Pyramid-based Visual Tracking Using Sparsity Represented Mean Transform

Zhe Zhang Kin Hong Wong
Department of Computer Science and Engineering
The Chinese University of Hong Kong
{zzhang, khwong}@cse.cuhk.edu.hk

Abstract

In this paper, we propose a robust method for visual tracking relying on mean shift, sparse coding and spatial pyramids. Firstly, we extend the original mean shift approach to handle orientation space and scale space and name this new method as mean transform. The mean transform method estimates the motion, including the location, orientation and scale, of the interested object window simultaneously and effectively. Secondly, a pixel-wise dense patch sampling technique and a region-wise trivial template designing scheme are introduced which enable our approach to run very accurately and efficiently. In addition, instead of using either holistic representation or local representation only, we apply spatial pyramids by combining these two representations into our approach to deal with partial occlusion problems robustly. Observed from the experimental results, our approach outperforms state-of-the-art methods in many benchmark sequences.

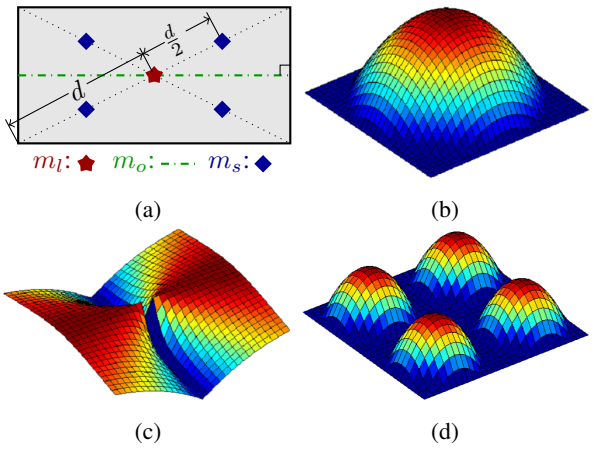


Figure 1: The visualized center points of the object window in location, orientation and scale spaces are shown in (a). The equivalent probability density functions in the Euclidean space are presented in (b), (c) and (d) when the Epanichinikov kernel is applied to three spaces respectively.

1. Introduction

Visual tracking is one of the most popular research topics in computer vision which has promising applications in many fields, such as surveillance, stabilization and video retrieval. Most proposed approaches in visual tracking can be generally classified into two groups, either discriminative methods or generative methods.

Representative discriminative methods are ensemble tracker [2], on-line boosting tracker [8], multiple instance learning [3] and structured SVM [9]. Most discriminative methods regard tracking as a classification problem and build a classifier to distinguish the object from the background. On the other hand, generative methods concentrate on modeling the appearance of the target such as histogram-based tracker [6], subspace represented tracker[16], $\mathcal{WSL}$ appearance tracker [10] and sparsity-based tracker [15].

From the literature, there are several main problems in object tracking, namely, motion estimation, illumination change and partial occlusion. In order to solve these problems, people have proposed many approaches. For example, fragment-based tracker [1] and structural local sparse tracker [11] are reported to be robust to partial occlusions. Mean shift tracker [7] and particle filter tracker [17] are stated to be able to estimate object motion well. The combination of sparse representation and trivial templates [15] is said to be less affected by illumination changes compared to other methods. Although many ideas have been proposed to overcome these problems, currently no one can solve all these problems perfectly. In the following paragraphs, we will analyze these problems and propose our corresponding solutions.

Motion estimation is one of the core components in visual tracking. The most frequently used method for motion estimation is particle filters [17].The idea of particle filters is to represent the probability densities by a set of random particles with associated weights. Through importance sampling of particle positions and Bayesian inference of the corresponding weights, particle filters can be used to esti-

mate the motion of the object. In order for good accuracy, a large number of particles are required to be sampled , leading to a high computation load. Another popular method for motion estimation is the mean shift approach [6]. As a kernel density gradient-based method, mean shift can reach the maximum of the similarity function very quickly by climbing in the direction of the probability gradient. However, the original mean shift method only estimates the location of the object window but not the scale or the orientation measurement. This greatly limits the performance of mean shift based approaches. Collins [5] tries to extend the mean shift method to handle scale space, but the additional spatial kernel makes the procedure tedious. In this paper, we extend the mean shift to both scale and orientation space and name this new approach as mean transform.

