

Transferring Visual Prior for Online Object Tracking

Qing Wang Feng Chen Jimei Yang Wenli Xu Ming-Hsuan Yang

Abstract—Visual prior from generic real-world images can be learned and transferred for representing objects in a scene. Motivated by this, we propose an algorithm that transfers visual prior learned offline for online object tracking. From a collection of real-world images, we learn an over-complete dictionary to represent visual prior. The prior knowledge of objects is generic and the training image set does not necessarily contain any observation of the target object. During the tracking process, the learned visual prior is transferred to construct an object representation by sparse coding and multi-scale max pooling. With this representation, a linear classifier is learned online to distinguish the target from the background, and to account for the target and background appearance variations over time. Tracking is then carried out within a Bayesian inference framework, in which the learned classifier is used to construct the observation model and a particle filter is used to estimate the tracking result sequentially. Experiments on a variety of challenging sequences with comparisons to several state-of-the-art methods demonstrate that more robust object tracking can be achieved by transferring visual prior.

Index Terms—Visual prior, object tracking, object recognition, sparse coding, transfer learning.

I. INTRODUCTION

Object tracking has been an important and active research topic in computer vision with numerous applications including surveillance, traffic control, human-computer interfaces, motion analysis, to name a few. The main challenge in developing a robust tracking algorithm is to account for large appearance variations of the target object and background over time. In this paper, we tackle this problem with both prior and online visual information. By learning visual prior from real-world images and transferring it to the tracking task, we propose an adaptive tracking algorithm to account for appearance variations of the target and background.

The central theme of our approach is to exploit generic visual prior for object tracking. Although object tracking is usually an online task and visual information of the target may be scarce before the task starts, some useful prior can still be exploited offline especially on patch level. We note that there is a huge amount of real-world image data at our disposal. These images are likely to include similar holistic observations of the target object in the tracking task, but most

Q. Wang, F. Chen and W. Xu are with National Laboratory for Information Science and Technology, Department of Automation, Tsinghua University, Beijing, 10084, P.R. China. E-mail: qing-wang07@mails.tsinghua.edu.cn; chenfang@mail.tsinghua.edu.cn; xuwl@mail.tsinghua.edu.cn. Q. Wang and F. Chen are supported in part by NSFC (grant 61071131) and Beijing Natural Science Foundation (grant 4122040). W. Xu is supported in part by National Key Basic Research and Development Program of China (grant 2009CB320602).

J. Yang and M.-H. Yang are with Electrical Engineering and Computer Science, University of California, Merced, CA, 95344. E-mail: jyang44@ucmerced.edu; mhyang@ucmerced.edu. M.-H. Yang and J. Yang are supported in part by NSF CAREER Grant #1149783 and NSF IIS Grant #1152576.

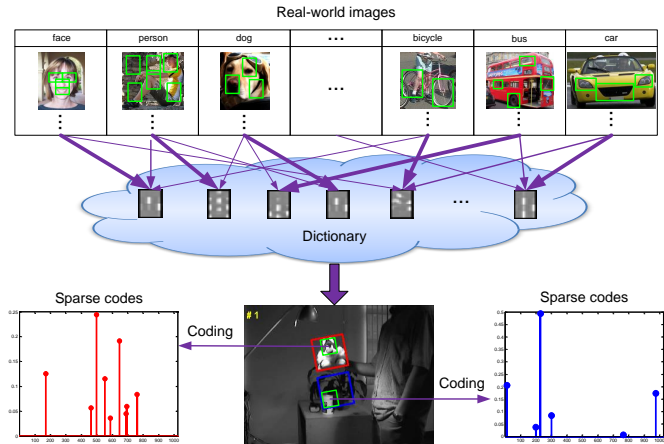


Fig. 1: Transferring visual prior to object tracking. We learn visual prior from real-world images and represent it with an over-complete dictionary based on SIFT feature. The width of each arrow to the dictionary illustrates the contribution of one object class to one basis in the dictionary. With ℓ_1/ℓ_2 sparse coding on the learned dictionary, visual prior is transferred from the real-world images to the target object for tracking. The coding coefficients measure the contribution of a basis to represent an object patch, and reflect the resemblance of the target and real-world object classes.

likely they may not. Nevertheless, local patches from these images often share great similarity. Motivated by this, we learn generic visual prior from a large set of image data with an over-complete dictionary and sparse coding.

For visual tracking, we transfer the learned visual prior for object representation by sparse coding. Figure 1 shows the prior representation and transfer processes in the proposed tracking algorithm. By filtering the sparse representation results at different scales, the corresponding object level representation is obtained. With some samples from the target and background in the first frame, a classifier is initialized to distinguish them and the tracking task is formulated within the Bayesian inference framework. To account for the appearance changes of the target object and background during tracking, we update the classifier when new tracking results are obtained. Furthermore, to alleviate the visual drift problem during classifier update, we retain the initial classifier as a detector. In each frame, the observation model for Bayesian inference is constructed by a combination of the detector and the online classifier.

For most tracking algorithms in the literature, either strong prior information of the target object is assumed or no prior knowledge is exploited. When an algorithm depends heavily on the prior, all views of the target should be known or detailed geometric model is assumed before tracking, and the

application domains of such algorithms are limited. In contrast, online tracking methods can be applied to numerous problems as no prior is required. However, the tracking results after a long duration are usually unpredictable as only online visual information is used. Our algorithm exploits the strength of both approaches. In particular, we propose an algorithm that learns generic visual prior offline and transfers such knowledge to online object tracking.

The contributions of our tracking algorithm are summarized as follows. First, we learn a generic visual prior offline without assuming any specific knowledge of the target object for tracking. On the patch level, small images often share structural similarity, which motivates us to exploit such prior information offline and use it for online visual tracking. Second, the prior is represented by an over-complete dictionary and learned by sparse coding from local patches. It is different from the widely used orthogonal dictionary (subspace) learned from holistic images by Principal Component Analysis (PCA) or its variants. The non-orthogonal, over-complete dictionary learned from local patches makes our visual prior more effective for object description. Third, with the learned dictionary, ℓ_1/ℓ_2 sparse coding and multi-scale max pooling, a high-level object representation is constructed, and a simple classifier is capable of separating the target from the background.

II. RELATED WORK

There is a rich literature in object tracking and a thorough review on this topic can be found in [28]. To deal with the problem of large object and background appearance variations, most recent tracking algorithms focus on developing robust object representation schemes.

Since it is difficult to find a set of features that are invariant to appearance variations of target objects and backgrounds, learning algorithms have been adopted for this task. Based on a specific prior of the target, an object model can be learned offline. Black et al. [4] learn a subspace model to represent target objects at fixed views. In [5], Black et al. extend their subspace representation method to a mixture model which can better account for object appearance. Avidan [1] uses a set of vehicle and non-vehicle images collected offline to learn a classifier for car tracking. All these methods depend heavily on the specific prior. That is, these methods are developed for specific objects of interest. When all possible views of the target are known before tracking, object appearance models can be well constructed. However in most real-world tracking applications, it is difficult to enumerate all possible appearance variations of objects. Therefore, such tracking algorithms have limited application domains.

