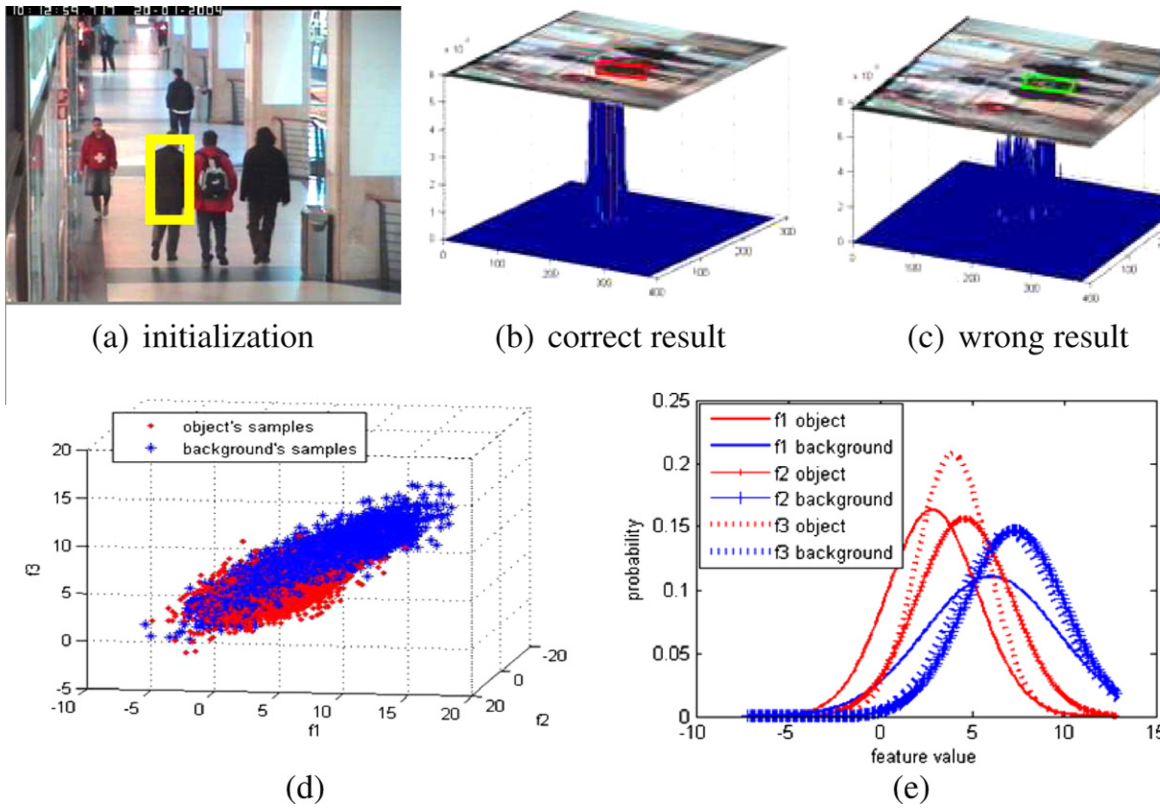



**Fig. 1.** (a) is the initialization for tracking. (b) and (c) are positions and weights of particles obtained using our feature subset selection and Collins' feature selection method [1]. In our selected feature space, the distribution of the samples that belong to object class and background class is shown in (d). Samples' distribution with respect to each feature is provided in (e). We can see that each single feature is not a good one for classifying the tracked target and background. However, when they used together, they generate good results as shown in (d) and (b).

embed our feature subset selection method into particle filter framework and during tracking we re-select the feature subset to adapt to background's and the target's variations. The flowchart is shown in Fig. 3.

## 2. Feature subset selection for tracking

For object tracking, feature subset selection is to select the most representative $k$-feature subset that could separate the tracked target from the background. Apparently if the size of the set is $n$, the size of its subsets containing $m$ elements is $C_n^m$, which may be very large. For example, in [1] there are 49 linear combinations of $R$, $G$, $B$ and the number of its 5-element subsets is 1,906,884. Facing the large candidate space, we must handle two challenges. One is how to describe it so that it can be easily traversed. The other is how to find out the optimal subset fast and update it online in real-time tracking. In order to judge whether a feature subset is better than the others or not, a function $S$ is usually defined to provide feature subset (referred as $\Omega$) a quantitative evaluation (referred as $S(\Omega)$) and the goal of feature selection becomes finding out a feature subset of the given size that can minimize $S(\Omega)$.

### 2.1. Feature subset representation

As we discussed above, candidate subset space is the set of all possible feature subsets which may be very large. We propose to build a special tree $T$ (e.g. Fig. 2), whose leaves could represent those candidate subsets. Let $N_1, N_2, \ldots, N_r$ represent the node set of $T$. We label each node with a number (referred to as the label of the node) which corresponds to a feature except for zero. For example, number $i$ denotes feature $f_i$. The level of a node is the length of the shortest path from this node to the root and the largest level is defined as the depth of this tree. Then our tree should satisfy three following conditions:

(1) Each subtree is tree-ordered: the labels of children nodes are larger than that of the root.
(2) If the depth of the tree is $L$, then all of the leaf nodes should be on level $L$.
(3) Let $k$ represent the size of the feature subset that we want to select, then the depth of the tree should be $k$. Let $n$ denote the size of feature pool, if a node with label $i$ is on level $l$, it should have $n - i - k + l + 1$ branches. This condition can be derived from condition (1) and (2). In object tracking, since $k$ usually is much smaller than $n$, the depth of the this tree would be very small.

As we use this tree to denote the candidate feature subset space, we name this tree as "*feature subset tree*" (FST). For every node, there is a shortest path connecting it with the root (e.g. the red nodes in Fig. 2 constitute the shortest path that connects the root and a leaf). The label set of the nodes on this path denotes the set of eliminated features. So if the size of feature pool is $n$, then traversing such a tree whose depth is $k$ can get all of the subsets of size $k$. The pseudo-code for building FST is presented in Table 1.

### 2.2. Feature subset evaluation in object tracking

In our method, we use conditional entropy [8] to evaluate the discriminative ability of a feature subset, which quantifies the uncertainty of a random variable when another variable is given. At time $t$, let $\Omega_t$ represent a feature subset and $p(\boldsymbol{\omega_t})$ ($\boldsymbol{\omega_t}$ is a vector)

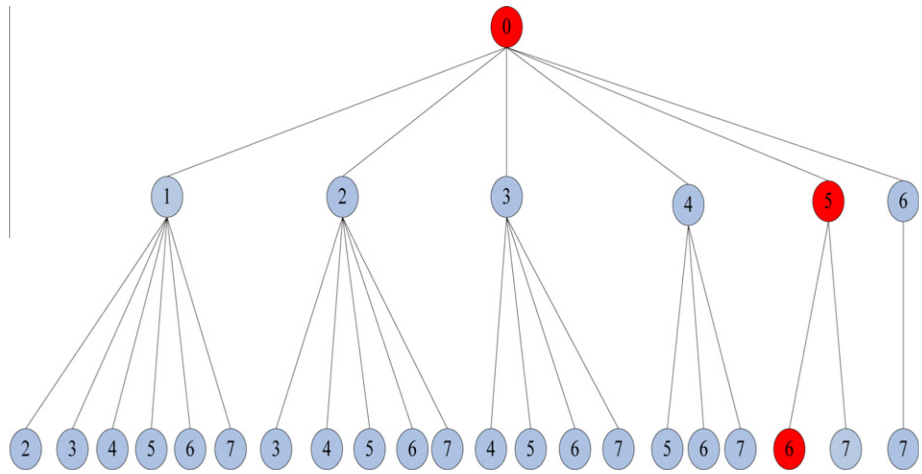



**Fig. 2.** The tree used to represent 2-feature subset space of a feature pool containing 7 features. The number labeled on every node denotes the feature's index, e.g. 6 denotes $f_6$.