Illumination change is another difficult problem during visual tracking. The combination of sparse representation and pixel-wise trivial templates in [15] performs well under illumination changes. As we know that the computation time for sparse coding is approximately linear to the size of the dictionary. In this paper, we propose to use region-wise trivial templates instead of pixel-wise trivial templates. Compared to the pixel trivial templates, block trivial templates can keep the dictionary size as compact as possible, which enables our algorithm to be very efficient.

Partial occlusion happens very often during visual tracking. The fragments-based tracker [1] and structural local sparse tracker [11] divide the object window into smaller patches and infer the true state of our objects only through local information. But these approaches are not so stable due to the lack of holistic information. Thus we propose the spatial pyramids technique which makes full use of holistic information and local information. The idea of spatial pyramids is first proposed in image classification [13]. We adopt the spatial pyramids method here and show that it can handle partial occlusions robustly.

There are totally four contributions of this paper:

1. A new kernel density gradient-based method named as mean transform is proposed to handle the motion of the object.
2. A spatial-pyramid-based model is introduced to maintain tracking even when partial occlusions occur.
3. A new design of region-wise trivial templates is used to increase the efficiency of sparsity-based models.
4. A pixel-wise dense patch sampling scheme is applied to ensure the high accuracy of our model.

The content of this paper is organized as follows: mean transform model, sparse representation and spatial pyramids model are respectively illustrated in Section 2, 3 and 4. Section 5 presents the details of our implementation. Experimental results are shown in Section 6. Our paper is concluded in Section 7.

2. Mean transform

The principle of our mean transform is originated from mean shift [6]. In contrast to mean shift which only considers updating in location space, the proposed mean transform takes into account the motion in two other spaces: orientation space and scale space. The whole procedure of mean transform can be divided into two steps. The first step is to transform the object into specific spaces. The second step is to apply the mean shift procedure to update the location, orientation and scale in the corresponding spaces.

2.1. Mean transform in the location space

Mean transform in the location space is the same as the original mean shift. Thus this section will give a brief review of the mean shift approach. Mean shift is a gradient-based approach used to find the mode of kernel densities. Because of its high efficiency and ease of use, mean shift enjoys a popularity in the field of visual tracking [6, 5]. The details of mean shift are described as follows:

Given a window centered at point m_l, the goal is to maximize the kernel density estimator formulated as:

$$f(l_c) = \mathcal{C}_l \sum_{l \in \mathcal{S}_l} K\Big(\frac{l - l_c}{b_l}\Big), \tag{1}$$

where $K(x)$ is the kernel function, l denotes the Euclidean coordinates of points in the window, l_c denotes the Euclidean coordinates of the mode point shown as m_l in Figure 1a, $\mathcal{S}_l$ denotes the Euclidean space of the window, b_l is the bandwidth and $\mathcal{C}_l$ is a constant number for normalization. In order to locate the window center l_c with the maximum kernel density $f(l_c)$, Mean Shift updates the location l_c with the gradient

$$\Delta l_c = \nabla f(l_c) = \mathcal{C}_l \sum_{l \in \mathcal{S}_l} K'\Big(\frac{l - l_c}{b_l}\Big). \tag{2}$$

One popular isotropic kernel is the Epanichinikov Kernel:

$$K_E(x) = \begin{cases} \frac{1}{2} c_d^{-1}(d+2)(1 - \|x\|^2) & \text{if } \|x\| \le 1 \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where c_d is the volume of unit space and d is the dimension. The PDF in Euclidean space is illustrated in Figure 1b when the above kernel is applied to the location space. When the Epanichinikov Kernel is applied, mean shift updates the center of the window in the location space in the way as:

$$\Delta l_c = \mathcal{C}_l \frac{\sum_{l \in \mathcal{S}_l} (l - l_c)}{\sum_{l \in \mathcal{S}_l} 1}. \tag{4}$$