Numerous adaptive appearance models have been recently proposed for object tracking. In these algorithms, object representation can be initialized and updated with online observations without any prior. Jepson et al. [13] learn a Gaussian mixture model via an online expectation maximization (EM) algorithm to account for target appearance variations during tracking. Aside from mixture models, incremental subspace methods based on PCA or its variants have been used for online object representation [24], [17]. To overcome the

problem of partial occlusion, sparse representation has also been utilized for object tracking [21]. In [15], the authors extend the conventional particle filtering framework with multiple dynamic and observation models to account for target appearance variation caused by change of pose, illumination, scale as well as partial occlusion.

Object tracking has also been posited as a binary classification problem. Collins et al. [7] propose a method to select discriminative color features online for tracking whereas Avidan [2] uses online boosting method to classify pixels belonging to foreground and background. Recently, numerous approaches have been proposed to deal with the drift problem when updating the learned appearance model or classifier online with newly obtained tracking results. Grabner et al. [11] regard all the object information corresponding to the tracking results as unlabeled data and adapt a classifier within the semi-supervised learning framework. Babenko et al. [3] use Multiple Instance Learning (MIL) to handle ambiguously labeled positive and negative data obtained online to reduce visual drift. Kalal et al. [14] propose a method to handle unbalanced samples which exploits the underlying structure to select positive and negative samples for online update. All these tracking algorithms do not assume any prior regarding the target object class and can be applied to numerous problems. However, persistent object tracking with these methods is difficult as it is not clear whether the updated visual information is correct or not (e.g., new observations may contain image regions from the background and thus incorrect information is updated).

Sparse coding algorithms model an observed example as a linear combination of a few elements from an over-complete dictionary. The recent development of sparse coding/representation has attracted much interest and has been used in image denoising [8], [20], image classification [23], [27] and object tracking [21]. These methods have proven that learning dictionary from data outperforms pre-chosen (fixed) ones (e.g., wavelet) since the former can significantly reduce reconstruction error [8]. Different from the representations based on PCA and its variants [24], [17], such sparse models do not impose that the bases in the dictionary be orthogonal, which allows more flexibility to adapt the representation to the data [19]. In [21], the sparse representation of a target object is achieved by optimizing an objective function which includes two terms: one measures the reconstruction error and the other measures the sparsity. However, these methods are generative at its core (based on reconstruction error) for determining tracking results and are not equipped to distinguish target and background patches. In [23], the authors carry out sparse coding on raw image patches for image classification. In [27], the authors perform sparse coding on SIFT features [18] and achieves state-of-the-art performance for image classification on public benchmarks.

III. LEARNING VISUAL PRIOR WITH SPARSE CODING

We first present how generic visual prior can be learned from numerous images of diverse object classes. Although we can get a large number of real-world images, there is no straightforward method to exploit and represent generic visual

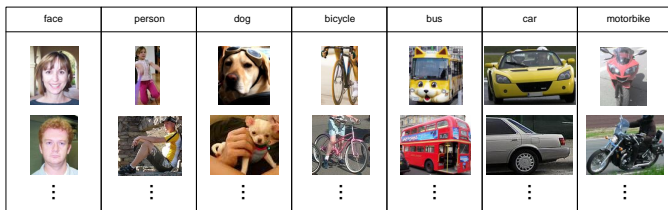


Fig. 2: Sample images for learning visual prior.

prior in the tracking literature. In this paper, we use sparse coding to learn the visual prior from large image sets of diverse objects with an over-complete dictionary.

A. Image set

This work aims to bridge the gap between object recognition (based on visual prior) and object tracking (based on prior or online information). On the patch level, small images often share structural similarity. This is why we exploit such prior information offline from existing data sets and use it for online visual tracking. The VOC2010¹ and Caltech101² data sets which consist of a large variety of objects are used for learning visual prior. Without loss of generality, we use object classes which are common in surveillance scenarios from these two data sets, including non-rigid (face, person and dog) and rigid (bicycle, bus, car and motorbike) objects. Some images of these classes are shown in Figure 2. It is worth noting that other related object images can also be used to learn a prior for specific tracking tasks.

B. Learning dictionary

Since sparse coding based on SIFT descriptor has been proved to outperform sparse coding on raw image patches in computer vision [27], we also choose SIFT as the basic appearance descriptor in our tracking method. We extract the SIFT descriptors from overlapped patches of each gray scale image, and learn the dictionary in an unsupervised manner. Let $X = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{m \times n}$ be the SIFT descriptors we extract from the image set, where m and n are the dimensionality of each SIFT descriptor and the number of SIFT descriptors, respectively. Denote $D = [\mathbf{d}_1, \dots, \mathbf{d}_k] \in \mathbb{R}^{m \times k}$ ($k \gg m$) as the dictionary we want to learn, this problem can be formulated as :

$$\min_{D, \{\mathbf{a}_i\}} \frac{1}{2} \sum_{i=1}^n \|\mathbf{x}_i - D\mathbf{a}_i\|^2 + \beta \sum_{i=1}^n \|\mathbf{a}_i\|_1 \quad (1)$$

$$s.t. \quad \|\mathbf{d}_j\|_2 \leq 1, \forall j \in \{1, \dots, k\}$$

where $\mathbf{a}_i \in \mathbb{R}^k$ is the sparse coefficient vector of \mathbf{x}_i when encoded by the dictionary D . The parameter β is a trade-off between reconstruction error and sparsity. To enlarge the sparsity of the learned coefficient vector, we can increase β and vice versa. The ℓ_2 -norm constraint on \mathbf{d}_j is to prevent arbitrarily small values of $\mathbf{a}_i, \forall i \in \{1, \dots, n\}$. Although there are a large number of SIFT descriptors extracted from the data set, D is learned offline with the sparse coding method proposed in [16]. Some bases (column vectors) in D are shown in the image form in Figure 1. This dictionary D contains generic structured information of different objects

from numerous classes and is used to encode generic visual prior of objects.

Different from pre-chosen or randomly selected codebook, the learned dictionary contains structural information of object appearance and can represent an object with less reconstruction error. Although it may be possible to represent an object with pre-chosen or randomly selected codebook [10], [22], the coding coefficient vectors are likely to be denser and more sensitive to noise.

IV. TRANSFERRING VISUAL PRIOR FOR OBJECT REPRESENTATION

The learned visual prior is represented by the over-complete dictionary D . We transfer this prior for object tracking by representing an object with D . For each SIFT descriptor inside an object region, a sparse coefficient vector is learned by performing ℓ_1/ℓ_2 sparse coding on the dictionary D . Then an object is represented by applying multi-scale max pooling on the coding results of all the local SIFT descriptors in their corresponding image region.

