Robust Online Visual Tracking

Zhibin Hong

Faculty of Engineering and Information Technology

University of Technology, Sydney

A thesis submitted for the degree of

Doctor of Philosophy

2015

Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Student: Zhibin Hong
Date: 22/09/2015

I would like to dedicate this thesis to my loving parents
Liduan Lin and *Cheng Hong*

Acknowledgements

I would like to take this good opportunity to appreciate my advisors, several professors, my colleagues, my friends and my family for their significant help during my doctoral study in University of Technology (UTS), Sydney, Australia.

First of all, I would like to express my sincere appreciation and deep gratitude to my advisor supervisor **Prof. Dacheng Tao**. He is the one who led me to the field of academic research. He always gives me plenty of freedom to explore and timely constructive suggestions to help me out of difficulties. I can always benefit and learn a lot from various detailed discussions with him. It is hard to imagine I could have finished this thesis without his high scientific standards, unlimited patience, generous support, constant encouragement and guidance. I also wish to express my sincere appreciation to **Prof. Lianwen Jin** who was my advisor when I was in South China University of Technology for master study. I am so grateful for his generosity, guidance and support. He is the one so important to my career and life track. I would not have come to the field of computer vision and would not have come to UTS and met Prof. Dacheng Tao without him. I am also deeply indebted to **Dr. Xue Mei** who led me to the field of visual tracking. His expertise, guidance and encouragement significantly helped me for the completion of this thesis and kept me away from many detours during the journey of exploration. I sincerely appreciate him for his valuable time for beneficial discussions, constructive suggestions and timely support. I would also like to express my appreciation to **Dr. Chaohui Wang** as a close mentor, collaborator and elder brother, for his kindly support and timely help. In addition, I appreciate Prof. Dacheng Tao, Prof. Lianwen Jin, Dr. Xue Mei and Dr. Chaohui Wang for their guidance, suggestions and help to my future career in the final stage of PhD study.

I have been fortunate to work in UTS and Centre for Quantum Computation and Intelligent Systems (QCIS). I wish to express my appreciation to Prof. Chengqi Zhang, for his full support for me to attend those top conferences including ECCV, ICCV and CVPR, and for his amazing work of gathering so many brilliant researchers in QCIS. I am grateful to Prof. Xingquan Zhu, Prof. Massimo Piccardi, Prof. Maolin Huang, Dr. Danil Prokhorov, Dr. Lin Chen, Dr. Jun Li for their help during my doctoral study. Moreover, I also want to give special thanks to my excellent collaborators: Jia Wu, Zhe Xu, Zhe Chen, for their brilliant work and timely support, and to my dear colleagues and friends I met in QCIS: Prof. Weifeng Liu, Dr. Wei Bian, Dr. Tianyi Zhou, Dr. Naiyang Guan, Dr. Nannan Wang, Dr. Meng Fang, Dr. Mingsong Mao, Dr. Hongshu Chen, Dr. Shirui Pan, Mingming Gong, Tongliang Liu, Changxing Ding, Maoying Qiao, Dr. Guodong Long, Dr. Jing Jiang, Barbara Munday, Chunyang Liu, Dr Shengzheng Wang, Dr. Yong Luo, Lianyang Ma, Xiaoyan Li, Fei Gao, Jie Gui. Bozhong Liu, Prof. Bo Du, Dianshuang Wu, Dr. Ting Guo, Dr. Lianhua Chi, Kailing Guo, Ruxin Wang, Qiang Li, Weilong Hou, Shaoli Huang, Chang Xu, Chen Gong, Sujuan Hou, Haishuang Wang, Qin Zhang, Yali Du, Zhongwen Xu, Dr. Yi Yang, Hao Xiong, Prof. Shigang Liu, Prof. Xianhua Ben, Prof. Wankou Yang, Xiyu Yu, Jiang Bian, Guoliang Kang, Liu Liu, for the inspiring discussions, kind support and companionship.

I am also grateful to all the other friends: Zhida Guo, Qihao Chen, Xueshen Xian, Shixin Lin, Hepeng Ling, Yingying Li, Ming Chen, Sheng Dai, YingLi Li, Ruifang Chen, Shi Chen, Jiawei Xuan, Yanhua Zhang, Shanlin Ye, Xueying Chen, Tao Wang, Weiqi Tang, Chunyu Wang, Ang Li, Qihan Zhao, Danyan Lei, for their support and company during both joyful and stressful times.

Finally, I would like to express my deeply felt gratitude to my family: my parents, my grandparents, my lovely brother, my uncles and aunties, my cousins, for their endless love, encouragement and full support throughout my study and life.

Abstract

Visual tracking plays a key role in many computer vision systems. In this thesis, we study online visual object tracking and try to tackle challenges that present in practical tracking scenarios. Motivated by different challenges, several robust online visual trackers have been developed by taking advantage of advanced techniques from machine learning and computer vision.

In particular, we propose a robust distracter-resistant tracking approach by learning a discriminative metric to handle distracter problem. The proposed metric is elaborately designed for the tracking problem by forming a margin objective function which systematically includes distance margin maximization, reconstruction error constraint, and similarity propagation techniques. The distance metric obtained helps to preserve the most discriminative information to separate the target from distracters while ensuring the stability of the optimal metric.

To handle background clutter problem and achieve better tracking performance, we develop a tracker using an approximate Least Absolute Deviation (LAD)-based multi-task multi-view sparse learning method to enjoy robustness of LAD and take advantage of multiple types of visual features. The proposed method is integrated in a particle filter framework where learning the sparse representation for each view of a single particle is regarded as an individual task. The underlying relationship between tasks across different views and different particles is jointly exploited in a unified robust multi-task formulation based on LAD. In addition, to capture the frequently emerging outlier tasks, we decompose the representation matrix to two collaborative components which enable a more robust and accurate approximation.

In addition, a hierarchical appearance representation model is proposed for non-rigid object tracking, based on a graphical model that exploits shared information across multiple quantization levels. The tracker aims to find the most possible position of the target by jointly classifying the pixels and superpixels and obtaining the best configuration across all levels. The motion of the bounding box is taken into consideration, while Online Random Forests are used to provide pixel- and superpixel-level quantizations and progressively updated on-the-fly.

Finally, inspired by the well-known Atkinson-Shiffrin Memory Model, we propose MUlti-Store Tracker, a dual-component approach consisting of short- and long-term memory stores to process target appearance memories. A powerful and efficient Integrated Correlation Filter is employed in the short-term store for short-term tracking. The integrated long-term component, which is based on keypoint matching-tracking and RANSAC estimation, can interact with the long-term memory and provide additional information for output control.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

With the exponential growth in computing power of modern computers and the significant cost-reduction of high quality camera devices, the development of intelligent computer vision applications, such as robotics, video surveillance, automatic control, vehicle navigation, and human computer interaction (HCI), has aroused increasing interest in both academic and industry. In particular, a lot of intelligent visual systems are designed based on object analysis. For example, in a video surveillance system, it is often necessary to analyze the trajectories, behaviors and identities of pedestrians in order to understand what is happening in a controlled area. In general, these systems consist of three components, which are the detector for finding the objects of interest, the tracker for recovering trajectories of detected objects in consecutive frames, the high-level model for identity recognition, behavior recognition, or system control [186]. Visual tracking plays a key role in all these systems, since the performance of them depends heavily on the information provided by a robust visual object tracker.

In a computer vision system, object detection, which is also called object-class detection, is used to discover objects of interest in a given video source. A lot of research aims at developing accurate detectors for specific objects, such as face [191], pedestrian [46] and vehicle [162]. On the other hand, some works focus on the detection of generic object [55, 164] and make no restriction on

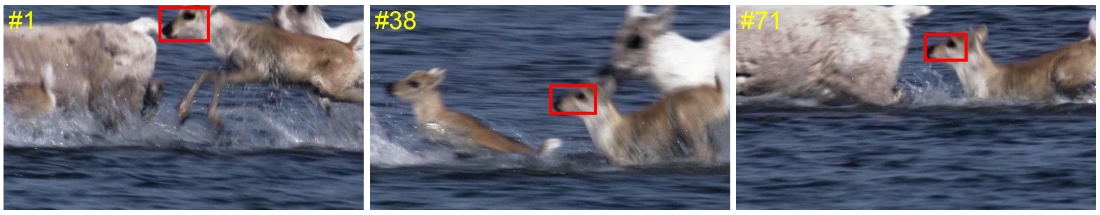


Figure 1.1: An example of online visual object tracking discussed in this thesis. Given a bounding box of an object in the first frame of a video, the task of a tracker is to locate the target in subsequent video frames.

the object to detect. Although some knowledge-based methods (e.g. based on symmetry or color) [33, 128] and template-based methods [17] are proposed for specific object-class detection, most of modern detectors [46, 164] make use of a large number of training samples and adopt varied training methods to obtain the model parameters off-line.

Given the detections found by an employed detector, it is also critical for the system to recover the trajectories of respective objects (i.e. tracking) in order to conduct more meaningful analysis and give appropriate response. Practically, it is not appropriate to simply combine near detections for tracking since the outputs of detectors are usually unreliable and sparse. Therefore, tracking is generally done by a certain tracker (or trackers). In particular, some tracking approaches are constructed based on associating the data of detections and global trajectory optimization over a temporal window [18, 80]. On the other hand, a large group of works [8, 68, 131], namely online object tracking, assume that the ground-truth location of a target is already given in a particular frame of video. For example, the ground-truth can be obtained by a detector or manual setting. Therefore, the goal of online tracking remains to estimate the locations of the target in the subsequent frames.

In this thesis, we mainly study the online visual object tracking, namely model-free tracking. In particular, we focus on generic object tracking and assume that there is no available prior knowledge about the target. Also, we study only Single Object Tracking (SOT), which can be regarded as a special case of Multiple Object Tracking (MOT) [19]. Given an annotation of the target (i.e. bounding box in this thesis) in the first frame, the task of a tracker is to estimate

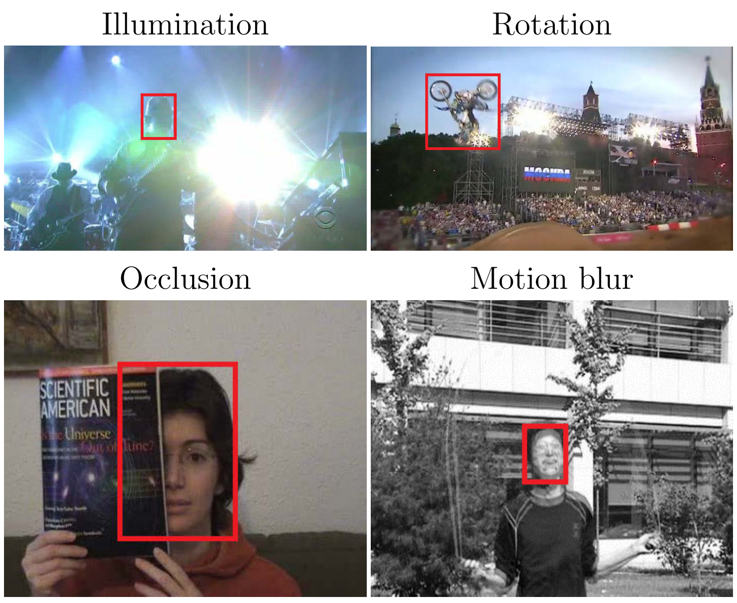


Figure 1.2: Some examples of challenges in online visual object tracking.

the target locations using the same annotation in subsequent video frames. An example of online visual object tracking studied in this thesis is shown in Fig. 1.1. It should be noted that online visual trackers do not employ off-line training samples, which is different from the ones in [18,80]. Therefore, in order to handle the appearance changes of objects, all parameter updates are performed on-the-fly by using the samples selected by the tracker itself, which can be regarded as a self-learning problem [200].

Online visual tracking has been studied for several decades. Many trackers have been proposed [181, 185] and show promising results. However, designing a universally effective tracker remains extremely difficult due to the presence of various challenges and the difficulty to well balance them simultaneously. These challenges are frequently referred to:

- Robustness to occlusion [96]
- Robustness to scale changes [42]
- Robustness to illumination variation [56]
- Robustness to motion blur [182]
- Robustness to rotation

- Robustness to noises
- Tracking objects Leaving the Field of View (FOV) [88]
- Tracking non-rigid objects [60]
- Background clutter
- The presence of distracters [45]
- Update problem (drift) [129]

1.2 Literature survey of Online Visual Tracking

There are extensive papers related to tracking as well as some comprehensive reviews that can be found in the literature [34, 115, 160, 185, 186]. However, the field has grown at an amazing speed over the past decades, hence making a completed review throughout all tracking methods intractable. In this section, we instead focus on the review of some representative works and the existing works that are most related to this thesis. In particular, we summarize existing work according to their mechanisms in appearance modeling.

Typically, an online visual object tracker consists of three important components [115], which are: motion model, searching model, and appearance model. In general, a motion model uses a series of measurements observed over time, containing noise, and performs the dynamic state estimation during tracking. It is usually constructed by employing some stochastic filters such as Kalman filters [89], or Particle Filters (PFs) [48, 82], while a searching model is used to provide the output, which is usually completed by maximizing a posterior estimation based on prediction of motion model [131] or exhausting search in a local region [8].

Given specific motion model and searching model, the performance of a tracker highly depends on the construction of appearance model. Therefore, a large amount of recent works focus on developing a robust appearance model that is able to differentiate the target from background and adapts appearance changes over time. Specifically, existing methods can be generally categorized into two groups, generative and discriminative methods.

1.2.1 Generative Methods

Generative methods, such as subspace learning-based methods [75,150] and sparse representation-based methods [131], aim to build an appearance model to represent the tracking target. In this framework, tracking can be considered as searching a candidate with the most similarity to the target. The appearance models can be learned or updated to capture the appearance changes in consecutive frames. In practice, generative trackers often have better descriptive power and demand a small training set; however, since these methods focus on only the target domain and never consider the impact of background, they are usually hard to survive in a cluttered scene, especially with the presence of distracters.

1.2.1.1 Subspace Learning-based Methods

In general, subspace learning methods treat the target as a whole "thing" instead of a set of independent features [150]. In the process of subspace learning, the relevant covariance of all features are learned and thus embedded into a compact subspace. The likelihood of a candidate being the target is measured as the probability that the candidate is generated from the learned subspace. It is related to the distance from the candidate to the learned subspace, i.e. the candidate with a smaller distance to the subspace is more likely to be the target.

Black and Jepson [23] proposed an eigentracker that used an eigen subspace learned by well-known Principal Component Analysis (PCA) to track a rigid object and get promising performance. However, the limitation of their framework is obvious that the subspace needs to be pre-trained offline, thus the various views of the target should be collected and fed to the training phase in advance, which makes it hard to track unseen objects and unable to update itself adaptively.

To overcome the adaptability problem, Ross *et al.* [150] introduced incremental PCA [29] to learn low-dimensional subspace representation of the target online. In their framework, the tracking results are successively collected and used to update the eigen subspace in an incremental manner, which enable it to utilize the information available online and significantly improve the computational efficiency. For the observation model, the likelihood of a candidate generated from the learned subspace is jointly governed by two Gaussian distributions related to the specific distances, which are respectively the distance from the candidate to the subspace and the distance from the projected sample to the subspace center.

Traditional PCA methods treat images as vectors, which may neglect the spatial information within each image matrix and the local spatial relationships [174]. Therefore, Li *et al.* [116] employed high-order tensor analysis to treat images as matrices and developed a tracking approach based on Incremental Tensor Subspace Analysis (ITSA). It is well known that tensor subspace analysis can further reduce the spatial-temporal redundancies and therefore obtain more compact low-order representations. Moreover, Gai and Stevenson [57] argued that the Probabilistic Principal Component Analysis (PPCA) adopted in [150] is based on classic Gaussian density, which has light tails and is sensitive to outliers. To resolve this problem, they developed a visual tracking system based on a full Bayesian Student's t-distribution PCA. The shape of the observation distribution can be adaptively adjusted by updating the set of auxiliary latent variables and thus gets heavier tails to resist outlier noise.

By contrast to the methods [116, 150] that use the image-as-vector or image-as-matrix representations, Li *et al.* [117] and Wu *et al.* [180] turned to represent object appearance as covariance matrices that are constructed by image features. Given an image region R, the corresponding covariance matrix can be represented as

$$C_R = \frac{1}{N-1}\sum_{i=1}^{N}(\boldsymbol{f}_i - \boldsymbol{\mu}_N)(\boldsymbol{f}_i - \boldsymbol{\mu}_N)^T \tag{1.1}$$

where $\{\boldsymbol{f}_i\}_{i=1,\ldots,N}$ denote the d-dimensional feature points obtained by a feature extraction function, $\boldsymbol{\mu}_N$ is the statistic mean of $\boldsymbol{f}_i$, and N is the number of pixels within R. In Riemannian geometry, the distance between two points corresponding to two covariance matrices can be more easily computed using the Log-Euclidean Riemannian Metric compared to the original Riemannian Metric. Based on this covariance matrix descriptor of image, Li *et al.* [117] developed an incremental algorithm to learn the Log-Euclidean Riemannian subspace, which was similar to the incremental PCA approach presented in (Ross *et al.* 2008). By contrast, Wu *et al.* [180] suggested an incremental method to update the covariance matrices instead and gave the samples different weights according to their importances varying over time.

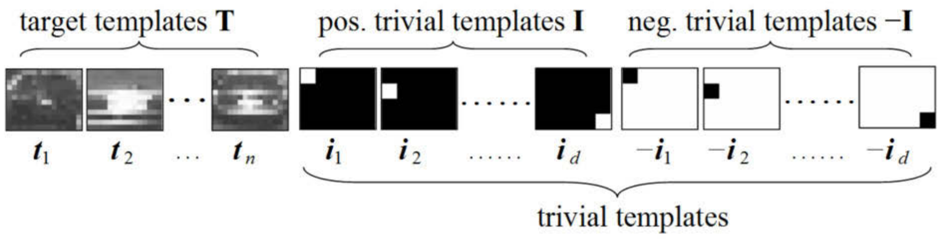


Figure 1.3: An example of target templates and trivial templates used in L1 tracker [131]. Original figure is from (Mei & Ling 2009) [131].

1.2.1.2 Sparse Representation-based Methods

Sparse representation [47], which is also referred to as Compressive Sensing (CS), has recently aroused considerable attention in the computer vision community. One of the most representative works was presented in [179], which successfully employed sparse representation to tackle the face recognition problem. Based on the compressive sensing theory, if a testing face belongs to a specific image class (a specific person) from the training images, then this testing face can be sparsely represented by a linear combination of training face images from an over-complete dictionary; meanwhile the coefficients should concentrate on the images from the same class as the testing image, while other coefficients of the images from different classes should tend to be sparse and zeros. Consequently, the proposed framework can recognize the identity of the testing face image based on prior knowledge of the training dictionary. Comparing with previous face recognition algorithms, sparse representation demonstrates superior performance as well as simplicity in feature extraction [179].

Inspired by the success in face recognition, Mei and Ling [131] first introduced the sparse representation framework to the tracking domain and sparsely represented the tracking target by a linear combination of target templates and trivial templates, as

$$\boldsymbol{y} = \mathbf{T}\boldsymbol{c} + [\mathbf{I}, -\mathbf{I}]\boldsymbol{e} \quad , \quad \text{s.t.} \quad \boldsymbol{w} \succcurlyeq 0 \quad , \tag{1.2}$$

where $\mathbf{T} \in R^{d\times n}$ are n columns of target templates, $\mathbf{I} \in R^{d\times d}$ and $-\mathbf{I}$ are trivial templates referred to as a set of unit vectors (both positive and negative), $\boldsymbol{c}$

represents the sparse coefficients corresponding to the target templates, $\boldsymbol{e}$ is the coefficients for trivial templates, and and $\succcurlyeq$ is an element-wise operator, which constrains each element at the left side to be greater or equal to the one at the right side. Fig. 1.3 visualizes a set of example target templates and trivial templates.

In order to obtain a sparse solution of $\boldsymbol{c}$, Mei and Ling [131] proposed to solve the L1-regularized least squares problem

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \|\mathbf{M}\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{w}\|_1 \quad , \quad \text{s.t.} \quad \boldsymbol{w} \succcurlyeq 0 \quad , \tag{1.3}$$

where $\mathbf{M} = [\mathbf{T}, \mathbf{I}, -\mathbf{I}]$ is an over-complete dictionary, $\boldsymbol{w} = [\boldsymbol{c}^T, \boldsymbol{e}^T]^T$ is the obtained sparse representation, $\boldsymbol{y} \in R^d$ is a given candidate of target, λ is the parameter to control the sparsity of $\boldsymbol{w}$. The observation likelihood of a candidate is measured by the loss function based on the reconstruction error, i.e. Euclidean distance between the derived combination of target templates and the candidate patch. For the motion model, they employ a particle framework to guide the tracking process and iteratively generate candidate patches according to observation likelihoods. A template update strategy is also proposed, which enables the tracker to handle slight appearance variation and pose changes. Comparing to previous trackers, L1 tracker experimentally shows promising performance, especially in some scenarios including occlusion and corruption due to the employment of trivial templates that model noise and occluded pixels.

Although L1 tracker [131] shows its robustness in some scenarios, solving L1 minimization (1.3) is computationally complex, and the computation time grows linearly along with the increase of particle sample number, which make L1 tracker fail to achieve real-time performance. Therefore, some researches focus on improving the computation efficiency of L1 framework, such as [113], [133] and [13]. In [133], Mei *et al.* exploited the reconstruction error bound of L2 norm and utilize it to develop a new resampling approach named Bounded Particle Resampling (BPR), which effectively excludes some unpromising particle samples, decreases the amount of samples fed to L1 minimization and results in accelerating the tracking speed as well. An occlusion detection method, which tries to utilize the information given by trivial coefficients (the coefficients of trivial templates obtained by L1 minimization), has also been proposed to improve the template update strategy, based on the fact that occlusion would result in partial density in trivial coefficients. In contrast, Li *et al.* [113] focused on decreasing the dimension of image representation (image features) by transforming original features to a low dimension compressive feature space through a random generated orthogonal matrix, which is based on the superior signal recovery ability of CS. Meanwhile, a customized Orthogonal Matching Pursuit (OMP) algorithm is also adopted to accelerate the computation of L1 minimization in their framework. As a consequence, their proposed tracker demonstrates a real-time performance as well as an acceptable tracking ability. Subsequently, Bao *et al.* [13] proposed a real-time L1 tracker by introducing the accelerated proximal gradient approach to solve the L1 minimization problem with guaranteed quadratic convergence.

Instead of improving the computation efficiency of L1 tracker, some work aims to improve the robustness. In [195], a multi-task learning [37] approach is applied to tracking by learning a joint sparse representation of all the particles in a particle filter framework. Compared to the original L1 tracker [131] that pursues the sparse representation independently, Multi-Task Tracking (MTT) achieves more robust performance by exploiting the interdependency between particles. In addition, [196] also tries to exploit the interdependency between particles and cast the tracking problem as a low-rank matrix learning problem, where a better performance over L1 tracker is obtained.

1.2.1.3 Kernel-based Methods

In [40], a novel generative appearance model is proposed by Comaniciu *et al.* to tracking by employing kernel density estimation to construct kernel-based visual representations. Spatial kernels are used to regularize the color histogram-based feature representation of the target by intuitively giving bigger weights to the features near the target center, and thus to spatially smooth the similarity functions. In this way, tracking can be reformulated as a gradient-based optimization problem instead of exhaustive searching or employing a stochastic framework, e.g. particle filter, which significantly improves the computational efficiency. Mean Shift (MS) is usually adopted to solve the gradient ascent optimization algorithm, which iteratively calculates the mean shift vector and moves the candidate position towards the maximum similarity until convergence is achieved.

However, the approach proposed by Comaniciu *et al.* [40] has several drawbacks: 1) the target template is formed by a single frame, which apparently restricts the adaptability of the tracker and leads it to fail in some scenarios where the target undergoes significant appearance variation or pose change; 2) the tracking result is sensitive to the selected kernel. In particular, the selection of bandwidth demonstrates a great impact on the tracker's performance in a specific scenario; 3) the MS algorithm can be adopted to search the local optimum as the tracking result, which is based on the assumption that the displacement between successive frames is not significant. However, this assumption can be violated in some cases.

To address the fixed template problem and place further emphasis on the contour of the tracking target, Yilmaz employed a modified level set method to get the contour of the tracking object and then built an asymmetric kernel based on the learned level set function, which is different from the traditional symmetric kernel-based methods [187]. Moreover, the object centroid, orientation and scale are all successively updated during the mean shift iteration, which enables better performance when the target undergoes gentle appearance changes. In [107], Leichter *et al.* proposed an affine kernel based method instead of the early isotropical kernel. Similar to [187], the tracker in [107] also exploits both color and color edge information provided by the tracked object and adaptively fits the target's affine transformation in the course of tracking. In addition, an extension of their method was proposed in [108]. In [108], the target reference color is gradually updated on-the-fly, which effectively enhances the adaptive ability of the previous method [107]. On the other hand, Shen *et al.* [157] proposed a multi-bandwidth MS procure to overcome the sensitivity problem of [40]. In order to achieve a real-time performance, an accelerated version of the algorithm is proposed as well. In this version, they over-relax the step adaptively, which significantly reduces the number of iterations required to achieve convergence.

1.2.2 Discriminative Methods

In contrast to generative trackers that try to model the target appearance only, discriminative methods aim to learn some decision bounds and thus separate targets from background. A number of discriminative approaches, which are frequently used in machine learning area, have been introduced to tracking, such as Support Vector Machine (SVM) [7], Boosting [63], Random forest [86], decision tree. In this section, we will have a brief review of some representative trackers related to discriminative models.

1.2.2.1 SVM-based Methods

Support Vector Machines (SVMs) are powerful supervised learning approaches that have been extensively used for classification and regression in many practical applications and have proven to be effective. In [7], Avidan developed a gradient-based tracking approach termed Support Vector Tracking (SVT), which combined an offline-trained SVM classifier with an optic-flow-based tracker. In contrast to an optic-flow-based tracker that minimizes an intensity difference loss function between a pair of successive frames, SVT tries to iteratively maximize the SVM classification score in a coarse-to-fine manner. If the motions of the target are small, one can maximize classification score by iteratively computing the motion parameters based on the first-order Taylor approximation. However, when the motions are assumed to be large, a pyramid scheme should be adopted, and the optimization problem starts from a subsampled version problem. SVT takes advantage from both computational efficiency of optic-flow-based tracking [11] and the discriminative power of SVM classifiers.

Although SVT demonstrated success in tracking moving vehicles, it still remained with many problems to solve. One of its main problems is that the employment of a pre-trained SVM classifier makes it hard to implement SVT in the general tracking problem, since training a SVM classifier requires a large training set, which is usually hard to obtain and expensive. Tang *et al.* [166] accounted for this by employing a semi-supervised SVM in a co-training framework. By using a co-training framework, a few labeled samples are initially fed to train SVM classifiers based on two types of relatively independent features, respectively. The target positions are then found as the global maximum of a combined confident map built by weighted outputs of the two classifiers. Subsequently, the result with a confidence level higher than a manually defined threshold is added to update the classifiers respectively so that the classification ability of the proposed tracker is generally improved. Based on a similar SVM model used in [166], Li *et al.* [114] further developed a graph mode-based contextual kernel for SVMs, which captures the higher-order contextual information among both labeled and unlabeled samples.

By contrast to [7, 11, 166] that uses binary labels for SVM training, Hare *et al.* [68] elaborately considered the structural information of training samples by assigning them with structured labels and presented a tracker named Struck based on a kernelized structured output SVM classifier. In particular, Struck is one of the best performing trackers and has been highlighted in several recent studies [146, 160, 181].

1.2.2.2 Boosting-based Methods

Boosting is another popular tool for classification. It strategically selects a set of weak learners to create a single strong one [155] and has been widely used in the computer vision community due to its powerful discriminative learning capability. Specifically, Viola and Jones [171] successfully introduced Adaptive Boosting (AdaBoost) for face detection, which is undoubtedly a great achievement in computer vision.

In terms of tracking, Grabner *et al.* [63] proposed a tracker based on AdaBoost and the mechanism of ensemble tracking presented in [6]. The tracker selects discriminative features online and uses them to construct several weak classifiers, which are the basic components of a strong ensemble classifier. It considers both the information from the foreground and background and therefore has superior discriminative ability compared to some generative methods. Nevertheless, in [63], the positive and negative samples are selected based on results of the previous trained classifier. Therefore, errors may be accumulated gradually due to the continual updates of the classifier, and the tracker is prone to suffer from the drifting problem.

Later, Grabner *et al.* [64] and Li *et al.* [112] tackled the drifting problem by using the popular semi-supervised learning schemes. In [64], the labeled samples are first used to construct a fixed prior for the online classifier training, while Li *et al.* [112] specifically considered the distribution changes of the tracking target, which was called "Covariate Shift", and developed a new semi-supervised online boosting method, i.e. CovBoost. Instead of using the semi-supervised methods, Babenko *et al.* [8] introduced Multiple Instance Learning (MIL) method to tracking. The proposed method not only takes advantage from the traditional boosting methods in feature selection ability, but also overcomes the ambiguous labeling problem by intuitively putting all promising positive samples into a single positive bag during training.

1.2.2.3 Randomized Learning-based Methods

Random forests [30] and random ferns [145] are also popular ensemble learning methods for classification. In principle, these methods randomly select features to build multiple decision units and outputs the labels based on the weighted-sum. Random forests and random ferns have been widely used in the computer vision community due to the following merits: First, they are powerful for nonlinear classification without the use of kernels; Second, they are excellent in accuracy and very efficient in both the training and testing frames. In particular, they can be easily parallelized and further speeded up with parallel implementations. In addition, these methods are more robust to the label noise compared to Boosting [30], which is more desirable in some scenarios, such as online tracking.

To take advantage of randomized learning methods, Saffari *et al.* [153] extended the classic off-line random forest to the online version and use it for tracking. In particular, they propose a novel on-line growing procedure for decision trees, which is based on on-line bagging and extremely randomized forests. The proposed tracker achieved better performance compared to the on-line Boosting-based trackers according to their report. In [58], a random forest is combined with generalized Hough transform and a class-specific decision forest, i.e. Hough forest, is trained for detection. The latter, Godec *et al.* [59] extended the off-line Hough forest and developed an on-line Hough forest for tracking. In order to overcome the drawback of bounding-box representation, which may include too many background pixels, they also adopt the GrabCut [151] approach to segment the target within the detected bounding-box to get a better update.

In [86, 88], Tracking-Learning-Detection (TLD), developed by Kalal *et al.*, employs random ferns to train an ensemble classifier online due to its computational efficiency and flexibility. The independence of each component classifier is enforced by using different pixel comparison features. Moreover, Kalal *et al.* introduced a theory called P-N learning to improve the performance of the classifier by exploiting the structured unlabeled data. The structured data is defined as examples with no prior-known labels but their labels are restricted and can be inferred during training. In P-N learning, the learning process is guided by two constraints, i.e. positive (P) constraints and negative (N) constraints, which are the spatial occurring conditions of respective labels. The learned classifier is iteratively performed on the structured data and is updated by the labeling results that are restricted by P-N constraints.

1.3 Summary of Contributions

In this thesis, we aim to tackle challenges that are present in practical tracking scenarios in videos and to develop robust online visual trackers that achieve superior performance over existing trackers, by taking advantage of advanced techniques in machine learning including distance metric learning [124], sparse representation [179], multi-view learning [184], multi-task learning [36], conditional random field [72]. We elaborately design trackers that specifically handle distracter, background clutter, non-rigid deformation, appearance update problems and present them respectively in Chapter 2, Chapter 3, Chapter 4 and Chapter 5.