2.2. Mean transform in the orientation space

Yilmaz [19] is the first who proposed the idea of space transformation to extend mean shift. In order to model the

orientation properly, we first need to transform the points in the object from the location space into the orientation space as follows:

$$\mathcal{S}_l \rightarrow \mathcal{S}_o : o = atan\Big(\frac{y - y_c}{x - x_c}\Big), \tag{5}$$

where $[x, y]$ denotes the Euclidean coordinates of points in the window and $[x_c, y_c]$ are the Euclidean coordinates of m_l. Given the object window in the orientation space, the kernel density estimator can be formulated as follows:

$$f(o_c) = \mathcal{C}_o \sum_{o \in \mathcal{S}_o} K\Big(\frac{o - o_c}{b_o}\Big), \tag{6}$$

where $\mathcal{S}_o$ denotes the orientation space of the window, b_o is the bandwidth and $\mathcal{C}_o$ is a constant number for normalization. o_c is chosen to be the orientation value of points on the line of m_o as shown in Figure 1a. This is because in order to make mean shift valid, the distribution of the value $o - o_c$ should be isotropic in accordance with the distribution of the kernel K. This alignment has been ensured in our paper while there is no such treatment in [19].

Assume that the Epanichinikov Kernel is also applied in the orientation space. If we project the corresponding PDF in the orientation space back into the Euclidean space, the equivalent PDF of the window is shown as Figure 1c. Mean shift updates the center of the window in the orientation space as follows:

$$\Delta o_c = \mathcal{C}_o \frac{\sum_{o \in \mathcal{S}_o} (o - o_c)}{\sum_{o \in \mathcal{S}_o} 1}. \tag{7}$$

Each time after update, we rotate the candidate object window to keep the orientation of m_o horizontal. In this way, we constrain the result of Equation 5 in the range of $[-\frac{\pi}{2}, \frac{\pi}{2}]$.

2.3. Mean transform in the scale space

Similar to the treatment in orientation, for mean transform in the scale space, we first need to transform the points in the object from the location space to the scale space as:

$$\mathcal{S}_l \rightarrow \mathcal{S}_s : s = \|[x - x_c, y - y_c]\|. \tag{8}$$

The result scale coordinates s ranges in $[(0, 0), (\frac{w}{2}, \frac{h}{2})]$, where $[w, h]$ is the size of the window. The kernel density estimator can be formulated as:

$$f(s_c) = \mathcal{C}_s \sum_{s \in \mathcal{S}_s} K\Big(\frac{s - s_c}{b_s}\Big), \tag{9}$$

where s_c denotes the scale coordinates $[\frac{w}{4}, \frac{h}{4}]$ of mode points shown as m_s in Figure 1a, $\mathcal{S}_s$ denotes the scale space of the window, b_s is the bandwidth and $\mathcal{C}_s$ is a constant number for normalization.

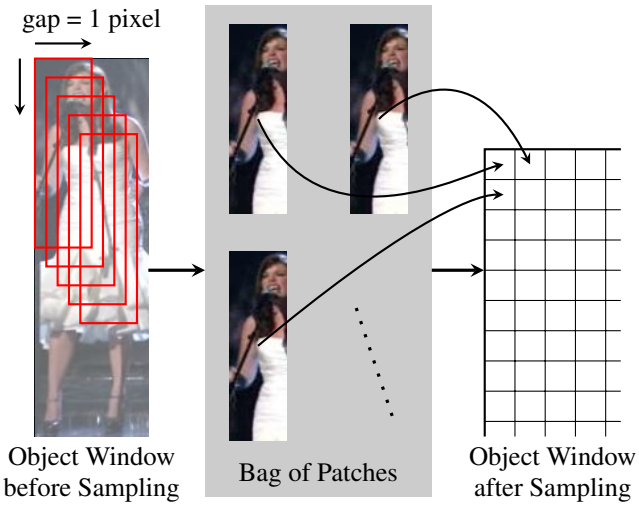


Figure 2: Illustration of dense patch sampling model

Assume that the Epanichinikov Kernel is also applied in the scale space. If we project the corresponding PDF in the scale space back into the Euclidean space, the equivalent PDF of the window is shown in Figure 1d. Mean shift updates the center of window in the scale space as:

$$\Delta s_c = \mathcal{C}_s \frac{\sum_{s \in \mathcal{S}_s} (s - s_c)}{\sum_{s \in \mathcal{S}_s} 1}. \tag{10}$$

3. Sparse representation

Similar to other mean shift based approaches, the object windows in our approach are also represented by a histogram. However, we adopt some techniques which enable the mean transform to update accurately and efficiently.