A. ℓ_1/ℓ_2 sparse coding

To represent an object, we first extract the SIFT descriptors from its image patches, and then encode them with the learned dictionary. Let $X = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{m \times N}$ denote the SIFT descriptors extracted from an object image, the ℓ_1/ℓ_2 sparse coefficient vector $\mathbf{a}_i \in \mathbb{R}^k$ for coding \mathbf{x}_i by the learned dictionary D can be calculated by [29]:

$$\min_{\mathbf{a}_i} \frac{1}{2} \|\mathbf{x}_i - D\mathbf{a}_i\|_2^2 + \lambda_1 \|\mathbf{a}_i\|_1 + \frac{\lambda_2}{2} \|\mathbf{a}_i\|_2^2 \quad (2)$$

where λ_1 and λ_2 are regularization parameters. When $\lambda_2 = 0$, it leads to the ℓ_1 -norm sparse coding problem which has been widely used in [21], [27]. The choice of the $\lambda_2 > 0$ makes the problem of Eq. 2 becomes strictly convex. The coding results of all the descriptors in X are denoted by a sparse coefficient matrix $A = [\mathbf{a}_1, \dots, \mathbf{a}_N] \in \mathbb{R}^{k \times N}$, where each column of A denotes the coding result of the SIFT descriptor for an image patch.

With ℓ_1/ℓ_2 sparse coding, the SIFT descriptors from different objects can be encoded by different bases in the dictionary. The sparse coefficients from the dictionary reflect some structured information of the image, thereby facilitating inference of their class label (see Figure 1). From this sparse coding process, the visual prior of generic objects is transferred to the tracking task.

B. Comparison with other decomposition methods

The sparse coding method we use for object representation is different from the PCA method and vector quantization (VQ) codebook. For object tracking, PCA has been widely used to learn specific prior (with orthogonal dictionary/subspace) [4], [5] and to model the target appearance online [24]. It is well known that in PCA the training data is assumed to be Gaussian distributed and solved by ℓ_2 optimization. When this assumption does not hold (e.g., due to occlusion), object representation using the dictionary learned by PCA is not effective. On the other hand, VQ based codebook is also widely used for dictionary learning

¹<http://pascal.in.ecs.soton.ac.uk/challenges/VOC/voc2010/index.html>

²http://www.vision.caltech.edu/Image_Datasets/Caltech101/

and encoding. Since VQ focuses on minimizing the overall reconstruction of data points, it is not effective in representing a diverse collection of image as there is a trade-off between ℓ_2 error and codebook size. Sparse coding can be regarded as a generalized VQ in which the ℓ_1 regularization allows \mathbf{a}_i of Eq. 2 to have more than one zero elements. Thus, sparse coding can achieve a much lower reconstruction error due to this less restrictive constraint. Compared to the ℓ_1 sparse coding formulation, the use of ℓ_1/ℓ_2 constraints can lead to more stable coding results [29].

C. Multi-scale max pooling

For the tracking task, we need to define object level feature for a target or background sample over the sparse representation matrix \mathbf{A} . There exist numerous methods for representing an object with a set of descriptors, and here we use a pooling function which operates on each row of \mathbf{A} and obtain a vector $\mathbf{b} \in \mathbb{R}^k$. Since each row of \mathbf{A} corresponds to the response of all local SIFT descriptors in X to one specific basis in dictionary D , different pooling functions may generate different image statistics. To make the representation more robust to local spatial translations, we use the max pooling function on the absolute sparse codes:

$$\mathbf{b}_i = \max \left\{ |\mathbf{a}_{i,1}|, \dots, |\mathbf{a}_{i,N}| \right\}, \quad (3)$$

where \mathbf{b}_i is the i -th element of \mathbf{b} and $\mathbf{a}_{i,j}$ is the element of i -th row and j -th column of \mathbf{A} . As discussed in [26], [27], the max pooling process is well established with biophysical evidence in visual cortex and has been shown to be effective for object representation with local responses.

To preserve the spatial information and local invariance, we use multi-scale max pooling to obtain the object level representation. This pooling process searches across different locations and over different scales of the object image and combines all local maximum responses. In this work, it is implemented by dividing the whole object image into M non-overlapped spatial cells, applying max pooling on the coding results of descriptors in each cell and concatenating the pooled features from all the spatial cells:

$$\mathbf{z} = [\mathbf{b}_1^T, \dots, \mathbf{b}_M^T]^T, \quad (4)$$

where \mathbf{b}_i is the max pooling result of the i -th spatial cell, M is the number of spatial cells, and $\mathbf{z} \in \mathbb{R}^{Mk}$. With this process, we obtain a pyramid representation of an object which is robust to local transformation.

V. LEARNING CLASSIFIER

After extraction of SIFT features, ℓ_1/ℓ_2 sparse coding and multi-scale max pooling, we obtain a spatial pyramid representation for each object image. With the over-complete dictionary D , SIFT descriptors either from a target or a background object image can be represented well by its sparse coefficient vector with low reconstruction error. Therefore, different from [24], [21], it is less effective to use observation models based on the reconstruction error for object tracking. With sparse coding and multi-scale max pooling, images of different object classes are often represented by different bases

of the learned dictionary. Therefore, it is easier to separate the samples from different classes. In this paper, we pose tracking as a binary classification problem, in which a linear classifier is learned to separate the target from the background. We use the logistic regression to learn the classifier and use the classification score as our similarity measure for object matching. Although patches from different objects in the same category may be represented by similar sparse coefficients, the object level representation based on multi-scale max pooling exploits unique spatial characteristics of a target object. That is, our representation scheme consists of local feature response (based appearance) as well as their geometric shape. Thus, the learned classifier is target-specific which can be used to discriminate a target from objects in the same or different categories. To account for appearance variations of the target and background during tracking, the classifier is updated with the most recent observations.

A. Classifier initialization

To initialize the classifier, we need to collect some target (positive) and background (negative) samples with the sparse representation and multi-scale max pooling process. In the first frame, when the target is labeled manually or by an object detector, we can get a set of target images with small location perturbations. To collect background samples, we first randomly draw samples from an annular region defined by $\gamma < \|\mathbf{I} - \mathbf{I}_t\| < \eta$ (γ and η are inner and outer radiuses), in which $\mathbf{I}_t \in \mathbb{R}^2$ is the location of the target sample and $\mathbf{I} \in \mathbb{R}^2$ is the sample location. After extracting SIFT descriptors, representing these target and background images by sparse coding and multi-scale max pooling, we obtain a set of training data, denoted by $(\mathbf{z}, y) \in \mathbb{R}^{Mk} \times \{+1, -1\}$, where y is the class label. The linear classifier can be obtained by minimizing the cost function:

$$J(\mathbf{w}) = \frac{1}{n_s} \sum_{i=1}^{n_s} \ell(y_i, \mathbf{w}, \mathbf{z}_i) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2, \quad (5)$$

where \mathbf{w} is the classifier parameter set we want to learn, ℓ is a loss function, and n_s is the number of training samples. The parameter $\lambda > 0$ controls the strength of the regularization term. In our method, we use the logistic loss function:

$$\ell(y, \mathbf{w}, \mathbf{z}) = \log \left(1 + e^{-y\mathbf{w}^T \mathbf{z}} \right). \quad (6)$$

The corresponding classifier can be denoted by:

$$f(\mathbf{z}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{z}}}. \quad (7)$$

Once the classifier is initialized, the classification score can be utilized as the similarity measure for tracking.