**Table 1**
The pseudo-code for building FST.

```
Given k, feature pool F̃ = {f_{m_1}, f_{m_2}, ..., f_{m_n}}. Initiate node.level = 0,
node.label = 0;
function BuildFST(node,k,n,F)
if node.Level = = k break;
else
  for i = node.label + 1 : n – i
    node.child = new FSTNode[n – i – node.label];
    node.child[k].label=the index of feature F(i);
    node.child[k].level = nodel.level + 1;
    BuildFST(node.child[k],k,n,F);
  end
end
```

denote its probability density function. Let $c_1$ and $c_2$ represent target class and background class in object tracking and $C = \{c_1, c_2\}$. The evaluate function $S(\Omega_t)$ for $\Omega_t$ is defined as conditional entropy $H(C|\Omega_t)$.

$$S(\Omega_t) \models H(C|\Omega_t) = \int p(\boldsymbol{\omega_t}) H(C|\boldsymbol{\omega_t}) d\boldsymbol{\omega_t}, \quad (1)$$

$$= \sum_{i=1}^{2} \int p(\boldsymbol{\omega_t}, c_i) \left( -log \frac{p(\boldsymbol{\omega_t}|c_i) p(c_i)}{p(\boldsymbol{\omega_t})} \right) d\boldsymbol{\omega_t}, \quad (2)$$

$$= \sum_{i=1}^{2} \int p(\boldsymbol{\omega_t}, c_i)(-log\, p(\boldsymbol{\omega_t}|c_i) - log\, p(c_i) + log\, p(\boldsymbol{\omega_t})) d\boldsymbol{\omega_t}, \quad (3)$$

$$= \sum_{i=1}^{2} \int p(\boldsymbol{\omega_t}, c_i)(-log\, p(\boldsymbol{\omega_t}|c_i)) d\boldsymbol{\omega_t} + \sum_{i=1}^{2} (-log\, p(c_i)) \times \int p(\boldsymbol{\omega_t}, c_i) d\boldsymbol{\omega_t} - \int (-log\, p(\boldsymbol{\omega_t})) \sum_{i=1}^{2} p(\boldsymbol{\omega_t}, c_i) d\boldsymbol{\omega_t}, \quad (4)$$

$$= \sum_{i=1}^{2} \int p(\boldsymbol{\omega_t}, c_i)(-logp(\boldsymbol{\omega_t}|c_i)) d\boldsymbol{\omega_t} + \sum_{i=1}^{2} (-log\, p(c_i) p(c_i) - \int (-log\, p(\boldsymbol{\omega_t})) p(\boldsymbol{\omega_t}) d\boldsymbol{\omega_t}, \quad (5)$$

$$= \sum_{i=1}^{2} p(c_i) \int p(\boldsymbol{\omega_t}|c_i)(-log\, p(\boldsymbol{\omega_t}|c_i)) d\boldsymbol{\omega_t} + H(C) - H(\Omega_t) \quad (6)$$

with

$$H(C|\boldsymbol{\omega_t}) = \sum_{i=1}^{2} (p(c_i|\boldsymbol{\omega_t})(-logp(c_i|\boldsymbol{\omega_t}))). \quad (7)$$

$H(\Omega_t)$ and $H(C)$ are the entropy of $\Omega_t$ and $C$ respectively. In our experiments, we use Gaussian distribution $(N(\boldsymbol{\mu_i^t}, \boldsymbol{\Sigma_i^t}))$ to model $p(\boldsymbol{\omega_t}|c_i)$ $(i = 1,2)$. Then we could derive

$$\int p(\boldsymbol{\omega_t}|c_i) logp(\boldsymbol{\omega_t}|c_i) d\boldsymbol{\omega_t} \quad (8)$$

$$= \int p(\boldsymbol{\omega_t}|c_i) \frac{lnp(\boldsymbol{\omega_t}|c_i)}{ln10} d\boldsymbol{\omega_t} \quad (9)$$

$$= \frac{1}{ln10} \int p(\boldsymbol{\omega_t}|c_i) \left( -\frac{1}{2} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t})' (\boldsymbol{\Sigma_i^t})^{-1} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t}) \right) d\boldsymbol{\omega_t} - \frac{1}{ln10} \int p(\boldsymbol{\omega_t}|c_i)(\log(\sqrt{2\pi})^{N_{\Omega_t}} |\boldsymbol{\Sigma_i^t}|^{\frac{1}{2}})) d\boldsymbol{\omega_t} \quad (10)$$

$$= \frac{1}{ln10} E\left[ -\frac{1}{2} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t})' (\boldsymbol{\Sigma_i^t})^{-1} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t}) \right] - \frac{1}{ln10} ln\left( (\sqrt{2\pi})^{N_{\Omega_t}} |\boldsymbol{\Sigma_i^t}|^{\frac{1}{2}} \right). \quad (11)$$

Let $\left( (\boldsymbol{\Sigma_i^t})^{-1} \right)_{r,j}$ and $((\boldsymbol{\Sigma_i^t}))_{r,j}$ denote the $r$th row, $j$th column element of $(\boldsymbol{\Sigma_i^t})^{-1}$ and $(\boldsymbol{\Sigma_i^t})$ respectively, $\boldsymbol{\omega}_{\boldsymbol{t},r}$ and $\boldsymbol{\mu^t}_{\boldsymbol{i},r}$ represent the $r$th element of the vector $\omega_t$ and $\mu_i^t$. Since

$$(\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t})' (\boldsymbol{\Sigma_i^t})^{-1} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t}) = \sum_{r=1}^{N_{\Omega_t}} \sum_{j=1}^{N_{\Omega_t}} \left( (\boldsymbol{\Sigma_i^t})^{-1} \right)_{r,j} (\boldsymbol{\omega}_{\boldsymbol{t},r} - \boldsymbol{\mu}_{\boldsymbol{i},r}^t)(\boldsymbol{\omega}_{\boldsymbol{t},j} - \boldsymbol{\mu}_{\boldsymbol{k},j}^t), \quad (12)$$

$$E\left( (\boldsymbol{\omega}_{\boldsymbol{t},r} - \boldsymbol{\mu}_{\boldsymbol{i},r}^t) \left( \boldsymbol{\omega}_{\boldsymbol{t},j} - \boldsymbol{\mu}_{\boldsymbol{i},j}^t \right) \right) = (\boldsymbol{\Sigma_i^t})_{r,j} = (\boldsymbol{\Sigma_i^t})_{j,r}, \quad (13)$$

we can derive

$$E\left[ -\frac{1}{2} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t})' (\boldsymbol{\Sigma_i^t})^{-1} (\boldsymbol{\omega_t} - \boldsymbol{\mu_i^t}) \right] = -\frac{1}{2} \sum_{r=1}^{N_{\Omega_t}} \sum_{j=1}^{N_{\Omega_t}} \left( (\boldsymbol{\Sigma_i^t})^{-1} \right)_{r,j} (\boldsymbol{\Sigma_i^t})_{j,r} \quad (14)$$

$$= -\frac{1}{2} \sum_{r=1}^{N_{\Omega_t}} 1 = -\frac{N_{\Omega_t}}{2}, \quad (15)$$

substitute above two equations into Eq. (11), we could easily obtain

$$\int p(\boldsymbol{\omega_t}|c_i) logp(\boldsymbol{\omega_t}|c_i) d\boldsymbol{\omega_t} = -\frac{0.5N_{\Omega_t}}{ln10} - \frac{1}{ln10} ln\left( (2\pi)^{\frac{N_{\Omega_t}}{2}} |\boldsymbol{\Sigma_i^t}|^{\frac{1}{2}} \right) \quad (16)$$

$$= -\frac{0.5N_{\Omega_t}}{ln10} - 0.5log\left( (2\pi)^{N_{\Omega_t}} |\boldsymbol{\Sigma_i^t}| \right), \quad (17)$$

where $N_{\Omega_t}$ is the number of features in $\Omega_t$ and $|\boldsymbol{\Sigma_i^t}|$ represents the value of determinant $\boldsymbol{\Sigma_i^t}$.