In particular, in Chapter 2, we propose a robust distracter-resistant tracking approach by learning a discriminative metric that adaptively learns the importance of features on-the-fly. The proposed metric is elaborately designed for the tracking problem by forming a margin objective function which systematically includes distance margin maximization and reconstruction error constraint that acts as a force to push distracters away from the positive space and into the negative space. Due to the variety of negative samples in the tracking problem, we specifically introduce the similarity propagation technique that gives distracters a second force from the negative space. Consequently, the discriminative metric obtained helps to preserve the most discriminative information to separate the target from distracters while ensuring the stability of the optimal metric. We seamlessly combine it with the popular L1 minimization tracker. The proposed tracker is therefore not only resistant to distracters, but also inherits the merit of occlusion robustness from the L1 tracker.

In Chapter 3, we present a tracking approach using an approximate Least Absolute Deviation (LAD)-based multi-task multi-view sparse learning method to enjoy robustness of LAD and take advantage of multiple types of visual features. The proposed method is also integrated in a particle filter framework where learning the sparse representation for each view of a single particle is regarded as an individual task. The underlying relationship between tasks across different views and different particles is jointly exploited in a unified robust multi-task formulation based on LAD. In addition, to capture the frequently emerging outlier tasks, we decompose the representation matrix to two collaborative components which enable a more robust and accurate approximation. We show that the proposed formulation can be effectively approximated by Nesterov's smoothing method and efficiently solved using the Accelerated Proximal Gradient method.

In Chapter 4, we propose a hierarchical appearance representation model for tracking, based on a graphical model that exploits shared information across multiple quantization levels. The tracker aims to find the most probable position of the target by jointly classifying the pixels and superpixels and obtaining the best configuration across all levels. The motion of the bounding box is taken into consideration, while Online Random Forests are used to provide pixel- and superpixel-level quantizations and are progressively updated on-the-fly. By appropriately considering the multilevel quantizations, our tracker exhibits not only excellent performance in non-rigid object deformation handling, but also its robustness to occlusions.

In Chapter 5, we adopt cognitive psychology principles to design a flexible representation that can adapt to changes in object appearance during tracking. Inspired by the well-known Atkinson-Shiffrin Memory Model, we propose MUlti-Store Tracker (MUSTer), a dual-component approach consisting of short- and long-term memory stores to process target appearance memories. A powerful and efficient Integrated Correlation Filter (ICF) is employed in the short-term store for short-term tracking. The integrated long-term component, which is based on keypoint matching-tracking and RANSAC estimation, can interact with the long-term memory and provide additional information for output control.

1.4 Publications Related to this Thesis

1. **Zhibin Hong**, Zhe Chen, Chaohui Wang, Xue Mei, Danil Prokhorov, Dacheng Tao: MUlti-Store Tracker (MUSTer): a Cognitive Psychology Inspired Approach to Object Tracking, IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) 2015: 749-758.
2. Xue Mei, **Zhibin Hong (equal contributor)**, Danil Prokhorov, Dacheng Tao: Robust Multi-Task Multi-View Tracking in Videos, IEEE Transactions on Neural Networks and Learning Systems (TNNLS), 2015.
3. **Zhibin Hong**, Chaohui Wang, Xue Mei, Danil Prokhorov, Dacheng Tao: Tracking Using Multilevel Quantizations. European Conference on Computer Vision (ECCV) (6) 2014: 155-171
4. **Zhibin Hong**, Xue Mei, Danil Prokhorov, Dacheng Tao: Tracking via Robust Multi-task Multi-view Joint Sparse Representation. IEEE International Conference on Computer Vision (ICCV) 2013: 649-656
5. **Zhibin Hong**, Xue Mei, Dacheng Tao: Dual-Force Metric Learning for Robust Distracter-Resistant Tracker. European Conference on Computer Vision (ECCV) (1) 2012: 513-527

Chapter 2

Distracter-Resistant Tracking via Dual-Force Metric Learning

Distracters pose great challenge in designing a robust visual tracking algorithm. They are background regions that have similar appearance to the target. When the target undergoes appearance change such as out-of-plane rotation, the distance between the new appearance and target templates is getting larger while the distance between the new appearance and the distracter is getting smaller. Therefore, the tracker can easily lock onto the distracter and fail to track the target in the following frames. In this chapter, we propose a robust distracter-resistant tracking approach by learning a discriminative metric that adaptively learns the importance of features on-the-fly. The proposed metric is elaborately designed for the tracking problem by forming a margin objective function which systematically includes distance margin maximization and reconstruction error constraint that acts as a force to push distracters away from the positive space and into the negative space. Due to the variety of negative samples in the tracking problem, we specifically introduce the similarity propagation technique that gives distracters a second force from the negative space. Consequently, the discriminative metric obtained helps to preserve the most discriminative information to separate the target from distracters while ensuring the stability of the optimal metric. We seamlessly combine it with the popular L1 minimization tracker. Our tracker is therefore not only resistant to distracters, but also inherits the merit of occlusion robustness from the L1 tracker. Quantitative comparisons with several state-of-the-art algorithms have been conducted in many challenging video sequences. The results show that our method resists distracters excellently and achieves superior performance.

2.1 Introduction

Designing a robust tracker is a challenging task due to the extremely varied factors, such as illumination, rotation, occlusion, and background clutter. Fortunately, numerous tracking algorithms have been invented to overcome these difficult problems and provide many inspiring ideas [8, 98, 131].

Most of the existing tracking algorithms employ the Euclidean distance metric for template matching; for example, the tracking result of the L1 tracker [131] is the candidate that has the minimum reconstruction error, while the error is defined as the Euclidean distance $\| T\mathbf{a} - \mathbf{y} \|_2^2$ between the target candidate and the reconstructed image using target templates, where T is the target templates, $\mathbf{a}$ is the sparse coefficients obtained by solving the L1 minimization, and $\mathbf{y}$ is the tracking candidate. Euclidean distance is independent of the input data and hence weights features equally. This is problematic, especially when tracking the target in cluttered and distractive scenes. Distracters that have a similar appearance to the target constantly emerge from the background. When the target undergoes appearance change, the Euclidean distance between the target and the distracters can be smaller than the distance between the target and its templates. Fig. 2.1 (c) and (d) illustrate this problem. In this scenario, we track a multi-pose tiger doll (target) against a cluttered background in which the tiger and the board (distracter, marked by green rectangle) in the background are similar in appearance. When the tiger moves in the sequential frames and experiences out-of-plane rotation or transformation, the initial target becomes closer to the board in terms of appearance. Consequently, the tracker locks onto the board and obtains an inaccurate update in the target template set. This phenomenon frequently happens in the tracking process and therefore many trackers fail to achieve a satisfactory performance. This motivates us to learn the optimal distance metric that takes into account the second order statistics,

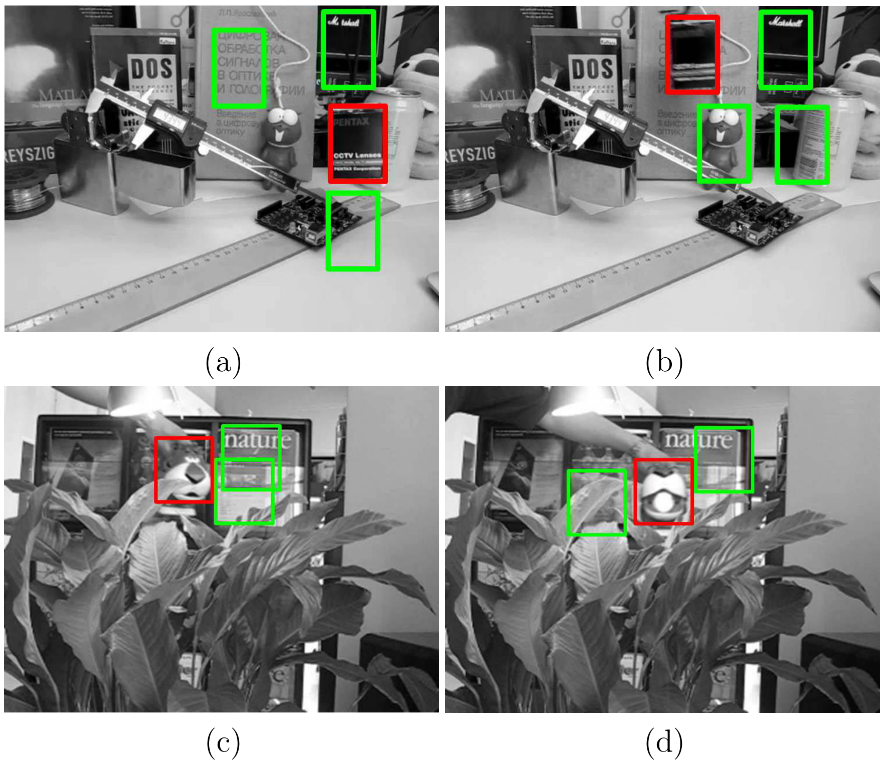


Figure 2.1: Some examples of distracters. Red windows highlight the tracking objects, while green windows are the areas which have the greatest similarity to the objects (calculated by Euclidean distance) and can easily be locked onto.

i.e. the covariance information carried by a collection of data, and assigns proper weights to features so that the learned metric can differentiate the target from distracters in the background.

Learning a distance metric can be regarded as finding the optimal selection matrix, because $\| \mathbf{x} - \mathbf{y} \|^2_{\Sigma \succcurlyeq 0} = (\mathbf{x} - \mathbf{y})^T \Sigma (\mathbf{x} - \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T U U^T (\mathbf{x} - \mathbf{y}) = \| U^T (\mathbf{x} - \mathbf{y}) \|^2_F$, where U is the feature transformation matrix, $\Sigma \succcurlyeq 0$ means Σ is a semi-positive definite matrix, and $\| \cdot \|_F$ is the Frobenius norm. Since a suitable distance metric encodes the geometric information of the data distribution and improves the performance of learning algorithms, learning a distance metric has attracted intensive attention. In recent years, dozens of algorithms have been developed for different purposes. For example, Principal Component Analysis (PCA) maximizes the mutual information between the original space and the transformed space. Linear Discriminant Analysis (LDA) maximizes the trace of between-class scatter matrix and minimizes the trace of within-class scatter matrix in the transformed space simultaneously. Both PCA and LDA assume samples are Gaussian distributed, so they cannot discover the local geometry of the non-Gaussian distributed data. Since data are intrinsically distributed on a low-dimensional manifold embedded in a high-dimensional ambient space, some manifold learning based approaches have been proposed, such as Locally Linear Embedding (LLE) [152] and Laplacian Eigenmaps (LE) [16]. However, existing distance metric learning approaches are not specifically designed for visual tracking and thus cannot solve problems encountered in tracking. In particular, positive samples for representing the target are usually scarce and are embedded in either a compact space or a local dimensional manifold (considering the transformation, rotation and illumination changes). In contrast, negative samples collected from the background are massive and various in visual tracking, and may contain samples (distracters) that are visually similar to the positive samples.

In this chapter, we solve the distracter problem by developing a dual-force metric learning algorithm that adaptively learns the importance of features. The metric is learned from image patches sampled from both the foreground and background. By incorporating distracters into the negative sample space, the metric automatically adjusts weights for different features to best differentiate the target from distracters. Since the old distracters can move out of view, and since new distracters may appear, the metric will adapt itself to the new environment by adding new distracters to the training samples. In particular, the proposed metric is induced from three functional parts which are 1) the margin maximization for enlarging the distance between the target and distracters, 2) the similarity propagation for simultaneously connecting distracters with negative samples and the target with positive samples, and 3) the neighborhood embedding for characterizing the local geometry of positive samples. The first two functional parts form the dual force to push distracters away from positive samples. The problem can be efficiently solved by employing eigenvalue decomposition. The dual-force metric can be seamlessly combined with the L1 tracker to take advantage of both the descriptive power of L1 minimization and the discriminative power of the metric. Our tracker is thus not only resistant to distracters, but also inherits the merit of occlusion robustness from the L1 minimization.

2.2 Related Work

Existing tracking algorithms can be generally classified into two groups: generative methods or discriminative methods. Generative trackers [12, 75, 150] try to build an appearance model to represent the tracking target. In this framework, tracking can be seen as a search for the candidate with the greatest similarity to the target. Template models are learned or updated to capture the appearance changes in consecutive frames. In [150], Ross *et al.* proposed a tracking method to incrementally learn a low-dimensional subspace of the target representation and efficiently update to adapt the appearance changes using the incremental singular value decomposition, the observation likelihood is then jointly governed by two Gaussian distributions related to the specific distances, the distance from the candidate observation to the subspace and the distance from the projected sample to the subspace center, respectively. Though generative trackers often have better descriptive power and demand a small training set, they only focus on the target domain and never consider the impact of the background, which makes it difficult for them to survive in a cluttered scene with distracters present, as shown in Fig. 2.2 (a).

In contrast to generative trackers, discriminative methods formulate tracking as a classification problem. They utilize both the foreground and background information and train a classifier online to extract the target from the background [5, 8]. In [8], Babenko *et al.* introduced the Multiple Instance Learning (MIL) method to overcome the ambiguous labeling problem in the tracking procedure. However, these models are vulnerable when drifting or occlusion occurs, since unexpected results are put into training, which may ruin the classification models. Because of the respective merits of generative and discriminative methods, Yu *et al.* [189] proposed a hybrid algorithm which simultaneously maintains these two types of models, generative model and discriminative model, and co-trains them incrementally to adapt the object appearance variations. Moreover, the use of generative model allows their tracker to reacquire the target after complete occlusion. Other algorithms also solved the tracking problem by adaptively learning a discriminative metric, such as methods presented in [85, 177]. In [85], Jiang *et al.* proposed a new tracking approach that incorporates an adaptive metric into a differential tracking method. It obtains a closed-form solution to the motion estimation of the moving object. Fig. 2.2 (b) shows the mutual effects of training samples in these methods.

Recently, Mei *et al.* [131] proposed a new generative L1 tracker based on sparse representation which is solved by employing L1 minimization with non-negativity constraints. Promising results have been reported and some extensions have followed [113, 120, 133]. In [133], Mei *et al.* exploited the reconstruction error bound of L2 norm and utilize it to develop a new resampling approach named Bounded Particle Resampling (BPR), which effectively excluded some unpromising particle samples, decreased the amount of samples fed to L1 minimization, resulted in accelerating the tracking speed. By contrast, Li *et al.* [113] focused on decreasing the dimension of image representative (image features) by transforming original features to a low dimension compressive feature space through a random generated orthogonal matrix, which was based on the superior signal recovery ability of CS. Meanwhile, a customized Orthogonal Matching Pursuit (OMP) algorithm was also adopted to accelerate the computation of L1 minimization in their framework. As a consequence, their proposed tracker demonstrated a real-time performance as well as an acceptable tracking ability. Although taking advantage of sparse representation helped the L1 tracker to overcome many challenging issues like occlusion, it was still fragile in resisting distracters in many tracking scenarios.

Our method is also related to the works in [45, 54]. In [45], distracters are detected and tracked along with the target in the sequence to prevent the tracker from drifting. This can add significant computational expense as the number of distracters increases in a cluttered background. To overcome the spatial distractions, Fan *et al.* [54] searched the attention regions (ARs) with discriminative power for Hough voting-based tracking. Moreover, an alternative class of approaches to the distracter challenge is presented in [24, 35, 39, 53, 83, 158]; these approaches make use of richer feature sets and focus on salient features. In [39], canonical Linear Discriminant Analysis (LDA) is applied to an extended feature set to find discriminative features for meanshift-based tracking. In [35], Cannons

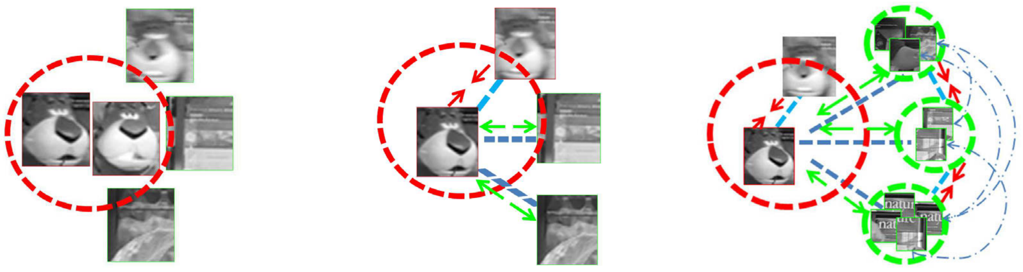

(a) (b) (c)

Figure 2.2: Mutual effects of negative and positive samples in different methods: (a) Typical generative trackers, which only focus on the positive space. Some patches from the background that are similar to the target may be selected. (b) Most discriminative methods, which only create mutual effects between positive and negative space. Some negative samples are pushed away from positive samples during optimization, while some hard negative samples may be still near the positive space. (c) Our metric. Distances between positive and negative samples are maximized. Similarity is propagated in the negative space and therefore connections are built between samples to effectively separate positive and negative samples.

et al. exploited both the appearance and motion information, and simultaneously modeled the spatial structure and dynamics of a tracking target by employing the spatiotemporal energy measurements.

Our work is mostly different from theirs in the formulation and original inspiration. We focus on designing a superior metric which well utilizes the information in the tracking procedure and can be seamlessly integrated into generative methods like [131]. Our method adds distracters to the negative sample space through similarity propagation and prevents them from being selected as the target during the tracking process. In the objective function of our metric, each negative sample can be regarded as receiving one force from the positive space (implemented by margin maximization) and another force from the negative space (implemented by similarity propagation). Therefore, we call our tracker the Dual-Force Metric Learning-based Distracter-Resistant Tracker (DFMLDR).

2.3 Particle Filter

The particle filter (PF) [48], also known as the sequential Monte Carlo method, is a Bayesian sequential importance sampling technique, which provides a convenient framework for estimating the posterior distribution of state variables and for simulating a dynamic system, such as the tracking process. The sophisticated PF has been extensively used in object tracking due to its nonlinear and non-Gaussian model assumption which regardless of the underlying distribution.

Let $\mathbf{y}_t$ denote the state variable describing the location and shape of a target at time frame t. Given all the available observations $\mathbf{x}_{1:t} = \{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$ until the current frame t, the predicting distribution of the target is inferred using the Bayes rule as

$$p(\mathbf{y}_t|\mathbf{x}_{1:t-1}) = \int p(\mathbf{y}_t|\mathbf{y}_{t-1})p(\mathbf{y}_{t-1}|\mathbf{x}_{1:t-1})d\mathbf{y}_{t-1}, \tag{2.1}$$

$$p(\mathbf{y}_t|\mathbf{x}_{1:t}) \propto p(\mathbf{x}_t|\mathbf{y}_t)p(\mathbf{y}_t|\mathbf{x}_{1:t-1}), \tag{2.2}$$

where $p(\mathbf{y}_t|\mathbf{y}_{t-1})$ is the state transition distribution, and $p(\mathbf{x}_t|\mathbf{y}_t)$ is the observation likelihood estimated by the appearance model. In practice, the posterior probability is approximated by a finite set of n samples, i.e. particles, $\{\mathbf{y}_t^i\}_{i=1}^n$ with importance weight w_t^i. At each frame, the weight w_t^i is updated by the observation likelihood $p(\mathbf{x}_t|\mathbf{y}_t^i)$ following the strategy of the bootstrap filter [48],

$$w_t^i \propto w_{t-1}^i p(\mathbf{x}_t|\mathbf{y}_t^i). \tag{2.3}$$

Subsequently, a set of n equally weighted particles are resampled according to the importance weights using state transition distribution $p(\mathbf{y}_t|\mathbf{y}_{t-1})$.

In this chapter, we let $\mathbf{y}_t = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, t_x, t_y)$ to describe the 2D affine transformation of the target, where $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ are the affine transformation parameters and (t_x, t_y) are the translation parameters, which is analogous to the previous work [133]. The state transition distribution $p(\mathbf{y}_t|\mathbf{y}_{t-1})$ is simulated independently by the Gaussian distribution model, while the observation likelihood $p(\mathbf{x}_t|\mathbf{y}_t)$ reflecting the similarity between a target candidate and the target template is estimated by the reconstruction error described in Section 2.5.2. To model the observation likelihood $p(\mathbf{x}_t|\mathbf{y}_t)$, a region corresponding to state $\mathbf{y}_t$ is first cropped from the current frame. Multiple features are then extracted from the region and normalized to form a 1D feature vector $\mathbf{x}_t$.

2.4 Dual-Force Metric Learning

In many generative methods, the similarity between a tracking candidate $\mathbf{u}$ and its approximation $\mathbf{v}$ using a combination of target templates is measured by Euclidean distance d as

$$d^2 = ||\mathbf{v} - \mathbf{u}||_2^2. \tag{2.4}$$

By introducing a measure matrix G to (2.4) as

$$d^2 = \parallel \mathbf{v} - \mathbf{u} \parallel_{\Sigma \succcurlyeq 0}^2 = (\mathbf{v} - \mathbf{u})^T \Sigma (\mathbf{v} - \mathbf{u}), \tag{2.5}$$

where (2.5) includes (2.4) as a special case by setting Σ to the identity matrix I. The Euclidean metric treats different features equally and assumes that different features are independent of one another. Apparently, the Euclidean metric is not optimal for visual tracking. Since $\Sigma \succcurlyeq 0$ is symmetric and positive semi-definite, then (2.4) can be rewritten as

$$d^2 = (\mathbf{v} - \mathbf{u})^T U U^T (\mathbf{v} - \mathbf{u}), \tag{2.6}$$

where $\Sigma = UU^T$, and U^T can be regarded as the projection matrix.

In this chapter, we learn an effective projection matrix U^T that 1) maximizes the margin between positive and negative samples, 2) propagates similarities for simultaneously connecting distracters with negative samples and the target with positive samples, and 3) embeds the neighborhood information to incorporate the local geometry of positive samples. In contrast to the proposed dual-force metric, Li *et al.* [113] replaced U^T with a hashing matrix for reducing the computational cost.

2.4.1 Dual-Force Formulation

2.4.1.1 Margin Maximization

We divide the training samples $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ into two sets: $\mathcal{X}_+ = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n_p}\}$ and $\mathcal{X}_- = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n_n}\}$, which respectively contains n_p positive samples from the target, and n_n negative samples from the background, where $n = n_p + n_n$. Analogous to [20,167], we define the margin maximization by maximizing the average distance between positive and negative samples and minimizing the average distance between positive samples in the transformed space

$$\begin{aligned} U^* = \arg\max_U \frac{1}{n_p \times n_n} \sum_{\mathbf{x}_i \in \mathcal{X}_+} \sum_{\mathbf{x}_j \in \mathcal{X}_-} \| U^T(\mathbf{x}_i - \mathbf{x}_j) \|^2 - \\ \frac{\lambda}{n_p \times n_p} \sum_{\mathbf{x}_i \in \mathcal{X}_+} \sum_{\mathbf{x}_j \in \mathcal{X}_+} c_{i,j} \| U^T(\mathbf{x}_i - \mathbf{x}_j) \|^2, \end{aligned} \tag{2.7}$$

where $c_{i,j}$ is set as $exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma})$ according to Laplacian Eigenmaps (LE) [16] for locality preservation, $\lambda \geq 0$ is the margin factor.

The distance maximization tries to find the most discriminative features between the target and distracters and thus enables our tracker to resist the selection of distracters. However, as we can see from (2.7), there is no restraint on the negative samples. More specifically, the margin maximization (2.7) only tries to maximize average pairwise distances among the positives and the negatives, while there is no reason to preserve the manifold of the negative space. Since negative samples collected from the background are massive and various, some negative samples from distracters can still be close to the positive space. Besides, although the collected positive samples come from the target space, they can be corrupted by noise or occlusions. We therefore consider similarity propagation to enhance the robustness.

2.4.1.2 Similarity Propagation

To obtain a more reliable distance metric for tracking, we should give more restraint to each sample. One common way to build the pairwise connection is to

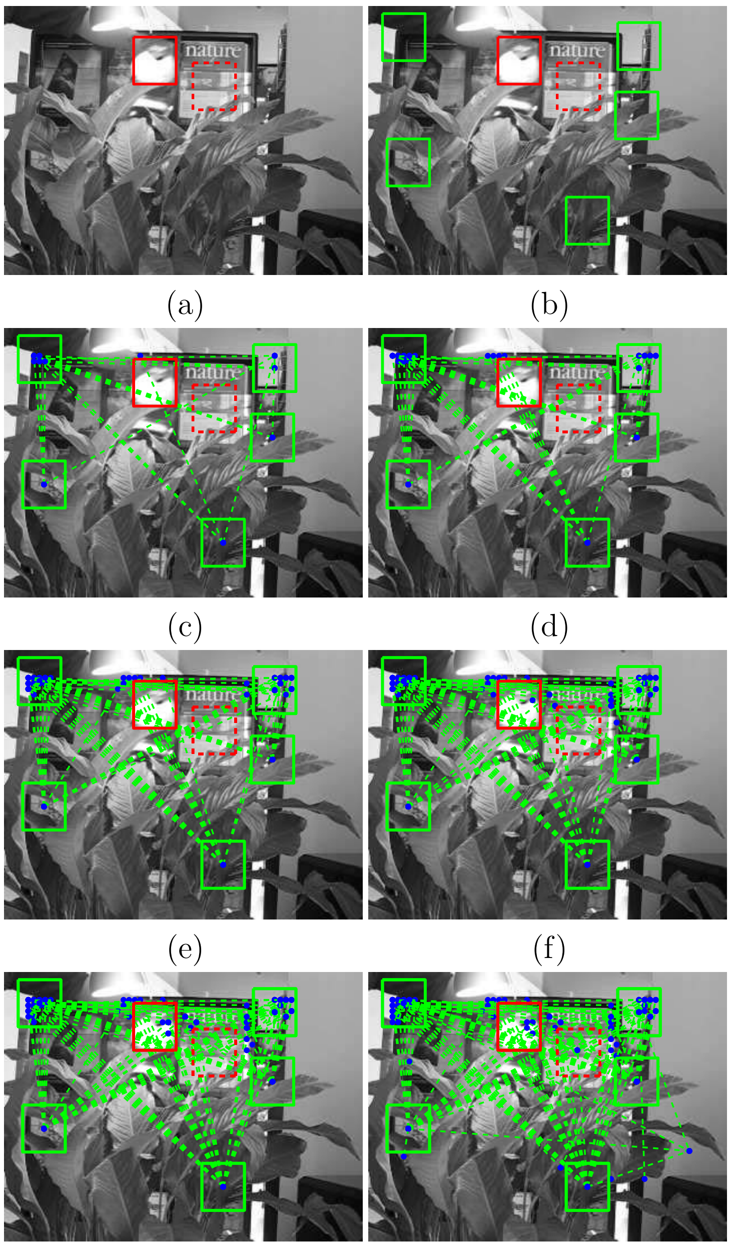


Figure 2.3: Figures to illustrate the effect of similarity propagation. (a) Without negative samples, the tracker locks onto the distracter (highlighted by dashed rectangle window). (b) to (h) Iteration of similarity propagation. The distracter is linked by other negative samples when convergence is reached. The cyan points denote the center locations of negative samples.

develop the neighborhood information by neighborhood preserving, so we begin by introducing a k-neighbor graph, which can be denoted by the neighborhood indicator matrix

$$N_{i,j}^{k} = \begin{cases} 1, & \mathbf{x}_j \text{ is the k nearest neighbor of } \mathbf{x}_i \\ 0, & \text{otherwise} \end{cases} . \tag{2.8}$$

Note that N^k is an asymmetric matrix and $N_{i,i}^k = 0$ for $i = 1, \cdots, n$. Then the original neighborhood preserving can be formulated as

$$U^* = \arg\min_U \frac{1}{2} \sum_{i,j=1}^{n} N_{i,j}^k \parallel U^T(\mathbf{x}_i - \mathbf{x}_j) \parallel^2 . \tag{2.9}$$

Although the above formula (2.9) tries to preserve neighborhood information, the setting of k is a problem. If k is large, it does not emphasize the more similar pairs in the k neighborhood, whereas if k is small, only limited connections are built. We therefore apply similarity propagation to build more connections according to the different pairwise similarity, and achieve a better result. The similarity propagation follows the general propagation scheme [124], [199] and is related to the spreading activation networks [159] and diffusion framework [94].

In (2.8), the $N_{i,j}^k$ can be considered as indicating the pairwise similarity between samples $\mathbf{x}_i$ and $\mathbf{x}_j$. Let us first set $k = 2$ to find the nearest neighbors in the respective set and build a strong similarity matrix $S = N_*^2$ (we denote it as N_*^2 rather than N^2, since it does not find the nearest neighbors in the whole set) . We then set $k = 6$ to build a weak similarity matrix $C = N^6$. Since we wish to build as many connections as possible, we propagate the strong similarity to all samples using the weak similarity and with the aim of learning an intrinsic similarity matrix N^*.

Before the propagation, we set all diagonal entries of S to 1. Now we regard any $S_{i,j} = 1$ in S as original positive energies, and we expect to propagate them to other entries where $S_{i,j} = 0$ following the paths built in the weak similarity matrix C. The propagation procedure can be formulated as [124]

$$S_i^{(t+1)} = (1-\alpha)S^{(0)} + \alpha \frac{\sum_{j=1}^{n_s} C_{i,j} S_j^{(t)}}{\sum_{j=1}^{n_s} C_{i,j}}, \tag{2.10}$$

where $S_i^{(t)}$ denotes the ith row of $S^{(t)}$, where $t = 0, 1, \cdots$ is the time stamp, α is the trade-off parameter and the initial state $S^{(0)} = S$. Let $P = D^{-1}C$ as the well-known transition probability matrix in Markov random walk models, where D is a diagonal matrix whose diagonal entries equal the sums of corresponding row elements in C, i.e., $D_{i,i} = \sum_{j=1}^{n} C_{i,j}$, then (2.10) can be rewritten as

$$S^{(t+1)} = (1-\alpha)S^{(0)} + \alpha P S^{(t)}. \tag{2.11}$$

Since $0 < \alpha < 1$ and all entries in $S^{(0)}$ are less than one, the convergence can be reached. It can be obtained by

$$S^* = \lim_{t\to\infty} S^{(t)} = (1-\alpha)(I - \alpha P)^{-1} S^{(0)}. \tag{2.12}$$

Note that S^* we obtain here is asymmetric, so we simply symmetrize it and remove small values of the symmetrized matrix to build our final similarity matrix N^*, i.e.,

$$N^* = [\frac{S^* + S^{*T}}{2}]_{\geq\theta}, \tag{2.13}$$

in which the operator $[S]_{\geq\theta}$ zeros out the entries of S whose absolute values are smaller than $0 < \theta < 1$. The effect of similarity propagation is shown in Fig. 2.3.

Then, we can replace N^k in (2.9) with N^* and add the similarity restraint to (2.7). Finally, we get the objective function of our distance metric:

$$\begin{aligned} U^* = \arg\max_U \frac{1}{n_p \times n_n} \sum_{\mathbf{x}_i \in \mathcal{X}_+} \sum_{\mathbf{x}_j \in \mathcal{X}_-} \parallel U^T(\mathbf{x}_i - \mathbf{x}_j) \parallel^2 - \\ \frac{\lambda}{n_p \times n_p} \sum_{\mathbf{x}_i \in \mathcal{X}_+} \sum_{\mathbf{x}_j \in \mathcal{X}_+} c_{i,j} \parallel U^T(\mathbf{x}_i - \mathbf{x}_j) \parallel^2 - \\ \frac{\beta}{n \times n} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{x}_j \in \mathcal{X}} N^*_{i,j} \parallel U^T(\mathbf{x}_i - \mathbf{x}_j) \parallel^2, \end{aligned} \tag{2.14}$$

where $\beta \geq 0$ is the margin factor. After adding the similarity propagation constraint, each negative sample can be regarded as being affected by two different types of forces. One force is from the positive space and pushes them into the negative space, while the other force is from the negative space and drags them into the negative space. Corrupted positive samples, such as targets with strong occlusion, may build connections with negative samples during the similarity propagation procedure and be dragged into the negative space. This helps to enhance the robustness of the obtained metric.