B. Online update of the classifier

The image appearance of target and background may change due to many factors such as pose, scale, illumination, and occlusion. Intuitively, object appearance remains the same only for a certain period of time, and eventually the initialized model will be no longer accurate as time progresses. To account for image variations for robust object tracking, we update the classifier adaptively with new observations. With

the recently obtained target observations and some negative samples extracted in the current frame, the classifier can be updated. We have experimented with incremental learning methods (e.g., stochastic gradient algorithm [6]) for online classifier update. However, such update methods are sensitive to the learning rate empirically, and thus in our method we update the classifier by retraining with the method presented in Section V-A. Since the number of samples for retraining is small, the classifier can be updated efficiently. As only the most recently obtained target observations are needed to be stored in the memory and the negative samples are collected in the current image frame, our algorithm does not have large memory requirement and is flexible to deal with long sequences.

VI. PROPOSED TRACKING ALGORITHM

In this paper, object tracking is carried out within the Bayesian inference framework. Given the observation set of the target $\mathbf{z}_{1:t} = \{\mathbf{z}_1, \dots, \mathbf{z}_t\}$ up to time t , the target state (motion parameter set) \mathbf{x}_t can be determined by the maximum a posteriori (MAP) estimation:

$$\hat{\mathbf{x}}_t = \arg \max_{\mathbf{x}_t} p(\mathbf{x}_t | \mathbf{z}_{1:t}), \quad (8)$$

where $p(\mathbf{x}_t | \mathbf{z}_{1:t})$ can be inferred by the Bayesian theorem in a recursive manner (with Markov assumption):

$$p(\mathbf{x}_t | \mathbf{z}_{1:t}) \propto p(\mathbf{z}_t | \mathbf{x}_t) p(\mathbf{x}_t | \mathbf{z}_{1:t-1}), \quad (9)$$

where $p(\mathbf{x}_t | \mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t | \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1} | \mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1}$. The tracking process is governed by a dynamic model, $p(\mathbf{x}_t | \mathbf{x}_{t-1})$, and an observation model, $p(\mathbf{z}_t | \mathbf{x}_t)$.

A particle filter method [12] is adopted here to estimate the target state. In the particle filter, $p(\mathbf{x}_t | \mathbf{z}_{1:t})$ is approximated by a finite set of samples $\{\mathbf{x}_t^i, i = 1, \dots, N_s\}$ with importance weights $\{\omega_t^i, i = 1, \dots, N_s\}$. The candidate sample \mathbf{x}_t^i is drawn from an importance distribution $q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{z}_{1:t})$ and the weight of the i -th sample is:

$$\omega_t^i = \omega_{t-1}^i \frac{p(\mathbf{z}_t | \mathbf{x}_t^i) p(\mathbf{x}_t^i | \mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i | \mathbf{x}_{t-1}^i, \mathbf{z}_{1:t})}, \quad (10)$$

where $q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{z}_{1:t}) = p(\mathbf{x}_t | \mathbf{x}_{t-1})$ in this work.

The dynamic model $p(\mathbf{x}_t | \mathbf{x}_{t-1})$ delineates the temporal correlation of the target states in consecutive frames. In our algorithm, we approximate the motion of a target between two consecutive frames with affine transformation. Let \mathbf{x}_t be the six-dimensional vector where each parameter is modeled independently by a scalar Gaussian distribution centered at its counterpart of \mathbf{x}_{t-1} . Thus, the dynamic model is formulated as $p(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}_{t-1}, \Sigma)$, where Σ is a diagonal covariance matrix whose elements are the variances of the affine parameters.

The observation model $p(\mathbf{z}_t | \mathbf{x}_t)$ denotes the likelihood of \mathbf{x}_t generating observation \mathbf{z}_t . It plays a key role for robust tracking because it directly corresponds to the core challenge of tracking, i.e., unpredictable variations such as appearance or background changes. When the classifier is available, the observation model can be constructed as

$$p(\mathbf{z}_t | \mathbf{x}_t) \propto c_t \quad (11)$$

Algorithm 1 Summary of the Online Tracking Algorithm.

- 1: **Input:** Video frames F_1, \dots, F_T .
 - 2: **Output:** Target states $\mathbf{x}_1, \dots, \mathbf{x}_T$.
 - 3: **for** $t = 1, \dots, T$ **do**
 - 4: **if** $t = 1$ **then**
 - 5: Transfer prior for object representation.
 - 6: Initialize the classifier with parameter set \mathbf{w}_1 .
 - 7: **else**
 - 8: Transfer prior for object representation.
 - 9: Estimate \mathbf{x}_t using particle filtering.
 - 10: Store target observation corresponding to \mathbf{z}_t .
 - 11: **if** The number of target observations is equal to some predefined threshold **then**
 - 12: Collect a number of negative samples in the current frame.
 - 13: Use the target observations (positive samples) and negative samples to update \mathbf{w}_t .
 - 14: Clear the target observation set.
 - 15: **else**
 - 16: $\mathbf{w}_t = \mathbf{w}_{t-1}$
 - 17: **end if**
 - 18: **end if**
 - 19: **end for**
-

where $c_t = f(\mathbf{z}_t)$ is the classification score at time t . To alleviate the visual drift problem during update, we retain the initial classifier trained in the first frame as a detector. In each frame, the total classification score can be calculated by

$$c_t^* = (1 - r)c_1 + rc_t \quad (12)$$

where c_1 and c_t are the classification scores of the detector initialized in the first frame and the adaptive classifier at time t , respectively. We denote c_t^* as the final classification score at time t and r is a predefined constant, which determines the confidence of our adaptive classifier. That is, the adaptive classifier helps account for rapid appearance change while the detector alleviates the drift problem. With the prior learned offline, the online tracking algorithm is carried out as summarized in Algorithm 1.

VII. EXPERIMENTS

We evaluate our tracker on twelve challenging image sequences (some of them are publicly available) against several state-of-the-art algorithms. The challenging factors in these sequences include pose change, illumination change, occlusion, cluttered background, image blur, and camera motion.

A. Implementation

The proposed algorithm consists of an offline prior learning module and an online object tracking component. In the offline phase, the SIFT descriptors are densely extracted from 16×16 patches on a grid with step size of 8 pixels from each selected image (based on intensity). With about 200,000 SIFT descriptors, and learn a 128×1024 dictionary. For convenience of further analysis, we add a label to each basis vector in the dictionary to show which object class contributes most to this

basis (Note that a basis vector may be learned by the SIFT descriptors from different object classes, which is illustrated in Figure 1).