As $p(\boldsymbol{\omega_t}) = p(\boldsymbol{\omega_t}|c_1)p(c_1) + p(\boldsymbol{\omega_t}|c_2)p(c_2)$, and $p(\boldsymbol{\omega_t}|c_i)$, $(i = 1,2)$ follows Gaussian distribution, $p(\boldsymbol{\omega_t})$ follows Gaussian mixture distribution $G(\boldsymbol{\omega_t})$.

$$G(\boldsymbol{\omega_t}) = p(c_1)N(\boldsymbol{\mu_1^t}, \boldsymbol{\Sigma_1^t}) + p(c_2)N(\boldsymbol{\mu_2^t}, \boldsymbol{\Sigma_2^t}). \tag{18}$$

According to [10], the entropy for $\Omega_t$ could be approximated by,

$$H(\Omega_t) \approx -\sum_{i=1}^{2} p(c_i)logG(\boldsymbol{\mu_i^t}) - \sum_{i=1}^{2} 0.5p(c_i)F(\boldsymbol{\mu_i^t}) \odot \boldsymbol{\Sigma_i^t} \tag{19}$$

with

$$F(\boldsymbol{\mu_i^t}) = \frac{1}{G(\boldsymbol{\mu_i^t})}\sum_{j=1}^{2} p(c_j)\left(\boldsymbol{\Sigma_j^t}\right)^{-1} N\left(\boldsymbol{\mu_i^t}; \boldsymbol{\mu_j^t}, \boldsymbol{\Sigma_j^t}\right) \left(\frac{1}{G(\boldsymbol{\mu_i^t})}\Delta\boldsymbol{\mu_{ij}^t}(\nabla G(\boldsymbol{\mu_i^t}))^T + \Delta\boldsymbol{\mu_{ij}^t}\left(\left(\boldsymbol{\Sigma_j^t}\right)^{-1}\Delta\boldsymbol{\mu_{ij}^t}\right)^T - I\right), \tag{20}$$

where

$$\Delta\boldsymbol{\mu_{ij}^t} = \boldsymbol{\mu_i^t} - \boldsymbol{\mu_j^t}, \tag{21}$$

$$\nabla G = \frac{\Delta G(\Omega_t)}{\Delta \Omega_t}, \tag{22}$$

$$N\left(\boldsymbol{\mu_i^t}; \boldsymbol{\mu_j^t}, \boldsymbol{\Sigma_j^t}\right) = \frac{1}{\sqrt{(2\pi)^{N_{\Omega_t}}|\boldsymbol{\Sigma_j^t}|}} \exp\left(-\frac{1}{2}(\Delta\boldsymbol{\mu_{ij}^t})'\left(\boldsymbol{\Sigma_j^t}\right)^{-1}\Delta\boldsymbol{\mu_{ij}^t}\right), \tag{23}$$

the operator $\odot$ of two matrices $\boldsymbol{A}(a_{ij})$ and $\boldsymbol{B}(b_{ij})$ is defined as:

$$\boldsymbol{A} \odot \boldsymbol{B} \models \sum_{i,j} a_{ij}b_{ij}. \tag{24}$$

We can see that the larger the $\|\Delta\boldsymbol{\mu_{12}^t}\|$, the smaller $N(\boldsymbol{\mu_1^t}; \boldsymbol{\mu_2^t}, \boldsymbol{\Sigma_2^t})$ and $N(\boldsymbol{\mu_2^t}; \boldsymbol{\mu_1^t}, \boldsymbol{\Sigma_1^t})$ are, leading to smaller $F(\boldsymbol{\mu_1^t})$ and $F(\boldsymbol{\mu_2^t})$.

Substituting Eqs. (19) and (17) into Eq. (6), we can derive that,

$$S(\Omega_t) \approx 0.5p(c_1)log((2\pi)^{N_{\Omega_t}}|\boldsymbol{\Sigma_1^t}|) + 0.5p(c_2)log\left((2\pi)^{N_{\Omega_t}}|\boldsymbol{\Sigma_2^t}|\right) + H(C) + \sum_{i=1}^{2}(p(c_i)logG(\boldsymbol{\mu_i^t}) + 0.5p(c_i)F(\boldsymbol{\mu_i^t}) \odot \boldsymbol{\Sigma_i^t}) + \frac{N_{\Omega_t}}{2ln10}. \tag{25}$$

In information theory, conditional entropy $H(C|\Omega_t)$ reflects the uncertainty of $C$ when $\Omega_t$ is known. More concretely, if $C$ is only relevant with $\Omega_t$, $H(C|\Omega_t)$ will be zero. This is because if $\Omega_t$ is known, no more information is needed to decide the value of $C$. If $C$ is independent with $\Omega_t$, $H(C|\Omega_t)$ will be the entropy of $C$ as $\Omega_t$ could not provide any information about $C$ at all. So the smaller it is, the better the feature subset is. Feature subset selection is to find out the optimal one ($\Omega_t^*$) that has the smallest conditional entropy.

$$\Omega_t^* = \arg\min_{\Omega_t}\{S(\Omega_t)\}. \tag{26}$$

From Eq. (25), we can see that the smaller $|\boldsymbol{\Sigma_1^t}|$ and $|\boldsymbol{\Sigma_2^t}|$ are, the smaller $S(\Omega_t)$ is. Meanwhile, the larger the $\|\Delta\boldsymbol{\mu_{12}^t}\|$, the smaller $F(\boldsymbol{\mu_1^t})$ and $F(\boldsymbol{\mu_2^t})$ are, therefore the smaller $S(\Omega_t)$ is. In other words, if a feature subset $\Omega_t$ leads to smaller within class scatter matrixes and larger between class scatter matrix, $S(\Omega_t)$ will be smaller. Intuitively, using the features obtained through our feature subset selection method, the distribution of samples within the same class is more concentrated while samples belonging to different classes are more scattered. This means, with the selected feature subset, the tracked target and background are very easy to discriminate.

*2.3. Feature subset searching algorithm*

Here let $R(N_i)$ denote the label set on the shortest path connecting the root and node $N_i$, we use $S(R(N_i))$ to score $R(N_i)$. Then $k$-feature subset problem can be regarded as finding a leaf that has the smallest score. Apparently, through traversing FST, we can easily obtain all the feasible subsets and then select the best one. However, the FST could be very large, so traversing the whole tree to find out the best $k$-feature subset is very time-consuming. Therefore, propose to search this tree in a greedy way [9]. we Every time, we select the best node to extend until the selected node is on $k$th level. Then many nodes are not searched and evaluated when we get an optimal feature subset. The details are presented in Table 2.