To solve the objective function, we first build a unified matrix M, in which each entry $M_{i,j}$ encodes a pairwise weighting factor in (2.14). This can easily be obtained by building a weighting matrix separately for each part and then adding them, as

$$M_{i,j} = \begin{cases} \frac{\beta N^*_{i,j}}{n \times n} + \frac{\lambda c_{i,j}}{n_p \times n_p}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_+ \\ \frac{\beta N^*_{i,j}}{n \times n}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_- \\ \frac{\beta N^*_{i,j}}{n \times n} + \frac{1}{n_p \times n_n}, & \text{others} \end{cases} . \tag{2.15}$$

Then, we rewrite (2.14) as

$$\begin{aligned} U^* &= \arg\max_U \frac{1}{2} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{x}_j \in \mathcal{X}} M_{i,j} \parallel U^T(\mathbf{x}_i - \mathbf{x}_j) \parallel^2 \\ &= \arg\max_U tr[U^T X(H - M)X^T U] \\ &= \arg\max_U tr[U^T X L X^T U] \\ &= \arg\max_U tr[U^T A U], \end{aligned} \tag{2.16}$$

where H is a diagonal matrix whose diagonal entry equals the sum of the corresponding row elements in M, i.e., $H_{i,i} = \sum_{j=1}^n M_{i,j}$, $L = (H - M)$ is known as the graph Laplacian, and $A = XLX^T$.

2.4.2 Reconstruction Error Constraint

Since many generative tracking algorithms utilize combination of templates to reconstruct the target model, and the geometry of positive samples should be preserved in the measure space, we apply reconstruction error constraint to positive samples $\mathcal{X}_+$. According to Neighborhood Preserving Embedding (NPE) [71], we minimize the reconstruction error for $\mathbf{x}_i$ to obtain the neighbor reconstruction weight $w_{i,j}$ of $\mathbf{x}_j$, i.e.,

$$w_{i,j} = \arg\min_{w_{i,j}} \sum_{i=1}^{n_p} \| \mathbf{x}_i - \sum_{1 \le j \ne i \le n_p} w_{i,j}\mathbf{x}_j \|^2 \, . \tag{2.17}$$

We impose $\sum_{1 \le j \ne i \le n_p} w_{i,j} = 1$ and $w_{i,i} = 0$, then the solution of $w_{i,j}$ can be obtained by constructing the the local Gram matrix $C^{(i)}$ of sample $\mathbf{x}_i$, where $C^{(i)}_{m,n} = (\mathbf{x}_i - \mathbf{x}_m)^T(\mathbf{x}_i - \mathbf{x}_n)$ is the entry (m,n) in $C^{(i)}$ and $1 \le n, m \ne i \le n_p$. So $w_{i,j} = \sum_n C^{-1}_{j,n} / \sum_{m,n} C^{-1}_{m,n}$, where the $C^{-1}_{m,n}$ is the entry (m,n) in the inverse matrix of $C^{(i)}$.

In the transformed space, we use $w_{i,j}$ to reconstruct $U^T\mathbf{x}_i$ and minimize the reconstruction error, as

$$\begin{aligned} U^* &= \arg\min_U \sum_{i=1}^{n_p} \| U^T\mathbf{x}_i - \sum_{1 \le j \ne i \le n_p} w_{i,j}U^T\mathbf{x}_j \|^2 \\ &= \arg\min_U tr[U^T X_p(I - W^T)(I - W^T)^T X_p U] \\ &= \arg\min_U tr(U^T B U), \end{aligned} \tag{2.18}$$

where $B = X_p(I - W^T)(I - W^T)^T X_p$.

To combine (2.16) with (2.18), we have the final metric objective function as

$$\begin{aligned} U^* &= \arg\max_U tr(U^T A U) - \gamma(U^T B U) \\ &= \arg\max_U tr[U^T(A - \gamma B)U] \\ &= \arg\max_U tr(U^T C_0 U), \end{aligned} \tag{2.19}$$

where γ is the margin factor, and $C_0 = (A - \gamma B)$. By imposing $U^T U = I_d$, we obtain U by applying the standard eigenvalue decomposition on C_0, and the solution comprises the d largest eigenvectors associated with the d largest eigenvalues.

2.5 Distracter-Resistant Tracker

Though the proposed metric learning method has already obtained robust discriminative power, it still should be combined with an existing tracking framework to cast tracking problems. In this section, we begin with a brief review of the L1 minimization framework. We then introduce the integration of our Dual-force distance metric and L1 framework to develop our robust distracter-resistant tracker.

2.5.1 L1 Minimization Tracking

In the L1 tracking framework [131], the appearance of a target object observation $\mathbf{y}$ is assumed to be approximately expressed by the combination of templates through a regularized L1 minimization function with nonnegativity constraints

$$\min_{\mathbf{c}} \parallel B\mathbf{c} - \mathbf{y} \parallel_2^2 + \lambda \parallel \mathbf{c} \parallel_1 \quad , \quad \text{s.t.} \ \ \mathbf{c} \succcurlyeq 0 \quad , \tag{2.20}$$

where $B = \Big[T, I, -I\Big]$ is composed of target template set T and trivial template sets I and $-I$. Each column in T is a target template generated by reshaping the pixels of a candidate patch into a column vector; and each column in the trivial template sets is a unit vector that has only one nonzero element. $\mathbf{c} = \Big[\mathbf{a}^\top, \mathbf{e}^{+\top}, \mathbf{e}^{-\top}\Big]^\top$ is composed of target coefficients $\mathbf{a}$ and positive and negative trivial coefficients $\mathbf{e}^+$ and $\mathbf{e}^-$, respectively. The ℓ_1-regularized least squares problem (2.20) can be efficiently addressed by the interior-point method described in [91] or the accelerate gradient algorithm presented in [37].

Finally, the observation likelihood is derived from the reconstruction error of $\mathbf{y}$ as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{\Gamma} \exp\{-\rho \parallel T\mathbf{a} - \mathbf{y} \parallel^2\} \ , \tag{2.21}$$

where $\mathbf{x}$ is the state variables of the candidate, $\mathbf{a}$ is obtained by solving the L1 minimization (2.20), ρ is a constant controlling the shape of the Gaussian kernel, and Γ is a normalization factor. Then the target is predicted as the candidate with the maximum observation likelihood, while the observation likelihoods of all the candidates are fed to update the particle filter and generate the particles for next frame. For more details of the original L1 tracker, please refer to [131].

2.5.2 Distracter-Resistant Tracker

To train our proposed metric learning method, we should maintain two different training sets, i.e., positive samples $\mathcal{X}_+$, negative samples $\mathcal{X}_-$, respectively. We impose the template set as positive samples directly. For further positive samples, they can be obtained by sliding the output window and collecting the samples near the target like many other tracking algorithms [8, 85]. However, this method sometimes would be unreliable if the output is not the real target. In our observation from the experiments, the first m candidates with the most likelihood always locate very near to the real target even when the wrong result occasionally appears. Therefore, we collect further positive samples by exploiting the information provided by (2.21), and these samples can be regarded as the bound of positive and negative space. Note that we only maintain the positive samples collected from last frame. For negative ones, we collect them from the background in each frame, and then put them into the larger negative set, which we call total negative set $\mathcal{X}_{t-}$. The $\mathcal{X}_{t-}$ would be updated gradually, since the volume is fixed and the oldest samples would be removed in each frame. For efficiency, we obtain $\mathcal{X}_-$ by sampling n_n negative samples from $\mathcal{X}_{t-}$ instead of using the whole $\mathcal{X}_{t-}$.

To implement our proposed metric to the L1 framework, we replace the Euclidean distance metric in (2.20), (2.21) with our dual-force distance metric. However, we rebuild the identity matrix in the transformed space instead of transforming the trivial templates because we expect it to bring the same function as it does in the original space. The objective function of our tracker is written as

$$\mathbf{c}^* = \arg\min_{\mathbf{c}} \parallel B^*\mathbf{c} - U^T\mathbf{y} \parallel_2^2 + \mu \parallel \mathbf{c} \parallel_1, \ \text{s.t.} \ \mathbf{c} \succeq 0, \tag{2.22}$$

where $B^* = [U^T T, I, -I]$ is the new template set. Meanwhile (2.21) should be modified to

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{\Gamma} \exp\{-\rho \parallel U^T T\mathbf{a} - U^T\mathbf{y} \parallel^2\} \ , \tag{2.23}$$

and we still update the template weights and target template as [131].

To increase efficiency, we also employ Bound Particle Resampling (BPR), proposed in [133], to our tracker. By adopting BPR, only candidates with high likelihood computed by Least Squares (LS) will be subjected to solving the computationally expensive L1 minimization without sacrificing resampling precision. Since the amount of time saved depends on the dissimilarity among the candidates, our metric preserves the discriminative information and therefore enhances the time-reducing power of BPR; thus, more samples would be excluded from solving the L1 minimization. Furthermore, after projecting both the target templates and candidate samples in (2.20) to the low dimensional transformed space, the computation time of BPR and L1 minimization for each sample is saved. As a result, though training a distance metric costs extra time in each frame, the total time cost does not increase significantly compared to BPR-L1 in [133], while our tracker is much faster than the original L1 tracker [131] and achieves a remarkable performance improvement.

2.6 Experiments

We implemented our Dual-Force Metric Learning-based Distracter-Resistant Tracker by combining the proposed metric learning method with the L1 tracker [133] as presented in section 2.5. Without code optimization, the tracker runs at 0.8s per frame on average on a PC (2.9GHz Intel Xeon E5-2690,32GB RAM). We tested the performance on several challenging sequences[1]. In some of the sequences, many existing trackers, such as L1 tracker, failed due to the presence of background distracters. The compared trackers included Fragments-based tracking (Frag) [2] , Increment Visual Tracking (IVT) [150], Multiple Instance Learning (MIL) tracker [8], Visual Tracking Decomposition (VTD) [98] and Bounded Particle Resampling-based L1 tracker (BPR-L1T) [133]. All of these trackers are popular and extensively mentioned in the literature. We obtained the compared tracking results by running the code provided by the original authors. Note that

[1]The box and board sequences were from `http://gpu4vision.icg.tugraz.at/index.php?content=subsites/prost/prost.php`. The tiger1 and tiger2 sequences were from `http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml`. The david_indoor sequence was from `http://www.cs.toronto.edu/~dross/ivt/`. animal and shaking sequences were from `http://cv.snu.ac.kr/research/~vtd/`. Our thanks to the authors for their contribution.

all image are resized to 320×240 for comparison and all trackers start from a manually selected position of the target in the first frame.

2.6.1 Qualitative Comparison

The Board sequence tracks a board from a cluttered background. The board is hung by a rope and rotated by around 40 degrees. The traditional rectangular window used by Frag, VTD and MIL may contain many pixels from the background and result in locking on the background or drifting from the center of the target. BPR-L1T, IVT and our tracker-adopting affine window can all effectively track the target until fully out-of-plane rotation occurs. However, our tracker can track the board well even if it undergoes the rotation, because we adaptively learn the metric by selecting the important features and update the template. A number of tracking results are presented in Fig. 2.4 (A).

The Box sequence tracks a box from the same background as the board sequence. Although there is no severe rotation in this sequence, it is also nevertheless challenging due to the existence of many traps in the background, i.e. distracters. Specifically, the MIL and BPR-L1T lock on the distracter even from frame 33, as shown in Fig. 2.4 (B). Frag loses the target when the box undergoes slightly out-of-plane rotation. All trackers lose the target between frame 460 and frame 486 because of the full occlusion; however, VTD and our tracker wait for the reappearance of the target, while IVT starts to track the background and drifts away. With the exception of our tracker, BPR-L1T and other trackers continuously lock on the distracters for the whole sequence. This demonstrate the superiority of our proposed metric.

The David_Indoor sequence is a classical sequence which contains many challenges such as illumination and pose changes. Most trackers successfully track the target when the illumination changes, except for VTD. Following the person's change of pose, IVT also shrinks to a smaller window and tracks part of the target. Compared to BPR-L1T, our tracker performs more stably when faced with the pose changes. A number of tracking results are presented in Fig. 2.4 (C).

The Tiger1 and Tiger2 sequences track a tiger doll from a cluttered background undergoing many challenges, i.e. occlusion, multiple poses, motion blur

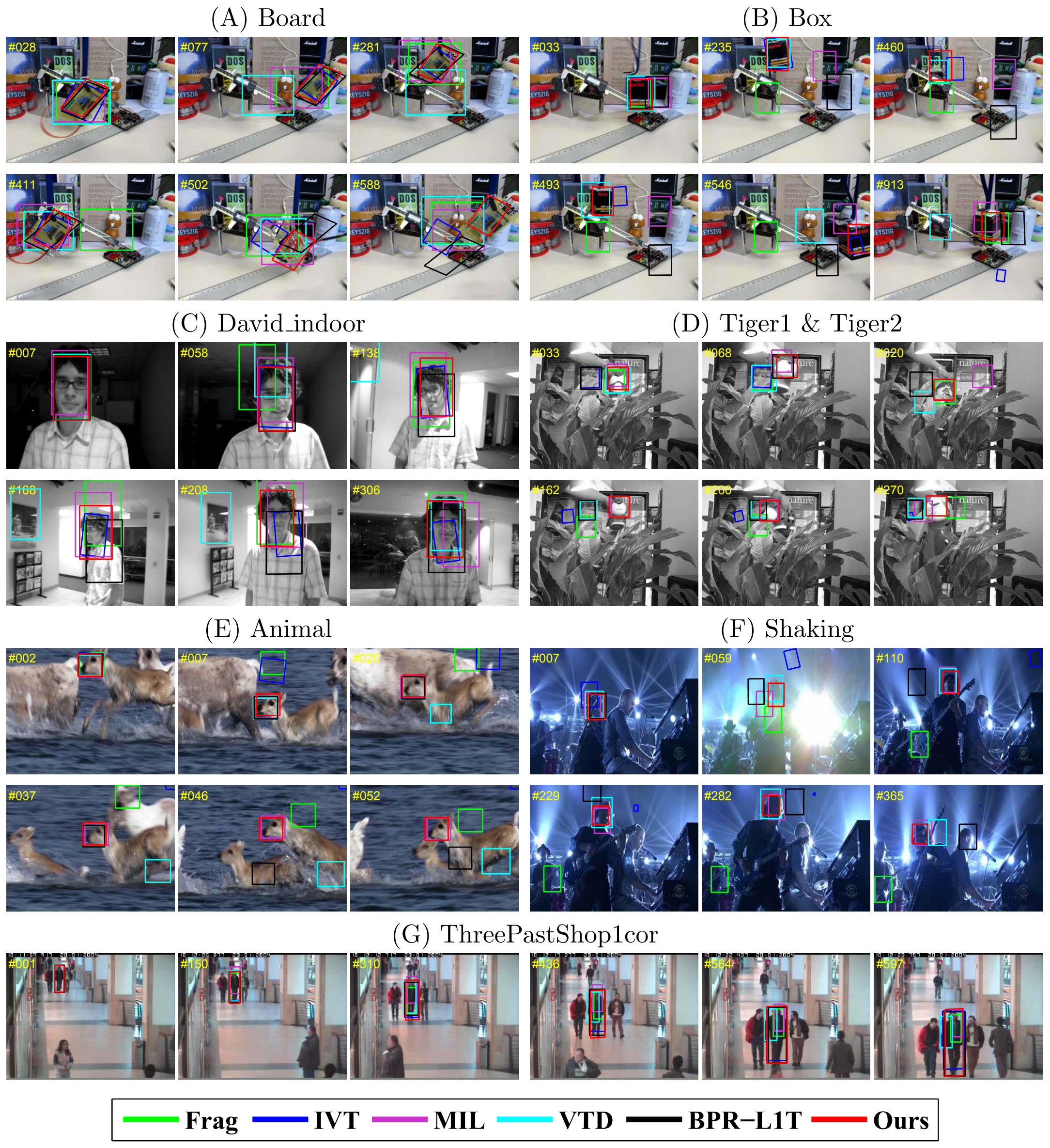


Figure 2.4: Qualitative results of DFMLDR compared with different algorithms. Frame numbers are shown in the top left of each figure. Note that (D) contains the results of two sequences, i.e. the first row for Tiger1 and the second row for Tiger 2.

and out-of-plane rotations. Frag, IVT, DVT, MIL and BPR-L1T all suffer from these challenges and lock on the background, which has a similar appearance to the target. In Tiger1, IVT locks on the background and even moves off-screen. Our tracker outperforms other trackers in both these sequences due to the robust distracter resistance of our proposed metric. A number of clips of the results are shown in Fig. 2.4 (D).

The results of the Animal and Shaking sequences are shown in Fig. 2.4 (E) and Fig. 2.4 (F) respectively. In these two sequences, the target has severe motion blur caused by abrupt movement. Only MIL and our tracker perform well in the animal sequence. The shaking sequence is much harder to track due to the significant pose and illumination changes. Frag, IVT, BPR-L1T all fail to track the singer in this sequence when these challenges occur. Our tracker survives because the metric discriminatively resists distracters and the NPE part in our metric can well preserve the manifold of the target space.

The ThreePastShop1cor sequence tracks one of three men walking through a corridor. The results are shown in Fig. 2.4 (G). As the target moves from far to near, IVT, BPR-L1T and our tracker gradually enlarge the tracking window and result in a satisfactory performance. Other trackers drift from the center and lock ons part of the person's body.

2.6.2 Quantitative Comparison

To quantitatively evaluate the performance of each tracker, two types of evaluation methodologies have been adopted in the section. As done in [8] and [131], we calculate the distance between the centers of the tracking result and the ground truth for each frame and plot this center location error versus frame number. We refer to this as "position error". For a perfect result, the position error should be zero. As shown in Fig. 2.5, the error curves of our tracker are generally lower than those of other trackers and are more closer to zero. This means our tracker effectively tracks the target in all tested sequences. Moreover, the average position errors are summarized in Tab. 2.1 for the clearer comparison. It shows that our tracker achieves the best performance in six tested sequences and overall beats the compared trackers the other five selected trackers.

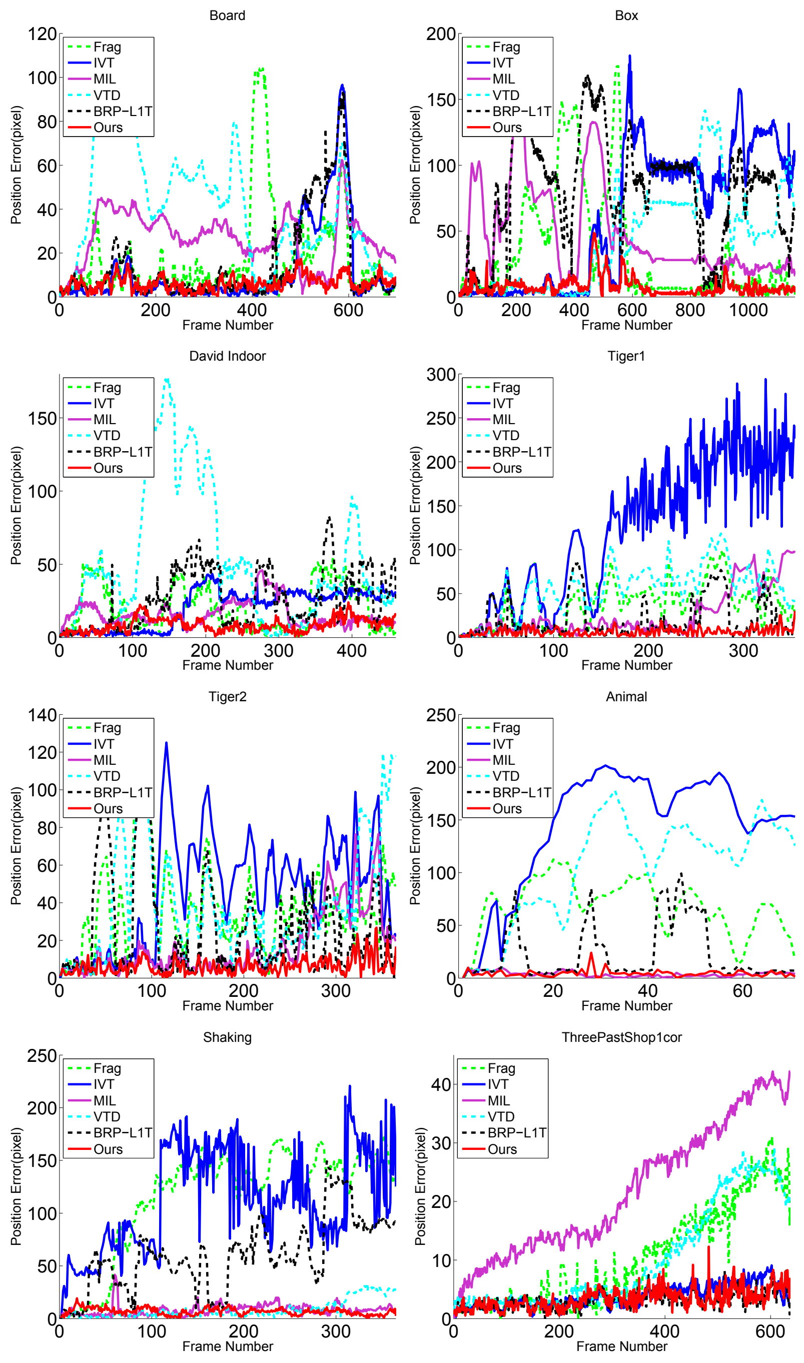


Figure 2.5: Position error (in pixel) plot of each tracker on eight tested sequences for quantitative comparison.

Table 2.1: **Average position error** (pixels). Bold indicates best performance.

	Frag	IVT	MIL	VTD	L1T	Ours
Board	17.9	11.9	26.8	41.3	14.5	**6.3**
Box	34.1	59.8	48.0	42.8	80.1	**7.9**
David_indoor	20.5	20.1	16.0	53.8	26.1	**8.6**
Tiger1	38.6	122.3	26.3	55.5	20.7	**6.7**
Tiger2	37.5	46.1	16.9	38.5	25.8	**6.2**
Animal	67.6	143.9	**3.7**	104	23.1	4.3
Shaking	115.4	112.2	8.2	8.4	59.9	**6.7**
ThreePastShop1cor	10.0	3.8	21.9	10.0	**3.1**	3.4

2.7 Conclusion

In this chapter, we have presented a novel dual-force distance metric which is elaborately designed for distracter-resistant tracking. The proposed metric systematically includes the normalized margin maximization, the similarity propagation and the reconstruction error constraint, in which the normalized margin maximization gives a force to separate positive samples and negative ones while the similarity propagation gives another force to drag negative samples into negative space. Thus we call it Dual-Force distance metric. We seamlessly integrate our metric with the L1 minimization framework and takes advantage perfectly of both the descriptive power of L1 minimization with occlusion robustness and the discriminative power of our metric with distracter resistance. We tested our tracker on several challenging sequences and compared it with other five popular trackers including original BPR-L1 tracker to validate its superiority.

Chapter 3

Robust Tracking via Multi-Task Multi-View Joint Representation

3.1 Introduction

Various sparse representation based methods have been proposed to solve tracking problems and most of them employ Least Squares (LS) criteria to learn the sparse representation. In many tracking scenarios, traditional LS based methods may not perform well due to the presence of heavy tailed noise. In this chapter, we present a tracking approach using an approximate Least Absolute Deviation (LAD)-based multi-task multi-view sparse learning method to enjoy robustness of LAD and take advantage of multiple types of visual features, such as intensity, color and texture. The proposed method is integrated in a particle filter framework where learning the sparse representation for each view of the single particle is regarded as an individual task. The underlying relationship between tasks across different views and different particles is jointly exploited in a unified robust multi-task formulation based on LAD. In addition, to capture the frequently emerging outlier tasks, we decompose the representation matrix to two collaborative components which enable a more robust and accurate approximation. We show that the proposed formulation can be effectively approximated by Nesterov's smoothing method and efficiently solved using the Accelerated Proximal Gradient method. The presented tracker is implemented using four types of features and is tested on numerous synthetic sequences and real-world video sequences including the CVPR2013 Online Object Tracking Benchmark (OOTB) [181] and ALOV++ dataset [160]. Both the qualitative and quantitative results demonstrate the superior performance of the proposed approach compared with several state-of-the-art trackers.

Online object tracking is an important research topic in computer vision and is related to many practical applications, such as video surveillance and vehicle perception. Given an annotation of the object in the first frame, the task of a tracker is to estimate the target locations using the same annotation in subsequent video frames. Many model-free trackers [115], [181] have been designed to handle generic object tracking where the prior knowledge about the target is absent. Generally, designing a universally effective tracker is extremely difficult due to the presence of various challenges, such as appearance variations, occlusions and illumination changes.

In the community of visual tracking, the Least Squares (LS) criterion, i.e. Euclidean distance, is usually used to approximate the sparse representation for tracking [133], [195]. LS performs well when the data distribution is Gaussian. It is popular because of its differentiability and smoothness properties and it can be efficiently solved by gradient-based methods [138]. However, the noise in many real tracking scenarios is heavy-tailed, such as in the cases of background clutter, Laplace noise and salt & pepper noise, where LS based methods may degrade seriously since these kinds of noise can not be well estimated by LS. According to [67], [69], Least Absolute Deviation (LAD) is much more robust than LS, especially in the presence of heavy-tailed noise.

On the other hand, tracking problems can involve data that is represented by multiple views[1] of various types of visual features including intensity [134], color [39], edge [98], wavelet [83] and texture. Relying on these multiple sources of information can significantly improve tracking performance as a result of their complementary characteristics [10, 49, 98, 123]. Given these cues from multiple views, an important problem is how to integrate them and build an appropriate

[1]Regarding the term multi-view learning [149, 183], we follow the machine learning convention, in which views refer to different feature subsets used to represent particular characteristics of an object.

model to explore their mutual dependencies and independencies.

Sparse representation has recently been introduced for tracking [133], in which a tracking candidate is sparsely represented as a linear combination of target templates and trivial templates. In particle filter-based tracking methods, particles around the current state of the target are randomly sampled according to a zero-mean Gaussian distribution. Each particle shares dependencies with other particles. Original multi-task learning in [36] aims to improve the performance of multiple related tasks by exploiting the intrinsic relationship among them. In [195], learning sparse representation of each particle is viewed as an individual task. However, [195] assumes that all tasks share a common set of features, which generally does not hold in visual tracking applications, since outlier tasks often exist. Outlier tasks are a set of minority tasks that do not share a common set of features with the majority of tasks. Furthermore, [133] and [195] only use the intensity feature to model the appearance change of the target. The intensity appearance model with L1 minimization is very robust to partial occlusion and other tracking challenges [133]. However, it is very sensitive to shape deformation of targets such as non-rigid objects.

To overcome the above problems, we propose to take advantage of LAD and employ other visual features such as color, edge, and texture to complement intensity in the appearance representation, and to combine a multi-view representation with a robust multi-task learning [61] (Fig. 3.1). Within our proposed framework, the sparse representation for each view is learned as a linear combination of atoms from an adaptive feature dictionary which enables the tracker to capture different statistics carried by different views. To exploit the interdependencies shared between different views and particles, we impose the $\ell_{1,2}$-norm group-sparsity regularization on the representation matrix to learn the joint sparse representation for all views and over all particles, where learning the sparse representation for each view of a single particle is regarded as an individual task. The LAD instead of LS reconstruction error is used to learn the sparse representation and to improve the robustness of the learned representation. We decompose the sparse representation into two collaborative parts, thereby enabling them to learn representative coefficients and detect outlier tasks simultaneously. The proposed LAD formulation is effectively approximated by Nesterov's smoothing method [139].

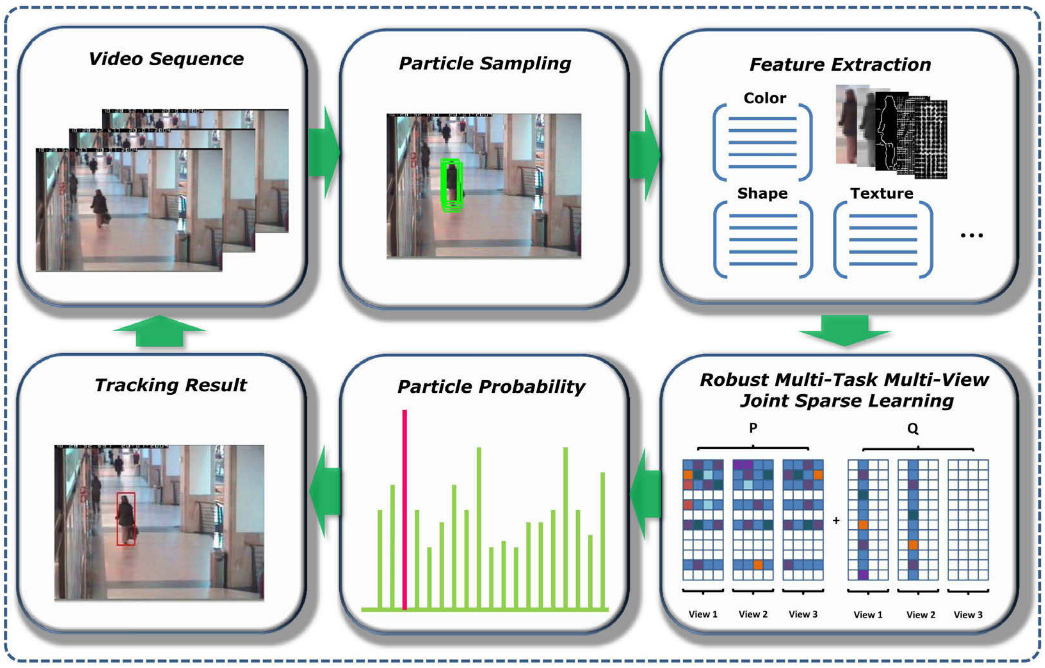


Figure 3.1: Flowchart to illustrate the proposed multi-task multi-view tracking framework.

An efficient Accelerated Proximal Gradient (APG) [138] scheme is employed to obtain the optimal solution via a sequence of closed-form updates. Although discriminative approaches can be sometimes more effective, generative methods often have better performance when the size of labeled data is small [140]. Many trackers are built on discriminative approaches [9, 39, 68, 79, 194], but there are also many generative [13,84,133,195] or even hybrid [198] methods which demonstrate superior performance in various scenarios. It should be noted that this work aims to improve the previous generative approaches [133, 195] by considering the multi-view setting in the sparse representation framework and exploring the relationship between different views among different particles. Although employing the discriminative setting in the current framework might further improve the performance, it is beyond the scope of this chapter.

Our contribution is four-fold: 1) we utilize multiple types of features in a sparse representation-based framework for tracking. Compared to previous related trackers [113] [132] [195], the new tracker is not only able to take advantage of the robustness to occlusion from sparse representation, but also introduces complementary multiple-view representation for robust appearance modeling; 2) we treat every view in each particle as an individual task and jointly consider the underlying relationship shared among different views and different particles in a multi-task learning framework; 3) to capture the outlier tasks that frequently emerge in the particle sampling process, we employ a robust multi-task scheme by decomposing the coefficient matrix into two collaborative components; and 4) outlier rejection helps identify outlier tasks and improves resampling efficiency by setting posterior probabilities of outliers to zero and making sure they are not sampled in the resampling process; 5) instead of using the popular LS criterion, we propose to take advantage of LAD which is more robust than the LS method especially when the data is contaminated by outliers and noise and use LAD reconstruction error during the sparse representation learning; 6) due to the non-smoothness of Manhattan norm used in the LAD criterion, the APG method cannot be directly used, which is different from the case in [77]. Therefore, we approximate LAD using Nesterov's smoothing method [139], and then efficiently solve the optimization using the APG scheme.