For multi-scale max pooling in the online tracking phase, we use 3 spatial levels, and set the numbers of cells on 3 levels to be 1, 4 and 9, respectively. Therefore, the dimensionality of each object representation is $\mathbf{z} \in \mathbb{R}^{14336}$. In the first frame, the linear classifier f is initialized with 60 target samples and 60 background samples. The classifier parameter \mathbf{w} is updated every 10 frames as a trade-off of effectiveness and efficiency. The confidence ratio r is set to be 0.8, and the number of particles is 300 in all experiments. The videos and source codes are available at <http://faculty.ucmerced.edu/mhyang/papers/tip11a.html>.

B. Baseline experiments

In the *Sylvester* sequence [24], the target (plush toy) undergoes pose and illumination change. The results in Figure 3 (a) show that the proposed algorithm is able to track the target well. In the *Wall-E* sequence shown in Figure 3 (b), the proposed algorithm is able to account for appearance variation due to drastic scale and pose change (3-D motion) of the target. Note that we do not have the same object images in the training set. These two experiments demonstrate that the learned generic visual prior can be used to represent different kinds of objects.

To illustrate how the visual prior is transferred for object tracking, we plot the probability of learned bases of each object class being selected to represent the target. Since we can determine which object class contributes most to one basis when learning the dictionary offline, we mark each basis with the corresponding class label. Thus during tracking, the class label of each basis being used to encode each local SIFT descriptor can be identified. With the coding coefficients, we can compute the probability of the visual prior transferred from each training object class to the corresponding local descriptor. For the *Sylvester* and the *Wall-E* sequences, we select one image patch (marked by a blue polygon) from the target region and plot the sparse coding results on the dictionary and the histograms of prior being transferred from each training object class in Figure 3 (a) and Figure 3 (b). We note that the visual prior of the *dog* object class is used in the *Sylvester* sequence, the visual prior of the *bus* is exploited most in the *Wall-E* sequence. The results can be explained by the fact that some patches of these targets bear significant resemblance to the object classes in the training images (i.e., plush toy vs. dog, robot car vs. bus).

C. Qualitative evaluation

We compare our tracker with several object tracking algorithms: the incremental visual tracker (IVT) [24], the variance ratio tracker (VRT) [7], the L1 tracker (L1T) [21], the multiple instance learning tracker (MILT) [3], the visual tracking decomposition tracker (VTD) [15], and the tracking-learning-detection (TLD) method [14]. We obtain the tracking results of these compared methods with the code provided by the authors. For the IVT, L1T, VTD and our methods, we use the same parameters for particle filtering. Both the IVT and L1T

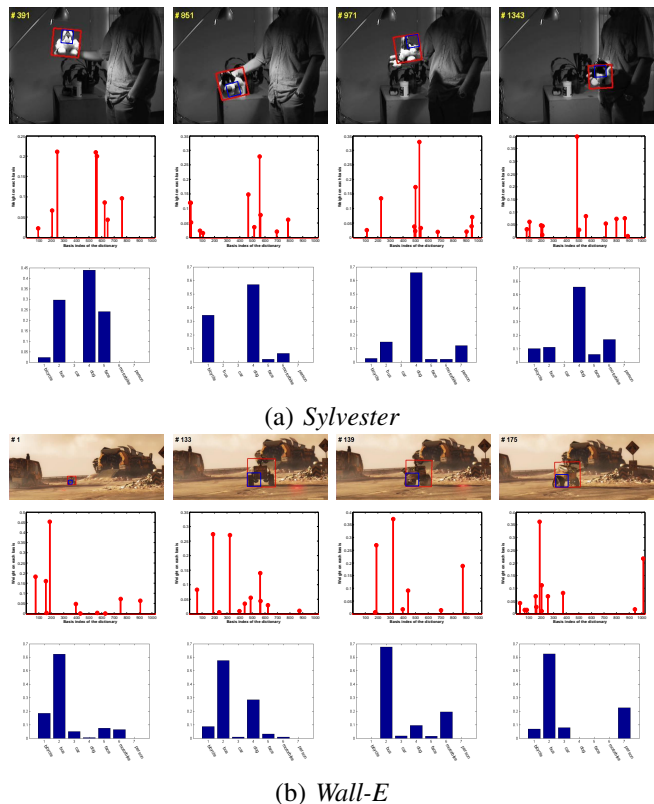
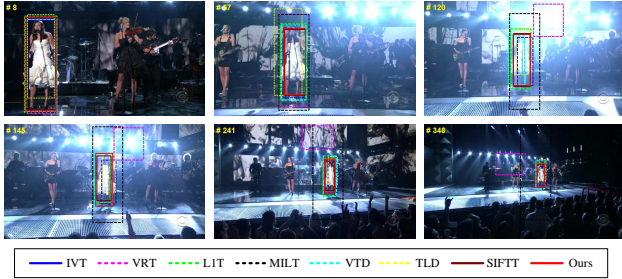


Fig. 3: Tracking results and prior transfer of the *Sylvester* and *Wall-E* sequences. The second row of (a)/(b) shows the coding results of an image patch inside the object region over time, which reflects the weight changes of each basis in the dictionary that an image patch is dependent on. The third row of (a)/(b) illustrates the prior transfer from the real-world images to the target object (SIFT descriptor extracted from the image patch in the blue polygon). The numbers on the x -coordinate refer to different classes of training data. The y -coordinate illustrates how much prior is transferred to the tracking object from each object class.

methods learn dictionaries online from test videos. The bases learned in the IVT method are orthogonal to each other and the number of bases is not larger than the dimensionality of the feature vector. On the contrary, the dictionary in the L1T method is over-complete. The TLD method is equipped with a detection procedure to help find the object after occlusion. The IVT, L1T and VTD trackers are generative methods, while the others are discriminative trackers. We also use experiments to show that the prior transfer is vital for robust object tracking. To this end, we implement a tracking algorithm (referred to as SIFTT) wherein the SIFT descriptors are directly used for object representation by multi-scale max pooling. All the other modules are the same as our algorithm. In the following sections, we present some representative tracking results.