3.1. Dense patch sampling

Instead of using the raw pixels, we sample grid patches from the original object window and regard them as the basic points for the new object window. Thus, each basic point in the new object window contains much texture information. The sampling gap between two neighboring local patches is only 1 pixel, enabling the accuracy of our approach to be pixel-wise. An illustration of our sampling process is shown in Figure 2.

3.2. Dictionary learning

Given the local patches $X = [x_1, ..., x_n]$ in $R^{m \times n}$ (m is the length of each linearized patch x_i) extracted from the object window in the first frame, we learn a dictionary $\mathcal{D}$ in $R^{m \times k}$ by solving the ℓ_1-sparse coding problem:

$$\mathcal{D} = \arg\min_{D,\alpha} \sum_{i=1}^{n} (\frac{1}{2}\|x_i - \mathcal{D}\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1), \tag{11}$$

where k denotes the size of our dictionary and λ denotes the regularization constant. After learning the dictionary, we can compute the coefficients α_i of each local patch y_i extracted from the windows in the subsequent frames as:

$$\alpha_i = \arg\min_{\alpha_i} \frac{1}{2}\|y_i - \mathcal{D}\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1. \quad (12)$$

Max-pooling is used in our method, thus α_i satisfies:

$$max(\alpha_{i,j}) = 1 \quad \&\& \quad \text{others} = 0. \quad (13)$$

3.3. Histogram representation

After the coefficients are computed, we can compute the statistical histograms by applying Epanichinikov Kernel in three spaces. In the location space, $\widetilde{h}^l = [h_1^l, ..., h_k^l]$ are computed as:

$$h_j^l = \mathcal{C}_l \sum_{l_i \in \mathcal{S}_l} K\Big(\frac{l_i - l_c}{b_l}\Big)\alpha_{i,j}. \quad (14)$$

In the orientation space, $\widetilde{h}^o = [h_1^o, ..., h_k^o]$ are computed as:

$$h_j^o = \mathcal{C}_o \sum_{o_i \in \mathcal{S}_o} K\Big(\frac{o_i - o_c}{b_o}\Big)\alpha_{i,j}. \quad (15)$$

In the scale space, $\widetilde{h}^s = [h_1^s, ..., h_k^s]$ are computed as:

$$h_j^s = \mathcal{C}_s \sum_{s_i \in \mathcal{S}_s} K\Big(\frac{s_i - s_c}{b_s}\Big)\alpha_{i,j}. \quad (16)$$

The above b_l, b_o and b_s are the bandwidths used to normalize the x in $K(x)$ to make it in the range of $[-1, 1]$.

3.4. Trivial templates

Trivial templates are very useful when noise occurs or illumination changes. The trivial templates used in [15] will form an identity matrix I with each column denotes one trivial template. Such a procedure is accurate but may increase the size of the dictionary too much when the length of each patch is large. It will severely affect the efficiency during the learning of the coefficients α_i for each patch. Additionally, the fact is that noises, illumination changes and partial occlusions always occur in regions other than in pixels. Thus, we assert that it is reasonable to use block trivial templates rather than pixel-wise trivial templates. The block trivial templates are shown in Figure 3, where $[u, v]$ are the size of each region.

After trivial templates are added to the dictionary, the new dictionary will become:

$$\mathcal{D}' = [\mathcal{D}, \mathcal{T}, -\mathcal{T}] \quad (17)$$

and it will substitute $\mathcal{D}$ in 12. But only the first k elements of α_i will be kept during the subsequent computations. Through experiments, we have found that such trivial templates can handle noises, illumination changes and partial occlusions well.