3.2 Related Work

An extensive review on tracking and multi-view learning is beyond the scope of this chapter. We refer readers to some recently published surveys [115, 185] for more details about existing trackers, and an extensive survey on multi-view learning can be found in [184]. In this section, we review the works of relavance to our method including popular single-view based trackers, multi-view based trackers and sparse representation based trackers, multi-task learning, and LAD.

Numerous existing trackers use single feature only and solve tracking in various ways. For instance, Comaniciu *et al.* [40] introduced a spatial kernel to regularize the color histogram-based feature representation of the target, which enables tracking to be reformulated as a gradient-based optimization problem solved by mean-shift. Multiple Instance Learning (MIL) tracker proposed by Babenko *et al.* [9] is equipped with a Haar feature pool to model the target appearance. In [150], Ross *et al.* presented a tracking method that incrementally learns a low-dimensional subspace representation based on intensity features. In [86], Kalal *et al.* proposed a new tracking paradigm that combines the classical Lucas-Kanade based tracker with an online learned random-forest based detector using pixel-wise comparison of features. The learned detector is notable for enabling reacquisition following tracking failures.

The above trackers nevertheless tend to be vulnerable in particular scenarios due to the limitations of the adopted features. Various methods aim to overcome this problem by taking advantage of multiple types of features to enable a more robust tracker [98, 136, 161, 188]. In [21], two complementary features, color histogram and intensity gradient, are jointly considered to track a person's head. In [136], Moreno-Noguer *et al.* proposed a probabilistic framework allowing the integration of multiple features for tracking by considering cue dependencies. Kwon and Lee [98] proposed a method termed *Visual Tracking Decomposition* (VTD) which employs Sparse Principal Component Analysis (SPCA) to construct multiple basic observation models (basic trackers) based on multiple types of features. In [100], the *visual tracker sampler* (VTS) is further proposed by the authors to sample the basic trackers and probabilistically determine the acceptance of them.

Sparse representation was recently introduced for tracking in [133] which casts tracking as a sparse representation problem in a particle filter framework [82] which was later used in [78, 113, 119, 135]. In [195], a multi-task learning [37] approach is applied to tracking by learning a joint sparse representation of all the particles in a particle filter framework. Compared to the original L1 tracker [133] that pursues the sparse representation independently, Multi-Task Tracking (MTT) achieves more robust performance by exploiting the interdependency between particles. In addition, [196] also tries to exploit the interdependency between particles and cast the tracking problem as a low-rank matrix learning problem. Multi-task learning has also been successfully applied to face recognition [44] and image classification [190]. In [190], a multi-task covariate selection model is used to classify a query image using multiple features from a set of training images, and a class-level joint sparsity regularization is imposed on class-level representation coefficients.

A well-known alternative of the popular Least Squares (LS) is the Least Absolute Deviation (LAD) [90]. In [69], Harter comprehensively discussed the method of LS and its alternatives, and proposed that LAD is more robust than the LS method especially when the data is contaminated by outliers. In addition, a lot of works have been proposed to exploit the robustness of LAD in the linear approximation problems [14, 67, 156, 175]. In [175], Wang *et al.* proposed the LASSO Regularized Least Absolute Deviation (RLAD) regression that takes advantage of both the LAD and the LASSO. In [67], Guan *et al.* also exploited the LAD and propose a Manhattan Non-negative Matrix Factorization (MahNMF) for modeling the heavy-tailed Laplacian noise. The effectiveness of the proposed method was tested on both synthetic and real-world datasets, such as face images, natural scene images, surveillance videos and multi-model datasets.

Motivated by the above advances, in this chapter, we propose a Multi-Task Multi-View Tracking (MTMVT) method based on sparse representation to exploit the related information shared between particles and views in order to obtain improved performance. Moreover, we propose to employ the LAD and minimize the Sum of Absolute Errors (SAE) regularized by the group sparsity for the joint spare representation learning. In the rest of this chapter, we denote the proposed Multi-Task Multi-View Tracking method using Least Squares as MTMVTLS and the Multi-Task Multi-View Tracking method using Least Absolute Deviation as MTMVTLAD.

3.3 Multi-task Multi-view Sparse Tracker

The L1 tracker [133] tackles tracking as finding a sparse representation in the template subspace. The representation is then used in a particle filter framework for visual tracking. However, appearance representation based only on intensity is prone to failure in difficult scenarios such as tracking non-rigid objects. Employing multiple types of features has proven to be beneficial for tracking because the ensemble of multiple views provides a comprehensive representation of the target appearance undergoing various changes such as illumination and deformation. However, combining multiple views by simply concatenating features into a high-dimensional feature vector is not a good option, since different features have different statistical properties [184]. Inspired by previous works [190, 195], the dependencies of these views as well as the intrinsic relationship of sampled particles should be jointly considered. In this section, we propose to employ other visual features such as color, edge, and texture to complement intensity in the target appearance representation, and to combine a multi-view representation with a robust multi-task learning [61] to solve the visual tracking problem.

3.3.1 Sparse Representation-based Tracker

In [133], the sparse representation of intensity feature x is formulated as the minimum error reconstruction through an L1-regularized minimization problem with nonnegativity constraints

$$\min_{\mathbf{w}} \parallel \mathbf{Mw} - \mathbf{x} \parallel_2^2 + \lambda \parallel \mathbf{w} \parallel_1 \quad , \quad \text{s.t.} \;\; \mathbf{w} \succcurlyeq 0 \quad , \tag{3.1}$$

where $\mathbf{M} = \left[\mathbf{D}, \mathbf{I}, -\mathbf{I}\right]$ is an over-complete dictionary that is composed of target template set $\mathbf{D}$ and positive and negative trivial template sets $\mathbf{I}$ and $-\mathbf{I}$, and $\succcurlyeq$ is an element-wise operator, which constrains each element at the left side to be greater or equal to the one at the right side. Each column in $\mathbf{D}$ is a target template generated by reshaping pixels of a candidate region into a column vector; and the trivial templates $\mathbf{I}$ is modeled using an identity matrix. $\mathbf{w} = \left[\mathbf{a}^\top, \mathbf{e}^{+\top}, \mathbf{e}^{-\top}\right]^\top$ is composed of target coefficients $\mathbf{a}$ and positive and negative trivial coefficients $\mathbf{e}^+$, $\mathbf{e}^-$ respectively.

Finally, the observation likelihood is derived from the reconstruction error of $\mathbf{x}$ as

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{\Gamma} \exp\{-\alpha \parallel \mathbf{Da} - \mathbf{x} \parallel^2\} \ , \tag{3.2}$$

where $\mathbf{a}$ is obtained by solving the L1 minimization (3.1), α is a constant controlling the shape of the Gaussian kernel, and Γ is a normalization factor.

3.3.2 Robust Multi-task Multi-view Sparse Learning with Least Absolute Deviation

We consider n particle samples, each of which has K different views (e.g., color, shape and texture). Learning the sparse representation for each view of a single particle is regarded as an individual task, so there are a total of nK tasks to tackle

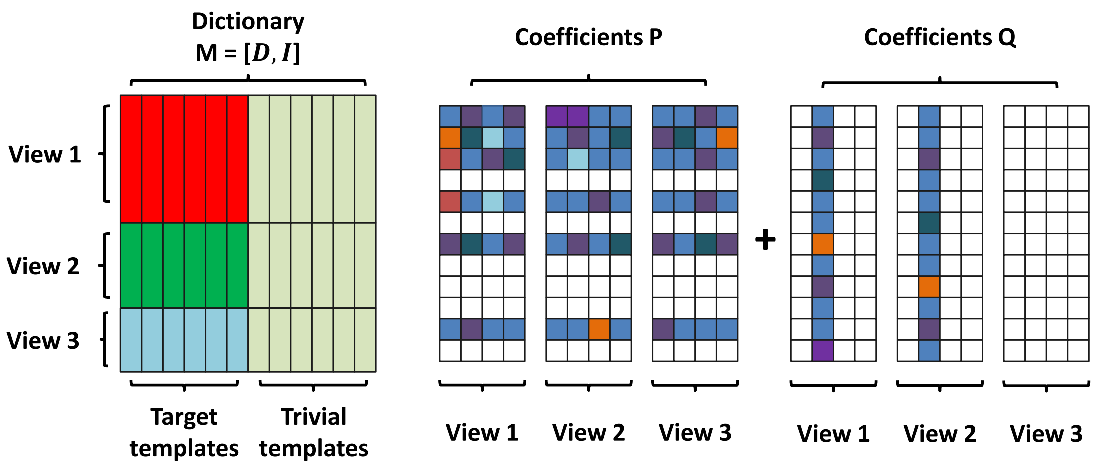


Figure 3.2: Illustration for the structure of the learned coefficient matrices $\mathbf{P}$ and $\mathbf{Q}$, where entries of different color represent different learned values, and the white entries in $\mathbf{P}$ and $\mathbf{Q}$ indicate the zero rows and columns. Note that this figure demonstrates a case that includes four particles and three views, where the second particle is an outlier whose coefficients in $\mathbf{Q}$ comprise nonzero values.

for the joint sparse representations. For each view index $k = 1, \dots, K$, denote $\mathbf{X}^k \in \mathbb{R}^{d_k \times n}$ as the feature matrix which is a stack of n columns of normalized particle image feature vectors of dimension d_k, where d_k is the dimension for the kth view. We denote $\mathbf{D}^{(k)} \in \mathbb{R}^{d_k \times N}$ as the target dictionary in which each column is a target template from the kth view, where N is the number of target templates. The target dictionary is combined with trivial templates $\mathbf{I}_{d_k} \in \mathbb{R}^{d_k \times d_k}$ to construct the complete dictionary $\mathbf{M}^{(k)} = [\mathbf{D}^{(k)}, \mathbf{I}_{d_k}] \in \mathbb{R}^{d_k \times h_k}$, where $h_k = N + d_k$.

Motivated by [61], we jointly evaluate K feature view matrices $\{\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(K)}\}$ with n particles and learn the latent representations $\{\mathbf{W}, \dots, \mathbf{W}^{(K)}\}$. The decomposed matrices $\mathbf{W}^{(k)}$s enable different views of particles to have different learned representations, and therefore exploit the independency of each view and capture the different statistical properties. Moreover, each representation matrix $\mathbf{W}^{(k)}$ is constructed by two collaborative components $\mathbf{P}^{(k)}$ and $\mathbf{Q}^{(k)}$, where $\mathbf{P}^{(k)}$ is regularized by row sparse constraint, which assumes that all particles share the same basis, while $\mathbf{Q}^{(k)}$ is regularized by column sparse constraint, which enables the capture of outlier tasks.

The same columns from each view in the dictionary should be activated to represent the particle in a joint sparse manner, since the corresponding columns represent the same sample of the object. Therefore, the corresponding decomposed weight matrices $\mathbf{P}^{(k)}$s and $\mathbf{Q}^{(k)}$s from all the views can be stacked horizontally to form two bigger matrices $\mathbf{P}$ and $\mathbf{Q}$, respectively. Each of them consists of the coefficients across all the views. Group lasso penalty $\ell_{1,2}$ is applied to row groups of the first component $\mathbf{P}$ for capturing the shared features among all tasks over all views, where we define $\|\mathbf{P}\|_{1,2} = \sum_j (\sum_i \mathbf{P}_{j,i}^2)^{1/2}$, and $\mathbf{P}_{j,i}$ denotes the entry in the jth row and ith column in the matrix $\mathbf{P}$. The same group lasso penalty is imposed on column groups of the second component $\mathbf{Q}$ to identify the outlier tasks simultaneously. The multi-view sparse representations for all particles can be obtained by solving the following problem

$$\min_{\mathbf{W},\mathbf{P},\mathbf{Q}} \sum_{k=1}^{K} f_L(\mathbf{M}^{(k)}\mathbf{W}^{(k)} - \mathbf{X}^{(k)}) + \lambda_1\|\mathbf{P}\|_{1,2} + \lambda_2\|\mathbf{Q}^\top\|_{1,2}\ , \tag{3.3}$$

where $f_L(\mathbf{X})$ is a cost function measuring the reconstruction errors during the representation learning, $\mathbf{W}^{(k)} = \mathbf{P}^{(k)}+\mathbf{Q}^{(k)}, \mathbf{P} = [\mathbf{P}^{(1)}, \ldots, \mathbf{P}^{(K)}]$, $\mathbf{Q} = [\mathbf{Q}^{(1)}, \ldots, \mathbf{Q}^{(K)}]$, and λ_1 and λ_2 are the parameters controlling the sparsity of $\mathbf{P}$ and $\mathbf{Q}$, respectively. Fig. 3.2 illustrates the structure of the learned matrices $\mathbf{P}$ and $\mathbf{Q}$.

Note that the stacking of $\mathbf{P}^{(k)}$s and $\mathbf{Q}^{(k)}$s requires that $\mathbf{M}^{(k)}$s have the same number of columns. However, we can pad the matrices $\mathbf{M}^{(k)}$s with zero columns to make them the same number of columns in order to apply (3.3). The coefficients associated with the zero columns will be zeros based on the sparsity constraints from ℓ_1 regularization and do not impact the minimization function in terms of the solution. Without loss of generality, we assume $\mathbf{M}^{(k)}$s are sorted in descending order of the number of columns h_k, that is, $h_1 \geq h_2 \geq \ldots \geq h_K$. The new $\hat{\mathbf{M}}^{(k)}$ is defined as the zero padded matrix of $\mathbf{M}^{(k)}$, that is, $\hat{\mathbf{M}}^{(k)} = [\mathbf{M}^{(k)}, \mathbf{0}^{(k)}]$, where $\mathbf{0}^{(k)} \in \mathbb{R}^{d_k \times (h_1 - h_k)}$ and every element in $0^{(k)}$ is zero. We can replace $\mathbf{M}^{(k)}$ in (3.3) with $\hat{\mathbf{M}}^{(k)}$ and solve the same minimization problem.

For the cost function $f_L(\mathbf{MW} - \mathbf{X})$, a conventional selection is based on the Frobenius norm $f_L(\mathbf{MW} - \mathbf{X}) = \frac{1}{2}\|\mathbf{MW} - \mathbf{X}\|_F^2$, which employs the Euclidean distance to measure the reconstruction error, i.e., minimizing the problem based

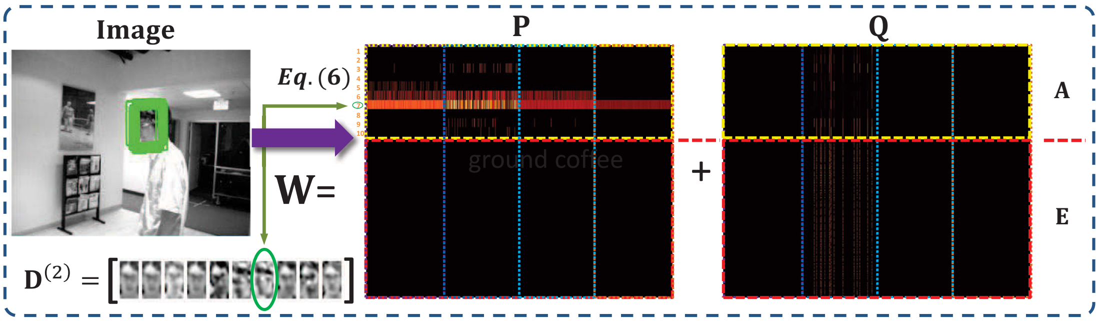


Figure 3.3: A schematic example of the learned coefficients. We visualize the learned coefficient matrices **P** and **Q** for all particles across all views, which are color histograms, intensity, HOG and LBP, respectively. Each matrix consists of four column parts corresponding to four different views, where the brighter color represents larger value in the corresponding matrix element. The seventh template in the dictionary is the most representative (which is circled in green in the shown intensity templates $\mathbf{D}^{(2)}$) and results in brighter values in the seventh row of **P** across all views (they are associated by the line with two arrows), while some columns in **Q** have brighter values which indicate the presence of outliers.

on LS criterion. Then, the problem in (3.3) can be explicitly written as

$$\min_{\mathbf{W},\mathbf{P},\mathbf{Q}} \sum_{k=1}^{K} \frac{1}{2} \|\mathbf{M}^{(k)}\mathbf{W}^{(k)} - \mathbf{X}^{(k)}\|_F^2 + \lambda_1 \|\mathbf{P}\|_{1,2} + \lambda_2 \|\mathbf{Q}^\top\|_{1,2} \; . \tag{3.4}$$

The LS is popular due to its useful properties, namely smoothness and differentiability, which enables application of efficient gradient based methods, such as APG [37], as presented in our previous work [77]. However, as discussed in [67, 69], LAD is much more robust than the ordinary LS in many applications, especially in the presence of heavy-tailed noise. The LAD estimate also arises as the maximum likelihood estimate if the errors have a Laplace distribution. To use LAD, we replace the Frobenius norm with the Manhattan norm, the problem (3.3) becomes

$$\min_{\mathbf{W},\mathbf{P},\mathbf{Q}} \sum_{k=1}^{K} \|\mathbf{M}^{(k)}\mathbf{W}^{(k)} - \mathbf{X}^{(k)}\|_{\mathbf{M}} + \lambda_1 \|\mathbf{P}\|_{1,2} + \lambda_2 \|\mathbf{Q}^\top\|_{1,2} \; . \tag{3.5}$$

In particular, the problem in (3.5) minimizes the Sum of Absolute Errors (SAE) regularized by the group sparsity prior. Although the problem (3.5) is still convex, each term in (3.5) is typically non-smooth. The APG method can no longer be used. Fortunately, we will show Nesterov's smoothing method [139] can be used to smooth the Manhattan norm in Section 3.3.4, and thus the problem (3.5) can still be solved efficiently.

For a more intuitive view of the proposed formulation, we visualize a schematic example of the learned sparse coefficients in Fig. 3.3, The $\mathbf{W}$ can be decomposed in both horizontal and vertical directions. Vertically, $\mathbf{W} = [\mathbf{A}^\top, \mathbf{E}^\top]^\top$ consists of target coefficients $\mathbf{A}$ and trivial coefficients $\mathbf{E}$ respectively, while horizontally, $\mathbf{W} = \mathbf{P} + \mathbf{Q}$ consists of information sharing matrix $\mathbf{P}$ and outlier identification matrix $\mathbf{Q}$.

3.3.3 The General Form and Special Cases

Before presenting the optimization method of (3.5), we would like to have a brief discussion about the proposed problem (3.3) in this section. The proposed optimization problem (3.3) can be generalized as

$$\min_{\mathbf{W},\mathbf{P},\mathbf{Q}} \sum_{k=1}^{K} f_L(\mathbf{M}^{(k)}\mathbf{W}^{(k)} - \mathbf{X}^{(k)}) + \lambda_1 ||\mathbf{P}||_{p,q} + \lambda_2 ||\mathbf{Q}^\top||_{p,q} \ , \tag{3.6}$$

where $\|\mathbf{P}\|_{p,q} = (\sum_j ((\sum_i (\mathbf{P}_{j,i})^q)^{1/q})^p)^{1/p}$ is the $\ell_{p,q}$ norm of $\mathbf{P}$, and $\mathbf{P}_{j,i}$ represents the element in the jth row and ith column of $\mathbf{P}$. To restrict a small number of dictionary templates to be selected by all particles across all views, let $p = 1$, then we get $\|\mathbf{P}\|_{1,q} = \sum_j (\sum_i (\mathbf{P}_{j,i})^q)^{1/q}$, which encourages $\mathbf{P}$ to be row sparse. For the options of q, we select three widely studied mixed norms $q \in \{1, 2, \infty\}$ as discussed in MTT [195]. Now we discuss (3.6) with different combinations of λ_2, q, K and f_L, which yields different trackers. If we restrict our tracker to the case of $\lambda_2 = +\infty$ and $K = 1$ for a single view multi-task problem, then we get $\mathbf{Q} = 0$. So, (3.6) is degenerated to

$$\min_{\mathbf{P}} f_L(\mathbf{MP} - \mathbf{X}) + \lambda_1 ||\mathbf{P}||_{1,q} \ , \tag{3.7}$$

where both the LS and LAD can be used. We note that if the LS is employed, where $f_L(\mathbf{MP} - \mathbf{X}) = \frac{1}{2}\|\mathbf{MP} - \mathbf{X}\|_F^2$, (3.7) is exactly the same as the objective function used in MTT [195]. Furthermore, if we let $q = 1$, the obtained formulation is intrinsically the same as (3.1), which is used in the L1 tracker [133]. In this way, both the MTT tracker and the L1 tracker can be regarded as special cases of the proposed MTMVT in the single view setting. Meanwhile, the LAD versions of MTT and L1 trackers can be naturally obtained within the proposed formulation in (3.7).

Here we discuss another single view version ($K = 1$) of MTMVT by appropriately setting $\lambda_2 > 0$, in which some nonzero columns of $\mathbf{Q}$ will be obtained if outliers exist. Specifically, if we set $q = 2$, the MTT tracker with outlier handling can be obtained as follows

$$\min_{\mathbf{P},\mathbf{Q},\mathbf{W}} f_L(\mathbf{MW} - \mathbf{X}) + \lambda_1||\mathbf{P}||_{1,2} + \lambda_2||\mathbf{Q}^\top||_{1,2} \ , \tag{3.8}$$

where $\mathbf{W} = \mathbf{P} + \mathbf{Q}$, and the component $\mathbf{P}$ exploits the underlying relationships of majority particles, while the component $\mathbf{Q}$ is able to capture the outlier tasks simultaneously, which yields more robust representations.

3.3.4 Optimization with Approximated Least Absolute Deviation

In this section we show how to solve the proposed problem (3.5) efficiently. Firstly, the Manhattan norm is approximated by a smooth function using the method presented in [67, 139], and then gradient-based method is applied to obtain the solution using a small number of closed-form updates.

Due to the separability property of Manhattan norm, we consider the following loss function of a single task

$$g(\mathbf{M}, \mathbf{w}, \mathbf{x}) = \|\mathbf{Mw} - \mathbf{x}\|_1 \ , \tag{3.9}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the single view observation for a particle, $\mathbf{M} \in \mathbb{R}^{d\times h}$ is the dictionary, and $\mathbf{w} \in \mathbb{R}^h$ is the sparse representation. As shown in [67], Nesterov's smoothing method [139] can be used to approximate (3.9) and obtain a closed-form smoothed function as

$$g_\theta(\mathbf{M}, \mathbf{w}, \mathbf{x}) = \sum_{j=1}^{d} \|\mathbf{M}_{j,\cdot}\|_1 \psi_\theta(\frac{|\mathbf{M}_{j,\cdot}\mathbf{w} - \mathbf{x}_{(j)}|}{\|\mathbf{M}_{j,\cdot}\|_1}) \ , \tag{3.10}$$

where $\mathbf{M}_{j,\cdot}$ is the jth row of $\mathbf{M}$, $\mathbf{x}_{(j)}$ is the jth entry of vector $\mathbf{x}$, and $\theta > 0$ is a parameter controls the smoothness. The larger the parameter θ the smoother the approximate function $g_\theta(\mathbf{M}, \mathbf{w}, \mathbf{x})$, but the worse the approximate accuracy. The function ψ_θ is a piecewise function defined as

$$\psi_\theta(\delta) = \begin{cases} \frac{\delta}{2\theta}, & 0 \le \delta \le \theta \\ \delta - \frac{\theta}{2}, & \delta > \theta \end{cases} \ . \tag{3.11}$$

According to [139], $g_\theta(\mathbf{M}, \mathbf{w}, \mathbf{x})$ is well defined and continually differentiable at any $\mathbf{w} \in \mathbb{R}^h$. Moreover, $g_\theta(\mathbf{M}, \mathbf{w}, \mathbf{x})$ is convex and its gradient with respect to $\mathbf{w}$ can be obtained as

$$\nabla_w g_\theta = \mathbf{M}^\top \boldsymbol{\mu} \ , \tag{3.12}$$

where $\boldsymbol{\mu} \in \mathbb{R}^d$ is the Lagrange multiplier vector and

$$\boldsymbol{\mu}_{(j)} = \text{med}(-1, +1, \frac{\mathbf{M}_{j,\cdot}\mathbf{w} - \mathbf{x}_{(j)}}{\theta\|\mathbf{M}_{j,\cdot}\|_1}) \ , \tag{3.13}$$

where med$(\cdot)$ is the median operator.

By applying Equation (3.10), the problem (3.5) can be approximated as

$$\min_{\mathbf{W},\mathbf{P},\mathbf{Q}} \sum_{k=1}^{K} G_\theta(\mathbf{M}^{(k)}, \mathbf{W}^{(k)}, \mathbf{X}^{(k)}) + \lambda_1\|\mathbf{P}\|_{1,2} + \lambda_2\|\mathbf{Q}^\top\|_{1,2} \ , \tag{3.14}$$

where $G_\theta(\mathbf{M}^{(k)}, \mathbf{W}^{(k)}, \mathbf{X}^{(k)}) = \sum_{i=1}^{n} g_\theta(\mathbf{M}^{(k)}, \mathbf{w}_i^{(k)}, \mathbf{x}_i^{(k)})$ is the cost function for the kth view of the n particles.

Let us denote by

$$\ell(\mathbf{P}, \mathbf{Q}) = \sum_{k=1}^{K} G_\theta(\mathbf{M}^{(k)}, \mathbf{W}^{(k)}, \mathbf{X}^{(k)}) \ , \tag{3.15}$$

$$r(\mathbf{P},\mathbf{Q}) = \lambda_1\|\mathbf{P}\|_{1,2} + \lambda_2\|\mathbf{Q}^\top\|_{1,2} \ . \tag{3.16}$$

Note that now the objective function in (3.14) is a composite function of two parts, a differential empirical loss function $\ell(\mathbf{P},\mathbf{Q})$ and a convex non-smooth regularization $r(\mathbf{P},\mathbf{Q})$, which has been extensively studied [37,61,138]. The Accelerated Proximal Gradient (APG) method [37] is employed because of its well-known efficiency. In contrast to traditional subgradient-based methods that converge at sublinear rate, APG can obtain the globally optimal solution at quadratic convergence rate, which means APG achieves $O(1/m^2)$ residual from the optimal solution after m iterations.

We can apply the *composite gradient mapping* [138] to (3.14) and construct the following function

$$\begin{aligned}\Phi(\mathbf{P},\mathbf{Q};\mathbf{R},\mathbf{S}) =& \ell(\mathbf{R},\mathbf{S}) + \langle\nabla_{\mathbf{R}}\ell(\mathbf{R},\mathbf{S}),\mathbf{P}-\mathbf{R}\rangle \\ &+ \langle\nabla_{\mathbf{S}}\ell(\mathbf{R},\mathbf{S}),\mathbf{Q}-\mathbf{S}\rangle + \frac{\eta}{2}\|\mathbf{P}-\mathbf{R}\|_F^2 \\ &+ \frac{\eta}{2}\|\mathbf{Q}-\mathbf{S}\|_F^2 + r(\mathbf{P},\mathbf{Q}) \ .\end{aligned} \tag{3.17}$$

In $\Phi(\mathbf{P},\mathbf{Q};\mathbf{R},\mathbf{S})$ comprises the regularization term $r(\mathbf{P},\mathbf{Q})$ and the approximation of $\ell(\mathbf{P},\mathbf{Q})$ by the first order Taylor expansion at point $(\mathbf{R},\mathbf{S})$ regularized as the squared Euclidean distance between $(\mathbf{P},\mathbf{Q})$ and $(\mathbf{R},\mathbf{S})$, where η is a parameter controlling the step penalty and $\nabla_{\mathbf{R}}\ell(\mathbf{R},\mathbf{S})$ and $\nabla_{\mathbf{S}}\ell(\mathbf{R},\mathbf{S})$ denote the partial derivatives of $\ell(\mathbf{R},\mathbf{S})$ with respect to $\mathbf{R}$ and $\mathbf{S}$. Recall that $\nabla_{\mathbf{R}}\ell(\mathbf{R},\mathbf{S}) = \nabla_{\mathbf{S}}\ell(\mathbf{R},\mathbf{S}) = \nabla_{\mathbf{W}}\ell(\mathbf{R},\mathbf{S})$, so the partial derivatives $\nabla_{\mathbf{R}}\ell(\mathbf{R},\mathbf{S})$ and $\nabla_{\mathbf{S}}\ell(\mathbf{R},\mathbf{S})$ can be computed in closed-form by Equation (3.12).

In the mth APG iteration, $(\mathbf{R}^{(m+1)},\mathbf{S}^{(m+1)})$ is computed as a linear combination of $(\mathbf{P}^{(m)},\mathbf{Q}^{(m)})$ and $(\mathbf{P}^{(m-1)},\mathbf{Q}^{(m-1)})$, so $(\mathbf{R}^{(m+1)},\mathbf{S}^{(m+1)})$ stores the historical aggregation of $(\mathbf{P},\mathbf{Q})$ in the previous iterations, which is conventionally called *aggregation step.* As suggested in [37], we set

$$\begin{aligned}\mathbf{R}^{(m+1)} &= \mathbf{P}^{(m)} + \alpha_m(\frac{1-\alpha_{m-1}}{\alpha_{m-1}})(\mathbf{P}^{(m)} - \mathbf{P}^{(m-1)}) \ , \\ \mathbf{S}^{(m+1)} &= \mathbf{Q}^{(m)} + \alpha_m(\frac{1-\alpha_{m-1}}{\alpha_{m-1}})(\mathbf{Q}^{(m)} - \mathbf{Q}^{(m-1)}) \ ,\end{aligned} \tag{3.18}$$

where α_m can be set to $\alpha_0 = 1$ for $m = 0$ and $\alpha_m = \frac{2}{m+3}$ for $m \geq 1$, and $\mathbf{P}^{(0)}$, $\mathbf{Q}^{(0)}$, $\mathbf{R}^{(1)}$ and $\mathbf{S}^{(1)}$ are all set to zero matrix for the initialization. Once given the aggregation $(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$, the solution for the mth iteration is obtained by computing the following *proximal operator*

$$(\mathbf{P}^{(m)}, \mathbf{Q}^{(m)}) = \arg\min_{\mathbf{P},\mathbf{Q}} \Phi(\mathbf{P}, \mathbf{Q}; \mathbf{R}^{(m)}, \mathbf{S}^{(m)}) \ . \tag{3.19}$$

With simple manipulations, the optimization problem (3.19) can be decomposed into two subproblems for $\mathbf{P}$ and Q respectively, as

$$\mathbf{P}^{(m)} = \arg\min_{\mathbf{P}} \frac{1}{2}\|\mathbf{P} - \mathbf{U}^{(m)}\|_F^2 + \frac{\lambda_1}{\eta}\|\mathbf{P}\|_{1,2} \ , \tag{3.20}$$

$$\mathbf{Q}^{(m)} = \arg\min_{\mathbf{Q}} \frac{1}{2}\|\mathbf{Q} - \mathbf{V}^{(m)}\|_F^2 + \frac{\lambda_2}{\eta}\|\mathbf{Q}^\top\|_{1,2} \ , \tag{3.21}$$

where $\mathbf{U}^{(m)} = \mathbf{R}^{(m)} - \frac{1}{\eta}\nabla_{\mathbf{R}}\ell(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$ and $\mathbf{V}^{(m)} = \mathbf{S}^{(m)} - \frac{1}{\eta}\nabla_{\mathbf{S}}\ell(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$.