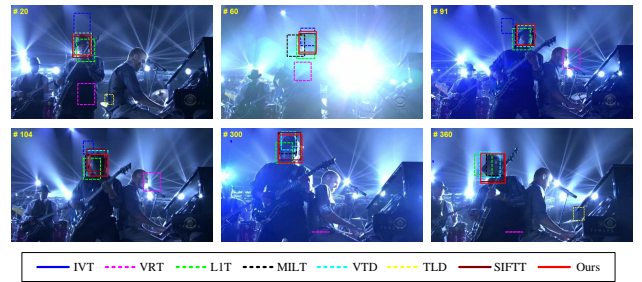
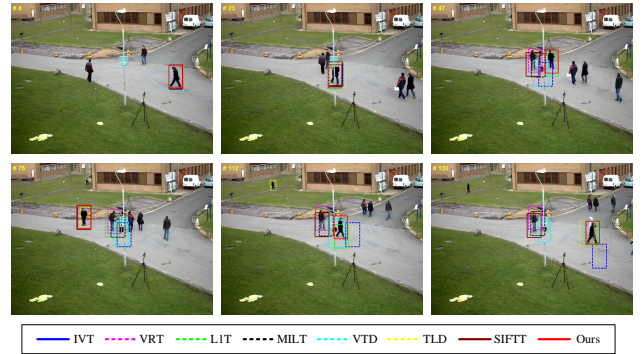
Pose change: The *Freeman* sequence is used to test the performance of our tracker in handling pose change. There is also significant scale change when the target walks towards the camera. From the tracking results illustrated in Figure 4, we note our method and the TLD method perform well while the IVT, L1T, MILT, VTD and SIFTT have some tracking

Fig. 4: Tracking results of the *Freeman* sequence.Fig. 5: Tracking results of the *singer* sequence.

errors. The VRT method fails gradually.

Illumination change: We use the *singer* and *shaking* sequences [15] to evaluate whether our tracker is able to handle drastic illumination change. Some representative tracking results are shown in Figure 5 and Figure 6. In both sequences, our tracker performs well when compared with the VTD method. In the *singer* sequence, there are large scale changes of the target and unknown camera motion in addition to illumination change. The VRT method fails to track the target when the illumination changes. The MILT and TLD algorithms also succeed in tracking the target but does not deal with the scale changes well. The IVT, LIT and SIFTT methods have similar tracking results as our tracker. In the *shaking* sequence, the target undergoes pose variation besides illumination change. The LIT, MILT and SIFTT algorithms are also able to track the target whereas the IVT, VRT and TLD methods drift from the target quickly. Our tracker uses an online update mechanism to account for the appearance variation of the target and background over time, and retains a detector to alleviate visual drift problem. In addition, the object representation based on sparse coding and multi-scale max pooling are less sensitive to illumination and pose change, thereby achieving good tracking performance.

Occlusion: We use the *PETS2009* sequence and the *CAVIAR* sequence to test the performance of our tracker when the target object undergoes heavy occlusion. In the *PETS2009* sequence, there is also out-of-plane pose change besides the heavy occlusion. The tracking results presented in Figure 7 show that our tracking method and the TLD tracker succeed in tracking the target object after the heavy occlusion, while the others all fail. In this sequence, the geometric shapes and local responses of the target objects are different from the other objects. Consequently, they can be used to differentiate the other objects. For the TLD method, it has a detection

Fig. 6: Tracking results of the *shaking* sequence.Fig. 7: Tracking results of the *PETS2009* sequence.

procedure, thereby also succeeding in tracking after occlusion. In the *CAVIAR* sequence, it is difficult to keep track of the target after occlusion because there are other objects with similar appearances in the scene. The IVT, LIT, MILT, VTD, TLD and SIFTT methods do not perform well whereas the VRT algorithm performs slightly better. In contrast, our method exploits visual prior, and represents objects by coding results of local image patches described by SIFT feature descriptors for learning a target-specific classifier. Thus, our method is able to discriminate the target from others in the same object category and performs well in this sequence. The initial classifier also facilitates the proposed method to keep track of the target when heavy occlusion occurs.

Background clutter: We use the *box* [25], *board* [25] and *NBA* sequences to evaluate our tracker in handling background clutter. In the *box* sequence shown in Figure 9, there are also partial occlusions, which adds difficulty for object tracking. From the tracking results in Figure 9, we observe that our method and the TLD tracker perform better than the other trackers. In the *board* sequence shown in Figure 10, our tracker performs well when the target undergoes out-of-plane rotation (3D-motion). All the other methods lose track of the target in some frames. In the *NBA* sequence, the target object is similar to other object in the scene. The IVT, VRT, LIT, MILT, VTD, TLD and SIFTT methods all lose track of the target gradually whereas our tracker succeeds in most of the frames except toward the end when the target undergoes occlusion for numerous frames.

Image blur and low contrast: Figure 12 and Figure 13 demonstrate the performance of these trackers in handling scenarios with image blur and low foreground-background contrast. In the *car* sequence shown in Figure 12 the target also undergoes partially occlusion and pose variation other than

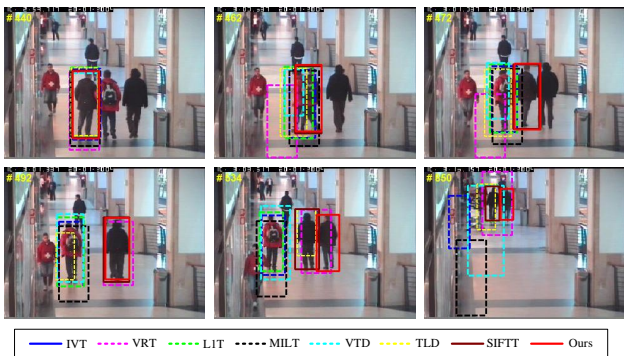
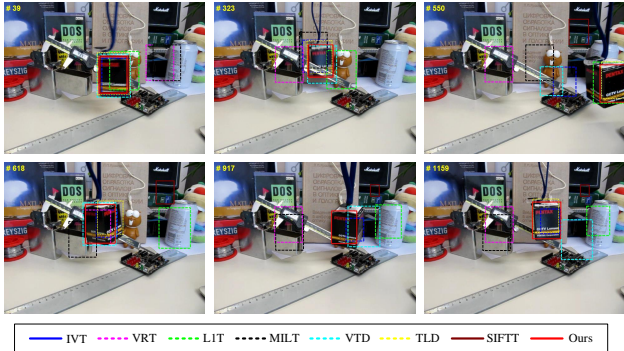
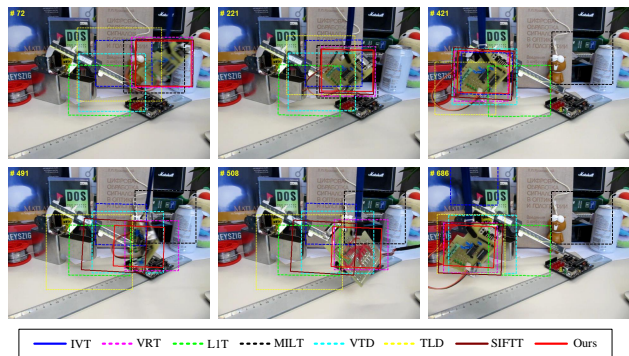
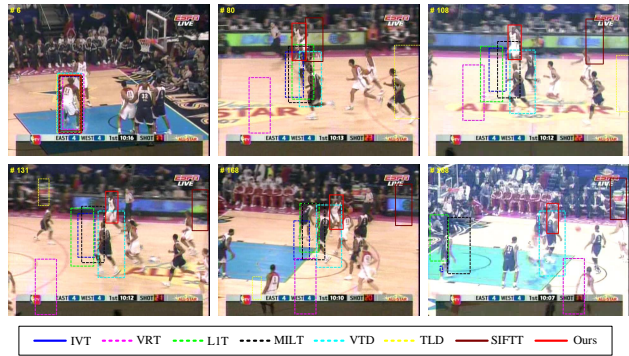
Fig. 8: Tracking results of the *CAVIAR* sequence.Fig. 9: Tracking results of the *box* sequence.