**Table 2**
The procedure of the proposed searching algorithm.

Given the feature pool $F = \{f_1, f_2, \ldots, f_n\}$, the size $k$ of the target feature subset, the parameters $(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1})$, $(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})$ of $p(z^F|c_1)$ and $p(z^F|c_2)$, $p(c_1)$, $p(c_2)$.

$$//\boldsymbol{\mu}_i = (\mu_{i,1}, \mu_{i,2}, \ldots, \mu_{i,n}), \boldsymbol{\Sigma}_i = \begin{bmatrix} \Sigma_{i,1,1} & \cdots & \Sigma_{i,1,n} \\ \vdots & \ddots & \vdots \\ \Sigma i,n,1 & \cdots & \Sigma_{i,n,n} \end{bmatrix}, \quad p(z^F|c_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), i = 1, 2$$

//$z^F$ denote the observation in feature space $F$
*root.level* = 0, *root.label* = 0;
BuildFST(*node*, $k$, $n$, $F$) according to Table 1;
// Search FST in a greedy method to find out the optimal $k$-feature subset.
set *queue* = *empty*, *node* = *root*;
while *node.level* $\leqslant k$
  For each children node $N_i$ of *node*
    Obtain $R(N_i) = \{f_{i_1}, f_{i_2}, \ldots, f_{i_{N_i}}\}$;
    //Extract $\tilde{\boldsymbol{\mu}}_j$ and $\widetilde{\boldsymbol{\Sigma}}_j$, $p(R(N_i)|c_j) \sim N(\tilde{\boldsymbol{\mu}}_j, \widetilde{\boldsymbol{\Sigma}}_j)$, $j = 1, 2$:

$$\tilde{\boldsymbol{\mu}}_j = \{\mu_{j,i_1}, \mu_{j,i_2}, \ldots, \mu_{j,i_{N_i}}\}, \quad j = 1, 2$$

$$\widetilde{\boldsymbol{\Sigma}}_j = \begin{bmatrix} \Sigma_{j,i_1,i_1} & \cdots & \Sigma_{j,i_1,i_{N_i}} \\ \vdots & \ddots & \vdots \\ \Sigma_{j,i_{N_i},i_1} & \cdots & \Sigma_{j,i_{N_i},i_{N_i}} \end{bmatrix}, j = 1, 2$$

    Compute $S(R(N_i))$ according to Eq. (25);
    Insert $N_i$ into *queue* in ascending order according to $S(R(N_i))$ ;
  end for
  *node* = Pop out the first node in *queue*;
end while

*2.4. Online feature subset selection for object tracking*

Particle filtering is a method for estimating the future unknown state $s_t$ from the sequential observations $z_{0:t} = \{z_0, z_1, \ldots, z_t\}$.

$$p(s_t|z_{0:t}) = \frac{p(s_t, z_{0:t})}{p(z_{0:t})}, \tag{27}$$

$$= \frac{p(s_t|z_{0:t-1})p(z_{0:t-1})p(z_t|s_t, z_{0:t-1})}{p(z_t|z_{0:t-1})p(z_{0:t-1})}, \tag{28}$$

$$= \frac{p(s_t|z_{0:t-1})p(z_t|s_t, z_{0:t-1})}{p(z_t|z_{0:t-1})}, \tag{29}$$

$$= k_t p(z_t|s_t)\int_{s_{t-1}} p(s_t|s_{t-1})p(s_{t-1}|z_{0:t-1})ds_{t-1}, \tag{30}$$

$$\propto p(z_t|s_t)\int_{s_{t-1}} p(s_t|s_{t-1})p(s_{t-1}|z_{0:t-1})ds_{t-1}, \tag{31}$$

where $k_t$ is an normalization constant which is independent of $s_t$. Provided the state transition model $p(s_t|s_{t-1})$ and the observation model $p(z_t|s_t)$, particle filter aims to approximate the posterior distribution function $p(s_t|z_{0:t})$ by a set of weighted particles $\{(s_t^i, w_t^i)\}_{i=1}^N$. The general procedure of particle filter algorithm is provided in Table 3.

For object tracking, we embed our feature subset selection into a simple particle filter tracking framework [11]. The state of the particle is denoted by $s_t(x_t, y_t)$, where $x_t$ and $y_t$ are the coordinates

**Table 3**
The procedure of the general particle filter algorithm.

1. **Initialization:**
   Generate a sample set $\{s_0^i, w_0^i\}_{i=1}^N$ according to the initial distribution $p(x_0)$, set $t = 1$.
2. **Resampling:**
   Generate a new sample set $\{s_{t-1}'^i, \frac{1}{N}\}_{i=1}^N$ by resampling $\{s_{t-1}^i, w_{t-1}^i\}_{i=1}^N$.
3. **Prediction:**
   Predict $\{s_t^i\}_{i=1}^n$ for $\{s_{t-1}'^i\}_{i=1}^N$ according to $p(s_t|s_{t-1})$.
4. **Update:**
   Compute the weight of the new particles $\tilde{w}_t^i = p(z_t^i|s_t^i)/N$ and $w_t^i = \frac{\tilde{w}_t^i}{\sum_{j=1}^N \tilde{w}_t^j}$.

of particles. The observation of every particle is denoted by $z_t$. Then the posterior probability $p(s_t|z_{0:t})$ of every particle is computed according to Eqs. (27)–(32), considering the probability $p(\boldsymbol{\omega}_i^t|c_1)$ and $p(\boldsymbol{\omega}_i^t|c_2)$.

$$p(z_t|s_t) \propto p(z_t|c_1)(1 - p(z_t|c_2)) = N(z_t; \boldsymbol{\mu}_1^t, \boldsymbol{\Sigma}_1^t)(1 - N(z_t; \boldsymbol{\mu}_2^t, \boldsymbol{\Sigma}_2^t)), \tag{32}$$

where

$$N(z_t; \boldsymbol{\mu}_i^t, \boldsymbol{\Sigma}_i^t) = \frac{1}{\sqrt{(2\pi)^{N_{\Omega_t}} |\boldsymbol{\Sigma}_i^t|}} \exp\left(-\frac{1}{2}(z_t - \boldsymbol{\mu}_i^t)'(\boldsymbol{\Sigma}_i^t)^{-1}(z_t - \boldsymbol{\mu}_i^t)\right) \quad (i = 1, 2).$$

To adapt to the variations of the target and background, during tracking, we collect new samples from the current frame when the target is located. At time $t$, let $\widetilde{N_i^t}$, $\widetilde{\boldsymbol{\mu}_i^t}$, $\widetilde{\boldsymbol{\Sigma}_i^t}$ denote the number, mean and covariance of these new samples that belong to class $c_i$ respectively, and $N_i^t$ represent the number of old samples in $c_i$, then $\boldsymbol{\mu}_i^t$ and $\boldsymbol{\Sigma}_i^t$ could be updated as follows,