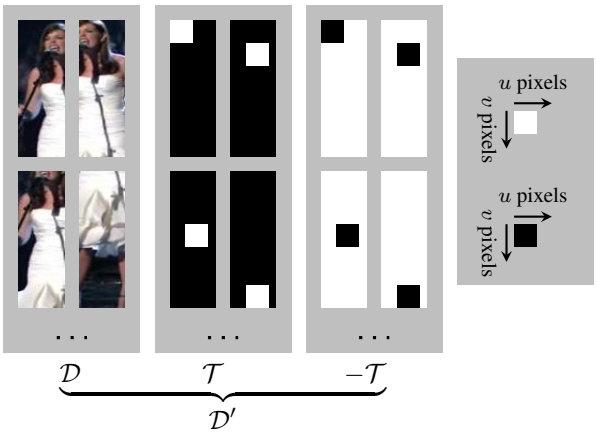


Figure 3: Illustration of block trivial templates

4. Spatial pyramids

The holistic models [15, 6] are robust to appearance changes but cannot handle the partial occlusions well. On the contrary, the local models [1] are less affected by partial occlusions but sensitive to appearance changes. If combining holistic models and local models together, we can make it robust to appearance changes and at the same time able to handle the partial occlusions. The spatial pyramids model satisfies our requirement. Figure 4 gives an illustration of our spatial-pyramid-based model. In the first level of pyramids, the object window after sampling is regarded as a unit and based on which the histograms $\widetilde{h}_1$ are computed. In the higher levels of pyramids, the object window is divided into smaller regions. The number smaller regions in each level is computed as:

$$n = 4^{r-1}, \quad (18)$$

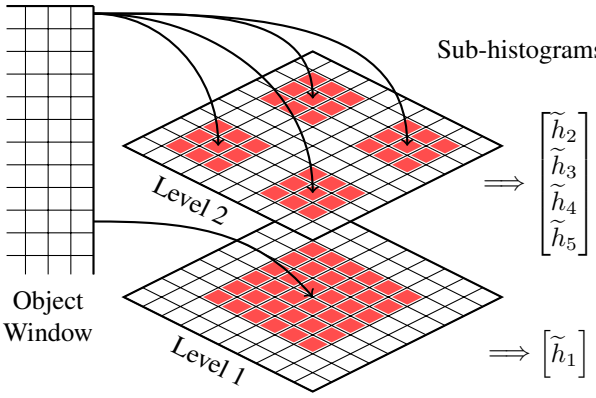
where r denotes the number of levels.

At last, we concatenate all the histograms computed from different levels of pyramids and form a long histogram shown as $\mathcal{H}$ in Figure 4.

5. Implementation

The whole tracking procedure can be divided into two parts. The first part is to learn the dictionary for sparse coding and compute the target histogram for the initial object window. The second part is to infer the new state of the object through candidate histogram in the subsequent frames. Assume the target histograms and the candidate histograms are denoted by $\mathcal{Q}^{l,o,s}$ and $\mathcal{P}^{l,o,s}$ respectively where l, o, s represent three spaces. Then our goal is to maximize the Bhattacharyya coefficient between $\mathcal{Q}^{l,o,s}$ and $\mathcal{P}^{l,o,s}$ which can be formulated as:

$$\rho(\mathcal{Q}, \mathcal{P}) = \sum_{u \in \{l,o,s\}} \sum_{v=1}^{z} \sum_{j=1}^{k} \sqrt{\mathcal{Q}_{v,j}^u \mathcal{P}_{v,j}^u} \quad (19)$$



Total-histogram $\mathcal{H} = [\widetilde{h}_1, \widetilde{h}_2, \widetilde{h}_3, \widetilde{h}_4, \widetilde{h}_5]$

Figure 4: Illustration of spatial pyramids model

where z denotes the total number of regions from pyramids, k denotes the size of dictionary.

Taylor expansion is then applied to linearly approximate the $\rho(\mathcal{Q}, \mathcal{P})$. Assume the previous values for l_c, o_c and s_c are l_0, o_0 and s_0 respectively, then we have:

$$\begin{aligned}\rho\big(\mathcal{Q}, \mathcal{P}(u_c)\big) &\approx \frac{1}{2}\sum_{u,v,j}\sqrt{\mathcal{Q}^u_{v,j}\mathcal{P}^u_{v,j}(u_c)} \\ &\quad + \frac{1}{2}\sum_{u,v,j}\mathcal{P}^u_{v,j}(u_c)\sqrt{\frac{\mathcal{Q}^u_{v,j}}{\mathcal{P}^u_{v,j}(u_0)}} \\ &= \frac{1}{2}\sum_{u,v,j}\sqrt{\mathcal{Q}^u_{v,j}\mathcal{P}^u_{v,j}(u_c)} \\ &\quad + \sum_u \frac{C_u}{2}\sum_{u_i\in\mathcal{S}_u} w^u_i K\Big(\frac{u_i - u_c}{b_u}\Big), \end{aligned} \tag{20}$$