image blur caused by camera motion. Our tracker performs well in this sequence whereas the LIT method fails quickly when the car is partial occluded by a bus. It can be explained by the fact that global intensity features used in the LIT method cannot discriminate the car object from buses that have similar holistic appearance. The IVT and VRT methods perform better than LIT, but lose track of the target when image blur occurs. The MILT, VTD, and TLD methods are able to track the target in this sequence although with some errors in the last frames. The SIFTT method provides similar tracking result as ours. In the *David* sequence shown in Figure 13, the contrast between the target and the background is low. In addition, there is severe occlusion and a large pose change of the target. The IVT method fails quickly after he is partially occluded by a pole, and the VTD and TLD methods gradually fail when David walks in front of the van. MILT method fails after the target walks behind a tree. The VRT and LIT methods perform better but also fail gradually after the target turns around. On the other hand, our tracker and the SIFTT method succeeds throughout this sequence.

D. Quantitative evaluation

Aside from the qualitative comparison, we compute the tracking success rate and center location error using the ground truth labeled manually at every 5 frames. We employ the criterion used in the PASCAL VOC challenge [9] to determine whether each tracking result is a success. Given the tracked bounding box ROI_T and the ground truth bounding box ROI_G , the score is defined as $score = \text{area}(ROI_T \cap ROI_G) / \text{area}(ROI_T \cup ROI_G)$. The tracking result in one frame is considered as a success when this score is

Fig. 10: Tracking results of the *board* sequence.Fig. 11: Tracking results of the *NBA* sequence.

above 0.5. Table I shows the tracking results in terms of success rates. The center location error is defined as the distance between the central locations of the tracked target and the ground truth. The tracking results in terms of center location errors are illustrated in Figure 14. The results demonstrate that our tracking algorithm performs well against the other state-of-the-art methods. Note that in the *Sylvester*, *Wall-E*, *box* and *board* sequences, no samples of these object classes are included in the training set. The experimental results demonstrate that the learned visual prior is generic and can be applied to different tracking tasks. The comparisons also demonstrate the necessity of learning and transferring visual prior instead of using the SIFT descriptor directly for object representation.

VIII. CONCLUSION

This work exploits *generic* visual prior learned from real-world images for online tracking of *specific* objects. On the patch level, small images often share structural similarity, and such prior information can be learned offline and used for modeling objects in online visual tracking. We have presented an effective method that learns and transfers visual prior for robust object tracking. With a large set of natural images, we represent visual prior with an over-complete dictionary. We transfer the learned prior to tracking tasks by sparse coding and represent the object with the multi-scale max pooling method. With newly arrived samples of the target and background, a classifier is learned online to discriminate the target object from background and a particle filter algorithm is utilized to estimate the target state sequentially. Compared to the related state-of-the-art tracking methods, the proposed tracking

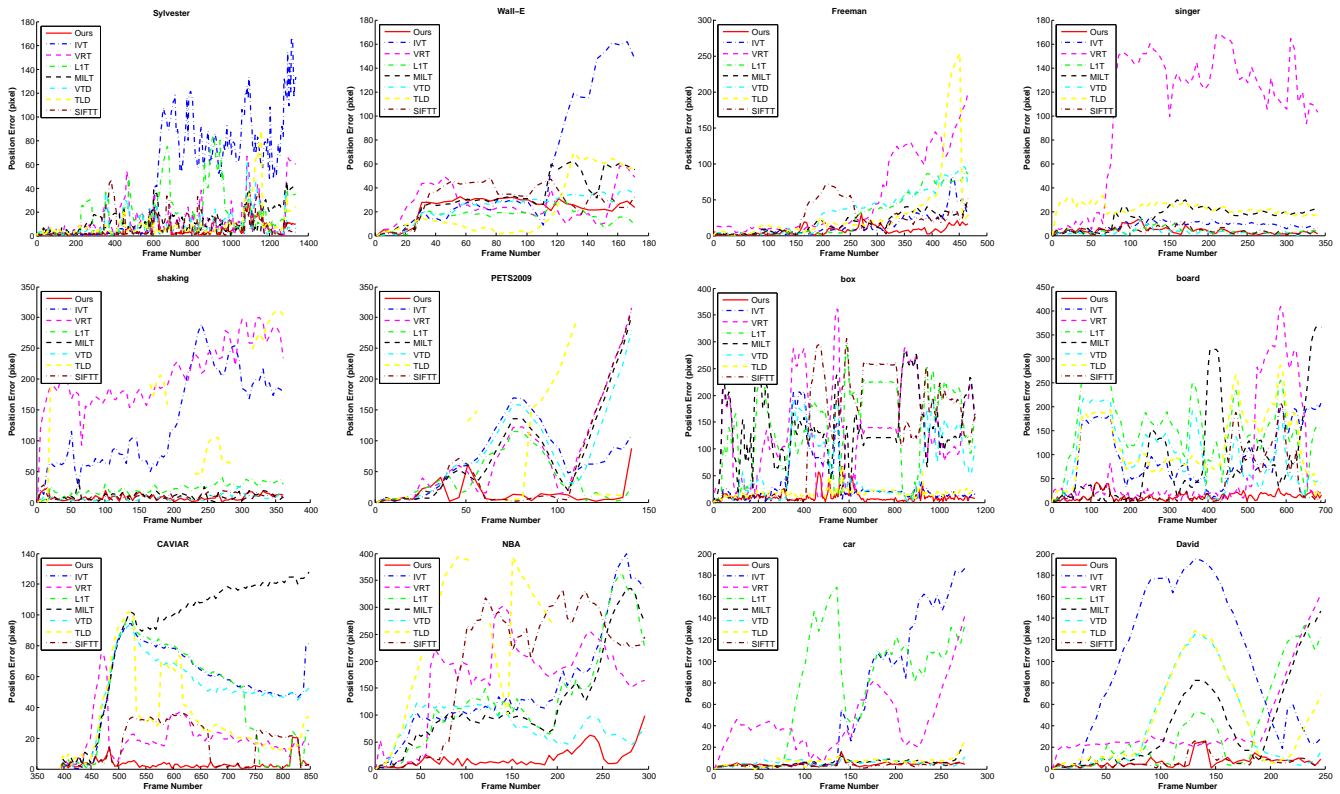


Fig. 14: Error plots of the test sequences.