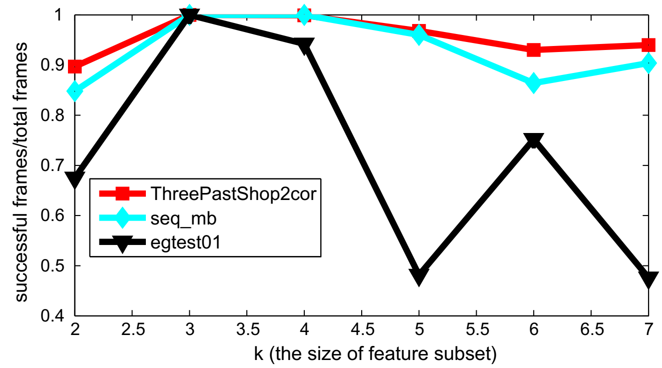



**Fig. 4.** Successful frames/total frames on test videos under different values of $k$.

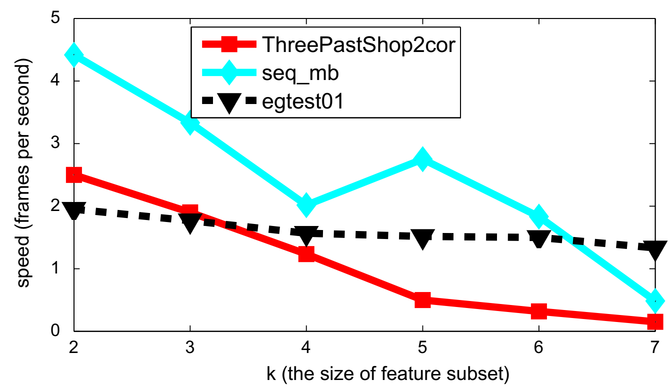



**Fig. 5.** Tracking speed on test videos under different values of $k$.

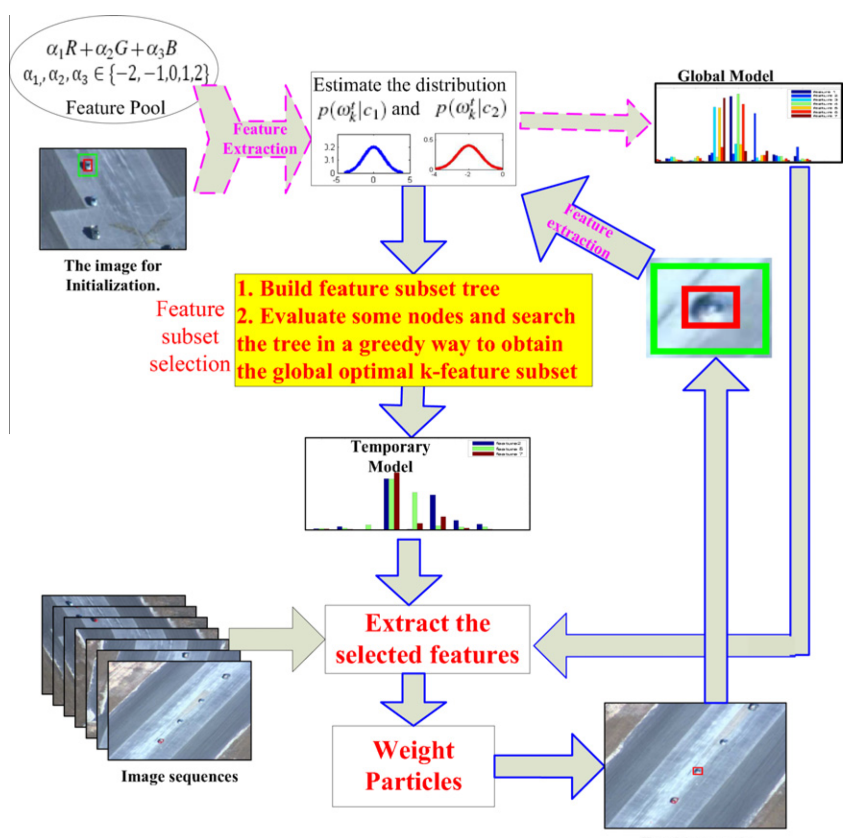



**Fig. 3.** The framework of our tracking algorithm.

$$\boldsymbol{\mu}_i^{t+1} = \frac{N_i^t \boldsymbol{\mu}_i^t + \widetilde{N_i^t}\widetilde{\boldsymbol{\mu}_i^t}}{N_i^t + \widetilde{N_i^t}}, \tag{33}$$

$$\boldsymbol{\Sigma}_i^{t+1} = \frac{1}{N_i^t + \widetilde{N_i^t}} \left( N_i^t \boldsymbol{\Sigma}_i^t + \widetilde{N_i^t}\widetilde{\boldsymbol{\Sigma}_i^t} + \frac{N_i^t \widetilde{N_i^t}}{N_i^t + \widetilde{N_i^t}} \left(\boldsymbol{\mu}_i^t - \widetilde{\boldsymbol{\mu}_i^t}\right)\left(\boldsymbol{\mu}_i^t - \widetilde{\boldsymbol{\mu}_i^t}\right)^T \right), \tag{34}$$

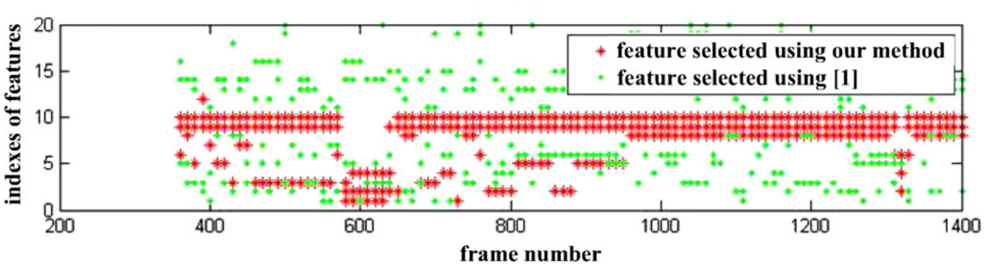
$$N_i^{t+1} = N_i^t + \widetilde{N_i^t}. \tag{35}$$

Fig. 3 provides the framework of our tracking algorithm. The best $k$-feature subset is re-selected according to Table 2 and used to compute the weight of every particle in current frame.

For the "drift" problem brought by model update [15], we collect samples (denoted by $S_0$) in the initial frame. When weighting the particle, we consider the similarity between its observation and $S_0$ (the pink and dotted arrows in Fig. 3 represent this procedure),

$$\tilde{w}(S_0, z_t, \Omega_t^*) = exp\left(-\lambda \| h(S_0, \Omega_t^*) - h(z_t, \Omega_t^*) \|^2\right), \tag{36}$$


(a) Tracking result of "*ThreePasstShop2cor*" (the 1$^{st}$ row: our result, the 2$^{nd}$ row: Collins' result, the 3$^{rd}$ row: OBT, the 4$^{th}$ row: CPT).




(b) The indexes of the features selected with our method and [1].

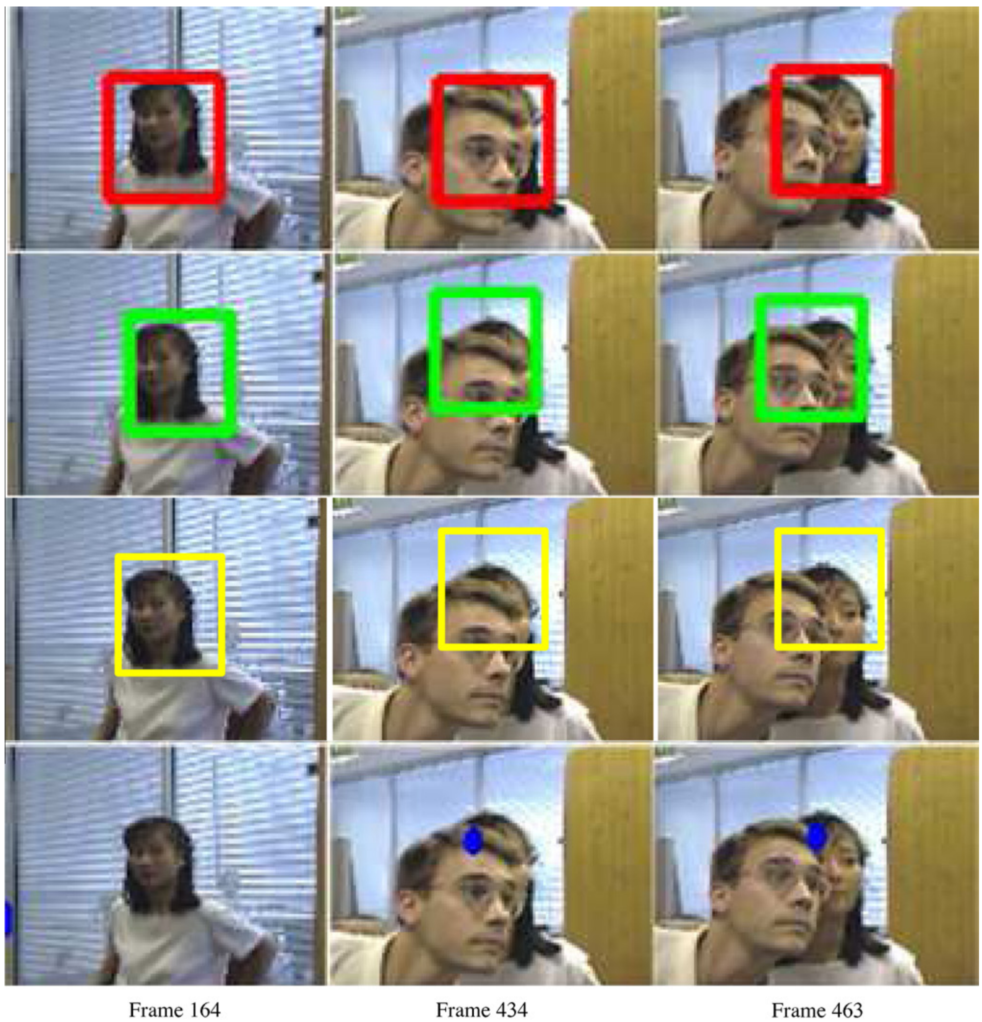
**Fig. 6.** Tracking results and the selected features on "*ThreePastShop2cor*".