where

$$w^u_i = \sum_{v,j}\alpha_{i,j}\sqrt{\frac{\mathcal{Q}^u_{v,j}}{\mathcal{P}^u_{v,j}(u_0)}}, \tag{21}$$

and $u \in \{l, o, s\}$, $v \in [1, z]$, $j \in [1, k]$. In the above formulas, the $\alpha_{i,j}$ are computed from equation 12 and 13. We address that three types of w^u_i where $u \in \{l, o, s\}$ should be distinguished but not regarded as the same as in [19]. Treating the three w^u_i as the same is quite inappropriate which may cause the failure of mean shift.

Finally, the solution to equation 20 can be computed by applying the mean transform procedure as:

$$\Delta u_c = \frac{\sum_{u_i\in\mathcal{S}_u} w^u_i(u_i - u_c)}{\sum_{u_i\in\mathcal{S}_u} w^u_i}. \tag{22}$$

The above $\Delta l_c, \Delta o_c$ and Δs_c can be estimated simultaneously, therefore the mean transform is also efficient as the original mean shift. After locating the u_c, the target histograms are updated as:

$$\mathcal{Q}(k) = (1-\beta)\mathcal{Q}(k-1) + \beta\mathcal{P}(k), \tag{23}$$

where β is set to be 0.02 when the similarity between $\mathcal{Q}^u_{v,j}$ and $\mathcal{P}^u_{v,j}$ is greater than the threshold, otherwise β is set to be 0. The whole pipeline of our approach is summarized in Algorithm 1.

Algorithm 1: The Procedure of Visual Tracking

Input: The object window in the start frame $\mathcal{W}_{start}$
Output: Tracked object windows in the following frames $\mathcal{W}_{start+1...end}$

```
Sample patches X according to Section 3.1 ;
Learn the dictionary D given X according to Section 3.2 ;
Divide the object window into pyramids regions [R_1, ...R_z] according to Section 4;
Calculate the target histograms H according to Section 3.3 and concatenating them to form Q;
for frame ← start + 1 to end do
    Use the tracked window in the previous frame as the initial window W_frame = W_{frame-1};
    for i ← 1 to m do
        Sample patches X according to Section 3.1 ;
        Divide the object window into pyramids regions [R_1, ...R_z] according to Section 4;
        Calculate the candidate histograms H according to Section 3.3 and concatenating them to form P;
        Apply mean transform to update W_frame according to Equation 22 ;
    end
    Update each segment of Q individually;
end
```

6. Experiment

In this part, we present our experimental settings as well as the evaluation criteria. Subsequently, we conduct quantitative and qualitative comparisons with other approaches.

6.1. Dataset and comparison methods

In order to evaluate our methods and other methods thoroughly, we have selected 10 representative sequences with different challenging properties from the benchmark paper [18] (http://visual-tracking.net/) and CAVIAR dataset (http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/). Details about the sequences are shown in Table 1.

	ASLA	Frag	IVT	L1APG	MTT	STRUCK	TLD	Ours
Car4	1.76	12.44	4.80	3.33	1.84	3.79	22.75	2.09
CarDark	1.19	24.52	1.54	1.30	1.45	1.49	30.88	1.18
David	4.20	39.36	3.60	53.61	27.36	9.70	39.04	5.00
Dudek	8.74	87.06	10.94	9.61	11.10	17.82	31.85	8.27
Faceocc2	22.72	14.17	6.13	9.32	5.90	6.02	17.60	3.93
Singer1	3.85	28.02	12.34	4.60	5.58	13.37	10.57	3.15
Caviar	3.62	19.87	70.59	65.22	61.88	65.06	62.48	3.59
Woman	145.75	110.39	137.71	122.52	130.58	3.36	59.17	3.21
David2	1.64	3.78	1.96	4.06	45.57	2.98	2.47	1.63
Walking2	3.13	57.30	2.63	2.52	2.10	11.96	60.55	4.85
STD	44.76	34.95	44.65	40.49	41.23	18.86	21.41	2.05

Table 2: Results of average center error (pixels). For each sequence, the rank-1st, rank-2nd and rank-3rd results are marked in red, green and blue respectively. The first row gives all the trackers and the first column shows all the sequences in our experiment. The last row is the standard deviation of the results for each tracker.

