Robust Visual Tracking via Multi-Task Sparse Learning

Tianzhu Zhang[1], Bernard Ghanem[1,2], Si Liu[3], Narendra Ahuja[1,4]
[1] Advanced Digital Sciences Center of Illinois, Singapore
[2] King Abdullah University of Science and Technology, Saudi Arabia
[3] Institute of Automation, Chinese Academy of Sciences, P. R. China
[4] University of Illinois at Urbana-Champaign, Urbana, IL USA
tz.zhang@adsc.com.sg, bernard.ghanem@kaust.edu.sa, sliu@nlpr.ia.ac.cn, n-ahuja@illinois.edu

Abstract

In this paper, we formulate object tracking in a particle filter framework as a multi-task sparse learning problem, which we denote as Multi-Task Tracking (MTT). Since we model particles as linear combinations of dictionary templates that are updated dynamically, learning the representation of each particle is considered a single task in MTT. By employing popular sparsity-inducing $\ell_{p,q}$ mixed norms ($p \in \{2, \infty\}$ and $q = 1$), we regularize the representation problem to enforce joint sparsity and learn the particle representations together. As compared to previous methods that handle particles independently, our results demonstrate that mining the interdependencies between particles improves tracking performance and overall computational complexity. Interestingly, we show that the popular L_1 tracker [15] is a special case of our MTT formulation (denoted as the L_{11} tracker) when $p = q = 1$. The learning problem can be efficiently solved using an Accelerated Proximal Gradient (APG) method that yields a sequence of closed form updates. As such, MTT is computationally attractive. We test our proposed approach on challenging sequences involving heavy occlusion, drastic illumination changes, and large pose variations. Experimental results show that MTT methods consistently outperform state-of-the-art trackers.

1. Introduction

The problem of tracking a target in video arises in many important applications such as automatic surveillance, robotics, human computer interaction, etc. For a visual tracking algorithm to be useful in real-world scenarios, it should be designed to handle and overcome cases where the target's appearance changes from frame-to-frame. Significant and rapid appearance variation due to noise, occlusion, varying viewpoints, background clutter, and illumination and scale changes pose major challenges to any tracker as shown in Figure 1. Over the years, a plethora of tracking methods have been proposed to overcome these challenges. For a survey of these algorithms, we refer the reader to [20].

Figure 1. Frames from a *shaking* sequence. The ground truth track of the head is designated in green. Due to fast motion, occlusion, and changes in illumination, scale, and pose, visual object tracking is a difficult problem.

Recently, sparse representation [5] has been successfully applied to visual tracking [15, 14]. In this case, the tracker represents each target candidate as a sparse linear combination of dictionary templates that can be dynamically updated to maintain an up-to-date target appearance model. This representation has been shown to be robust against partial occlusions, which leads to improved tracking performance. However, sparse coding based trackers perform computationally expensive ℓ_1 minimization at each frame. In a particle filter framework, computational cost grows linearly with the number of particles sampled. It is this computational bottleneck that precludes the use of these trackers in real-time scenarios. Consequently, very recent efforts have been made to speedup this tracking paradigm [13]. More importantly, these methods assume that sparse representations of particles are independent. Ignoring the relationships that ultimately constrain particle representations tends to make the tracker more prone to drift away from the target in cases of significant changes in appearance.

In this paper, we propose a computationally efficient multi-task sparse learning approach for visual tracking in a particle filter framework. Here, learning the representation of each particle is viewed as an individual task. Inspired by the above work, the next target state is selected to be the particle that has the highest similarity with a dictionary of

978-1-4673-1228-8/12/$31.00 ©2012 IEEE

target templates. Unlike previous methods, we exploit similarities between particles and, therefore, seek an accurate, *joint* representation of these particles w.r.t. the dictionary. In our multi-task approach, particle representations are jointly sparse – only a few (but the same) dictionary templates are used to represent all the particles at each frame. As opposed to sparse coding based trackers [15, 14] that handle particles independently, our use of joint sparsity incorporates the benefits of a sparse particle representation (e.g. partial occlusion handling), while respecting the underlying relationship between particles. Therefore, we propose a multi-task formulation (denoted as Multi-Task Tracking or MTT) for the robust object tracking problem. We exploit interdependencies among different particles to obtain their representations jointly. Joint sparsity is imposed on particle representations through an $\ell_{p,q}$ mixed-norm regularizer, which is optimized using an Accelerated Proximal Gradient (APG) method that guarantees fast convergence.

Contributions: The contributions of this work are threefold. **(1)** We propose a multi-task sparse learning method for object tracking, which is a robust sparse coding method that mines correlations among different tasks to obtain better tracking results than learning each task individually. To the best of our knowledge, this is the first work to exploit multi-task learning in object tracking. **(2)** We show that the popular L_1 tracker [15] is a special case of our MTT formulation. **(3)** Since we learn particle representations jointly, we can solve the MTT problem efficiently using an APG method. This makes our tracking method computationally attractive in general and significantly faster than the traditional L_1 tracker in particular.

The paper is organized as follows. In Section 2, we summarize the works most related to our work. The particle filter algorithm is reviewed in Section 3. Section 4 gives a detailed description of the proposed tracking approach, with the optimization details presented in Section 5. Experimental results are reported and analyzed in Section 6.