where $h(x, \Omega_t^*)$ is the histogram of the projection onto $\Omega_t^*$ of $x$. Combining with Eq. (27), the weight function of particle is finally defined as,

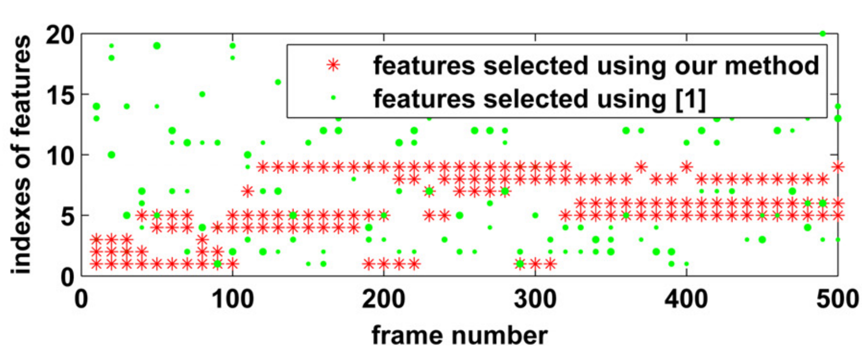
$$w(z_t) = p(s_t|z_{0:t}) + \tilde{w}(S_0, z_t, \Omega_t^*). \quad (37)$$

## 3. Experiments

In our experiments, our feature set includes 20 features which are randomly selected from 49 combinations $(\alpha_1 R + \alpha_2 G + \alpha_3 B, \alpha_1, \alpha_2, \alpha_3 \in \{-2, -1, 0, 1, 2\})$ of R, G and B [1]. We set $N_{particles} = 400$. For simplicity, motion model $p(s_{t+1}|s_t)$ is assumed to be




(a) Tracking result of "*seq_mb*" (the 1$^{st}$ row:our result, the 2$^{nd}$ row: Collins' result, the 3$^{rd}$ row: OBT, the 4$^{th}$ row: CPT).




(b) The indexes of the features selected with our method and [1].

**Fig. 7.** Tracking results and the selected features on "*seq_mb*".

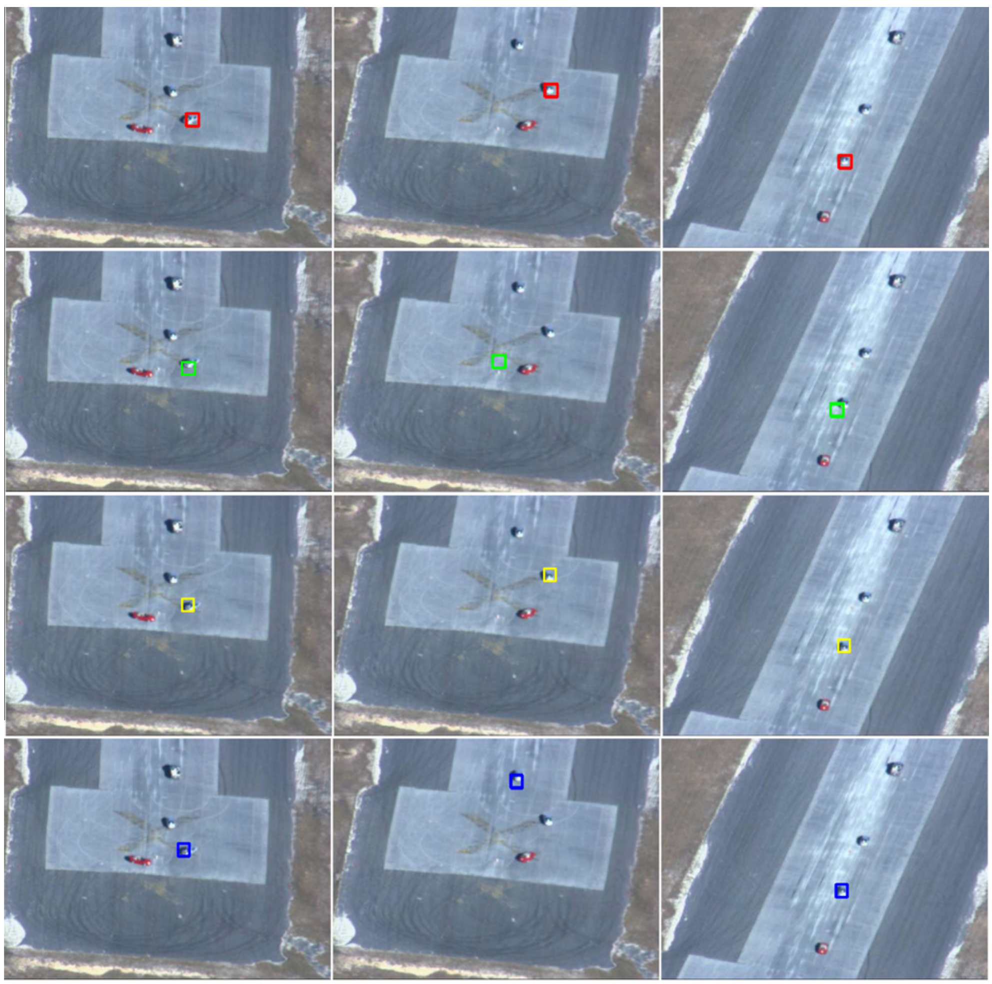
Gaussian distribution. After we locate the target object, we collect the target's rectangle area and surrounding background area as new samples. Then for every 10 frames, we add new samples into the old samples set and reselect the optimal $k$-feature subset.

### 3.1. Tracking result with different values of $k$

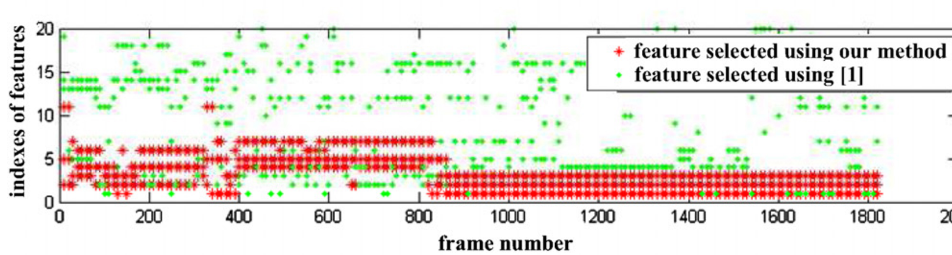
Our method could not select the most suitable value for $k$ automatically, so we run our tracker with different values of $k$ in order to obtain an appropriate value for it. Figs. 4 and 5 show the performance and tracking speed of our method on several test videos with different values for $k$. We can see that performance may not get better as $k$ gets larger. Besides, when $k$ gets larger, the tracking speed becomes slower. Since our tracker achieves the best performance on the three test videos with $k = 3$, we set $k = 3$ in our following experiment.