Following the decomposition, an efficient closed-form solution can be attained respectively for each row of $\mathbf{P}^{(m)}$ and each column of $\mathbf{Q}^{(m)}$ in the above subproblems (3.20) and (3.21) according to [122],

$$\begin{aligned}\mathbf{P}_{j,\cdot}^{(m)} &= \max(0, 1 - \frac{\lambda_1}{\eta\|\mathbf{U}_{j,\cdot}^{(m)}\|})\mathbf{U}_{j,\cdot}^{(m)} \\ \mathbf{Q}_{\cdot,i}^{(m)} &= \max(0, 1 - \frac{\lambda_2}{\eta\|\mathbf{V}_{\cdot,i}^{(m)}\|})\mathbf{V}_{\cdot,i}^{(m)} \ ,\end{aligned} \tag{3.22}$$

where $\mathbf{P}_{j,\cdot}^{(m)}$ denotes the jth row of $\mathbf{P}^{(m)}$ and $\mathbf{Q}_{\cdot,i}^{(m)}$ denotes the ith column of $\mathbf{Q}^{(m)}$. Finally, the solution of (3.14) can be obtained by iteratively computing (3.22) and updating ($\mathbf{U}^{(m)}$, $\mathbf{V}^{(m)}$) until the convergence of ($\mathbf{P}$, $\mathbf{Q}$). The procedure of the presented algorithm is summarized in the *Algorithm 1* .

Algorithm 1 Optimization algorithm

```
Input: Features of $K$ views for $n$ candidate samples $\mathbf{X}$, dictionary $\mathbf{M}$, $\mathbf{P}^{(0)}$ and $\mathbf{Q}^{(0)}$, $\eta$, $\lambda_1$, $\lambda_2$
Output: $\mathbf{P}$, $\mathbf{Q}$
 1: Initial $m = 1$, $\mathbf{P}^{(0)} = 0$, $\mathbf{Q}^{(0)} = 0$, $\mathbf{R}^{(1)} = 0$, $\mathbf{S}^{(1)} = 0$
 2: while not converged do
 3:     Compute $\nabla_{\mathbf{R}}\ell(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$ and $\nabla_{S}\ell(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$ using Equation (3.12)
 4:     Compute $\mathbf{U}^{(m)} = \mathbf{R}^{(m)} - \frac{1}{\eta}\nabla_{\mathbf{R}}\ell(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$
 5:     Compute $\mathbf{V}^{(m)} = \mathbf{S}^{(m)} - \frac{1}{\eta}\nabla_{\mathbf{S}}\ell(\mathbf{R}^{(m)}, \mathbf{S}^{(m)})$
 6:     Compute $\mathbf{P}^{(m)}$ and $\mathbf{Q}^{(m)}$ using Equation (3.22)
 7:     $\alpha_m = \frac{2}{m+3}$
 8:     $\mathbf{R}^{(m+1)} = \mathbf{P}^{(m)} + \alpha_m(\frac{1-\alpha_{m-1}}{\alpha_{m-1}})(\mathbf{P}^{(m)} - \mathbf{P}^{(m-1)})$
 9:     $\mathbf{S}^{(m+1)} = \mathbf{Q}^{(m)} + \alpha_m(\frac{1-\alpha_{m-1}}{\alpha_{m-1}})(\mathbf{Q}^{(m)} - \mathbf{Q}^{(m-1)})$
10:     $m = m + 1$
11: end while
```

3.3.5 Outlier Rejection

Although a majority of particles will share the same dictionary basis, some outlier tasks may exist. The proposed MTMVT in (3.3) is capable of capturing the outliers by introducing the coefficient matrix $\mathbf{Q}$. In particular, if the sum of the ℓ_1 norm of the coefficients for the corresponding ith particle is larger than an adaptive threshold γ, as

$$\sum_{k=1}^{K} \mid \mathbf{Q}_i^{(k)} \mid > \gamma \ , \tag{3.23}$$

where $\mathbf{Q}_i^{(k)}$ is the ith column of $\mathbf{Q}^{(k)}$, then it will be identified as an outlier and its observation likelihood will be set to zero. Therefore, the outliers will be ignored in the particle resampling process and the samples will be more efficiently used to focus on locating the target position. By denoting the number of detected outliers as n_o, the threshold γ is updated as follows

$$\begin{cases} \gamma_{new} = \gamma_{old}\kappa, \ \ n_o > N_o \\ \gamma_{new} = \gamma_{old}/\kappa, \ \ n_o = 0 \\ \gamma_{new} = \gamma_{old}, \ \ 0 < n_o \leq N_o \ , \end{cases} \tag{3.24}$$

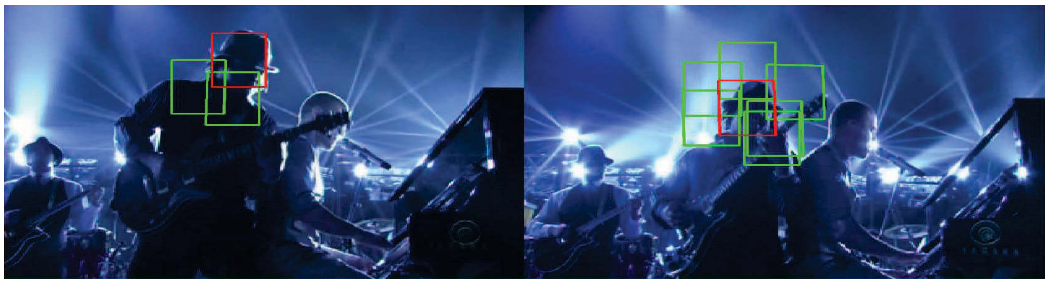


Figure 3.4: Examples of detected outliers. The green bounding boxes denote the outliers and the red bounding box denotes the tracked target. The outliers are detected out of 400 sampled particles. There are two outliers in the left frame and six outliers in the right frame.

where κ is a scaling factor, and N_o is a predefined threshold for the number of outliers. We select $\gamma = 1$, $\kappa = 1.2$ and $N_o = 20$ based on experiments. Fig. 3.4 illustrates examples with detected outliers.

3.3.6 Tracking using Robust Multi-task Multi-view Sparse Representation

In reference to the tracking result, the observation likelihood of the tracking candidate i is defined as

$$p_i = \frac{1}{\Gamma} \exp\{-\alpha \sum_{k=1}^{K} \| \mathbf{D}^{(k)}\mathbf{A}_i^{(k)} - \mathbf{X}_i^{(k)} \|^2\} \ , \tag{3.25}$$

where $\mathbf{A}_i^{(k)} \in \mathbb{R}^N$ is the coefficients of the ith candidate corresponding to the target templates of the kth view. The tracking result is the particle that has the maximum observation likelihood. It should be noted that both MTMVTLS and MTMVTLAD employ (3.25) to estimate the observation likelihood although different criterion are used to learn the sparse representation.

3.3.7 Template update

In the course of tracking, object appearance remains the same only episodically, but eventually the template is no longer an accurate model of the object appear-

Algorithm 2 Tracking via Robust Multi-Task Multi-View Sparse Representation

```
Input: Particle set $\mathcal{Y}_{t-1} = \{\mathbf{y}_{t-1}^i\}_{i=1}^n$, current complete dictionary $\mathbf{M}_{t-1} = \{\mathbf{M}_{t-1}^{(1)}, \cdots, \mathbf{M}_{t-1}^{(K)}\}$ for $K$ views
Output: Estimated target $\mathbf{y}_t^*$, particle set $\mathcal{Y}_t = \{\mathbf{y}_t^i\}_{i=1}^n$, updated complete dictionary $\mathbf{M}_t = \{\mathbf{M}_t^{(1)}, \cdots, \mathbf{M}_t^{(K)}\}$ for $K$ views
 1: /*$\mathcal{Y}_t \leftarrow \mathcal{Y}_{t-1}$ */
 2: for $i = 1$ to $n$ do
 3:     Draw particle $\mathbf{y}_t^i$ from $\mathbf{y}_{t-1}^i$ using the state transition distribution
 4: end for
 5: for $k = 1$ to $K$ do
 6:     Extract the features $\mathbf{X}^{(k)}$ according to $\mathcal{Y}_t$
 7: end for
 8: Estimate the robust joint sparse representation $\mathbf{W}, \mathbf{P}, \mathbf{Q}$ using (3.3)
 9: Detect outliers using (3.23) and set $p_i = 0$ for all outliers
10: For the remaining particles, compute $p_i$ using (3.25))
11: Find the best candidate $\mathbf{y}_t^*$ using $\arg\max_i p_i$
12: $\mathbf{M}_t \leftarrow$ Update templates
13: /*Resampling*/
14: $\mathcal{Y}_t \leftarrow$ Resample $\{\mathbf{y}_t^i\}_{i=1}^n$ with respect to $\{p_i\}_{i=1}^n$
```

ance. To handle appearance variations, the target dictionary $\mathbf{D}$ is progressively updated using an approach similar to [133, 195]. In particular, each target template in $\mathbf{D}$ is assigned a weight which represents its importance. At each frame, the norm of the learned coefficients $\mathbf{A}_i^{(k)}$s for the target particle is used to update the recorded weight of each template in $\mathbf{D}$. Once the angle between the tracked target and the most representative template (the template with the largest coefficient norm) is larger than a predefined threshold β, the template with the smallest recorded weight is replaced by the tracked target. We summarize the proposed tracking algorithm in *Algorithm* 2.

3.4 Experiments

In this section, we introduce the implementation details of the proposed trackers and report the experimental results by extensively evaluating the proposed track-

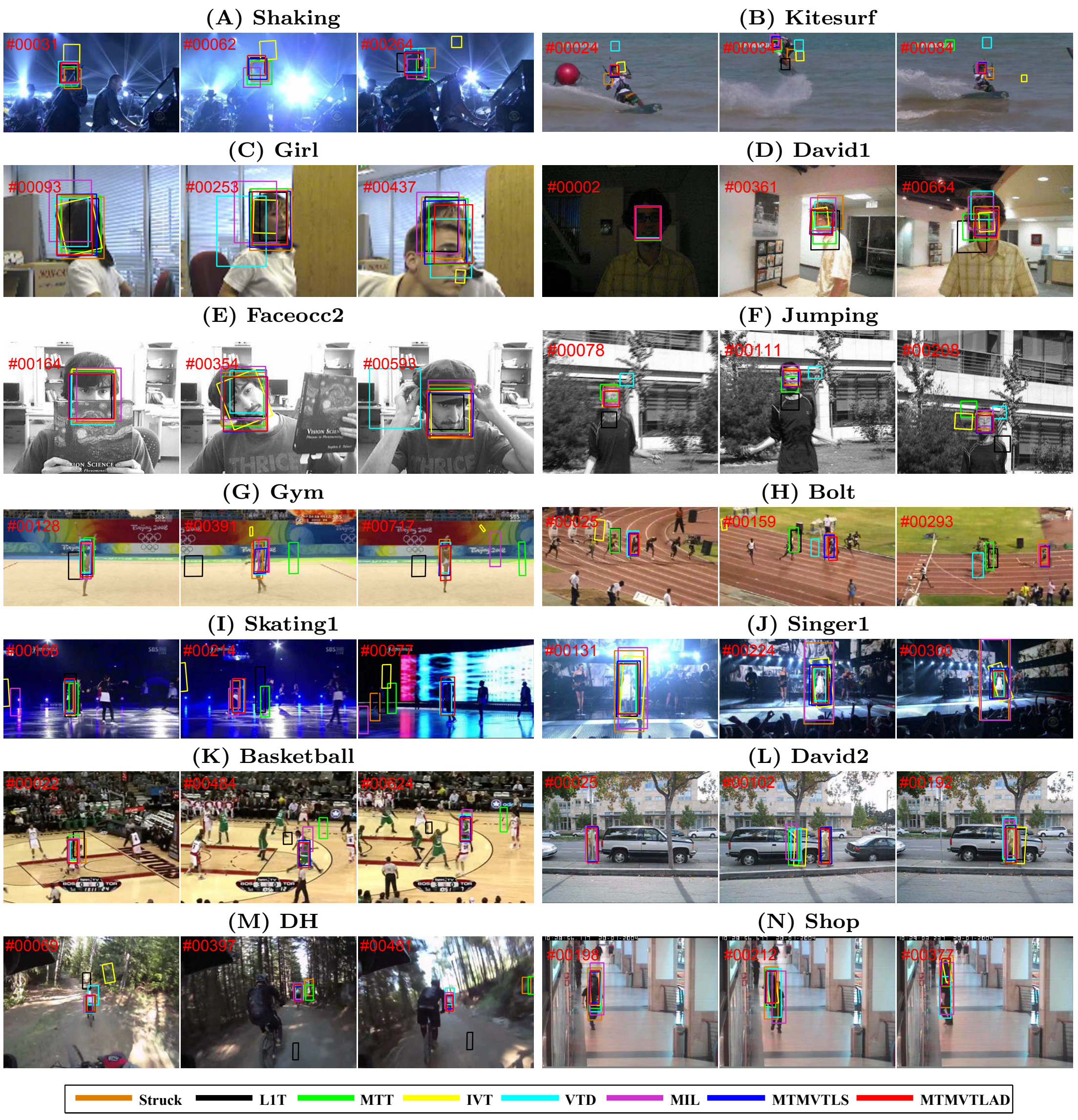


Figure 3.5: Qualitative results of MTMVTLS and MTMVTLAD compared to different algorithms. Frame indexes are shown in the top left of each figure.

ing methods on numerous video sequences[1] including a comprehensive tracking

[1] Some of the sequences are publicly available in the following websites: http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml;
http://www.cs.toronto.edu/~dross/ivt/; http://cv.snu.ac.kr/research/~vtd/;
http://www4.comp.polyu.edu.hk/~cslzhang/CT/CT.htm [193];
http://www.eng.tau.ac.il/~oron/LOT/LOT.html [144];

benchmark [181], which is recently published in CVPR2013 [1]. We also evaluated MTMVTLAD on the ALOV++ [160], which is another large dataset specifically constructed for tracking evaluations.

3.4.1 Implementation Details

To evaluate the effectiveness of the MTMVTLS and MTMVTLAD, they were implemented using four complementary features as four different views. We employed four popular features: color histograms, intensity, histograms of oriented gradients (HOG) [41] and local binary patterns (LBP) [142]. HOG is a gradient-based feature that captures edge distribution of an object. Local binary patterns (LBP) is powerful for representing object texture. Moreover, to ensure the quality of extracted features, a simple but effective illumination normalization method used in [165] is applied before the intensity feature extraction. The unit-norm normalization is applied to the extracted feature vector of each particle view respectively.

For all reported experiments, we set $\lambda_1 = \lambda_2 = 0.5$ for MTMVTLS, $\lambda_1 = 1.25$, $\lambda_2 = 1$ and $\theta = 0.1$ for MTMVTLAD, respectively. For both MTMVTLS and MTMVTLAD, we set the number of particles $n = 400$ (the same for L1T and MTT), the number of template samples $N = 10$. The template of intensity is set to one third of the size of the initial target (half size for those whose shorter side is less than 20). The color histogram, HOG, LBP are extracted in a larger template that doubles the size of the intensity template. The threshold for template update β is set to 60.

Currently, the proposed tracker MTMVTLAD is implemented using Matlab without special code optimization. The computational time of the proposed tracker mainly consists of two parts: feature extraction and the optimization solved by Algorithm 1. The feature extraction can be significantly accelerated using parallel programming based on GPU. However, we did not explore GPU programming, leaving this step for the future work. In Algorithm 1, the computational complexity of each iteration is dominated by the gradient computation

http://lrs.icg.tugraz.at/research/houghtrack/ [59]

[1]26th IEEE Conference on Computer Vision and Pattern Recognition, 2013

in Step 3 of complexity $O(nKdh)$. Therefore, the runtime of tracker depends on the dimensionality of features, the number of particles, and the number of views. Practically, in the experiment on *EXTsequences* reported in Tab. 3.4, it runs at 1.8s per frame on average on this multi-core system: 2.9GHz Intel Xeon E5-2690, 32GB RAM.

3.4.2 Evaluation on Publicly Available Sequences

In this section, we validate the effectiveness of the proposed trackers by extensively performing the experiments on 21 publicly available sequences. All original sized images are used in contrast with resizing to the same size implemented in [77]. The titles of used sequences are listed in Table 3.1. Firstly, we qualitatively compare MTMVTLS and MTMVTLAD with five other popular trackers: Struck [68], L1 Tracker (L1T) [133], Multi-Task Tracking (MTT) [195], tracking with Multiple Instance Learning (MIL) [9], Incremental Learning for Visual Tracking (IVT) [150], and Visual Tracking Decomposition (VTD) [98]. It should be noted that VTD is a multi-view tracker which employs hue, saturation, intensity, and edge templates for the features. We conducted the experiments by running source codes provided by the original authors. The recommended parameters are set for initialization.

The *Shaking*, *Kitesurf*, *Girl*, *Faceocc2*, *David1* and *Jumping* sequences track human faces under different circumstances and challenges. The experimental results show that both MTMVTLS and MTMVTLAD are able to handle the scale changes, pose changes, fast motion, occlusion, appearance variation, illumination change and angle variation problems encountered in face tracking tasks. For example, the *Shaking* sequence captures a person performing on stage. The task is to track his face under significant illumination changes and appearance variations. IVT drifts from the target quickly due to the severe appearance variation. Struck and MTT are prone to drift during the illumination change. In contrast, our trackers are more robust to the illumination changes as a result of the employment of rich feature types. In the *David1* sequence, a moving face is tracked, which presents many challenges such as pose and scale changes. Compared to L1T and MTT, MTMVTLS and MTMVTLAD successfully track the target under different challenges due to the robustness of the additional features. From the experiments, we find that IVT is vulnerable to the appearance variations, while VTD is prone to drift in occlusion scenarios. Some representative frames can be found in Fig. 3.5(A - F).

In Fig. 3.5(G - N), we show some example frames for a group of eight sequences, where the tasks are to track the human bodies in different settings. In particular, the *DH*, *Gym*, *Bolt*, *Skating1* and *Basketball* sequences track fast moving human bodies in sport scenarios. In the *DH* sequence, Struck, L1T , MTT and IVT lose the target because of the distracting background and fast motion. VTD is prone to drift and only track part of the target. In the *Gym*, *Bolt*, *Skating1* and *Basketball* sequences, the poses of targets changes rapidly and the appearance deforms frequently, which make them more challenging for existing trackers. Both L1T and IVT fail on all the sequences. Struck fails on the *Skating1* sequences due to the severe pose and illumination changes. MTT loses the targets soon on the *Bolt* and *Gym* sequences due to the deformation of the targets. VTD succeeds in the *Skating1* and *Basketball* sequences because of the benefit of multiple types of features but drifts apart from the target in the *Bolt* sequence. Besides, VTD fails in the *David2* and *Shop* sequences with the presence of occlusion. By contrast, MTMVTLS and MTMVTLAD successfully track all these targets in our experiments, which indicates the proposed tracker is not as sensitive to shape deformation as previous single view trackers, due to the effective use of the complementary features and the capability of detecting outliers. Moreover, MTMVTLAD appears to be more robust than MTMVTLS in the *Singer1* and *Basketball* sequences, where MTMVTLS tends to include some background into the bounding box.

In the last group of seven sequences, the tasks are varying from tracking animal in wild or car in road, to tracking moving dolls or object indoor. Some representative frames of these sequences are shown in Fig. 3.6(A - G). The *Animal* sequence shown in Fig. 3.6(A) tracks the head of a fast running deer. The main challenges are the fast motion and background clutter. In the *Animal* sequence, MTT, VTD, MTMVTLS and MTMVTLAD succeed in tracking the target over the whole sequence, while MIL and Struck are only able to track a part of the target though does not lose it. IVT gradually drifts from the target after the

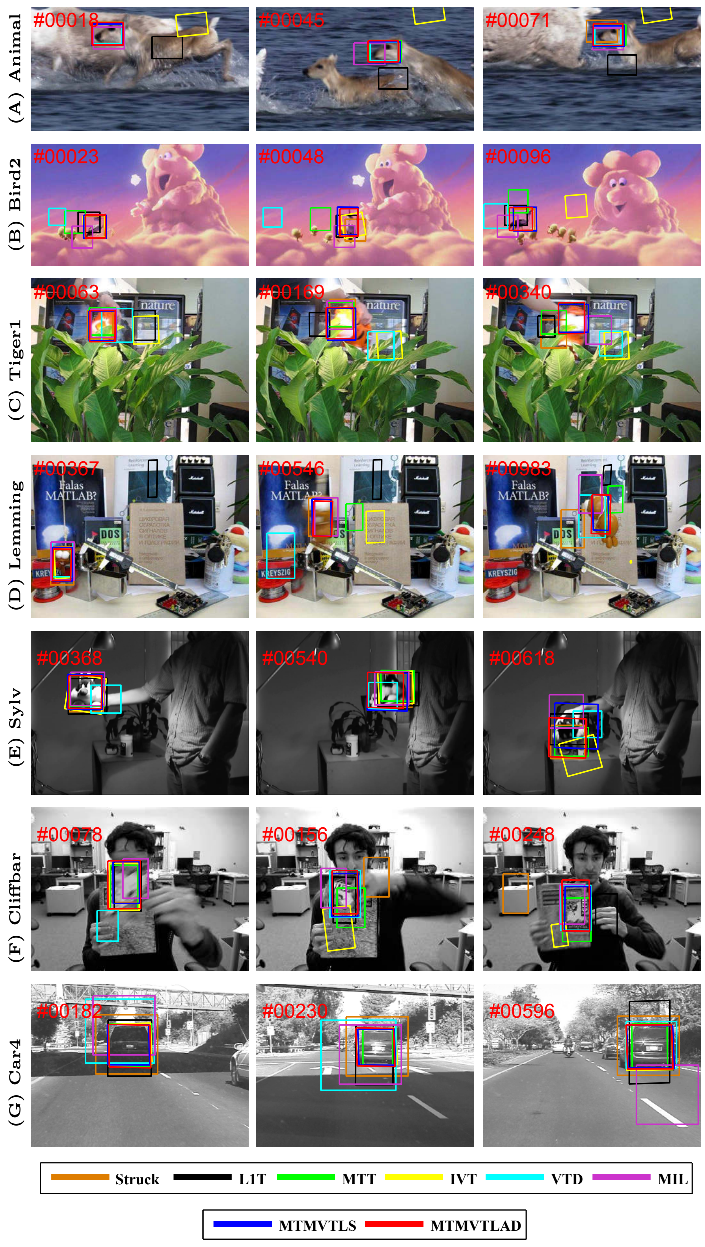


Figure 3.6: Qualitative results of MTMVTLS and MTMVTLAD compared to different algorithms. Frame indexes are shown in the top left of each figure.

third frame and totally loses the target in the sixth frame. L1T fails in the presence of fast motion and motion blur. The multi-task manner appears to make MTT, MTMVTLS and MTMVTLAD more robust than L1T. In the *Tiger1*,

Table 3.1: **Average overlap & success rates (percentages)**

Sequence	Struck	L1T	MTT	IVT	VTD	MIL	MTMVTLS	MTMVTLAD
Shaking	0.31 (0.15)	0.58 (0.59)	0.30 (0.12)	0.02 (0.01)	0.72 (0.96)	0.57 (0.58)	**0.75 (0.99)**	**0.76 (0.99)**
Kitesurf	0.17 (0.23)	0.49 (0.61)	0.31 (0.32)	0.20 (0.26)	0.03 (0.05)	0.74 (0.89)	**0.70 (0.95)**	**0.71 (0.95)**
Girl	0.73 (0.96)	0.60 (0.76)	**0.71 (0.98)**	0.34 (0.28)	0.34 (0.30)	0.37 (0.24)	**0.74 (0.97)**	0.70 (0.95)
David1	0.64 (0.83)	0.36 (0.36)	0.51 (0.53)	0.57 (0.55)	0.27 (0.22)	0.29 (0.05)	**0.72 (0.97)**	**0.72 (0.97)**
Faceocc2	**0.79 (1.00)**	0.68 (0.74)	**0.80 (1.00)**	0.68 (0.76)	0.60 (0.65)	0.72 (0.99)	0.77 (0.98)	0.77 (0.98)
Jumping	0.64 (0.83)	0.10 (0.07)	0.23 (0.15)	0.36 (0.47)	0.10 (0.10)	0.52 (0.45)	**0.70 (0.95)**	**0.71 (0.93)**
Gym	0.62 (0.75)	0.03 (0.03)	0.20 (0.16)	0.20 (0.19)	**0.66 (0.90)**	0.43 (0.43)	0.58 (0.78)	**0.62 (0.85)**
Bolt	0.49 (0.46)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.14 (0.09)	0.62 (0.82)	**0.67 (0.84)**	**0.62 (0.86)**
Skating1	0.46 (0.57)	0.29 (0.33)	0.17 (0.15)	0.06 (0.06)	**0.67 (0.92)**	0.19 (0.19)	0.67 (0.89)	**0.64 (0.91)**
Singer1	0.31 (0.22)	0.65 (0.92)	**0.77 (1.00)**	0.44 (0.37)	**0.79 (1.00)**	0.31 (0.21)	0.55 (0.50)	0.68 (0.94)
Basketball	0.38 (0.45)	0.12 (0.02)	0.17 (0.20)	0.16 (0.08)	**0.72 (0.94)**	0.26 (0.24)	0.62 (0.78)	**0.68 (0.92)**
David2	0.68 (0.91)	0.39 (0.52)	0.42 (0.56)	0.26 (0.35)	0.37 (0.51)	0.37 (0.46)	**0.69 (0.94)**	**0.67 (0.92)**
DH	0.45 (0.54)	0.14 (0.09)	0.47 (0.62)	0.07 (0.08)	0.53 (0.49)	0.53 (0.46)	**0.59 (0.83)**	**0.56 (0.67)**
Shop	0.45 (0.36)	0.76 (0.99)	**0.79 (0.99)**	0.57 (0.41)	0.32 (0.36)	0.25 (0.34)	**0.76 (1.00)**	0.76 (0.99)
Animal	0.56 (0.73)	0.05 (0.06)	0.60 (0.80)	0.03 (0.04)	**0.70 (0.97)**	0.37 (0.27)	0.58 (0.87)	**0.61 (0.89)**
Bird2	0.59 (0.55)	0.53 (0.57)	0.09 (0.09)	0.39 (0.48)	0.10 (0.13)	0.35 (0.19)	**0.67 (0.91)**	**0.71 (0.95)**
Tiger1	0.38 (0.47)	0.18 (0.15)	0.32 (0.33)	0.11 (0.13)	0.13 (0.12)	0.59 (0.70)	**0.71 (0.94)**	**0.71 (0.93)**
Lemming	0.52 (0.62)	0.11 (0.15)	0.27 (0.35)	0.26 (0.36)	0.45 (0.56)	0.35 (0.37)	**0.62 (0.81)**	**0.67 (0.89)**
Sylv	**0.75 (0.99)**	0.66 (0.91)	**0.76 (1.00)**	0.55 (0.76)	0.66 (0.78)	0.76 (0.96)	0.72 (0.94)	0.75 (0.96)
Cliffbar	0.34 (0.41)	0.42 (0.48)	0.59 (0.62)	0.34 (0.44)	0.48 (0.67)	0.53 (0.52)	**0.70 (0.88)**	**0.63 (0.80)**
Car4	0.49 (0.38)	0.54 (0.46)	0.74 (1.00)	0.85 (1.00)	0.53 (0.56)	0.28 (0.27)	**0.86 (1.00)**	**0.86 (1.00)**
Average	0.51 (0.59)	0.37 (0.42)	0.44 (0.52)	0.31 (0.34)	0.44 (0.54)	0.45 (0.46)	**0.68 (0.89)**	**0.69 (0.92)**

The quantitative comparison on the 21 sequences. The figures outside the brackets and the figures inside the brackets are the average overlap and the success rates, respectively. The **RED** number indicates the best performance, while the **GREEN** indicates the second best. The ranking is primarily based on the success rates. If the success rates scores are equal, then we compare the average overlap.

Lemming and *Sylv* sequences, the tasks are to track moving dolls in indoor scenes. Almost all the trackers compared can track the doll in the earlier part of the *Sylv* sequence. However, IVT loses the target when it undergoes pose changes. The *Tiger1* and *Lemming* sequences are much harder due to the significant appearance changes, occlusion, in-plane rotations and distractive background, so all trackers continuously lock in the background except MTMVTLS and MTMVTLAD. Our trackers faithfully tracks the dolls, and obtain the best performances. Some example shots of these three sequences are shown in Fig. 3.6(C - E). In the *Car4* sequence, MTT, IVT, MTMVTLS and MTMVTLAD perfectly track the moving car despite the dramatic illumination and scale changes, which are shown in Fig. 3.6(G). By contrast, VTD and MIL lose the target and L1T tends to include much of the background area into the bounding box when the car is moving under the bridge, which leads to significant illumination changes.

To quantitatively evaluate the performance of each tracker, we compute the bounding box overlap S_o of r_t and r_g in each frame, where r_t is the bounding box outputted by a tracker and r_g is the ground truth bounding box. The bounding box overlap is defined as $S_o = \frac{|r_t \cap r_g|}{|r_t \cup r_g|}$, where $\cap$ and $\cup$ denote the intersection and union of two regions, respectively. For more comprehensive comparison, we also compute the success rate R_o by counting the percentage of frames whose overlap S_o is bigger than a threshold $t_o = 0.5$. The average overlap as well as the success rate R_o of the eight comparative trackers on the 21 sequences are summarized in Tab. 3.1. It can be clearly seen that the proposed MTMVTLS and MTMVTLAD achieve the best average performances over all tested sequences compared to the other five popular trackers. Moreover, MTMVTLAD appears to be more robust than MTMVTLS and achieves a slightly better performance in this dataset.

3.4.3 Evaluation on Noisy Sequences

In the previous section, we compare the proposed trackers with five other trackers on 21 challenging sequences. Most of these sequences are captured under restricted environment without the contamination of noise. However, in the real-world setting, the video images may be contaminated by various of noise, which makes the tracking task even harder. In this section, we tested the proposed trackers on the sequences contaminated by different types of synthetic noise and real-world noise, e.g., snow and rain. The composition of the tested sequences and the qualitative comparison are detailed below.

3.4.3.1 Evaluation on Noisy Video Sequences

In reality, the targets and the scene can be contaminated by many kinds of noise. To evaluate robustness to noise, we contaminated the above 21 sequences with different types of synthetic noise including Gaussian, Laplace and salt & pepper noise to simulate the noise in real world and evaluate our proposed tracker on them. By synthesizing noisy images, we can choose different additive noise and control the noise level at the same time so we can better understand robustness of our method to noise. Similar approach that adds synthetic noise to the video sequences to test robustness of the tracking method has been adopted in [172]. Each type of noise is generated by four sets of different parameters indicating four light-to-heavy levels, and 12 additional groups of sequences are created, i.e., 252 (21×12) sequences in total. The parameters to generate the synthetic noise are summarized in Tab. 3.2. Some examples of the contaminated sequences

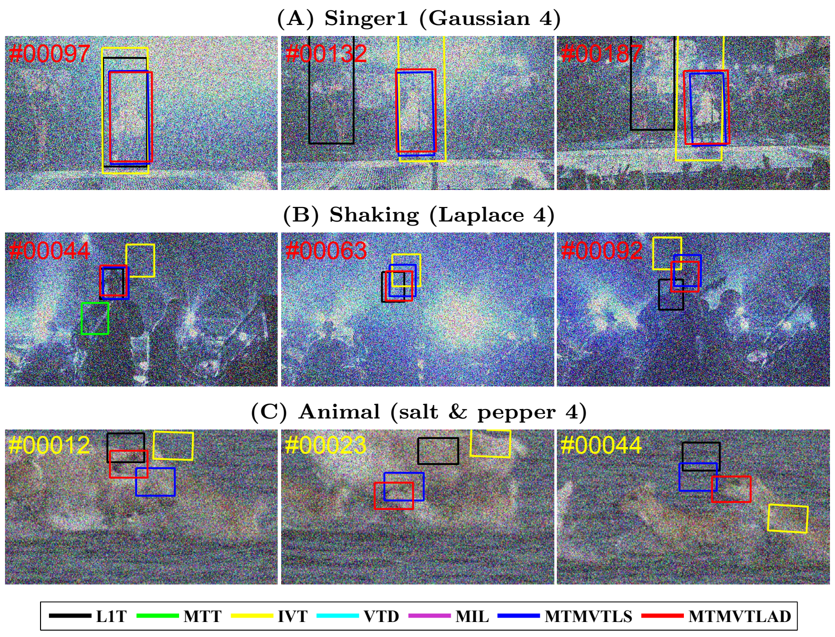


Figure 3.7: Some examples of the contaminated sequences.

as well as the qualitative results are illustrated in Fig. 3.7. To quantitatively compare the tracking performance on the 12 datasets, we summarized the average success rates in Tab. 3.3. From Tab. 3.3, we can see all trackers to some extent degraded in terms of performance on these noisy datasets. In particular, L1T and MTT appear to be more sensitive to the noise due to the use of single type of feature and the heuristic strategy used for template update. Interestingly, IVT may have sightly better performance in some of the contaminated video datasets compared to the performance on the original dataset. This phenomenon, in which addition of some noise to the input data during training may sometimes improve the generalization and therefore boost the performance, has been noted in [22]. VTD, MTMVTLS and MTMVTLAD achieve better performances on average compared to L1T, MTT and IVT because of the adoption of multiple types of features. MIL is comparable to VTD in terms of average performance since MIL appears to be insensitive to the noise levels. This suggests that the Haar feature associated with Multiple Instance Learning is just robust to our synthetic noise. On average, MTMVTLAD achieves better performance than MTMVTLS and obtain the best average performance over all comparative trackers and 12 tested datasets.