2. Related Work

There is extensive literature on object tracking. Due to space limitations, we only briefly review nominal tracking methods and those that are the most related to our own. For a more thorough survey of tracking methods, we refer the readers to [20]. Object tracking methods can be categorized as either generative or discriminative. In generative tracking methods, a generative (possibly dynamic) appearance model is used to represent target observations. Here, the tracker searches for a potential target location that is most similar in appearance to the generative model. Popular generative trackers include eigentracker [4], mean shift tracker [7], and incremental tracker [19]. Discriminative trackers formulate the tracking problem as a binary classification problem. In this case, the tracker finds the target location that best separates the target from the background. Popular discriminative methods include on-line boosting [10], ensemble tracking [2], and online MIL tracking [3].

Over the last decade, tracking methods using particle filters (also known as condensation or sequential Monte Carlo models) have demonstrated noteworthy success in visual object tracking [11]. The popularity of these methods stems from their generality, flexibility, and simple implementation. Increasing the number of particles sampled at each frame tends to improve tracking performance, accompanied by a linear increase in computational complexity. As a result, researchers have devised algorithms to improve the computational complexity of this tracking paradigm, e.g. the coarse-to-fine strategy in [8].

Motivated by its popularity in face recognition, sparse coding techniques have recently migrated over to object tracking [15, 13, 14]. Sparse coding based trackers use a sparse linear representation w.r.t. a set of target and occlusion templates to describe particles sampled in each frame. Particle representations are learned by solving a constrained ℓ_1 minimization problem for each particle independently. In [14], dynamic group sparsity is integrated into the tracking problem and very high dimensional image features are used to improve tracking robustness. Recent work has focused on making sparse coding based tracking more efficient by exploiting compressed sensing principles [13].

Multi-task learning [6] has recently received much attention in machine learning, and computer vision. It capitalizes on shared information between related tasks to improve the performance of each individual task, and it has been successfully applied to vision problems such as image classification [21] and image annotation [18]. The underlying assumption behind many multi-task learning algorithms is that the tasks are related. Thus, a key issue lies in how relationships between tasks are incorporated in the learning framework.

Our proposed method is inspired by the above works. To improve computational efficiency and to capitalize on the interdependence between particles (for additional robustness in tracking), we propose a multi-task sparse representation method for robust object tracking.

3. Particle Filter

The particle filter [9] is a Bayesian sequential importance sampling technique for estimating the posterior distribution of state variables characterizing a dynamic system. It provides a convenient framework for estimating and propagating the posterior probability density function of state variables regardless of the underlying distribution through a sequence of prediction and update steps. Let $\vec{\mathbf{s}}_t$ and $\vec{\mathbf{y}}_t$ denote the state variable describing the parameters of an object at time t (e.g. appearance or motion features) and its observation respectively. In the particle filter framework, the posterior $p(\vec{\mathbf{s}}_t|\vec{\mathbf{y}}_{1:t})$ is approximated by a finite set of n samples $\left\{\vec{\mathbf{s}}_t^i\right\}_{i=1}^n$ (called particles) with importance weights w_i. The particle samples $\vec{\mathbf{s}}_t^i$ are drawn from an importance dis-

tribution $q(\vec{\mathbf{s}}_t|\vec{\mathbf{s}}_{1:t-1}, \vec{\mathbf{y}}_{1:t})$, which for simplicity is set to the state transitional probability $p(\vec{\mathbf{s}}_t|\vec{\mathbf{s}}_{t-1})$. In this case, the importance weight of particle i is updated by the observation likelihood as: $w_t^i = w_{t-1}^i p(\vec{\mathbf{y}}_t|\vec{\mathbf{s}}_t^i)$.

Particle filters have been extensively used in object tracking [20]. In this paper, we also employ particle filters to track the target object. Similar to [15], we assume an affine motion model between consecutive frames. Therefore, the state variable $\vec{\mathbf{s}}_t$ consists of the six parameters of the affine transformation (2D linear transformation and translation). By applying an affine transformation using $\vec{\mathbf{s}}_t$ as parameters, we crop the region of interest $\vec{\mathbf{y}}_t$ from the image and normalize it to the same size as the target templates in our dictionary. The state transition distribution $p(\vec{\mathbf{s}}_t|\vec{\mathbf{s}}_{t-1})$ is modeled to be Gaussian, with the the dimensions of $\vec{\mathbf{s}}_t$ assumed independent. The observation model $p(\vec{\mathbf{y}}_t|\vec{\mathbf{s}}_t)$ reflects the similarity between a target candidate (particle) and target templates in the dictionary. In this paper, $p(\vec{\mathbf{y}}_t|\vec{\mathbf{s}}_t)$ is inversely proportional to the reconstruction error obtained by linearly representing $\vec{\mathbf{y}}_t$ using the template dictionary.

4. Multi-Task Tracking (MTT)

In this section, we give a detailed description of our particle filter based tracking method that makes use of multi-task learning to represent particle samples.

4.1. Multi-Task Representation of a Tracking Target

In the multi-task learning (MTL) framework, tasks that share dependencies in features or learning parameters are jointly solved in order to capitalize on their inherent relationships. Many works in this domain have shown that MTL can be applied to classical problems (e.g. image annotation [18] and image classification [21]) and outperform state-of-the-art methods that resort to independent learning. In this paper, we formulate the tracking problem as a MTL problem, where learning the representation of a particle is viewed as a single task. Usually, particle representations in tracking are computed independently (e.g. L_1 tracker [15]). In this paper, we show that by representing particles jointly in an MTL setting, tracking performance and tracking speed can be significantly improved.