### 3.2. Comparison experiments

We test our method with $k = 3$ on several image sequences from different database and provide results on three challenging sequences to illustrate the benefits brought by our feature subset




(a) Tracking result of "*egtest01*" (the 1$^{st}$ row: Our result, the 2$^{nd}$ row: Collins' result, the 3$^{rd}$ row: OBT, the 4$^{th}$ row: CPT).




(b) The indexes of the features selected with our method and [1].

**Fig. 8.** Tracking results and the selected features on "*egtest01*".

selection method. We also implement Collins' [1] feature selection method and replace the feature subset selection module with it in our tracking framework. Besides, we also compare our proposed tracker with color-based probabilistic tracker (referred as CPT) [11] and online boosting tracker (referred as OBT) [4]. Both qualitative and quantitative comparisons are made between the tracking results of those methods.

### 3.2.1. Collins' feature selection method

Given a feature $f_i$, feature values of the pixels on the target and background are used to form target's histogram $H^i = [h_1^i, h_2^i, \ldots, h_b^i]$ and background's histogram $\overline{H}^i = [\bar{h}_1^i, \bar{h}_2^i, \ldots, \bar{h}_b^i]$ respectively, with $\sum_{j=1}^{b} h_j^i = 1$, $\sum_{j=1}^{b} \bar{h}_j^i = 1$. Here $b$ is the length of histogram. Let $\epsilon = 0.0001$, $L(j) = \log\left(max\{h_j^i, \epsilon\}/max\{\bar{h}_j^i, \epsilon\}\right)$ and $V(L; H^i) = \sum_{j=1}^{b} h_j^i L^2(j) - \left(\sum_{j=1}^{b} h_j^i L(j)\right)^2$. Then the discriminability of $f_i$ is evaluated separately according to $\overline{S}(f_i)$.

$$\overline{S}(f_i) = \frac{V(L; (H^i + \overline{H}^i)/2}{V(L; H^i) + V(L; \overline{H}^i)}. \tag{38}$$

The larger $\overline{S}(f_i)$, the better feature $f_i$ is. The best $k$ features are used for object tracking.

### 3.2.2. Qualitative comparison experiments and analysis

The first image sequence is from "*ThreePastShop2cor*" [12]. The greatest challenge of this sequence is that a very distractive person gets close to the interested person and they exchange their positions during frame 450–505. As shown in Fig. 6(a), our tracker tracks the target in frame 844 while the other three trackers begin to lose the target in frame 443 because target's appearance is not discriminative, according to the features they select. Fig. 1(b) shows the distribution and weights of particles using feature selected by our method while Fig. 1(c) is about that using features selected by Collins' algorithm. Tracking results in that frame are labeled by red rectangle and green rectangle in the top images of Fig. 1(b) and Fig. 1(c) respectively. We can see that our results are accurate and the compared tracker fails.

The second image sequence is "*seq_mb*". The challenge of this sequence is that it includes 360 degree out-of-plane rotation of the head, partial and complete occlusion of the woman's head by another man's head, etc. The woman rotates so fast that CPT begins to lose the target in frame 164 and tracking result shrinks to a blue dot (as shown in the 4th row of Fig. 7(a)). We can also see that when the man's head gets close to the tracked target, OBT and the tracker using Collins' feature selection method starts to drift and the tracking results are less accurate than ours. Fig. 7(a) presents some tracking examples of this image sequence and Fig. 7(b) are the indexes of the selected features.

The third image sequence is named as "*egtest01*" [13] and the tracked car loops around on a runway. Its pose and size vary noticeably. As the appearances of the road and some cars are very similar, CPT and Collin's lose the target in frame 419. Thanks to online feature selection, our tracker tracks that car successfully in frame 419. Later, due to camera's movement, CPT and Collins' method recover the correct tracking. However, in frame 550, because of error accumulation and the similar background, features selected by [1] could not track the car accurately and "drift' begins to happen. Nonetheless, our feature subset selection method succeeds in selecting the distinguishable feature subset for tracking the target car. Fig. 8 shows some representative tracking results of "*egtest01*" as well as the results of our feature selection method.

From the results of feature selection, we can see that in most time, the intersection between our result and Collins' is empty. This means the optimal $k$-feature subset is not combined with the best $k$ features. For example, in frame 440 of "*ThreePastShop2cor*", $\{f_7, f_9, f_{10}\}$ is the global optimal 3-feature subset, while they are ranked *No.* 1, *No.* 14 and *No.* 17 according to their own discriminative abilities. Fig. 9(a) shows the samples distribution in the selected feature space and in Fig. 9(b), $p(f_i|c_1)$ ($i = 7, 9, 10$) and $p(f_i|c_2)$ ($i = 7, 9, 10$) are provided. We can see that, $p(f_{10}|c_1)$ and $p(f_{10}|c_2)$ are a little similar, so $f_{10}$ is not a good feature for classifying the two classes. However, when it is combined with $f_7$ and $f_9$, the generated feature subset could distinguish the target from the background as shown in Fig. 9(a), and they lead to good tracking results in following frames (see frame 443 in Fig. 6). In frame 350 of "*egtest01*", the optimal 3-feature subset is combined with $f_1, f_5$ and $f_7$, while they are ranked *No.* 9, *No.* 20 and *No.* 7 evaluating by method in [1]. Fig. 10(b) shows the probability density function of each feature and apparently their discriminative abilities are not so good. However, as shown in Fig. 10(a), the samples in the selected feature space are easily to be separated and they generate accurate tracking result in frame 351 (see the upper left image in Fig. 8).

### 3.2.3. Quantitative comparison of tracking results

We use overlap ratio [14] to evaluate the tracking results quantitatively. For single target tracking task, "*Overlap Ratio*" (OR) is defined as,

$$OR_t = \frac{|T_t \bigcap G_t|}{|T_t \bigcup G_t|}, \quad t = 1, 2, \ldots, N_{frame} \tag{39}$$

where $N_{frame}$ is the number of frames, $T_t$ and $G_t$ denote the tracking result and ground truth in frame $t$. Obviously, the higher the overlap

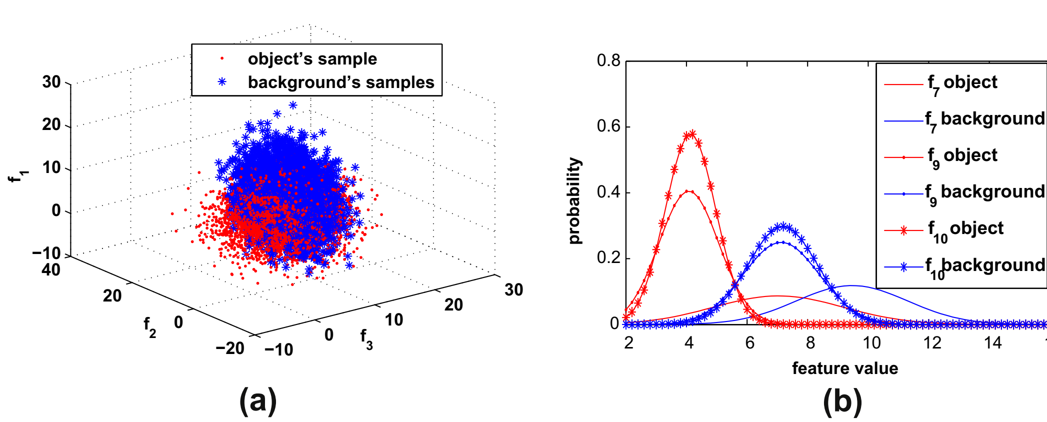



**Fig. 9.** In frame 440 of "*ThreePastShop2cor*", Samples' distribution in the selected feature space is shown in (a) and the probability density function of each selected feature is provided in (b).