Table 3.2: Parameters of synthetic dataset

Dataset	Mean	Variance	Noise density
Gaussian 1	0.02	0.01	–
Gaussian 2	0.02	0.05	–
Gaussian 3	0.02	0.1	–
Gaussian 4	0.02	0.5	–
Laplace 1	0.02	0.01	–
Laplace 2	0.02	0.05	–
Laplace 3	0.02	0.1	–
Laplace 4	0.02	0.5	–
salt & pepper 1	–	–	0.01
salt & pepper 2	–	–	0.05
salt & pepper 3	–	–	0.1
salt & pepper 4	–	–	0.5

3.4.3.2 Evaluation on EXTsequences

To further evaluate the robustness of the proposed tracker to noise in real world, we collect nine more video sequences which are taken in extreme weather. We call this set of sequences as *EXTsequences*. The first group of six sequences deals with tracking moving vehicles in bad weather (e.g. storm, snow) and associated challenges, e.g. occlusion by the windshield wiper, illumination changes and scale changes. Some example frames as well as the tracking results can be found in Fig. 3.8(A - F). The second group of three sequences deals with tracking human faces undergoing appearance variation due to in-plane rotation. For some example

Table 3.3: **Average success rates** in the contaminated datasets.

Sequence	L1T	MTT	IVT	VTD	MIL	MTMVTLS	MTMVTLAD
Gaussian1	0.36	0.50	0.36	0.53	0.41	**0.73**	**0.83**
Gaussian2	0.42	0.35	0.38	0.49	0.37	**0.69**	**0.70**
Gaussian3	0.34	0.29	0.37	0.47	0.42	**0.65**	**0.66**
Gaussian4	0.15	0.03	0.29	0.29	0.41	**0.42**	**0.45**
Laplace1	0.45	0.45	0.34	0.54	0.40	**0.83**	**0.80**
Laplace2	0.36	0.37	0.35	0.53	0.42	**0.73**	**0.73**
Laplace3	0.34	0.31	0.37	0.49	0.41	**0.59**	**0.63**
Laplace4	0.29	0.11	0.29	0.39	0.43	**0.51**	**0.49**
Sault & Peppr1	0.38	0.58	0.42	0.54	0.44	**0.87**	**0.88**
Sault & Peppr2	0.36	0.43	0.39	0.45	0.44	**0.82**	**0.82**
Sault & Peppr3	0.33	0.36	0.39	0.51	0.42	**0.68**	**0.78**
Sault & Peppr4	0.21	0.04	0.25	0.26	**0.43**	**0.38**	0.37
Average	0.33	0.32	0.35	0.46	0.42	**0.66**	**0.68**

The **RED** number indicates the best performance, while the **Green** indicates the second best.

frames, see Fig. 3.8(G - H). In these sequences, the visibility of faces is severely affected by snowstorm and spray. However, MTMVTLS and MTMVTLAD are able to track the targets faithfully. To quantitatively evaluate the performance, we again summarize the average overlap and the success rates in Tab. 3.4. The quantitative results demonstrate that MTMVTLAD is robust to noise and it obtains the best average performance compared to other state-of-the-art trackers.

Table 3.4: **Average overlap & success rates (percentages)**

Sequence	Frames	L1T	MTT	IVT	VTD	MIL	MTMVTLS	MTMVTLAD
Storm	455	0.44 (0.33)	0.63 (0.54)	0.45 (0.48)	0.83 (0.98)	0.56 (0.46)	**0.86 (1.00)**	**0.85 (1.00)**
Winter1	998	0.81 (1.00)	**0.84 (1.00)**	0.81 (1.00)	0.67 (0.76)	0.60 (0.62)	0.81 (1.00)	**0.83 (1.00)**
Winter2	533	0.86 (1.00)	0.85 (1.00)	0.73 (1.00)	0.68 (1.00)	0.63 (0.88)	**0.88 (1.00)**	**0.88 (1.00)**
Snow1	290	0.05 (0.06)	0.35 (0.46)	0.20 (0.23)	0.24 (0.08)	0.47 (0.48)	**0.72 (0.98)**	**0.80 (0.96)**
Snow2	4435	0.86 (1.00)	0.72 (1.00)	0.74 (0.85)	0.86 (1.00)	0.24 (0.29)	**0.88 (1.00)**	**0.89 (1.00)**
DarkCar	147	**0.76 (1.00)**	0.53 (0.39)	0.55 (0.50)	0.62 (0.56)	0.45 (0.32)	**0.62 (0.89)**	0.61 (0.83)
Antarctica	580	0.50 (0.58)	0.24 (0.25)	0.01 (0.01)	0.61 (0.75)	0.17 (0.10)	**0.60 (0.92)**	**0.62 (0.93)**
Skiing1	486	0.74 (0.99)	0.77 (0.98)	0.46 (0.67)	0.73 (0.95)	0.31 (0.36)	**0.78 (0.99)**	**0.82 (1.00)**
Skiing2	846	0.51 (0.38)	0.72 (0.89)	0.02 (0.02)	0.17 (0.21)	0.04 (0.04)	**0.76 (0.89)**	**0.74 (0.90)**
Average	-	0.61 (0.70)	0.63 (0.72)	0.44 (0.53)	0.60 (0.70)	0.39 (0.39)	**0.77 (0.96)**	**0.78 (0.96)**

The quantitative comparison on EXTsequences. The figures outside the brackets and the figures inside the brackets are the average overlap and the success rates, respectively. The **RED** number indicates the best performance, while the **GREEN** indicates the second best. The ranking is primarily based on the success rates. If the success rates scores are equal, then we compare the average overlap.

3.4.4 Evaluation on CVPR2013 OOTB

To evaluate the overall performance of the proposed tracker under different scenarios and demonstrate the improvement with respect to previous methods, in this section, we conduct the experiments on the the CVPR tracking benchmark [181] and compare the proposed tracker with numerous state-of-the-art trackers and its own baseline methods. The CVPR tracking benchmark is a comprehensive tracking benchmark, which is designed for tracking performance evaluation. It consists of 50 fully annotated sequences. Each sequence is tagged with the attributes indicating to the presence of different challenges: Illuminative Variation (IV), Scale Variation (SV), Occlusion (OCC), Deformation (DEF), Motion Blur (MB), Fast Motion (FM), In-Plane-Rotation (IPR), Out-of-Plane-Rotation (OPR), Out-of-View (OV), Background Clutters (BC) and Low Resolution (LR). To evaluate the strength and weakness of different methods, the sequences are categorized according to the attributes, and 11 challenge subsets

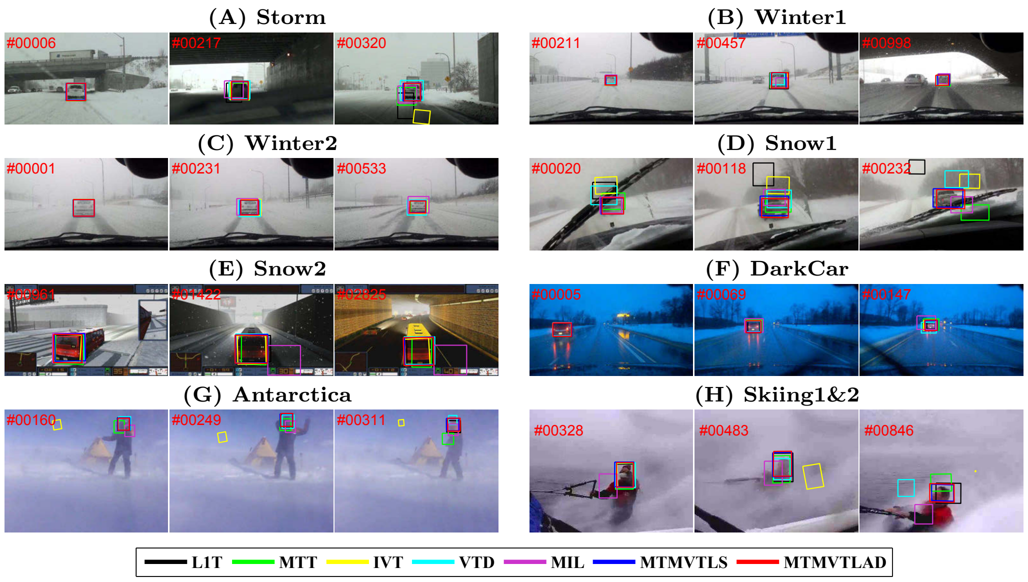


Figure 3.8: Qualitative results of MTMVTLS and MTMVTLAD compared to different algorithms on *EXTsequences*. Frame indexes are shown in the top left of each figure.

are created. In [181], the evaluation is based on two kinds of metrics, i.e., the precision plot and success plot. To obtain the precision plot, we calculate the Center Location Error (CLE), which is the distance between the centers of the tracking result and the manually labeled ground truth for each frame. The precision plot shows the percentage of frames whose CLE is within a given threshold and uses a representative precision score for ranking by choosing an appropriate threshold ($r = 20$). Another metric is to compute the bounding box overlap S_o which has been defined in Section 3.4.2. The number of frames whose overlap S_o is larger than the given threshold t_o is counted. The success plot shows the ratios of successful frames at the thresholds varied from 0 to 1. In success plot, the ranking is based on the area under curve (AUC) instead of a specific threshold. For the comparative trackers, it currently includes 29 popular tracking algorithms including the Struck, MTT, IVT, VTD, MIL, which have been tested in previous sections, and the L1APG [13] (a newer version of L1T). For more details about the benchmark, we refer readers to the original paper [181].

3.4.4.1 Comparison with trackers on CVPR2013 Tracking Benchmark

We run the one-pass evaluation (OPE) on the benchmark using the proposed trackers MTMVTLS and MTMVTLAD and compare them with the 29 popular tracking methods previously evaluated in [181]. We also compare the proposed trackers with the latest version of LRT [197], which has similar motivation as our our methods. In [197], the consensus between particles are enforced through low-rank minimization.

It should be noted that we strictly follow the protocol proposed by the authors and use the same parameters for all the sequences. For comparison, we use the on-line available [1] tracking results and the unified tool provided by [181] to compute the evaluation plots. For the results of [197], we downloaded the code from the authors' website [2] In the CVPR tracking benchmark, the proposed MTMVTLAD and MTMVTLS achieve overall the best and the second best performance using the precision plot as the metric, which is shown Fig. 3.9. MTMVTLS and MT-MVTLAD also rank in the top ten from all 32 trackers over all challenge subsets using either the measurement of precision plots or success plots. According to the results, MTMVTLS and MTMVTLAD are more robust to background clutter, deformation, in-plane rotation and out-plane rotation challenges compared to other 30 trackers because the proposed methods can effectively take advantage of complementary features. Moreover, MTMVTLAD takes the first places in 6 out of 11 challenge subsets when using the success plot as the metric because LAD is advantageous to learn more appropriate representations. We show the success plots of the BC and DEF subsets in Fig. 3.10, but omit other figures due to the space limits.

3.4.4.2 Comparison with baseline methods

In previous sections, we have demonstrated the superior performance of MTMVT-LAD compared to MTMVTLS and other state-of-art trackers. However, it is also important to compare MTMVTLAD with its baseline variants to demonstrate

[1] http://visual-tracking.net/

[2] http://nlpr-web.ia.ac.cn/mmc/homepage/tzzhang/Project_Tianzhu/zhang_IJCV14/RobustVisualTrackingViaConsistentLow-RankSparse.html and ran it on all sequences using the default parameters.

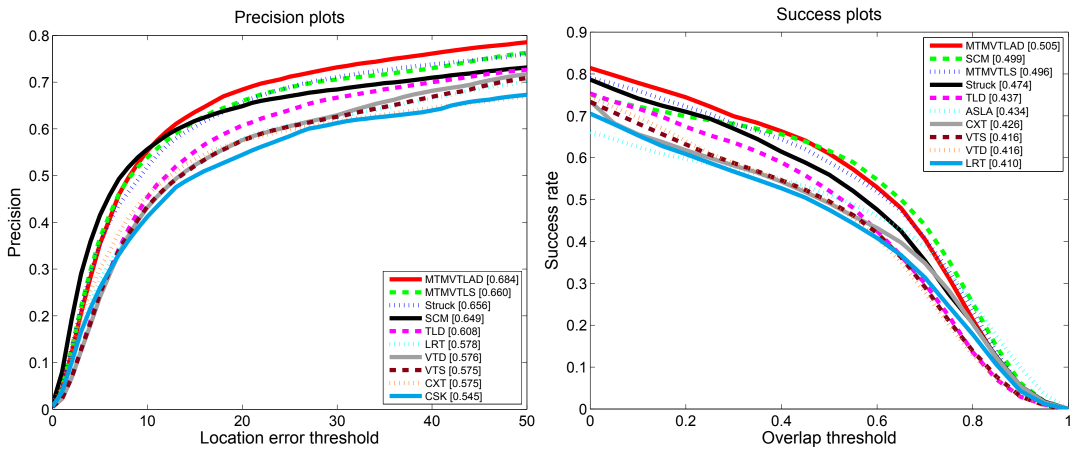


Figure 3.9: Precision plots and success plots on the CVPR2013 tracking benchmark. The values appearing in the legend of the precision plot are the precision scores in the threshold of 20, while the ones in Success plots are the AUC scores. Only the top 10 trackers are presented, while the other trackers can be found in [181]. The trackers appearing in the legend are as follows: Struck [68], SCM [198], TLD [86], LRT [197], VTD [98], VTS [99], CXT [45], CSK [74], ASLA [84].

the component-wise contributions to the performance of the proposed tracker.

Firstly, we validate the improvement brought by the robust multi-task multi-view representation by testing it in both single view and multi-view settings and comparing it with their corresponding baseline variants. To this end, we implement two multi-task single view ($K = 1$) trackers based on (3.4) and denote them respectively as MTSVTLS and MTSVTLS(-), where MTSVTLS(-) is constructed by removing the functional component $\mathbf{Q}$ (no outlier handling). It should be noted that the formulation of MTSVTLS(-) is the same as MTT [195] since MTT is a special case of the proposed general form (3.6) discussed in Section 3.3.3. MTSVTLS and MTSVTLS(-) are both using intensity feature only, similar to MTT [195]. We also implement a multi-task multi-view tracker MTMVTLS(-) similar to MTMVTLS but removing the functional component $\mathbf{Q}$. To test the improvement brought by the LAD formulation, we construct the corresponding LAD-based version and denote them by MTSVTLAD, MTSVTLAD(-) and MTMVTLAD(-) respectively. We ran these variants on the CVPR benchmark

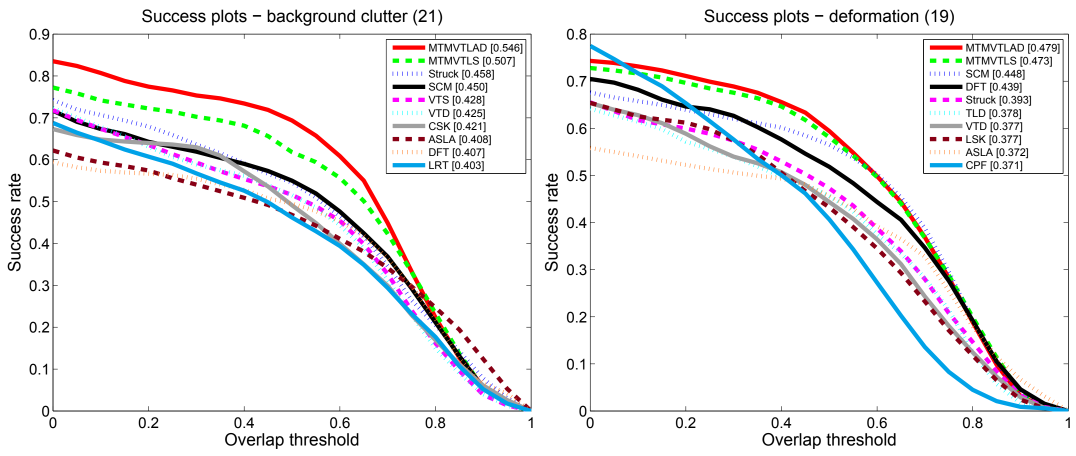
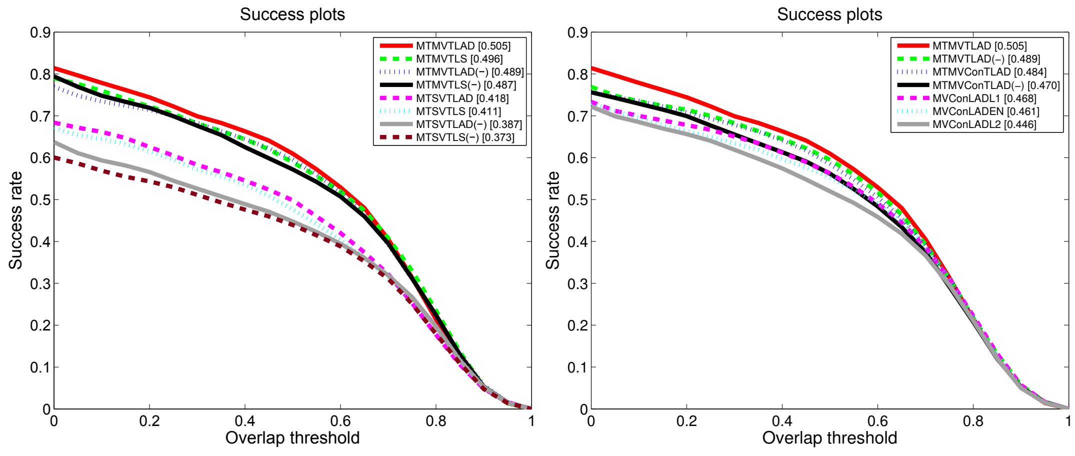


Figure 3.10: The success plots for BC and DEF subsets of CVPR2013 tracking benchmark. The value appearing in the title is the number of sequences in the specific subset. The values appearing in the legend are the AUC scores. Only the top 10 trackers are presented, while the other trackers can be found in [181]. The trackers appearing in the legend are as follows: DFT [105] , LSK [119], CPF [147].

2013 and compared MTMVTLAD with them using the success plot. As shown in Fig. 3.11 (a), the multi-view based trackers significantly outperform the trackers based on the single view, which demonstrates the advantage of using complementary information. Also, comparing to the trackers without the outlier handling, the trackers which explicitly take into account outliers generally achieve better AUC scores, which suggests that outliers should be specifically considered during the multi-task representation learning. Last but not least, the LAD based trackers outperform the corresponding LS based trackers, which validates the robustness of the learned representation based on LAD criterion.

As discussed previously, directly concatenating multiple features into a long feature vector is not a good way to handle multiple features. To validate this point, we concatenate multiple features and implement a baseline tracker based on the formulation (3.5), where we let $K = 1$. We call it as MTMVConTLAD. Using the concatenated features, we also implement several variants including MTMVConTLAD(-), which is the same as MTMVConTLAD but removes the functional component $\mathbf{Q}$ (no outlier handling), and three trackers MVConLADL1, MVConLADL2, MVConLADEN, which use L1, L2, and Elastic Net



(a) (b)

Figure 3.11: The success plots of MTMVTLAD and its baseline variants on the CVPR2013 tracking benchmark. The values appearing in the legend are the AUC scores.

regularizers [201], respectively. It should be noted that all these variants can be easily implemented based on the method presented in Section 3.3.4, along with the soft thresholding [15] for L1 regularizer. We also ran these variants on the CVPR benchmark 2013 and compare MTMVTLAD with them using the success plot, which is shown in Fig. 3.11 (b). It shows MTMVConTLAD and MTMVConTLAD(-) outperform MVConLADL1, MVConLADL2, MVConLADEN, which validates expected improvements brought by considering all particles in a multi-task setting. Also, MTMVConTLAD does not perform as good as the proposed MTMVTLAD, which suggests that the multiple features should not be concatenated directly.

3.4.5 Evaluation on ALOV++ Dataset

Recently, Smeulders *et al.* have developed the Amsterdam Library of Ordinary Videos dataset [160], named ALOV++, which consists of 14 challenge subsets, 315 sequences of which focuses on systematically and experimentally evaluating trackers' robustnesses in a large variety of situations including light changes, low contrast, occlusion, etc. In [160], survival curves based on F-score were proposed to evaluate trackers' robustnesses. To obtain the survival curve of a tracker, a F-score for each video is computed as $F = 2(\text{precision} \cdot \text{recall})/(\text{precision} + \text{recall})$, where $\text{precision} = n_{tp}/(n_{tp} + n_{fp})$, $\text{recall} = n_{tp}/(n_{tp} + n_{fn})$, and n_{tp}, n_{fp}, n_{fn} denote the number of true positives (overlap $S_o >= 0.5$), false positives and false negatives in a video. A survival curve shows the performance of a trackers on all videos in the dataset. The videos are sorted according to the F-score. By sorting the videos, the graph gives a comparative view in cumulative rendition of the quality of the tracker on the whole dataset. We refer the reader to the original paper [160] and the author's website[1] for details about the dataset and the evaluation tools.

To evaluate the proposed MTMVTLAD tracker on ALOV++ dataset, we downloaded the videos and ground truth data from the websitefootnote 1, and ran MTMVTLAD on all of the 315 sequences using the ground truth of the first frame as initialization. We compare our tracker with 19 popular trackers[2] evaluated in [160]. We show in Fig. 3.12 the survival curves of the top ten trackers and the average F-scores over all sequences. As shown in Fig. 3.12, the average F-score of MTMVTLAD in ALOV++ dataset is 0.67, which is better than Struck [68] with 0.66 and is also much better than 0.62 of VTS [99], another multi-view based tracker. In ALOV++ dataset, MTMVTLAD achieves the best overall performance over 20 compared trackers using the evaluation metric of average F-score. For better understanding of the overall performance of the proposed tracker, we also report the respective average F-scores of MTMVTLAD in 14 ALOV++ challenge subsets in Fig. 3.13.

3.4.6 Discussion

The experimental results demonstrate robust tracking performance of our approach. However, our tracker can indeed fail in some scenarios, which are shown in Fig. 3.14. Our tracker can fail when the objects undergo very large pose transformation caused by rotation or scale changes. For example, Fig. 3.14 (a) shows two failure cases of MTMVTLAD on *Skiing* and *MotorRolling* sequences

[1]http://imagelab.ing.unimore.it/dsm/

[2]Please refer to [160] and the references within for the details about the compared trackers. The evaluation results of these trackers were obtained from the authors of [160].

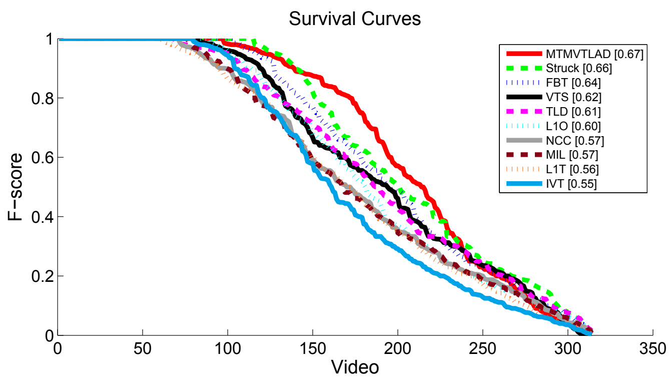


Figure 3.12: The survival curves for top ten trackers in ALOV++ dataset. The average F-scores over all sequences are specified in the legend. The trackers appearing in the legend are as follows: Struck [68], FBT [141], VTS [99], TLD [86], L1O [135], NCC [31], MIL [9], L1T [133], IVT [150]

of CVPR2013 benchmark where MTMVTLAD loses the targets when the targets undergo rotations and/or change their appearance and scale. Another failure case of MTMVTLAD is on the *LongDuration* subset of ALOV++ dataset. On this subset, the trackers run on 10 long sequences where some of targets may move completely out of the frame and then reappear. MTMVTLAD does not perform well and obtains a low F-score on this subset as shown in Fig. 3.14 (b). It is possible that the tracker locks on an irrelevant patch when the target is fully occluded. We expect our future investigation to address this failure mode of the proposed tracker.

3.5 Conclusion

In this chapter, we have presented a LAD-based robust multi-task multi-view sparse learning method for particle filter-based tracking. By appropriately introducing the $l_{1,2}$ norm regularization, the method not only exploits the underlying relationship shared by different views and different particles, but also captures the frequently emerging outlier tasks which have been previously ignored. The proposed regularized LAD problem is effectively approximated by Nesterov's smoothing method and efficiently solved by the APG. We implemented our method using

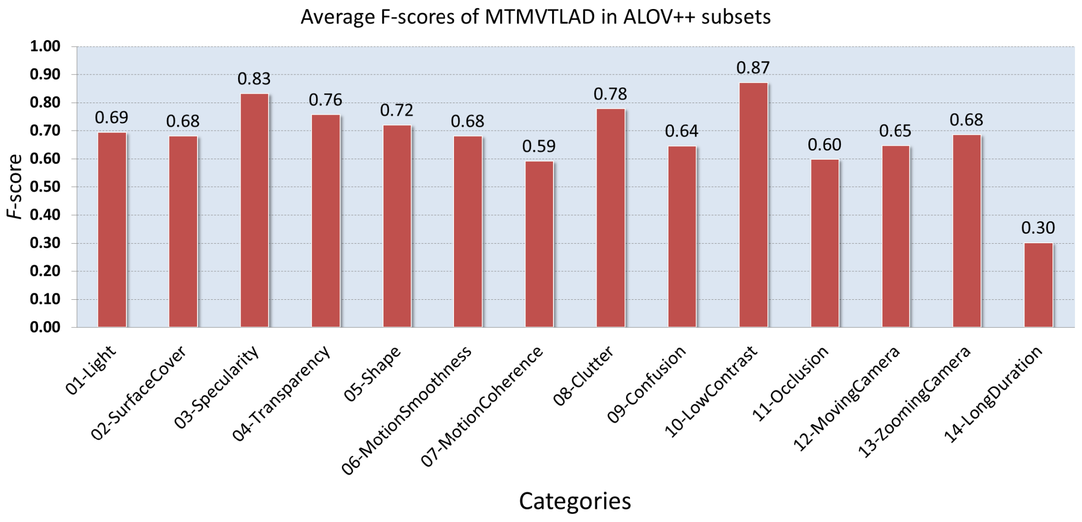


Figure 3.13: The respective average F-scores of the proposed MTMVTLAD tracker in 14 ALOV++ challenge subsets.

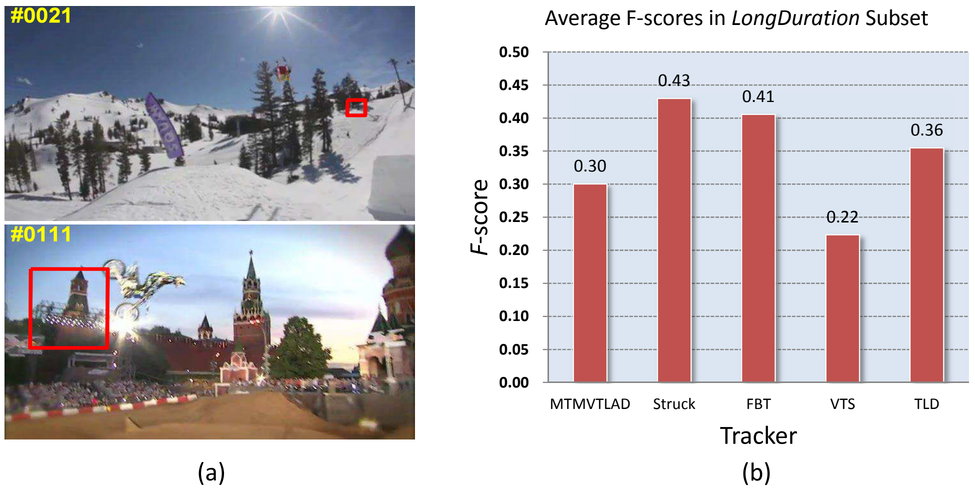


Figure 3.14: Failure cases of MTMVTLAD. (a) Failure cases on *Skiing* and *MotorRolling* sequences of CVPR2013 benchmark. (b) Failure case in the *LongDuration* subset of ALOV++ dataset. The numbers appear on the top of each bar is the tracker's average F-score over 10 sequences of the *LongDuration* subset.

four types of complementary features, i.e. intensity, color histogram, HOG and LBP, and extensively tested it on numerous challenging sequences including publicly available sequences, synthetic noisy sequences, real-world noisy sequences and two comprehensive tracking datasets. The experimental results demonstrate that the proposed method is capable of taking advantage of multi-view data and correctly handling the outlier tasks. Compared to several popular trackers, our tracker demonstrates superior performance. Moreover, the proposed method can potentially be extended to handle data obtained from sensors other than cameras.

Chapter 4

Non-rigid Object Tracking using Multilevel Quantizations

Most object tracking methods only exploit a single quantization of an image space: pixels, superpixels, or bounding boxes, each of which has advantages and disadvantages. It is highly unlikely that a common optimal quantization level, suitable for tracking all objects in all environments, exists. We therefore propose a hierarchical appearance representation model for tracking, based on a graphical model that exploits shared information across multiple quantization levels. The tracker aims to find the most possible position of the target by jointly classifying the pixels and superpixels and obtaining the best configuration across all levels. The motion of the bounding box is taken into consideration, while Online Random Forests are used to provide pixel- and superpixel-level quantizations and progressively updated on-the-fly. By appropriately considering the multilevel quantizations, our tracker exhibits not only excellent performance in non-rigid object deformation handling, but also its robustness to occlusions. A quantitative evaluation is conducted on two benchmark datasets: a non-rigid object tracking dataset (11 sequences) and the CVPR2013 Online Object Tracking Benchmark (OOTB). Experimental results show that our tracker overcomes various tracking challenges and is superior to a number of other popular tracking methods.