	ASLA	Frag	IVT	L1APG	MTT	STRUCK	TLD	Ours
Car4	0.86	0.46	0.73	0.77	0.87	0.48	0.61	0.84
CarDark	0.80	0.14	0.84	0.86	0.80	0.86	0.35	0.86
David	0.68	0.29	0.66	0.26	0.36	0.52	0.46	0.72
Dudek	0.77	0.49	0.79	0.77	0.72	0.66	0.60	0.81
Faceocc2	0.54	0.61	0.79	0.68	0.76	0.68	0.57	0.82
Singer1	0.77	0.31	0.47	0.77	0.44	0.35	0.69	0.82
Caviar	0.85	0.42	0.20	0.21	0.20	0.20	0.19	0.86
Woman	0.17	0.24	0.16	0.18	0.17	0.77	0.20	0.71
David2	0.79	0.65	0.75	0.69	0.32	0.71	0.67	0.81
Walking2	0.81	0.27	0.79	0.76	0.80	0.52	0.26	0.78
STD	0.21	0.17	0.25	0.27	0.27	0.20	0.20	0.05

Table 3: Results of average VOC overlap ratio (pixels). The table structure is the same as that of Table 2.

Properties	Sequences
Scale Variation	*Car4*, *Singer1*, *Walking2*
Orientation Variation	*Dudek*, *Faceocc2*, *David2*
Illumination Change	*CarDark*, *David*, *Singer1*
Partial Occlusion	*Faceocc2*, *Caviar*, *Woman*

Table 1: Challenging properties and the corresponding sequences

The ground truths of most sequences are inherited from the original datasets while ground truths of some other sequences such as *dudek*, *faceocc2* and *david2* are annotated by ourselves taking into account the rotation element. We have also applied some corrections for sequences such as *car4* and *caviar*.

We compare our method against 7 recent state-of-art visual tracking methods: ASLA tracker [11], Frag tracker [1], IVT tracker [16], L1APG tracker [4], MTT tracker [20], STRUCK tracker [9] and TLD tracker [12]. All these trackers are tested with tuned parameters to achieve their best performance.

6.2. Implementation details

All our experiments are tested in MATLAB R2010b on a PC with 3.1GHz Intel Core i5 CPU and 4GB memory. SPAMS package [14] is applied to learn the dictionary and solve the ℓ_1 minimization problems. In our approach, the object windows are resized to be 31×31 pixels and all the local patches are sampled with the size 16×16 pixels, thus in each window we can sample 256 patches. The size of our dictionary k is selected as 40. Surrounding the initial object position, about 5 particles are sampled in case of the abrupt motion of the object. All these settings are kept the same through all the sequences. Our source code and experimental result are available on the website `http://www.cse.cuhk.edu.hk/~khwong/demo/cvpr14`.

6.3. Quantitative analysis

We have employed two widely used criteria to evaluate the performance of all the trackers. The first criterion is the Average Center Error (ACE) which measures the average distance between the center of tracked object window

Figure 5: Tracking results of the trackers in all sequences

and the center of ground truth window. Results of ACE are summarized in Table 2. The second criterion is the PASCAL VOC Average Overlap Ratio (AOR) which is defined as $S_{AOR} = \frac{area(R_T \cap R_{GT})}{area(R_T \cup R_{GT})}$. Table 3 presents the results of AOR. Overall, our method outperforms state-of-the-art methods in most sequences and can achieve similar performance in the remaining sequences.

6.4. Qualitative analysis

Tracking results of all the trackers in ten sequences are shown in Figure 5. In the following, we give the detailed analysis on all the sequences and all the trackers according to the properties listed in the Table 1.