In our particle filter based tracking method, particles are randomly sampled around the current state of the tracked object according to a zero-mean Gaussian distribution. At instance t, we consider n particle samples, whose observations (pixel color values) in the t^{th} frame are denoted in matrix form as: $\mathbf{X} = [\vec{\mathbf{x}}_1, \vec{\mathbf{x}}_2, \cdots, \vec{\mathbf{x}}_n]$, where each column is a particle in $\mathbb{R}^d$. In the noiseless case, each particle $\vec{\mathbf{x}}_i$ is represented as a linear combination $\vec{\mathbf{z}}_i$ of templates that form a dictionary $\mathbf{D}_t = \left[\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2, \cdots, \vec{\mathbf{d}}_m\right]$, such that $\mathbf{X} = \mathbf{D}_t\mathbf{Z}$. The dictionary columns comprise the templates that will be used to represent each particle. These templates include visual observations of the tracked object (called target templates) possibly under a variety of appearance changes. Since our representation is constructed at the pixel level, misalignment between dictionary templates and particles might lead to degraded performance. To alleviate this problem, one of two strategies can be employed. **(i)** $\mathbf{D}_t$ can be constructed from an overcomplete sampling of the target object, which includes transformed versions of both. **(ii)** Columns of $\mathbf{X}$ can be aligned to columns of $\mathbf{D}_t$ as in [17]. In this paper, we employ the first strategy, which leads to a larger m but a lower overall computational cost.

We denote $\mathbf{D}_t$ with a subscript because the dictionary templates will be progressively updated to incorporate variations in object appearance due to changes in illumination, viewpoint, etc. Target appearance remains the same only for a certain period of time, but eventually the object templates are no longer an accurate representation of its appearance. A fixed appearance template is prone to the tracking drift problem, since it is insufficient to handle changes in appearance. In this paper, our dictionary update scheme is adopted from the work in [15]. Each target template in $\mathbf{D}_t$ is assigned a weight that is indicative of how representative the template is. In fact, the more a template is used to represent tracking results, the higher is its weight. When $\mathbf{D}_t$ cannot represent some particles well (up to a predefined threshold), the target template with the smallest weight is replaced by the current tracking result. To initialize the m target templates, we sample equal-sized patches at and around the initial position of the target. All templates are normalized.

In many visual tracking scenarios, target objects are often corrupted by noise or partially occluded. As in [15], this noise can be modeled as sparse additive noise that can take on large values anywhere in its support. Therefore, in the presence of noise, we can still represent the particle observations $\mathbf{X}$ as a linear combination of templates, where the dictionary is augmented with trivial (or occlusion) templates $\mathbf{I}_d$ (identity matrix of $\mathbb{R}^{d\times d}$), as shown in Eq (1). The representation error $\mathbf{e}_i$ of particle i using dictionary $\mathbf{D}_t$ is the i^{th} column in $\mathbf{E}$. The nonzero entries of $\vec{\mathbf{e}}_i$ indicate the pixels in $\vec{\mathbf{x}}_i$ that are corrupted or occluded. The nonzero support of $\vec{\mathbf{e}}_i$ can be different from other particles and is unknown a priori.

$$\mathbf{X} = [\mathbf{D}_t \;\; \mathbf{I}_d] \begin{bmatrix} \mathbf{Z} \\ \mathbf{E} \end{bmatrix} \Rightarrow \boxed{\mathbf{X} = \mathbf{B}\mathbf{C}} \tag{1}$$

4.2. Imposing Joint Sparsity via $\ell_{p,q}$ Mixed-Norm

Because particles are densely sampled around the current target state, their representations with respect to $\mathbf{D}_t$ will be sparse (few templates are required to represent them) and similar to each other (the support of particle representations is similar) in general. These two properties culminate in individual particle representations (single tasks) being *jointly sparse*. In other words, joint sparsity will encourage all particle representations to be individually sparse *and* share the same (few) dictionary templates that reliably represent them. This yields a more robust representation for the ensemble of particles. In fact, joint sparsity has been recently

employed to address MTL problems [18, 21]. A common technique to explicitly enforce joint sparsity in MTL is the use of sparsity-inducing norms to regularize the parameters shared among the individual tasks. In this paper, we investigate the use of convex $\ell_{p,q}$ mixed norms (i.e. $p \geq 1$ and $q \geq 1$) to address the problem of MTL in particle filter based tracking (denoted as Multi-Task Tracking or MTT). Therefore, we need to solve the convex optimization problem in Eq (2), where λ is a tradeoff parameter between reliable reconstruction and joint sparsity regularization. Note that we define $\|\mathbf{C}\|_{p,q} = (\sum_{i=1}^{m+d}(\|\mathbf{C}_i\|_p)^q)^{1/q}$, where $\|\mathbf{C}_i\|_p$ is the ℓ_p norm of $\mathbf{C}_i$, the i^{th} row of matrix $\mathbf{C}$.

$$\min_{\mathbf{C}} \|\mathbf{X} - \mathbf{B}\mathbf{C}\|_F^2 + \lambda\|\mathbf{C}\|_{p,q} \tag{2}$$