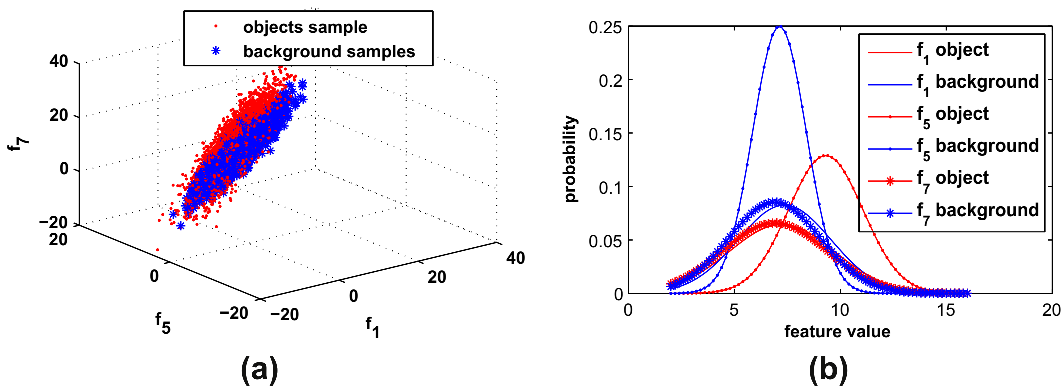



**Fig. 10.** In frame 350 of "*egtest01*", Samples' distribution in the selected feature space is shown in (a) and the probability density function of each selected feature is provided in (b).

**Table 4**
PFST on the three image sequences given different threshold.

| Threshold | 0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 |
|---|---|---|---|---|---|---|---|---|---|
| *PFST on the first image sequence* | | | | | | | | | |
| Our | 1.0 | 1.0 | 0.8 | 0.51 | 0.33 | 0.22 | 0.16 | 0.10 | 0.03 |
| Collins | 0.86 | 0.79 | 0.70 | 0.43 | 0.23 | 0.15 | 0.12 | 0.08 | 0.02 |
| OBT | 0.80 | 0.74 | 0.52 | 0.29 | 0.21 | 0.18 | 0.09 | 0.01 | 0.00 |
| CPT | 0.98 | 0.86 | 0.69 | 0.43 | 0.24 | 0.18 | 0.08 | 0.04 | 0.01 |
| *PFST on the second image sequence* | | | | | | | | | |
| Our | 1.0 | 0.93 | 0.93 | 0.87 | 0.66 | 0.48 | 0.23 | 0.07 | 0.01 |
| Collins | 0.98 | 0.87 | 0.80 | 0.66 | 0.48 | 0.30 | 0.15 | 0.04 | 0.01 |
| OBT | 0.91 | 0.91 | 0.89 | 0.86 | 0.77 | 0.54 | 0.35 | 0.14 | 0.02 |
| CPT | 0.98 | 0.86 | 0.70 | 0.43 | 0.25 | 0.18 | 0.10 | 0.04 | 0.01 |
| *PFST on the third image sequence* | | | | | | | | | |
| Our | 1.0 | 0.95 | 0.94 | 0.91 | 0.83 | 0.58 | 0.26 | 0.08 | 0.02 |
| Collins | 0.92 | 0.90 | 0.86 | 0.77 | 0.62 | 0.44 | 0.20 | 0.04 | 0.00 |
| OBT | 1.0 | 0.99 | 0.99 | 0.84 | 0.66 | 0.47 | 0.25 | 0.09 | 0.02 |
| CPT | 0.75 | 0.74 | 0.73 | 0.70 | 0.66 | 0.45 | 0.16 | 0.03 | 0.00 |

ratio is, the more accurate the tracking result is. If given a threshold, we could compute the "*Percentage of Frames Successfully Tracked*" (PFST) by the tracker.

$$PFST = \frac{\sum_{i=1}^{N_{frame}} (OR_t > threshold)}{N_{frame}}. \quad (40)$$

Table 4 displays PFST on the above three image sequences with different thresholds. We can see that under the 9 thresholds, our method performs best on every sequence. The average overlap ratios (defined as ATA in [14]) in the three image sequences are 0.38, 0.48, 0.53 and 0.32, 0.60, 0.59 and 0.39, 0.40, 0.47 using Collins' method, OBT and CPT respectively while they are 0.46, 0.58, 0.62 using our method. It has 21.1%, 20.8%, 17.0% improvement comparing to Collins' and 18.0%, 45.0%, 31.9% improvement comparing to CPT.

## 4. Conclusions

In this paper, online feature subset selection is proposed for object tracking. Evaluating feature subset as a whole can generate more discriminative features than the traditional feature selection methods for object tracking. We use FST to describe the whole *k*-feature subset space. To further reduce the cost of searching for the global optimal *k*-feature subset, we develop a greedy algorithm to search FST using conditional entropy as evaluating function. It is guaranteed to obtain the local optimal *k*-feature subset which could lead to better tracking results. The promising experimental results show that our method is more effective than the conventional feature selection method.

In our future work, we will focus on how to find out the global optimal *k*-feature subset faster, and how to decide *k* automatically. Besides, "drift" problem is still a challenge for our method and it is another topic of our future study.

## Acknowledgements

This work was supported in part by National Basic Research Program of China (973 Program): 2009CB320906, in part by National Natural Science Foundation of China: 61025011, 61035001, 61133003 and 61003165, and in part by Beijing Natural Science Foundation: 4111003.

## References

[1] R.T. Collins, Y. Liu, M. Leordeanu, Online selection of discriminative tracking features, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (10) (2005) 1631–1643.
[2] J. Wang, X. Chen, W. Gao, Online selecting discriminative tracking features using particle filter, IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2 (2005) 1037–1042.
[3] X. Liu, T.Yu, Gradient feature selection for online boosting, in: IEEE 11th International Conference on Computer Vision, 2007, pp. 1–8.
[4] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting, in: 17th British Machine Vision Conference 1, 2006, pp. 47–56.
[5] W. Liang, Q. Huang, S. Jiang, W. Gao, Mean shift blob trackingwith adaptive feature selection and scale adaption, IEEE International Conference on Image Processing 3 (2007) 369–372.
[6] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182. special Issue on variable and feature selection.
[7] X. Chen, J. Jeong, Minimum reference set based feature selection for small sample classifications, in: The 24th International Conference on Machine Learning, 2007, pp. 153–160.
[8] D. Jiang, et al., Information Theory and Coding, Second ed., pp. 77–95.
[9] J. Pearl, Heuristics: Intelligent Search Strategies for Computer Problem Solving, Addison-Wesley, 1984.
[10] M.F. Huber, T. Bailey, H. Durrant-Whyte, U.D. Hanebeck, On entropy approximation for Gaussian mixture random vectors, IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (2008) 181–188.
[11] P. Prez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, European Conference on Computer Vision (2002) 661–675.
[12] CAVIAR Test Case Scenarios at: <http://www.homepages.inf.ed.ac.uk/rbf/CAVIAR>.
[13] R.T. Collins, X. Zhou, S.K. Teh, An open source tracking testbed and evaluation web site, IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (2005).
[14] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, J. Zhang, Framework for performance evaluation of face, text, and vehicle detection and tracking in video: data, metrics, and protocol, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2009) 319–336.
[15] J. Tu, H. Tao, T. Huang, Online updating appearance generative mixture model for meanshift tracking, Machine vision and applications, 2009.