Scale variation: In the *Car4* sequence, the object car becomes smaller at first and becomes larger afterwards. In both *Singer1* and *Walking2* sequences, the scales of the object women decrease gradually and becomes less than half the size of the original objects eventually. Through all these sequences, ASLA, IVT, L1APG, MTT, TLD and our trackers can sense the scale change of the objects and adjust the window size accurately while Frag and STRUCK trackers do not perform well in this aspect. However, we do not use

the particle samplings (rather time consuming) like other methods but use the mean transform to find the scale modes the objects. Such procedure usually takes less than 5 steps in our approach, therefore our approach is very fast.

Orientation variation: The object faces in the *Dudek*, *Faceocc2* and *David2* sequences undergo great orientation variations. Only ASLA, IVT and our trackers can detect the rotation changes and locate the objects successfully. The performance of our tracker in these three sequences indicates that Mean Transform can also handle the rotation problems effectively and efficiently.

Illumination change: In the *CarDark* sequences, the object car suffers from great illumination changes. The Frag tracker fails to track the object from frame 80 and TLD tracker also drifts from frame 240. Other trackers can perform well in this sequence. In the *David* sequences, the man walks from a dark region to a bright region in the room. The Frag and TLD trackers lose the target from frame 380 and 540 respectively. The L1APG tracker also drifts from frame 546. The state light varies drastically in the *Singer1* sequence. In the frame 126, Frag, MTT, L1APG and IVT trackers heavily deviate from the object singer. Among all the trackers, our tracker is the most robust one in this set of sequences.

Partial occlusion: In the *Faceocc2* sequences, the object face is in turn heavily occluded by the book and the hat. In the frame 491, only ASLA and our tracker can infer the accurate location of the object face through partial information. But the ASLA tracker drifts off from the frame 531. In the *Caviar* sequence, the man in black walks from the leftmost to the rightmost location and he is occluded by two other people consecutively. From frame 85, most trackers drift to the man in red. At last, only ASLA, STRUCK and our tracker can successfully track the object through the sequence. In the *Woman* sequence, nearly half of the object woman is occluded by the car. From the frame 130, most trackers start to drift off and at the end only STRUCK and our tracker can accurately locate the object. The ASLA tracker can perform well in most cases but very sensitive to the initializations and not so robust to partial occlusions. Overall, our tracker can handle partial occlusions very well.

7. Conclusion and discussion

In summary, we propose a robust method for visual tracking combining techniques of mean transform, sparse coding and spatial pyramids. The newly proposed mean transform can model the location change, scale change and rotation change effectively and efficiently. Additionally, by merging the holistic model and local model together, the spatial pyramids model can handle partial occlusions well. Lastly, the dense patch sampling technique and the trivial template designing scheme make our method more robust to noise. The experimental results indicate that our approach outperforms state-of-the-art methods in many benchmark sequences.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, pages 798–805, 2006.

[2] S. Avidan. Ensemble tracking. *PAMI*, 29(2):261–271, 2007.

[3] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, pages 983–990, 2009.

[4] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *CVPR*, pages 1830–1837, 2012.

[5] R. Collins. Mean-shift blob tracking through scale space. In *CVPR*, pages 234–240, 2003.

[6] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002.

[7] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *PAMI*, 25(5):564–577, 2003.

[8] H. Grabner and H. Bischof. On-line boosting and vision. In *CVPR*, volume 1, pages 260–267, 2006.

[9] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *ICCV*, pages 263–270, 2011.

[10] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. *PAMI*, 25(10):1296–1311, 2003.

[11] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, pages 1822–1829, 2012.

[12] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *PAMI*, 34(7):1409–1422, 2012.

[13] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, pages 2169–2178, 2006.

[14] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *ICML*, pages 689–696, 2009.

[15] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *PAMI*, 33(11):2259–2272, 2011.

[16] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008.

[17] M. Sanjeev Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *SP*, 50(2):174–188, 2002.

[18] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, pages 2411–2418, 2013.

[19] A. Yilmaz. Object tracking by asymmetric kernel mean shift with automatic scale and orientation selection. In *CVPR*, pages 1–6, 2007.

[20] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via multi-task sparse learning. In *CVPR*, pages 2042–2049, 2012.