To encourage a sparse number of dictionary templates to be selected for all particles, we restrict our choice of mixed norms to the case of q=1, where $\|\mathbf{C}\|_{p,1} = \sum_{i=1}^{m+d}\|\mathbf{C}_i\|_p$. Among its convex options, we select three popular and widely studied $\ell_{p,1}$ norms: $p \in \{1, 2, \infty\}$. The solution to Eq (2) for these choices of p and q is described in Section 5. Note that each choice of p yields a different tracker, which we will denote as the L_{p1} tracker. In Figure 2, we present an example of how the L_{21} tracker works. Given all particles $\mathbf{X}$ (sampled around the tracked car) and based on a dictionary $\mathbf{B}$, we learn the representation matrix $\mathbf{C}$ by solving Eq (2). Note that smaller values are darker in color. Clearly, columns of $\mathbf{C}$ are jointly sparse, i.e. a few (but the same) dictionary templates are used to represent all the particles together. Particle $\vec{\mathbf{x}}_i$ is chosen as the current tracking result $\vec{\mathbf{y}}_t$ because its reconstruction error w.r.t. to the target templates is smallest among all particles. Since particles $\vec{\mathbf{x}}_j$ and $\vec{\mathbf{x}}_k$ are misaligned versions of the car, they are not represented well by $\mathbf{D}_t$ (i.e. $\vec{\mathbf{z}}_j$ and $\vec{\mathbf{z}}_k$ have small values). This precludes the tracker from drifting into the background.

Here, we note that $\|\mathbf{C}\|_{1,1} = \sum_{i=1}^{m+d}\|\mathbf{C}_i\|_1 = \sum_{j=1}^{n}\|\vec{\mathbf{c}}_j\|_1$, where $\vec{\mathbf{c}}_i$ denotes the i^{th} column in $\mathbf{C}$. This equivalence property between rows and columns (i.e. the sum of the ℓ_p norms of rows and that of columns are the same) only occurs when p=1. In this case, Eq (2) is equivalent to Eq (3), which is no longer an MTL problem, since the n representation tasks are solved independently. Interestingly, Eq (3) is the same formulation used in the popular $L1$ tracker [15], which can be viewed as a special case of our proposed family of MTT algorithms (specifically the L_{11} tracker). In fact, using the optimization technique in Section 5, our L_{11} implementation leads to a speedup of one order of magnitude over the L_1 tracker.

$$\min_{\vec{\mathbf{c}}_1,\ldots,\vec{\mathbf{c}}_n} \sum_{j=1}^{n}\left(\|\vec{\mathbf{x}}_j - \mathbf{B}\vec{\mathbf{c}}_j\|_2^2 + \lambda\|\vec{\mathbf{c}}_j\|_1\right) \tag{3}$$

5. Solving Eq (2)

To solve Eq (2), we employ the Accelerated Proximal Gradient (APG) method, which has been extensively used to efficiently solve convex optimization problems with non-smooth terms [6]. In our case, the $\ell_{p,q}$ mixed norm is the convex non-smooth term. As compared to traditional projected subgradient methods that have sublinear convergence properties, APG achieves the global solution with quadratic convergence, i.e. it achieves an $\mathcal{O}(\frac{1}{k^2})$ residual from the optimal solution after k iterations [16]. APG iterates between updating the current representation matrix $\mathbf{C}^{(k)}$ and an aggregation matrix $\mathbf{V}^{(k)}$. Each APG iteration consists of two steps: **(1)** a generalized gradient mapping step that updates $\mathbf{C}^{(k)}$ keeping $\mathbf{V}^{(k)}$ fixed, and **(2)** an aggregation step that updates $\mathbf{V}^{(k)}$ by linearly combining $\mathbf{C}^{(k+1)}$ and $\mathbf{C}^{(k)}$.

(1) Gradient Mapping: Given the current estimate $\mathbf{V}^{(k)}$, we obtain $\mathbf{C}^{(k+1)}$ by solving Eq (4), where $\mathbf{H} = \mathbf{V}^{(k)} - \eta\nabla^{(k)} = \mathbf{V}^{(k)} - 2\eta\mathbf{B}^T\left(\mathbf{B}\mathbf{V}^{(k)} - \mathbf{X}\right)$, η is a small step parameter, and $\tilde{\lambda} = \eta\lambda$.

$$\mathbf{C}^{(k+1)} = \arg\min_{\mathbf{Y}} \frac{1}{2}\|\mathbf{Y} - \mathbf{H}\|_2^2 + \tilde{\lambda}\|\mathbf{Y}\|_{p,q} \tag{4}$$

Taking joint sparsity into consideration, we set $q = 1$, which decouples Eq (4) into $(m + d)$ disjoint subproblems (one for each row vector $\mathbf{C}_i$), as shown in Eq (5). Each subproblem is a variant of the projection problem unto the ℓ_p ball. The solution to each subproblem and its time complexity depends on p. In Section 5.1, we provide the solution of this subproblem for popular ℓ_p norms: $p \in \{1, 2, \infty\}$.

$$\mathbf{C}_i^{(k+1)} = \arg\min_{\mathbf{Y}_i} \frac{1}{2}\|\mathbf{Y}_i - \mathbf{H}_i\|_2^2 + \tilde{\lambda}\|\mathbf{Y}_i\|_p \tag{5}$$

(2) Aggregation: We update $\mathbf{V}^{(k)}$ as follows: $\mathbf{V}^{(k+1)} = \mathbf{C}^{(k+1)} + \alpha_{k+1}(\frac{1}{\alpha_k} - 1)\left(\mathbf{C}^{(k+1)} - \mathbf{C}^{(k)}\right)$, where α_k is conventionally set to $\frac{2}{k+3}$. Our overall APG algorithm is summarized in Algorithm 1. Note that convergence is achieved when the relative change in solution or objective function falls below a predefined tolerance.