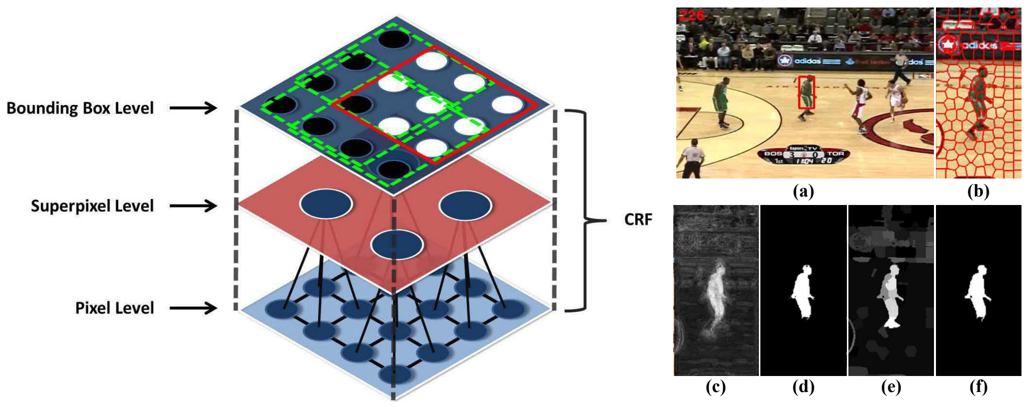


Figure 4.1: Illustration of the structure of the proposed hierarchical appearance representation model (left) and a practical example (right). In the proposed framework, a node in the Conditional Random Field (CRF) models each pixel, superpixel, and bounding box. At the pixel level, each pixel receives a measurement from a Random Forest and connects to the corresponding superpixel at the middle level. At the superpixel level, each superpixel also obtains a probability output by another Random Forest and suggests the pixels within the same superpixel to share the same label. At the bounding box level, different candidate bounding boxes (green) are considered, and the best position (red) with the best configuration is found. (a) shows the tracking result (in red bounding box) at Frame #226 in the *Basketball* sequence. (b) displays the superpixelization of the image. (c) and (d) are the output of the pixel-level RF and final labeling result, respectively, while (e) and (f) are the output of the superpixel-level RF and final labeling result.

4.1 Introduction

Online object tracking is a classic topic in computer vision and is used in many practical applications, such as video surveillance and autonomous driving. Given the position of the target in one frame, a tracker should be able to track the target in subsequent frames and be able to overcome various challenges, such as appearance variations, occlusions and illumination changes. Building an effective tracker is therefore extremely difficult, especially without prior knowledge of the appearance of the object to be tracked. However, a number of trackers have been proposed and show promising results [115, 181, 185].

Many tracking methods operate by making a single quantization choice in an image space, i.e., using pixels [50], superpixels [176], or bounding boxes [150], each of which has its pros and cons. For example, a tracker built on pixel-level quantization may be able to capture and thus better handle non-rigid deformation, but performs relatively poorly in scenarios where there is excessive background clutter due to the lack of holistic appearance of the target. In contrast, trackers that utilize higher-level quantization, such as bounding box-based matching, are robust to occlusions but tend to fail when the target undergoes non-rigid deformation. Therefore, a single optimal quantization level suitable for all objects in all environments is unlikely to exist.

Motivated by the above observation, in this chapter, we propose a novel hierarchical appearance representation formulation of object tracking based on Conditional Random Fields (CRFs), which unifies multiple disparate quantizations of the image space. Based on the information derived from different quantization levels (pixel, superpixel, bounding box), we integrate them into a principled framework to optimize the decision-making. At the lowest level, an Online Random Forest (ORF) [153] equipped with color-texture features is employed to provide a soft label of each pixel, which indicates the probability that the pixel belongs to the target. At the middle level, superpixels are generated by considering various cues such as the spatial relationship and feature similarity between pixels, which suggests a consistent pixel labeling within each superpixel. Besides, another ORF based on normalized histogram features of superpixels is also trained on the mid-level quantization. At the highest level, a bounding box-level regularization term is introduced, which enables to flexibly incorporate other information of a given bounding box, such as shape and motion, or even the measurement given by other trackers. The model bridges the hierarchical appearance representations by fusing multilevel quantization information and efficiently solves the optimization with the use of dynamic graph cuts [93]. However, the contribution of this chapter is not limited to the application of the novel hierarchical appearance representation framework to object tracking. We also address appearance variations by exploiting color-texture features and powerful, yet efficient, ORFs. These ORFs are strategically updated in the framework to capture appearance changes due to deformation or illumination over time. The proposed method is illustrated in Fig. 4.1.

4.2 Related Work

There has been recent progress in visual object tracking research, with several ideas that focus on various challenges being proposed; these are extensively discussed in elsewhere [115, 185]. In addition, standard benchmark datasets and quantitative evaluation metrics have been developed [59,181] to facilitate research in this area.

Some existing approaches use the pixel level of the image space to explore low-level cues for tracking. For instance, Avidan [6] proposed an ensemble tracker to track the object based on the result of the pixel-level classification. Although the discriminative setting enables the tracker to distinguish foreground from background, the pixel-based representation still limits robustness to a cluttered background and heavy occlusion [176]. More recently, Duffner and Garcia proposed the PixelTrack [50], which also addresses tracking at the pixel-level. The tracker works by combining a Hough voting-based detector with a soft segmentation approach similar to [3]. Although an efficient implementation is achieved, the tracker appears to be sensitive to the initialization and is prone to fail in grayscale sequences due to the dependence on the segmentation performance and the lack of global information.

Compared to pixel-level representations, mid-level visual cues provide more information about the local structure of images, while retaining the flexibility to model non-rigid deformation [2, 38, 97, 176]. In particular, Adam *et al.* [2] employed an appearance model and used local patches to handle partial occlusion. Superpixel tracking [176] aims to explore the mid-level cues and use the superpixel as the object representation. The normalized color histogram of each superpixel is extracted, and a confidence map is obtained by the superpixel-level, rather than pixel-level, classification [6]. In [38], the target is represented by a set of different regions. The regions are modeled by a Gaussian mixture model in a joint feature-spatial space, and the motion of the target is modeled by the level set evolution.

Many trackers are built to exploit high-level visual information using holistic appearance models [9, 74, 133, 150]. In [150], Ross *et al.* presented a tracking method that incrementally learns a low-dimensional subspace representation of the target. The L1 tracker, proposed by Mei *et al.* [133], and its variations [78, 113, 135, 195] appear to be robust to the illumination changes and occlusions but sensitive to non-rigid deformation. Babenko *et al.* [9] employ Multiple Instance Learning (MIL) to overcome the label ambiguity problem, in which the training samples are collected as bag of image patches. Random Forests (RFs) [30] have become increasingly popular in computer vision due to their attractive properties, and have been used in tracking [88, 153, 154]. Specifically, Saffari *et. al* [153] proposed an online version of RFs, which grows extremely randomized trees online, rather than offline. The Online RFs (ORFs) are then adopted for tracking by utilizing the features captured at the bounding box level, and this method has demonstrated better performance over the online boosting [62].

Multilevel data fusing has been exploited for image segmentation and labeling using Conditional Random Fields (CRFs) or Markov Random Fields (MRFs) [72, 81, 102, 173, 178]. In [102], a single optimization framework is presented in which a hierarchical random field model allows integration of features computed at different levels of the quantization hierarchy. In [72], labeling information from local image statistics, regional label features, and global label features are combined in order to label images with a predefined set of class labels. In [81], a hierarchical two-stage CRF model is used to combine parametric and nonparametric image labeling methods. In [103, 178], integration of object detection and pixel-wise scene labeling boosts the performance of both tasks, since they are mutually beneficial. Also, previous works have addressed tracking problems by combining multilevel information [92, 172, 198]. For example, in [198], a collaborative model is proposed to combine a Sparsity-based Discriminative Classifier (SDC) with Sparsity based Generative Model (SGM), which collaboratively considers holistic object templates and local image patches for target representations. Motivated by these advances, here we propose the Multilevel Quantization Tracker (MQT), which explores the quantization hierarchy from coarse to fine and unifies the information derived from multiple quantization levels in a coherent CRF framework. In this way, each quantization level benefits from other levels and, as a consequence, the overall performance of each individual level is enhanced.

4.3 Tracking with Multilevel Quantizations

The proposed tracker combines multilevel quantizations as a single graphical model to produce an efficient and robust solution to online object tracking. We first introduce the general multilevel quantization model and then describe other important components of the tracker, including feature extraction, online color-texture forests, ORF training, and occlusion handling.

4.3.1 Multilevel Quantizations Model

The whole model is built on multiple quantizations from three hierarchical appearance representation levels, namely pixels, superpixels and bounding boxes. We first extract information at each level before fusing them using a graphical model so as to perform inference.

Pixel is the finest quantization level in an image, and is the most obvious choice for quantization. Let each pixel $i \in \mathcal{P}$ ($\mathcal{P}$ denotes the set of pixels) be represented as a d-dimensional feature vector $\mathbf{f}_i \in \mathbb{R}^d$ that consists of some local information, and associated with a unique binary label $x_i \in \{0$ (background), 1 (foreground/object)$\}$. The pixel-level unary energy function is defined as:

$$\phi_i^{\mathrm{p}}(x_i) = -\log p(x_i; \mathbf{H}^{\mathrm{p}}), \tag{4.1}$$

where $p(x_i; \mathbf{H}^{\mathrm{p}})$ denotes the probability that pixel i is labeled as class x_i, output by an ORF with parameters $\mathbf{H}^{\mathrm{p}}$, which are updated online (Section 4.3.3). An example of $p(x_i; \mathbf{H}^{\mathrm{p}})$ output by an ORF is shown in Fig. 4.1(c).

Superpixels provide very useful mid-level support for image understanding tasks (e.g., [1, 111]). In order to exploit mid-level information, we employ the SLIC (Simple Linear Iterative Clustering) algorithm [1] to cluster the pixels and generate superpixels as shown in Fig. 4.1(b). For each superpixel $k \in \mathcal{S}$ ($\mathcal{S}$ denotes the set of superpixels), we also assign a binary label $y_k \in \{0, 1\}$, which is similar to x_i at pixel level. Again, an ORF is trained to output the probability that the superpixel belongs to the foreground or background, using the features extracted from each superpixel (Fig. 4.1(e)). Similarly, superpixel-level energy function is defined as:

$$\phi_k^{\mathrm{s}}(y_k) = -\log p(y_k; \mathbf{H}^{\mathrm{s}}), \tag{4.2}$$

where the symbols are analogous to those in (4.1).

At the highest level, like many existing online trackers [68, 150], we use a bounding box to delimit the object of interest. Let $\mathbf{B}(z)$ denote the bounding box with pose parameters z and energy function $\varphi(\mathbf{B}(z))$ encode the occurrence likelihood of the target in bounding box $\mathbf{B}(z)$. In contrast to other online trackers (e.g., [74,133,150]) which optimize merely $\varphi(\mathbf{B}(z))$ to get the tracking solution, we unify $\varphi(\mathbf{B}(z))$ with information from the other quantization levels, as explained above. The choice of $\varphi(\mathbf{B}(z))$ is modular and it can vary from simple matching techniques [32] to sophisticated classification models [68].

In our experiments, we use the Median Flow Tracker (MFT) [87] for the bounding box level quantization. MFT uses the feature matching to estimate the motion of target. Moreover, it measures the discrepancies of the forward and backward tracking in consecutive frames and reports failure when the target is lost [88]. We assign 0 to the tracking result z^M if failure is detected. The bounding box energy function $\varphi(\mathbf{B}(z))$ is defined as:

$$\varphi(\mathbf{B}(z)) = \begin{cases} 0 & , \quad z^M = 0 \\ D^2(\mathbf{B}(z), \mathbf{B}(z^M)), & \text{otherwise} \end{cases} \tag{4.3}$$

where $D(\mathbf{B}(z), \mathbf{B}(z^M))$ is the distance between the centers of two bounding boxes $\mathbf{B}(z)$ and $\mathbf{B}(z^M)$ (the results of MFT) in the image.

Given the above three levels, we adopt a Conditional Random Field (CRF) model to fuse the information from different levels. Each unit (pixel, superpixel, bounding box) at different levels is represented by a node in the graph, and use corresponding unary potential functions to encode those terms in (4.1), (4.2), and (4.3). Then we capture the interactions between these nodes via connecting them using CRF's edges with appropriate potential functions.

Firstly, we associate an edge between a pair of neighboring pixel nodes (4-neighborhood system is considered in the experiment) and the following potential function to encode the interaction between the labeling of the pixels:

$$\psi_{i,j}(x_i, x_j) = \begin{cases} \exp(-\frac{\|\mathbf{f}_i - \mathbf{f}_j\|^2}{\sigma^2}), & \text{if } \ x_i \neq x_j \\ 0 \qquad\qquad\qquad , & \text{otherwise} \end{cases} \tag{4.4}$$

where $\|\mathbf{f}_i - \mathbf{f}_j\|$ is the distance between x_i and x_j in the feature space, and σ is a parameter controlling the shape of the monotonically decreasing function, which is similar to [27]. We use $\mathcal{E}^{\text{pp}}$ to denote all such edges between neighboring pixels.

One important fact regarding the pixel- and superpixel-level quantizations is that the pixels in the same superpixel tend to share the same superpixel label. Hence, for each pixel i in superpixel k, we associate an edge using the Potts model as its potential function:

$$\xi_{i,k}(x_i, y_k) = \begin{cases} 1, & \text{if } \ x_i \neq y_k \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

which penalizes the inconsistency in labeling between superpixels and pixels. We use $\mathcal{E}^{\text{sp}}$ to denote all such edges.

We also connect all pixel nodes with the bounding box node. The pairwise potential function $w_i(z, x_i)$ is used to encourage consistency between pixel labeling and the pose of the bounding box:

$$w_i(z, x_i) = \begin{cases} d(z, i), & \text{if } (x_i = 1,\, i \in \mathcal{P}^{(\text{Out})}_{\mathbf{B}(z)}) \text{ or } (x_i = 0,\, i \in \mathcal{P}^{(\text{In})}_{\mathbf{B}(z)}) \\ 0 \qquad\ , & \text{otherwise} \end{cases} \tag{4.6}$$

where $d(z, i)$ represents the minimum normalized distance (which considers the size of bounding box and is detailed in Section 4.4.1) between the pixel i to the boundary of the bounding box $\mathbf{B}(z)$; $\mathcal{P}^{(\text{In})}_{\mathbf{B}(z)}$ and $\mathcal{P}^{(\text{Out})}_{\mathbf{B}(z)}$ denote the set of pixels inside and outside the bounding boxes, respectively. The choice of function is based on the observation that the pixels inside the bounding box tend to belong to the object, while the pixels outside the bounding box tend to belong to the background. Moreover, the closer the pixel is to the boundary, the more ambiguous the pixel label is. The pixel is penalized for having different label from what is expected, using a cost that is proportional to the distance between the pixel and the boundary of the bounding box, which is similar to the idea in [172].

Finally, given an image I, the joint probability of the realization $(z, \mathbf{x}, \mathbf{y}) = (z, \mathbf{x} = (x_i)_{i \in \mathcal{P}}, \mathbf{y} = (y_k)_{k \in \mathcal{S}})$ of all random variables in the CRF model is formulated as a Gibbs distribution $P(z, \mathbf{x}, \mathbf{y}|I) = e^{-E(z,\mathbf{x},\mathbf{y})}$. The corresponding Gibbs energy $E(z, \mathbf{x}, \mathbf{y})$ is defined as the sum of the above unary potentials and pairwise potentials:

$$\begin{aligned} E(z, \mathbf{x}, \mathbf{y}) =& \mu\varphi(\mathbf{B}(z)) + \sum_{i \in \mathcal{P}} \phi_i^{\mathrm{p}}(x_i) + \alpha \sum_{k \in \mathcal{S}} \phi_k^{\mathrm{s}}(y_k) + \lambda \sum_{i \in \mathcal{P}} \omega_i(x_i, z) \\ &+ \beta \sum_{\{i,k\} \in \mathcal{E}^{\mathrm{sp}}} \xi_{i,k}(x_i, y_k) + \gamma \sum_{\{i,j\} \in \mathcal{E}^{\mathrm{pp}}} \psi_{i,j}(x_i, x_j), \end{aligned} \tag{4.7}$$

where μ, α, λ, β, γ are the weight coefficients which balance the importance of each potential term.

In the tracking problem, we aim to determine the optimal pose parameters z for the bounding box. Since the minimization of $E(z, \mathbf{x}, \mathbf{y})$ with respect to $\mathbf{x}$ and $\mathbf{y}$ can be efficiently solved using the well-known graph cuts [28] for each possible z, we define an auxiliary function $\hat{E}(z)$ and search for the optimal z^* for $\hat{E}(z)$ using any off-the-shelf optimization algorithm:

$$z^* = \operatorname*{argmin}_z \{\hat{E}(z) = \min_{\mathbf{x} \in \{0,1\}^{|\mathcal{P}|}, \mathbf{y} \in \{0,1\}^{|\mathcal{S}|}} E(z, \mathbf{x}, \mathbf{y})\}. \tag{4.8}$$

For example, one can use the local dense sampling search as done in [9, 68]. In this chapter, we simply adopt the Nelder-Mead Simplex Method [104] to directly search for the solution. Note that during the search of z in the problem (4.8), the update of z only causes small change[1] in ω_i, which motivates the use of dynamic MRF algorithms [92] (e.g., dynamic graph cuts [93]) to efficiently obtain the value of $\hat{E}(z)$ and significantly accelerate the optimization.

4.3.2 Online Color-Texture Forests

The selection of features and an appropriate online learning process has been shown to be very important for tracker performance [6, 77, 150, 153]. In this section, we elaborate on online color-texture forests, which are used to obtain

[1]$\mu\varphi(\mathbf{B}(z))$ would change but would not affect the optimum of $E(z, \mathbf{x}, \mathbf{y})$ with respect to $\mathbf{x}$ and $\mathbf{y}$.

pixel- and superpixel-level potentials in (4.1) and (4.2).

The color feature is one of the most widely used visual features in tracking. The most important advantages of color feature are power of representing visual content of images, simplicity in extracting color information of images and high efficiency, independent of image size and orientation. However, only using color feature is difficult to tackle many real-world tracking scenarios, such as distractive background clutter and drastic illumination change. We combine texture with color as a complementary feature for tracking to better represent object appearance. For each pixel, we extract RGB (3-dim), CIELAB (3-dim) and texture features (48-dim) as the pixel-level representation with 54 dimensions. The texture feature is generated by the Leung-Malik (LM) Filter Bank [110], which consists of the first and second derivatives of Gaussians at 6 orientations and 3 scales, 8 Laplacian of Gaussian (LOG) filters, and 4 Gaussian filters. With respect to the superpixel level, we utilize normalized histogram-based features to capture the photometric properties of each superpixel, similar to [176].We extract a 64-bin normalized histogram in the HSV color space and a 10-bin normalized histogram based on uniform rotation-invariant local binary patterns (LBPs) [142], to form a 74 dimensional color-texture feature for each superpixel.

Random forests consist of a set of randomized decision trees. In each decision tree, an internal node corresponds to a random test on an input feature, which determines to which child node the feature should go. Therefore, a feature vector is presented to the root of a tree and it follows a specific path to a leaf node, which stores a histogram (occurrence frequency of each class) obtained during the training phase. Given a test sample $\mathbf{f}$, the probability is estimated by averaging the probabilities of all the trees:

$$p(\text{class} = c|\mathbf{f}) = \frac{1}{N}\sum_{n=1}^{N} p_n(\text{class} = c|\mathbf{f}),$$

where N denotes the number of the trees, and $p_n(\text{class} = c|\mathbf{f})$ is the probability that the feature belongs to class c output by the tree n.

RFs have demonstrated great promise in various computer vision tasks including object recognition [109] and image classification [26]. We adopt the Online

Algorithm 3 Tracking with Multilevel Quantizations

Input: The target bounding box $\mathbf{B}(z_1^*)$ in the first frame; T frames to track.
Output: Estimated target position $\mathbf{B}(z_t^*)$, where $t = 2, 3..., T$ is the frame index.

```
 1: /*Initialization*/
 2: Apply Grabcut [151] to find the positive and negative samples.
 3: Train pixel- and superpixel-level RFs using the collected samples.
 4: /*Start to track*/
 5: for $t = 2$ to $T$ do
 6:     /*Pixel level*/
 7:     Extract features for each pixel $i$ and obtain the pixel-level measurement
        $p(x_i; \mathbf{H}^{\mathrm{p}})$.
 8:     /*Superpixel level*/
 9:     Apply SLIC [1] to generate superpixels.
10:     Extract features for each superpixel $k$ and obtain the superpixel-level mea-
        surement $p(y_k; \mathbf{H}^{\mathrm{s}})$.
11:     /*Bounding box level and combine multilevel quantizations*/
12:     Estimate the motion of target using MFT [87] and obtain $\mathbf{B}(z_t^M)$.
13:     Find the target $\mathbf{B}(z_t^*)$ by solving (4.8) using [104] with dynamic graph
        cuts [93].
14:     if not occluded then
15:         Update $\mathbf{H}^{\mathrm{p}}$ of the pixel-level RF using $\mathcal{X}_p^+$, $\mathcal{X}_p^-$.
16:         Update $\mathbf{H}^{\mathrm{s}}$ of the superpixel-level RF using $\mathcal{X}_{sp}^+$, $\mathcal{X}_{sp}^-$.
17:     end if
18: end for
```

Random Forests [153] to incorporate the high-dimensional color-texture feature for our online tracking. The resulting online color-texture forest turns out to provide very discriminative classification results for our potential functions.

4.3.3 ORF Training and Occlusion Handling

To train the two RFs for pixels and superpixels, a key issue is how to get positive and negative samples for training. In the first frame, given the target bounding box, Grabcut [151] is adopted to automatically determine the pixels corresponding to the object which are then used as positive examples for training the RF for pixels. This generally improves the accuracy compared to treating all pixel inside the bounding box as foreground, since the object may not occupy the whole bounding box due to its shape. To deal with cases that object is not well segmented by Grabcut, we check the percentage of pixels with foreground labels in the bounding box. If it is greater than 70%, the result of Grabcut is accepted, otherwise it is rejected and all the pixels inside the bounding box are used as the positive samples. On the other hand, all the pixels outside the bounding box are used as negative samples. For superpixels, they are labeled using a voting scheme, i.e., the label of the superpixel is decided by the majority of the pixels inside it.

During the tracking, the ORFs are progressively updated to handle the appearance changes. Since pixels and superpixels are labeled in the whole formulation by jointly exploiting the information from multiple levels during the tracking, we only treat the pixels and superpixels as candidate positive samples if they are inside the target bounding box $\mathbf{B}(z^*)$ and labeled as positive by our tracker using (4.8). The pixels and superpixels outside the bounding box are treated as candidate negative samples. Moreover, only the candidate samples that are not classified with a high confidence or incorrectly classified by their respective RFs are assigned to RFs for updates. A similar strategy is employed in [88]. More specifically, the final positive sample set $\mathcal{X}_p^+$ and negative sample set $\mathcal{X}_p^-$ for the pixel-level RF update are respectively determined as:

$$\mathcal{X}_p^+ = \{i | x_i = 1, p(x_i = 1; \mathbf{H}^{\mathrm{p}}) < \varepsilon_p^+, i \in \mathcal{P}_{\mathbf{B}(z^*)}^{\mathrm{In}}\}, \tag{4.9}$$

$$\mathcal{X}_p^- = \{i | p(x_i = 1; \mathbf{H}^{\mathrm{p}}) > \varepsilon_p^-, i \in \mathcal{P}_{\mathbf{B}(z^*)}^{\mathrm{Out}}\}, \tag{4.10}$$

where ε_p^+, ε_p^- (and ε_{sp}^+, ε_{sp}^- below) are the predefined thresholds. For the superpixel-level RF, the positive sample set $\mathcal{X}_{sp}^+$ and negative sample set $\mathcal{X}_{sp}^-$ are similarly determined as

$$\mathcal{X}_{sp}^+ = \{k | y_k = 1, p(y_k = 1; \mathbf{H}^{\mathrm{s}}) < \varepsilon_{sp}^+, k \in \mathcal{S}_{\mathbf{B}(z^*)}^{\mathrm{In}}\} , \tag{4.11}$$

$$\mathcal{X}_{sp}^- = \{k | p(y_k = 1; \mathbf{H}^{\mathrm{s}}) > \varepsilon_{sp}^-, k \in \mathcal{S}_{\mathbf{B}(z^*)}^{\mathrm{Out}}\} , \tag{4.12}$$

where $\mathcal{S}_{\mathbf{B}(z^*)}^{\mathrm{in}}$ and $\mathcal{S}_{\mathbf{B}(z^*)}^{\mathrm{Out}}$ denote the set of superpixels inside and outside the bound-

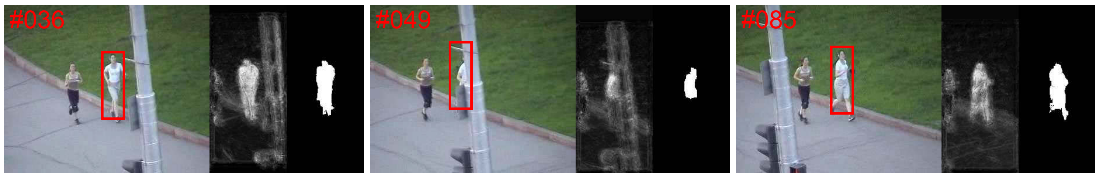


Figure 4.2: Occlusion handling on the *Jogging* sequence. The index is specified in the top-left of each frame, and the two figures between each frame are the corresponding outputs of the pixel-level RFs and labels x_i, respectively. The occlusion is detected from the Frame #049, from which point the RFs stop updating until the target moves out of occlusion.

ing box $\mathbf{B}(z^*)$, respectively. Noted that in (4.11) and (4.12), the voting scheme previously presented is still used to determine whether a superpixel is inside or outside the bounding box.

As discussed in previous works [6,176], it is also important to take occlusions into account during updates, especially when the target is temporarily out-of-view. The pixel labeling provided by our approach also can be used to handle occlusions: a flag of occlusion is trigged if the percentage of foreground pixels inside the found bounding box is less than a predefined threshold θ (0.3). In this case, the RFs are kept unchanged without update. An example of the occlusion handling is shown in Fig. 4.2. Finally, a systematic view of the whole algorithm is summarized in Algorithm 3.

4.4 Experiments

In this section, we first present implementation details about important aspects of MQT, including the parameter setting for evaluation. We then present a set of qualitative experimental results as well as quantitative comparison with several state-of-the-art trackers on two benchmarks.

4.4.1 Implementation Details

Similar to [6,176], the optimal tracking result at each frame is achieved by searching in a region centered around the target bounding box determined from the

Table 4.1: Non-rigid object tracking: percentage of correctly tracked frames.

	HT [59]	TLD [88]	PixelTrack [50]	SPT [176]	**MQT**	**P**	**S**	**B**	**P&S**	**P&B**	**S&B**
Cliff-dive 1	**100.00**	69.12	**100.00**	**100.00**	**100.00**	**100.00**	97.06	**100.00**	**100.00**	**100.00**	**100.00**
Motocross 1	**100.00**	15.38	57.69	29.49	43.59	30.13	30.13	40.38	44.87	66.67	35.26
Skiing	**100.00**	6.85	**100.00**	17.28	**100.00**	**100.00**	7.41	9.88	98.77	98.77	9.88
Mountain-bike	**100.00**	81.36	94.55	**100.00**	**100.00**	**100.00**	4.39	39.04	**100.00**	**100.00**	18.86
Cliff-dive 2	**100.00**	8.20	32.79	**100.00**	**100.00**	**100.00**	78.69	50.82	**100.00**	**100.00**	62.30
Volleyball	45.12	42.28	**100.00**	46.55	**100.00**	**100.00**	28.46	25.00	60.16	**100.00**	28.66
Motocross 2	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Transformer	38.71	33.06	94.35	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Diving	21.21	24.68	88.74	41.56	**97.84**	13.42	35.06	24.24	96.54	13.85	55.84
High Jump	77.87	35.25	94.26	19.67	**98.36**	84.43	8.20	8.20	90.16	91.80	41.80
Gymnastics	98.87	84.75	99.09	21.90	**100.00**	96.87	95.57	29.47	98.17	97.13	59.06
Average	80.16	45.54	87.41	61.50	**94.53**	84.08	53.18	47.91	89.88	88.02	55.60

Note: The left panel shows the comparison between different trackers. Right panel summaries the results of baseline performance, where P, S, B are the trackers using single quantization of pixel (RF), superpixel (RF), and bounding box (MFT), respectively, and P&S the tracker using the two quantizations from pixel and superpixel, etc.

previous frame, as illustrated in Fig. 4.1(b). As in [9, 68, 74], we use a bounding box with a fixed size during the tracking in the current implementation. In order to track objects with different resolutions using the same parameters, we resize the image and let the short side of the target bounding box in the first frame to have a length of 35 pixels. After tracking, the results of MQT are projected back to the original image for fair comparison. To obtain meaningful superpixels of appropriate size, the regularized size of SLIC [1] is set to 17. Regarding the parameters of the proposed model in (4.7), we set $\sigma = 0.1$, $\alpha = 5$, $\beta = 0.3$, $\lambda = 2$, $\gamma = 0.1$, and $\mu = \frac{w \times h}{100^2}$, where w and h are the width and the height of the target bounding box, respectively. The minimum normalized distance $d(z, i)$ in (4.6) is computed by measuring minimum distance between the pixel i and the boundary of bounding box $\mathbf{B}(z)$ in a resized coordinate system, in which the size of target bounding box becomes $w' = h' = 1$. The number of trees T is set to 15 for both pixel- and superpixel-level RFs. Other parameters specific to the sample selection in ORF model training are $\epsilon_p^+ = 0.8$, $\epsilon_p^- = 0.3$, $\epsilon_{sp}^+ = 0.5$, $\epsilon_{sp}^- = 0.5$. It should be noted that we strictly follow the protocols proposed in [50, 181] and fix all parameters for all video sequences in the following evaluations. We use the initial bounding box given by the dataset as the starting point for our tracker.

The tracker was implemented using Matlab & C++ without intensive program optimization. The average time cost for all testing sequences is 1.1s per frame on a cluster node (3.4GHz, 8 Cores, 32GB RAM, less than 19% CPU used), consisting of: feature extraction (0.13s), SLIC (0.10s), RF prediction (0.18s), RF update (0.38s), and dynamic graph cuts (0.30s). It should be noted that parallel programming can be easily adopted for some key components (e.g., feature extraction, graph cuts, and ORF) to significantly reduce the run-time.

4.4.2 Tracking non-rigid Objects

We first evaluate the performance of our tacker on the non-rigid object tracking dataset, which was first collected by [59] and was recently used to test a state-of-the-art tracking method [50]. This dataset consists of 11 sequences, where significant non-rigid deformation of objects is present. Quantitative results on a set of representative frames are shown in Fig. 4.3. For quantitative comparison, we compare MQT with a set of state-of-the-art methods[1], including PixelTrack [50], Hough Tracker (HT) [59], TLD [88], and the Superpixel Tracker (SPT) [176], by computing the success rate, defined as the percentage of frames in which the object is successfully tracked. In each frame, the overlap measure (i.e., half of the DICE coefficient) $S_o = \frac{|R_t \cap R_g|}{|R_t \cup R_g|}$ is computed, where R_t is the bounding box output by a tracker and R_g is the ground truth bounding box. The tracking is considered successful is S_o is larger than a given threshold t_o. For a fair comparison, we use the same protocol as in [50] by setting t_o to 0.1. The quantitative results of the comparative trackers are summarized in the left panel of Tab. 4.1.

As pointed out in [50], one of the most difficult videos for the PixelTrack is *Motocross 1*, where the motor-bike does a complete flip, changes its size rapidly, and the background is very cluttered. The rapid size change poses great challenge to our tracker given a fixed size bounding box we adopt. The quantitative evaluation shows that our tracker successfully tracks objects in almost all of the sequences in this dataset, and significantly outperforms the other four methods. Our tracker is demonstrated to be a promising method for tracking non-rigid objects, possessing the advantage of multilevel appearance representation incorporated in a graphical model, compared to the methods (e.g., PixelTrack, SPT, TLD) based on only a single level representation.