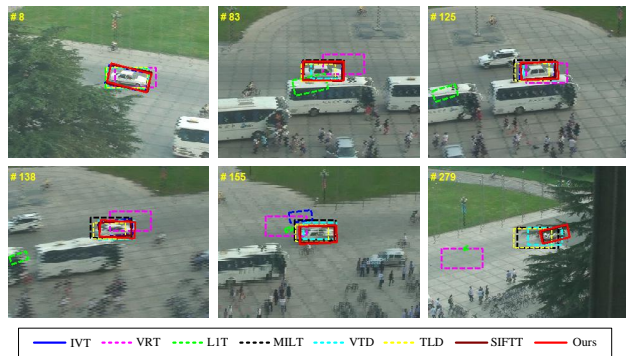


Fig. 12: Tracking results of the *car* sequence.

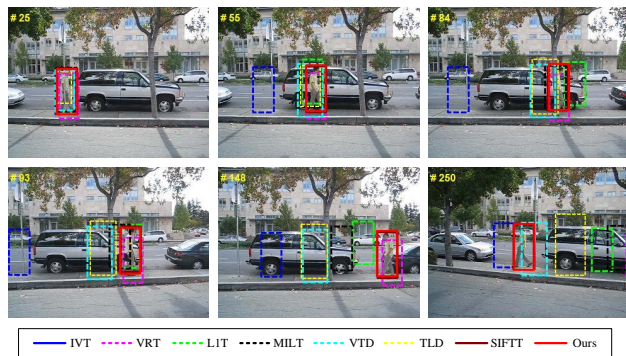


Fig. 13: Tracking results of the *David* sequence.

algorithm is demonstrated to perform robustly in complex environments where the target and background undergo different kinds of variations.

TABLE I: Success rates (%). The best two results are presented with bold face and italic fonts.

	IVT	VRT	LIT	MILT	VTD	TLD	SIFTT	Ours
<i>Sylvester</i>	45	72	36	68	79	86	81	91
<i>Wall-E</i>	11	8	51	8	20	70	17	84
<i>Freeman</i>	43	29	55	37	39	49	34	96
<i>singer</i>	56	20	100	23	99	36	92	100
<i>shaking</i>	3	0	27	88	96	5	93	93
<i>PETS2009</i>	21	21	24	24	14	41	34	59
<i>CAVIAR</i>	15	17	30	15	13	14	47	96
<i>box</i>	28	9	11	4	43	90	35	93
<i>board</i>	21	77	9	44	32	13	75	99
<i>NBA</i>	12	15	13	12	7	7	20	57
<i>car</i>	50	9	29	95	96	95	95	96
<i>David</i>	12	2	42	30	40	28	90	92

ACKNOWLEDGEMENTS

This research is supported in part by a Google Faculty Award, the US National Science Foundation (NSF) CAREER Grant #1149783, and NSF IIS Grant #1152576. Most part of this work was performed while Q. Wang was a visiting student at UC Merced.

REFERENCES

- [1] S. Avidan. Support vector tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 184–191, 2001.
- [2] S. Avidan. Ensemble tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 494–501, 2005.
- [3] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 983–990, 2009.
- [4] M. Black and A. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *Proceedings of European Conference on Computer Vision*, pages 329–342, 1996.

- [5] M. J. Black, D. J. Fleet, and Y. Yacoob. A framework for modeling appearance change in image sequences. In *Proceedings of IEEE International Conference on Computer Vision*, pages 660–667, 1998.
- [6] A. Bordes, L. Bottou, and P. Gallinari. Sgd-qn: Careful quasi-newton stochastic gradient descent. *The Journal of Machine Learning Research*, 10:1737–1754, 2009.
- [7] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1631–1643, 2005.
- [8] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.
- [9] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.
- [10] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.
- [11] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *Proceedings of European Conference on Computer Vision*, pages 234–247, 2008.
- [12] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.
- [13] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, 2003.
- [14] Z. Kalal, J. Matas, and K. Mikolajczyk. P-n learning: Bootstrapping binary classifiers by structural constraints. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–56, 2010.
- [15] J. Kwon and K. Lee. Visual tracking decomposition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1269–1276, 2010.
- [16] H. Lee, A. Battle, R. Raina, and A. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, 2007.
- [17] X. Li and W. Hu. Robust visual tracking based on incremental tensor subspace learning. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1–8, 2007.
- [18] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [19] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. Technical Report 7400, INRIA, 2010.
- [20] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proceedings of IEEE International Conference on Computer Vision*, pages 2272–2279, 2009.
- [21] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1436–1443, 2009.
- [22] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Advances in Neural Information Processing Systems*, pages 985–992, 2006.
- [23] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of International Conference on Machine Learning*, pages 759–766, 2007.
- [24] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1-3):125–141, 2008.
- [25] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. Prost: Parallel robust online simple tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 723–730, 2010.
- [26] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 994–1000, 2005.
- [27] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801, 2009.
- [28] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *Acm Computing Surveys*, 38(4):13, 2006.
- [29] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67(2):301–320, 2005.



Qing Wang is a Ph.D. candidate in the Department of Automation, Tsinghua University, China. He received the B.S. degree in automation from Northwestern Polytechnical University, China, in 2007. His current research interests include computer vision and machine learning.



Feng Chen is a professor of Tsinghua University. He received the B.S. and M.S. degrees in automation from Saint-Petersburg Polytechnic University, Russia, in 1994 and 1996, respectively, and the Ph.D. degree in automation from Tsinghua University, China, in 2000. His research interests are probabilistic graphical models and computer vision.



Jimei Yang received a B.S. degree in Electronics and Information Science from the China Agricultural University followed by a M.S. degree in Automation from the University of Science and Technology of China in 2006 and 2009. From 2007 to 2009, he worked at the Institute of Automation, Chinese Academy of Sciences as a research assistant. He is currently pursuing a Ph.D. in Computer Science at the University of California, Merced.



Wenli Xu is a professor of Tsinghua University. He received the B.S. degree in electrical engineering and the M.S. in automatic control engineering from Tsinghua University, in 1970 and 1980, respectively, the Ph.D. degree in electrical and computer engineering from the University of Colorado at Boulder, CO, in 1990. His research interests are mainly in automatic control, intelligent robot and computer vision.



Ming-Hsuan Yang is an assistant professor in Electrical Engineering and Computer Science at University of California, Merced. He received the PhD degree in computer science from the University of Illinois at Urbana-Champaign in 2000. He was a senior research scientist at the Honda Research Institute working on vision problems related to humanoid robots. He coauthored the book *Face Detection and Gesture Recognition for Human-Computer Interaction* (Kluwer Academic 2001) and edited special issue on face recognition for *Computer Vision and Image Understanding* in 2003. He serves as an area chair for the IEEE Conference on Computer Vision and Pattern Recognition in 2008 and 2009, as a publication chair in 2010, and an area chair for the Asian Conference on Computer in 2009 and 2010. He served as an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, and is currently an associate editor of the *Image and Vision Computing*. He is a recipient of the Google faculty award, UC Merced Distinguished Early Career Research Award and NSF CAREER award. He is a senior member of the IEEE and the ACM.