5.1. Solving Eq (5) for $p \in \{1, 2, \infty\}$

The solution to Eq (5) depends on the value of p. For $p \in \{1, 2, \infty\}$, we show that this solution has a closed form. Note that these solutions can be extended to ℓ_p norms beyond the three that we consider here.

- For $p = 1$: The solution is computed as $\mathbf{C}_i^{(k+1)} = \mathcal{S}_{\tilde{\lambda}}(\mathbf{H}_i)$, where $\mathcal{S}_{\tilde{\lambda}}$ is the soft-thresholding operator defined as $\mathcal{S}_{\tilde{\lambda}}(a) = \text{sign}(a)\max(0, |a| - \tilde{\lambda})$.
- For $p = 2$: Following [6], the solution is computed as $\mathbf{C}_i^{(k+1)} = \max(0, 1 - \frac{\tilde{\lambda}}{\|\mathbf{H}_i\|_2})\mathbf{H}_i$.
- For $p = \infty$: The solution is obtained via a projection onto the ℓ_∞ ball that can be done by a simple sorting procedure.

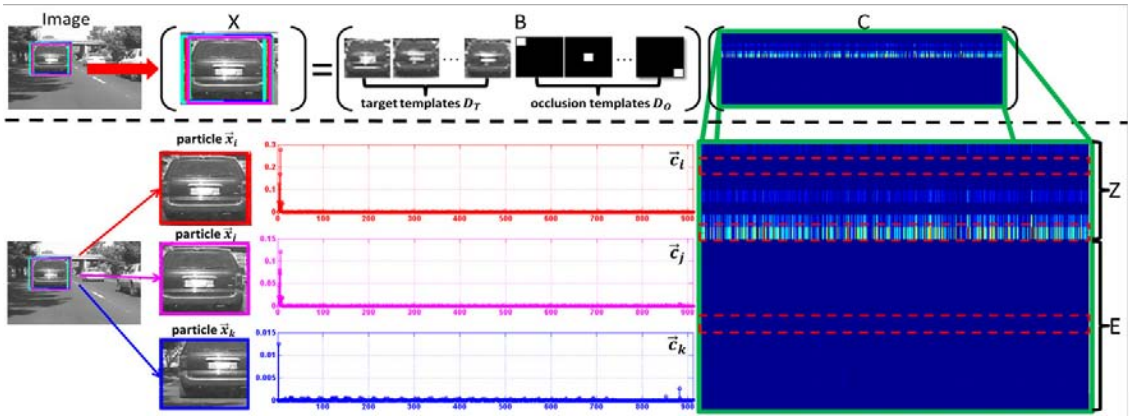

Figure 2. Schematic example of the L_{21} tracker. The representation $\mathbf{C}$ of all particles $\mathbf{X}$ w.r.t. dictionary $\mathbf{B}$ (set of target and occlusion templates) is learned by solving Eq (2) with $p = 2$ and $q = 1$. Notice that the columns of $\mathbf{C}$ are jointly sparse, i.e. a few (but the same) dictionary templates are used to represent all the particles together. The particle $\vec{\mathbf{x}}_i$ is selected among all other particles as the tracking result, since $\vec{\mathbf{x}}_i$ is represented the best by object templates only.

Algorithm 1: *Multi-Task Jointly Sparse Representation*

Input : $\mathbf{X}$, $\mathbf{B}$, p, $q = 1$, $\tilde{\lambda}$, and η
Output: $\mathbf{C}$

1 Initialize $k \leftarrow 0$, $\mathbf{V}^{(k)} = \mathbf{0}$, $\mathbf{C} = \mathbf{0}$, $\alpha_k = 1$
2 **while** *not converged* **do**
3 Compute $\mathbf{H} = \mathbf{V}^{(k)} - 2\eta\mathbf{B}^T\left(\mathbf{B}\mathbf{V}^{(k)} - \mathbf{X}\right)$
4 Solve Eq (5) $\forall i = 1, \ldots, (m+d)$ to obtain $\mathbf{C}^{(k+1)}$
5 $\alpha_{k+1} = \frac{2}{k+3}$
6 $\mathbf{V}^{(k+1)} = \mathbf{C}^{(k+1)} + \alpha_{k+1}(\frac{1}{\alpha_k} - 1)\left(\mathbf{C}^{(k+1)} - \mathbf{C}^{(k)}\right)$
7 $k \leftarrow k + 1$
8 **end**

In this case, the solution is $\mathbf{C}_i^{(k+1)} = \max(0, 1 - \frac{\tilde{\lambda}}{\|\mathbf{H}_i\|_1})\vec{\mathbf{a}}$, where $a_j = \text{sign}(\mathbf{H}_{ij}) \min(|\mathbf{H}_{ij}|, (\sum_{r=1}^{\hat{j}} u_r - \tilde{\lambda})/\hat{j})$, $j = 1, \ldots, n$. The temporary parameters u_r and $\hat{j}$ are obtained as follows. We set $u_j = |\mathbf{C}_{ij}|$ $\forall j$ and sort these values in decreasing order: $u_1 \geq u_2 \geq \ldots \geq u_n$. Then, we set $\hat{j} = \max\{j : \sum_{r=1}^{j}(u_r - u_j) < \tilde{\lambda}\}$.

The computational complexity of each iteration in Algorithm 1 is dominated by the gradient computation in Step 3 and the update of $\mathbf{C}_i^{(k+1)}$ in Step 4. Exploiting the structure of $\mathbf{B}$, the complexity of Step 3 is $\mathcal{O}(mnd)$, while that of Step 4 depends on p. The latter complexity is $\mathcal{O}(n(m+d))$ for $p \in \{1, 2\}$ and $\mathcal{O}(n(m+d)(1+\log n))$ for $p = \infty$. Since $d \gg m$, the per-frame complexity of the L_{11}, L_{21}, and $L_{\infty 1}$ trackers is $\mathcal{O}(mnd\epsilon^{-\frac{1}{2}})$, where the number of iterations is $\mathcal{O}(\epsilon^{-\frac{1}{2}})$. In comparison, the time complexity of the L_1 tracker (equivalent to our L_{11} tracker) is at least $\mathcal{O}\left(nd^2\right)$. In our experiments, we observe that the L_{11} tracker (that uses APG) is two orders of magnitude faster than the L_1 tracker (that solves n Lasso problems independently) in general. For example, when $m = 11$, $n = 200$, and $d = 32 \times 32$, the average per-frame run-time for L_{11} and L_1 are 1.1 and 179 seconds respectively. This is on par with the accelerated "real-time" implementation of the L_1 tracker in [13].

6. Experimental Results

In this section, we present experimental results that validate the effectiveness and efficiency of our MTT method. We also conduct a thorough comparison between MTT and state-of-the-art tracking methods where applicable.

6.1. Datasets and Baselines

To evaluate MTT, we compile a set of 15 challenging tracking sequences (e.g. *car4*, *david indoor*, *OneLeaveShopReenter2cor* (denoted as *onelsr* for simplicity), and *soccer* sequences) that are publicly available online[1]. Due to space constraints, we will only show results on 10 of these sequences, leaving the rest for the **supplementary material**. These videos are recorded in indoor and outdoor environments and include challenging appearance variations due to changes in pose, illumination, scale, and the presence of occlusion. We compare our MTT method ($p \in \{1, 2, \infty\}$) against 6 recent and state-of-the-art visual trackers denoted as: VTD [12], L_1 [15], IVT [19], MIL [3], Frag [1], and OAB [10]. We implemented these trackers using publicly available source codes or binaries provided by the authors. They were initialized using their default parameters.

6.2. Implementation Details

All our experiments are done using MATLAB R2008b on a 2.66GHZ Intel Core2 Duo PC with 6GB RAM. The template size d is set to half the size of the target initialization in the first frame. Usually, d is in the order of several hundreds of pixels. For all experiments, we model $p(\vec{\mathbf{s}}_t|\vec{\mathbf{s}}_{t-1}) \sim \mathcal{N}(\vec{\mathbf{0}}, \text{diag}(\vec{\sigma}))$, where $\vec{\sigma} = [0.005, 0.0005, 0.0005, 0.005, 4, 4]^T$. We set the number of particles $n = 400$, the total number of target templates $m = 11$ (same as L_1 tracker [15]), and the number of oc-

[1] vision.ucsd.edu/∼bbabenko/project_miltrack.shtml; www.cs.toronto.edu/ dross/ivt/; cv.snu.ac.kr/research/∼vtd/;

clusion templates to d. In Algorithm 1, we set $\eta = 0.01$ and $\tilde{\lambda}$ (by cross-validation) to $\{0.01, 0.005, 0.2\}$ for L_{21}, L_{11} and $L_{\infty 1}$ respectively. Each tracker uses the same parameters for all video sequences. In all cases, the initial position of the target is selected manually. In Sections 6.3 and 6.4, we give a qualitative and quantitative analysis of the MTT method, and compare it against the 6 baseline methods. Our experiments show that MTT produces more robust and accurate tracks, which are all made available in our website.

6.3. Qualitative Comparison

The *car4* sequence was captured in an open road scenario. Tracking results at frames {20, 186, 235, 305, 466, 641} for all 9 methods are shown in Figure 3(a). The different tracking methods are color-coded. OAB, Frag, and VTD start to drift from the target at frame 186, while MIL starts to show some target drifting at frame 200 and finally loses the target at frame 300. IVT and L_1 track the target quite well. The target is successfully tracked throughout the entire sequence by our L_{11}, L_{21}, and $L_{\infty 1}$ methods.

In the *car11* sequence, a car is driven into a very dark environment, while being videotaped from another moving car. Tracking results for frames $\{10, 110, 200, 250, 309, 393\}$ are presented in Figure 3(b). Frag starts to drift around frame 60. Due to changes in lighting, MIL starts to undergo target drift from frame 120. OAB and L_1 methods start to fail in frame 284. IVT and VTD can track the target through the whole video sequence; however, these tracks are not as robust or accurate as the proposed L_{21} and $L_{\infty 1}$ trackers.

The *coke11* sequence contains frequent occlusions and fast motion, which cause motion blur. The MTT trackers, L_1, OAB, and MIL can track the target almost throughout the entire sequence. The other trackers fails due to pose change and occlusion as shown in Figure 3(c).

In the *david* sequence, a moving face is tracked. The tracking results at frames {354, 423, 465, 502, 588, 760} are shown in Figure 3(d). Frag and VTD fail around frames 423 and 465 respectively. OAB starts to drift at frame 550. MIL and L_1 adequately track the face, but experience target drift, especially at frames 690 and 500, respectively. The IVT and MTT methods track the moving face accurately.

Figure 3(e) shows tracking results for the *girl* sequence. Performance on this sequence exemplifies the robustness of MTT to occlusion (complete occlusion of the girl's face as she swivels in the chair) and large pose change (the face undergoes significant 3D rotation). MTT and L_1 are capable of tracking the target during the entire sequence. Other trackers experience drift at different instances: Frag at frame 248, OAB and IVT at frame 436, and VTD at frame 477.

In the *shaking* sequence, the tracked object is subject to changes in illumination and pose. While the stage lighting condition is drastically changed, and the pose of the object is severely varied due to head shaking, our method successfully tracks the object (refer to Figure 3(f)). Compared with L_{11} and L_1, L_{21} and $L_{\infty 1}$ perform better because their joint particle representation is more robust to rapid changes. Other methods (OAB, IVT, L_1, and Frag) fail to track the object when these changes occur. VTD and MIL methods can track the object quite well except for some errors around frame 60.

In the *onelsr* sequence, the background color is similar to the color of the woman's trousers, and the man's shirt and pants have a similar color to the woman's coat. In addition, the woman undergoes partial occlusion. Some results are shown in Figure 3(g). While tracking the woman, IVT, MIL, Frag, OAB, and VTD start tracking the man when the woman is partially occluded around frame 200, and are unable to recover from this failure after that. The L_1 tracker tracks the woman quite well. Compared with other trackers, our L_{21} and $L_{\infty 1}$ trackers are more robust to the occlusion.

Results on the *soccer* sequence are shown in Figure 3(h). They demonstrate how our proposed method outperforms most of the state-of-the-art trackers when the target is severely occluded by other objects. The L_{21} and $L11$ methods accurately track the player's face despite scale and pose changes as well as occlusion/noise from the confetti raining around him. Other methods (IVT, L_1, $L_{\infty 1}$, OAB, MIL, and Frag) fail to track the object reliably. The VTD tracker can track the target in this sequence quite well.

Results on the *sylv* sequence are shown in Figure 3(i). In this sequence, a stuffed animal is being moved around, thus, leading to challenging pose, lighting, and scale changes. IVT fails around frame 613 as a result of a combination of pose and illumination change. The rest of the trackers track the target throughout the sequence, with Frag, MIL, VTD, OAB, L_{11} and L_1 veering off the target at certain instances.

The *trellis70* sequence is captured in an outdoor environment where lighting conditions change drastically. The video is acquired when a person walks underneath a trellis covered by vines. As shown in Figure 3(j), the cast shadow changes the appearance of the target face significantly. Furthermore, the combined effects of pose and lighting variations along with a low frame rate make visual tracking extremely difficult. Nevertheless, the L_{21} and $L_{\infty 1}$ trackers can follow the target accurately and robustly, while the other tracking methods perform below par in this case. TD and Frag fail around frame 185. L_1 starts drifting at frame 287, while MIL and OAB fail at frame 323. IVT starts drifting at frame 330.

6.4. Quantitative Comparison

To give a quantitative comparison between the 9 methods, we manually label the ground truth for 10 sequences. Tracker performance is evaluated according to the average per-frame distance (in pixels) between the center of the tracking result and that of ground truth. Clearly, this distance should be small. In Figure 4, we plot the distance of each tracker over time for 4 sample sequences. We see that MTT trackers consistently produce a smaller distance than other trackers in general. This implies that MTT can accurately track the target despite severe occlusions, pose variations, illumination changes, and abrupt motions.

Figure 3. Results of 9 tracking methods. Frame numbers are overlayed in red.

In Figure 5, we plot the average center distance for each tracker over the 10 sequences. It is clear that the MTT methods are consistently better than the other trackers in most sequences even though there are severe occlusions, pose variations, illumination changes, and abrupt motions. Among the MTT methods, L_{21} outperforms L_{11} and $L_{\infty 1}$ in general. In fact, except for the *girl*, *shaking* and *soccer* sequences, in which we obtain similar results as IVT and VTD, the L_{21} tracker does outperform the other methods. Frag and L_1 performs well under partial occlusion but tends to fail under severe illumination and pose changes. The IVT tracker is hardly affected by changes in appearance except those due to illumination. OAB is effected by background clutter, and it is easily drifts from the target. MIL performs well except when severe illumination changes force the tracker to drift into the background. VTD tends to be robust against illumination change, but it cannot handle severe occlusions and viewpoint changes adequately.

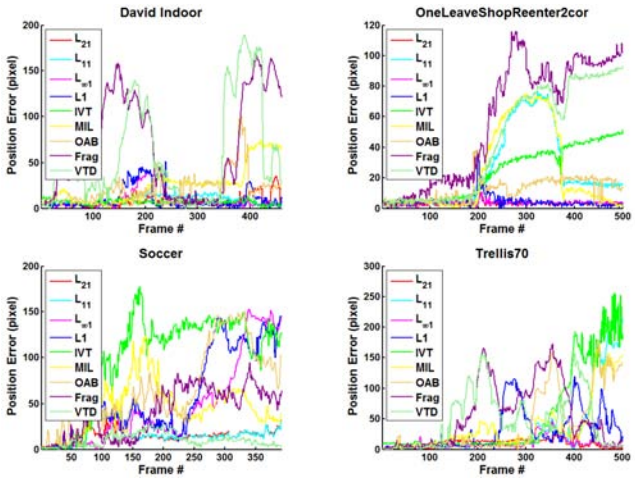

Figure 4. Center distance (in pixels) between tracking result and ground truth over time for 9 trackers applied to 4 video sequences.

Now, we compare the performance of the four trackers L_{21}, $L_{\infty 1}$, L_{11} and L_1 [15]. Based on the results in Figure 5, L_{21} and $L_{\infty 1}$ outperform L_{11} and L_1. That is because L_1 and L_{11} trackers represent particles independently, while L_{21} and $L_{\infty 1}$ capitalize on the dependencies among different particles to obtain a more robust joint representation. Our results demonstrate that it is useful for visual tracking to mine particle relationships. Moreover, in theory, the L_1 tracker is a special case of our MTT framework (refer to Eq (3)), and it should produce the same results as L_{11}. However, this is not reflected in our empirical results due to three reasons. **(a)** The L_1 tracker is forced to adopt a smaller template size ($d = 12 \times 15$) due to its high computational cost $\mathcal{O}(nd^2)$. A larger d leads to a richer representation and improved tracking performance. As mentioned earlier, MTT methods set d to half the size of the initial bounding box, which is generally more than 600 pixels. **(b)** In the public MATLAB implementation of L_1, the dictionary weights are used not only to update the target templates but also to multiply the templates themselves, which leads to an artificially sparser representation. For L_{11}, the weights are only used to update the target templates. In addition, MTT uses a more efficient solver (refer to Section 5.1) to learn particle representations, so L_{11} *can* reach a better solution than L_1 for the same stopping criterion at every frame. **(c)** Since the L_1 and L_{11} trackers both adopt the particle filter framework, their tracking results for the same sequence can be different due to random sampling of particles in the state space.

7. Conclusion

In this paper, we formulate particle filter based tracking as a multi-task sparse learning problem, where the representations of particles, regularized by a sparsity-inducing $\ell_{p,1}$ mixed norm, are learned *jointly* using an efficient Accelerated Proximal Gradient (APG) method. We show that the popular L_1 tracker [15] is a special case of our proposed formulation. Also, we extensively analyze the performance of our tracker on challenging real-world video sequences and show it outperforming 6 state-of-the-art tracking methods.

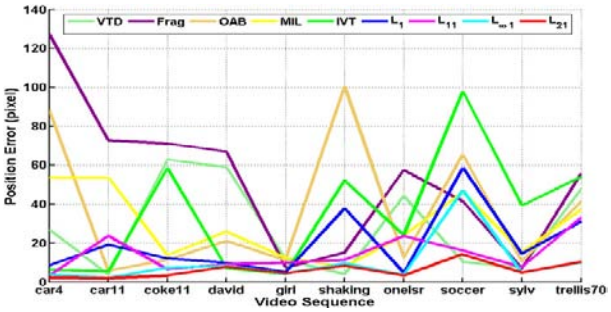

Figure 5. Average distance of 9 trackers applied to 10 sequences

Acknowledgment

This study is supported by the research grant for the Human Sixth Sense Programme at the Advanced Digital Sciences Center from Singapore's Agency for Science, Technology and Research (A*STAR).

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, pages 798–805, 2006. 5

[2] S. Avidan. Ensemble tracking. In *CVPR*, pages 494–501, 2005. 2

[3] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, 2009. 2, 5

[4] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *IJCV*, pages 63–84, 1998. 2

[5] E. J. Candès, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, Aug. 2006. 1

[6] X. Chen, W. Pan, J. Kwok, and J. Carbonell. Accelerated gradient method for multi-task sparse learning problem. In *ICDM*, 2009. 2, 4

[7] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-Based Object Tracking. *TPAMI*, 25(5):564–575, Apr. 2003. 2

[8] C.Yang, R. Duraiswami, and L. Davis. Fast multiple object tracking via a hierarchical particle filter. In *ICCV*, 2005. 2

[9] A. Doucet, N. De Freitas, and N. Gordon. Sequential monte carlo methods in practice. In *Springer-Verlag*. New York, 2001. 2

[10] H. Grabner, M. Grabner, and H. Bischof. Real-Time Tracking via On-line Boosting. In *BMVC*, 2006. 2, 5

[11] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998. 2

[12] J. Kwon and K. M. Lee. Visual tracking decomposition. In *CVPR*, 2010. 5

[13] H. Li, C. Shen, and Q. Shi. Real-time visual tracking with compressed sensing. In *CVPR*, 2011. 1, 2, 5

[14] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski. Robust and fast collaborative tracking with two stage sparse optimization. In *ECCV*, 2010. 1, 2

[15] X. Mei and H. Ling. Robust Visual Tracking and Vehicle Classification via Sparse Representation. *TPAMI*, 33(11):2259–2272, 2011. 1, 2, 3, 4, 5, 8

[16] Y. Nesterov. Gradient methods for minimizing composite objective function. In *CORE Discussion Paper*, 2007. 4

[17] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust Alignment by Sparse and Low-rank Decomposition for Linearly Correlated Images. *TPAMI (to appear)*, 2011. 3

[18] A. Quattoni, X. Carreras, M. Collins, and T. Darrell. An efficient projection for l 1,infinity regularization. In *ICML*, pages 857–864, 2009. 2, 3, 4

[19] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental Learning for Robust Visual Tracking. *IJCV*, 77(1):125–141, 2008. 2, 5

[20] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13+, Dec. 2006. 1, 2, 3

[21] X. Yuan and S. Yan. Visual classification with multi-task joint sparse representation. In *CVPR*, pages 3493–3500, 2010. 2, 3, 4