Moreover, to better understand the importance of different components in

[1]The performance of SPT is evaluated by using the code released by Wang *et al.* [176], and the data for the other three competitors' is from [50].

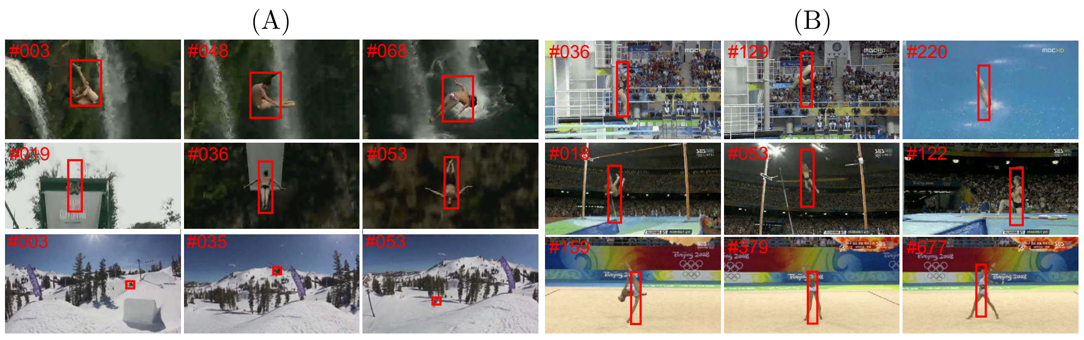


Figure 4.3: Qualitative results of MQT on the non-rigid object tracking dataset. Frame numbers are shown in the top left of each figure. Each column contains results of three sequences: (A) Cliff-dive 1, Cliff-dive 2, Mountain-bike; (B) Diving, High Jump, Gym.

the proposed framework, we also conducted the baseline experiments to evaluate performance on parts of our tracker by switching off some components and summarized the average performance in the right panel of Tab. 4.1.

4.4.3 Evaluation on CVPR2013 OOTB

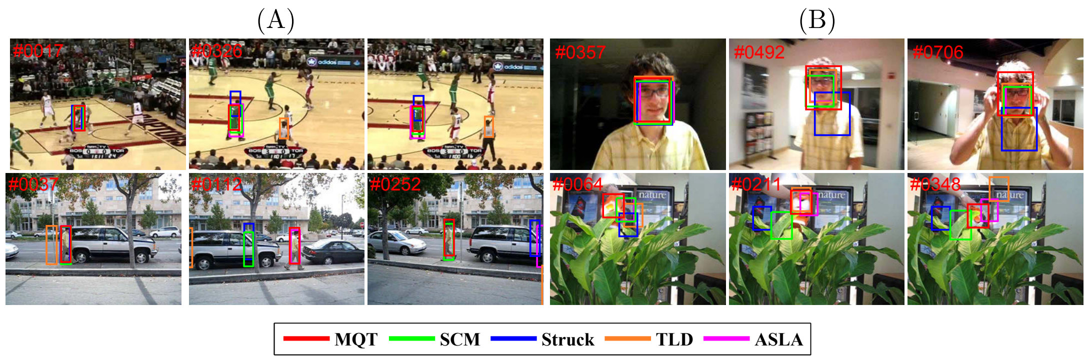


Figure 4.4: Qualitative results of MQT compared to different trackers on the CVPR2013 benchmark. Only top five trackers on success plots are presented. Frame numbers are shown in the top left of each figure. Each column contains results of two sequences: (A) Basketball, David3; (B) David, Tiger1.

The second experiment is conducted on the CVPR2013 tracking benchmark [181], which is an up-to-date comprehensive benchmark specifically designed for evaluation of tracking performance. The whole dataset consists of 50 fully annotated sequences. Each sequence is tagged with a number of attributes indicating to the presence of different challenges, e.g. Occlusion (OCC), Deformation (DEF). To evaluate the strength and weakness of different methods, the sequences are categorized according to those attributes and 11 challenge subsets are therefore created. In [181], the evaluation is based on two different metrics: the precision plot and success plot. The precision plot shows the percentage of frames on which the Center Location Error (CLE) of a tracker is within a given threshold r, where CLE is defined as the center distance between R_t and R_g, and a representative precision score ($r = 20$) is used for ranking. Another metric is to compute the bounding box overlap S_o introduced in the previous experiment (Section 4.4.2), and the success plot shows the ratios of successful frames at a given threshold t_o varied from 0 to 1. In success plot, the ranking is based on the area under curve (AUC) instead of using a specific threshold. For the comparative trackers, it currently includes 29 popular tracking algorithms, and most of them operated on a single choice of quantization. For more details about the benchmark, we refer readers to the original paper [181].

We run the One-Pass Evaluation (OPE) [181] on the benchmark using the proposed MQT. For comparison, we use the online available[1] tracking results and the unified tool provided by [181] to compute the evaluation plots. In this experiment, the proposed MQT achieves overall the best performance using both the metrics, which is shown in Fig. 4.5. MQT also ranks in the top ten from all 30 trackers over all challenge subsets using either the measurement of precision plots or success plots and takes the first places in the nine out of the eleven challenge subsets when using the success plots as measurement. According to the results, MQT is more robust to background clutter, deformation, occlusion challenges compared to the other 29 trackers. We show the success plots of the some challenge subsets in Fig. 4.6, but omit other figures due to the space limits. Finally, qualitative comparison with the top-rank trackers is shown in Fig. 4.4 for more intuitive demonstration. Note that, due to the adoption of a fixed-size bounding box and the lack of strong holistic-appearance model, the predicted bounding box will only partially capture the target in the presence of heavy

[1]http://visual-tracking.net/

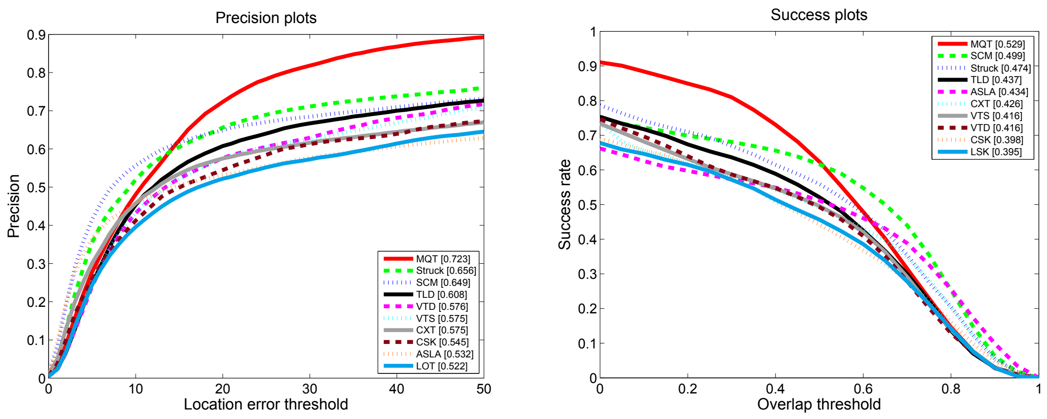


Figure 4.5: Quantitative comparison on CVPR2013 benchmark. The performance score for each tracker is shown in the legend. For each figure, only the top 10 trackers are presented. The trackers appearing in the legend are as follows: MQT (ours), Struck [68], SCM [198], TLD [88], VTD [98], VTS [99], CXT [45], CSK [74], ASLA [84], LOT [144], LSK [119].

occlusion and scale changes, which can be interpreted from Fig. 4.5 and Fig. 4.6.

4.5 Conclusions and Future Work

In this chapter, we propose a tracking method based on a hierarchical appearance representation using multilevel quantization. The different levels of the representation are incorporated into a Conditional Random Field model using a coherent framework. By exploiting all the quantization levels, the method utilizes and integrates the information contained at each representation level by explicitly modeling the interactions and constraints between them; this results in significantly improved performance compared to other state-of-the-art tracking methods based on a single quantization. Moreover, Online Random Forests are used to update the appearance model in different levels of the tracker, in order to capture changes in object appearance over time. The experimental results demonstrate that the proposed method is capable of taking advantage of multi-level information and significantly boosting tracking performance. In the future, we will improve our tracker by considering the scale change of the target and

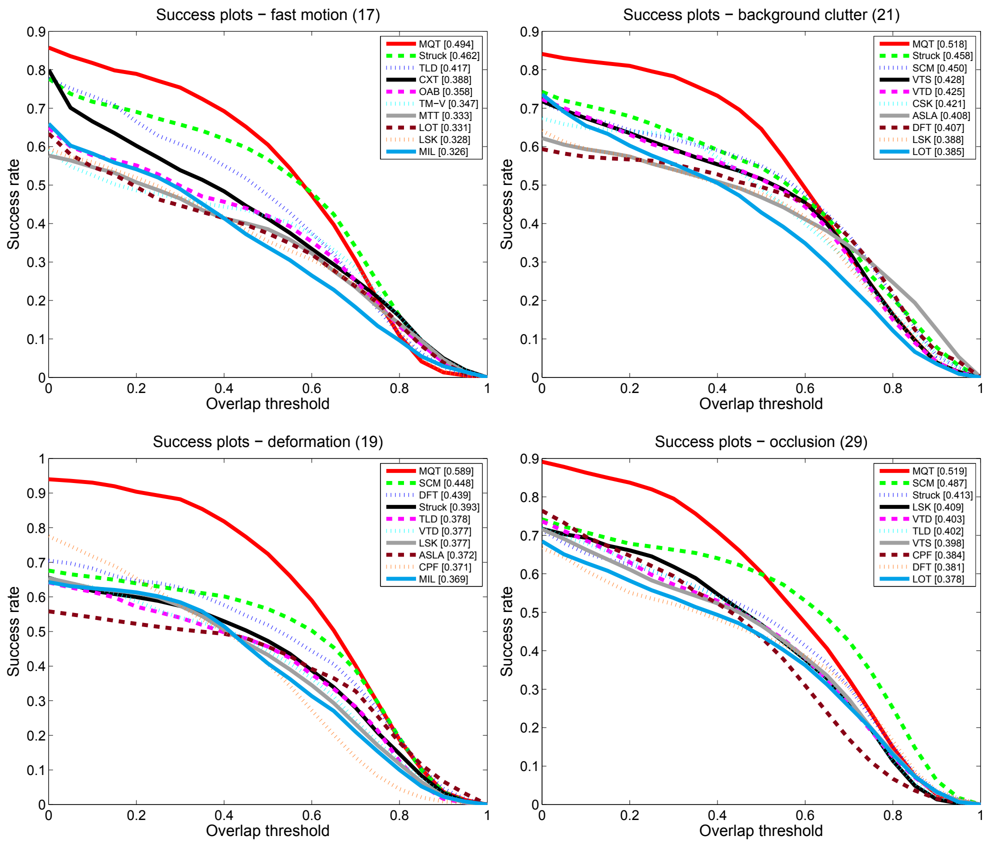


Figure 4.6: Success plots for some challenge subsets of CVPR2013 tracking benchmark. The performance score for each tracker is shown in the legend. The value appears in the title is the number of sequences in that subset. Only the top 10 trackers are presented. The trackers appearing in the legend are as follows: OAB [62], TM-V [39], DFT [105], CPF [147], MIL [9].

extend it by taking more sophisticated high-level information into consideration.

Chapter 5

Robust Tracking using Multi-store Memory Model

Variations in the appearance of a tracked object, such as changes in geometry/photometry, camera viewpoint, illumination, or partial occlusion, pose a major challenge to object tracking. Here, we adopt cognitive psychology principles to design a flexible representation that can adapt to changes in object appearance during tracking. Inspired by the well-known Atkinson-Shiffrin Memory Model, we propose MUlti-Store Tracker (MUSTer), a dual-component approach consisting of short- and long-term memory stores to process target appearance memories. A powerful and efficient Integrated Correlation Filter (ICF) is employed in the short-term store for short-term tracking. The integrated long-term component, which is based on keypoint matching-tracking and RANSAC estimation, can interact with the long-term memory and provide additional information for output control. MUSTer was extensively evaluated on the CVPR2013 Online Object Tracking Benchmark (OOTB) and ALOV++ datasets. The experimental results demonstrated the superior performance of MUSTer in comparison with other state-of-art trackers.

5.1 Introduction

Object tracking is relatively easy for humans. Humans respond quickly to visual information by recognizing temporal consistency and memorizing useful visual features to recover from tracking failures when the target leaves field-of-view. Memory is one of the most powerful, but least well understood, functions of the human brain. With the sophisticated memory system, humans are capable of adapting to complex environments and behaving stably and consistently when facing temporal issues. We hypothesize that the principles of biological memory can be exploited to design solutions to tracking problems, which can be regarded as a time-series motion estimation problem.

Although how human memory works is still not fully understood, several memory models have been proposed. We focus on the well-known Atkinson-Shiffrin Memory Model (ASMM, also known as the multi-store model, outlined in Figure 5.1), which was proposed by Atkinson and Shiffrin in 1968 [4] to explain the basic structure and function of memory. The ASMM proposes that memory exists as three separate stages: sensory memory, short-term memory, and long-term memory. First, external stimuli are delivered to the "sensory register", at which point the original input signals are transformed into chemical and physical signals for processing within the biological system. Acceptable input information is then sent to the *short-term store*, where information undergoes the processes of encoding, rehearsing, retrieving, and responding, after which the system can output a reasonable and appropriate response. However, the short-term store does not retain information for a long time. Long-term memorizing is actually performed by the *long-term store*; if a particular pattern is received repeatedly, the long-term store is activated and the pattern information is retained. Once memorized, the pattern is maintained for a certain period of time but forgotten if not reinforced. As a result, the information inside the long-term store is a stable and consistent representation of current event sequences.

By combining short-term processing and long-term maintenance, this memory model produces *sensitive* and *stable* responses to complex inputs. When the external environment is continuous and steady, short-term store processing is fast and produces immediate responses. On the other hand, if the model encounters

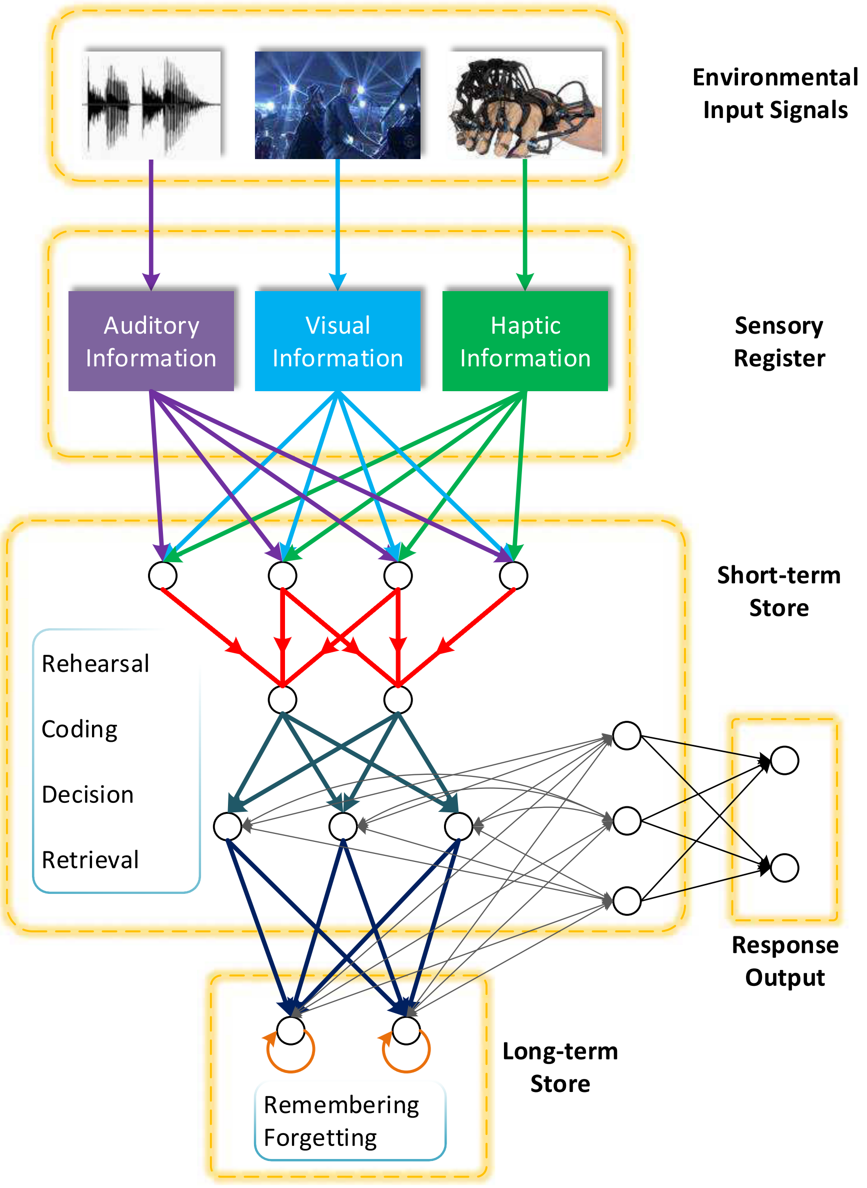


Figure 5.1: The Atkinson-Shiffrin memory model represented by an illustrative neural network. Nodes and their connections inside the short- and long-term stores represent the possible structure of the neural network inside the human brain.

a sudden change in input, information remembered by the long-term store is retrieved, which helps to stabilize the output. This cooperation allows humans to take reasonable actions in response to different and changing environments.

The online tracking research community have developed a number of trackers. Some [68,73,79,101,143] are highly sensitive and accurate in the short term, while others [88,106,148,163] are relatively conservative but robust over the long term. In other words, some trackers can be regarded as short-term systems while others can be regarded as long-term systems. The power of ASMM to track objects by co-operation between the long- and short-term memory stores has motivated us to design a tracker that integrates a short- and long-term system to boost tracking performance.

In this chapter, we propose the MUlti-Store Tracker (MUSTer) based on the ASMM. MUSTer consists of one short-term store and one long-term store that collaboratively process the image input and track the target. An Integrated Correlation Filter (ICF), which stores short-term memory and depends on spatiotemporal consistency, is employed in the short-term store to perform short-term tracking via two-stage filtering. In addition, we integrate a complementary component based on keypoint matching-tracking and RANSAC estimation that can interact with the keypoint feature database in the long-term store and control the final output and the short-term memory states. To maintain a reasonable keypoint feature database size, we adopt the well-known forgetting curve to model the remembering-forgetting loop and, in doing so, retain useful features.

5.2 Related Work

Online object tracking has long been a popular topic in computer vision. A large number of trackers have been proposed [115, 185], and the recent publication of benchmark datasets containing large numbers of sequences and standardized quantitative evaluation metrics is accelerating the pace of development in this field [95,160,181].

Various approaches that form the basis of existing trackers can be used to model short-term memory of the target appearance. In [150], Ross *et al.* proposed to incrementally learn a low-dimensional subspace of the target representation. Later, Mei *et al.* [133] introduced sparse representations for tracking, subsequently adopted in many trackers [77,195], in which the memory of the target appearance

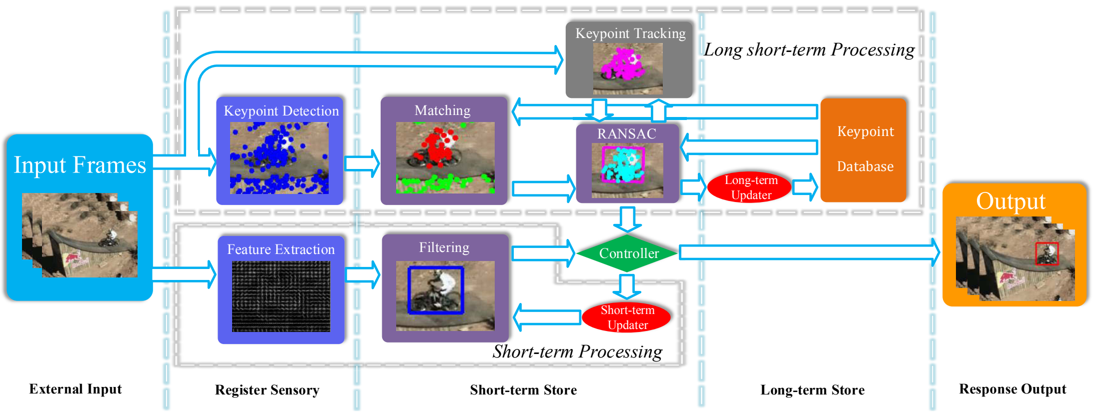


Figure 5.2: System flowchart of the proposed tracker based on the Atkinson-Shiffrin Memory Model. The short-term processing in short-term store is conducted by an ICF via two-stage filtering. Another set of short-term procedures including keypoint matching, keypoint tracking and RANSAC estimation is conducted by a conservative long-term component in short-term store, and it is able to interact with the long-term memory located in the long-term store. Both the results of short-term processing and long short-term processing are obtained by a controller, which decides the final output and the ICF update.

is modeled using a small set of target instances. In contrast to the generative approaches used in [150] and [133], discriminative methods [6, 7, 9, 68, 74, 78] have been proposed that consider both foreground and background information. In particular, Struck [68] is one of the best performing trackers and has been highlighted in several recent studies [146, 160, 181]. In [68], Hare *et al.* introduced structured SVM for tracking and trained a classifier using samples with structured labels. The correlation filter-based trackers [25, 42, 43, 73, 74, 118] are becoming increasingly popular due to their promising performance and computational efficiency. However, most of these trackers depend on the spatiotemporal consistency of visual cues and adopt relatively risky update schemes; therefore, they can only handle short-term tracking.

Some long-term tracking approaches [45, 88, 106, 148, 163] have also been proposed with promising results. For instance, TLD [88] employs two experts to identify the false negatives and false positives to train a detector. The experts are independent, which ensures mutual compensation of their errors to alleviate "drifting". In [148], Pernici and Bimbo modeled the target appearance using oversampled local features. They used transitive matching to find the target keypoints and, in addition, performed occlusion detection to avoided updating errors. In [163], long-term tracking was conducted based on a self-paced learning scheme in which the target appearance was conservatively learned by selecting trustworthy frames.

Some systems have a similar architecture to ASMM. In 1997, Hochreiter and Schmidhuber introduced a special kind of artificial neural network called *Long Short Term Memory* [76]. In this neural network, memory blocks are made up of two kind of units which are called "constant error carousels" and "multiplicative gate units". The constant error carousels memorize information and the multiplicative gate units control the information flow. These memory blocks can be regarded as the long-term store of ASMM. Together with pure feed-forward networks, which play the short-term role, long short-term memory systems show the great potential for various memory-intensive tasks, such as speech recognition [66] and music composition [52]. This makes the use of memory models for object tracking more convincing.

5.3 The Proposed Multi-store Tracker

The MUSTer framework is based on the ASMM (Figure 5.2), which consists of a short-term store, a long-term store, and the corresponding processing units. An Integrated Correlation Filter (ICF) is employed in the short-term store to perform short-term processing and track the target based on short-term memory and spatiotemporal consistency. This component generally works accurately and efficiently in relatively stable scenarios. In addition, another relatively conservative long-term component based on keypoint matching-tracking and RANSAC estimation is introduced to conduct the long short-term processing on the fly. This interacts with the short-term memory stored in an active set of keypoints using forward-backward tracking, and it also retrieves the long-term memory for matching and updates the long-term memory based on the RANSAC estimation results and the forgetting curve. During tracking, the outputs of both the short-term and long short-term processing are sent to a controller, which decides the final MUSTer output and the ICF update. Specifically, the short-term memory in ICF is reset when the short-term processing output is highly inconsistent with the long-term memory encoded by the output of the long short-term processing. This enables the recovery of the short-term tracking after dramatic appearance changes such as severe occlusion, the object leaving field-of-view, or rotation.

The following subsections detail successively all components in MUSTer: ICF short-term processing, the short-term processing of keypoints by the long-term component, the updating of long-term memory based on the forgetting curve, and finally, the output controller and ICF updater.

5.3.1 Short-term Integrated Correlation Filters

The short-term component is used to provide instant responses to the image input based on short-term memory. Recently, the robustness of correlation filter-based short-term trackers [42, 73, 118] has been recognized by [95]. For accurate and efficient short-term processing performance, we employ Integrated Correlation Filters (ICFs), which are based on the Kernelized Correlation Filters (KCFs) [73] and the Discriminative Scale Space Correlation Filter (DSSCF) [42]. The ICF is a two-stage filtering process that performs translation estimation and scale estimation, respectively, which is similar to the pipeline described in [42] and [118]. Here, ICF serves as the short-term component, where the short-term memory of ICF consists of the learned coefficients and templates for the filters.

In KCF, a classifier $f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle$ is trained on a $M \times N$ image patch $\mathbf{x}$ centered by the target bounding box $\mathbf{B}_{\mathrm{T}}$'s center and p times larger than $\mathbf{B}_{\mathrm{T}}$. Instead of using dense sliding windows to extract training samples, the classifier considers all the cyclic shift versions $\mathbf{x}_i$ for training, where $i \in \{0, ..., M-1\} \times \{0, ..., N-1\}$. Each example $\mathbf{x}_i$ is assigned with a score in $y_i \in [0, 1]$ generated by a Gaussian function in terms of the shifted distance, and the classifier is trained by minimizing the regression error:

$$\min_{\mathbf{w}} \sum_i (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle - y_i)^2 + \lambda \|\mathbf{w}\|^2, \tag{5.1}$$

where $\phi(\mathbf{x})$ is the mapping to a Hilbert space, and $\lambda \geq 0$ is the regularization parameter controlling the model simplicity. Employing a kernel $\kappa(\mathbf{x}, \mathbf{x}') = \langle\phi(\mathbf{x}), \phi(\mathbf{x}')\rangle$, the classifier can be derived as $f(\mathbf{x}) = \sum_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x})$, where $\boldsymbol{\alpha}$ is the dual variable of $\mathbf{w}$. Let us denote the Discrete Fourier Transform (DFT) of a vector with a hat "$\hat{}$", e.g., $\hat{\boldsymbol{\alpha}} = \mathscr{F}(\boldsymbol{\alpha})$, and denote the complex conjugate with $\hat{\boldsymbol{\alpha}}^*$. According to [43] and [73], if the employed kernel is shift invariant, e.g., an RBF kernel, $\hat{\boldsymbol{\alpha}}^*$ can be obtained based on the favorable properties of circulant matrices:

$$\hat{\boldsymbol{\alpha}} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{\mathbf{xx}} + \lambda}, \tag{5.2}$$

where $\mathbf{k}^{\mathbf{xx}}$ is a vector whose ith element is $\kappa(\mathbf{x}_i, \mathbf{x})$. In particluar, for image data with C feature channels, a concatenation $\mathbf{x} = [\mathbf{x}^1; ...; \mathbf{x}^C]$ can be constructed, and the kernel correlation $\mathbf{k}^{\mathbf{xx}}$ based on a Gaussian kernel can be efficiently computed by element-wise products and simple summation over the feature channels in the Fourier domain:

$$\mathbf{k}^{\mathbf{xx}'} = \exp(-\frac{1}{\sigma^2}(\|\mathbf{x}\|^2 + \|\mathbf{x}'\|^2 - 2\mathscr{F}^{-1}(\sum_{c=1}^{C} \hat{\mathbf{x}^c} \odot (\hat{\mathbf{x}'^c})^*))), \tag{5.3}$$

where $\odot$ denotes the operator of element-wise products, and c is the index of the feature channels.

During the first-stage filtering for translation estimation, given a $M \times N$ candidate image patch $\mathbf{z}$ as the search space, all cyclic patches of $\mathbf{z}$ can be evaluated via

$$\mathbf{f}(\mathbf{z}) = \mathscr{F}^{-1}((\hat{\mathbf{k}}^{\mathbf{xz}}) \odot \hat{\boldsymbol{\alpha}}), \tag{5.4}$$

where $\mathbf{f}(\mathbf{z})$ is the filtering response for all the cyclic versions of $\mathbf{z}$, and the translation is estimated by finding the location with the highest response. In our tracker, the candidate image patch $\mathbf{z}$ is centered by using the tracking result $\mathbf{B}_o$ of the last frame. To adapt to the short-term appearance changes of the target, the filter coefficients $\boldsymbol{\alpha}$ and the target template $\mathbf{x}$ are updated in an interpolating manner with learning rate γ. In our implementation, we employ the 31-dimensional HOG descriptors [55], as in [73]. Moreover, for color image sequences, we further extract 10-dimensional color attributes [43, 169] as complementary features and combine them with the HOG descriptors to further boost performance.

To cope with scale changes, a one-dimensional DSSCF [42] is also trained and the second-stage filtering is performed for scale estimation. To evaluate the trained DSSCF, S image patches centered around the location found by the KCF filter are cropped from the image, each of size $a^s M_\mathrm{T} \times a^s N_\mathrm{T}$, where $M_\mathrm{T} \times N_\mathrm{T}$ is the current size of the target, a is the scale factor, and $s \in \{-\frac{S-1}{2}, ..., \frac{S-1}{2}\}$. All S image patches are then resized to the template size for the feature extraction. Finally, the final output $\mathbf{B}_\mathrm{s}$ from the short-term processing is given as the image patch with the highest filtering response. Similar to KCF, the model parameters are also updated in an interpolating manner with learning rate μ. We refer readers to [42] for more details and the implementation of DSSCF.

5.3.2 Short-term Processing of Keypoints

The goal of the long-term component is to conservatively learn the appearance of the target and to refresh the short-term memory when a mistake made by the short-term component is detected.

Local Scale-Invariant Features [125] are a powerful tool used in various computer vision tasks including recognition [125] and scene alignment [121]. In particular, some promising trackers [65,137,148] have been proposed to model the object appearance based on Local Scale-Invariant Features (i.e., keypoints). Therefore, we employ a long-term component based on keypoint matching-tracking and RANSAC estimation to take advantage of the flexibity and shape generalizability of the keypoint-based appearance models.

The long-term memory of the target appearance is modeled by a total feature database $\mathcal{M} = \mathcal{T} \cup \mathcal{B}$ that consists of a foreground (target) feature database $\mathcal{T}$ and a background feature database $\mathcal{B}$:

$$\mathcal{T} = \{(\mathbf{d}_i, \mathbf{p}_i^o)\}_{i=1}^{N_\mathcal{T}}, \quad \mathcal{B} = \{\mathbf{d}_i\}_{i=1}^{N_\mathcal{B}} \ . \tag{5.5}$$

Here, $\mathbf{d}_i \in \mathcal{R}^{128}$ is the 128-dimensional Scale-invariant Feature Transform (SIFT) descriptors [126] of the keypoints. $N_\mathcal{T}$ and $N_\mathcal{B}$ are the respective numbers of descriptors. Each target descriptor $\mathbf{d}_i \in \mathcal{T}$ is also associated with the corresponding coordinates $\mathbf{p}_i^o \in \mathcal{R}^2$ that remember the keypoint location in the original target template, and can be used for estimating the transformation of target state. The background feature database is important to reduce erroneous matching of target keypoints and can help to detect the occlusions.

In addition to the filtering function of ICF, another set of short-term procedures conducted in the short-term store is to consecutively process the keypoints by retrieving the long-term memory stored in $\mathcal{M}$ and the short-term memory stored in an active set. In each frame, a SIFT detector based on the difference of Gaussians [126] is applied to an image search area to extract a set of keypoints with large responses. The set of detected keypoints associated with their SIFT descriptors is denoted as $\mathcal{P}_{\mathrm{D}} = \{(\mathbf{d}_k, \mathbf{p}_k)\}_{k=1}^{N_{\mathrm{D}}}$, where $\mathbf{p}_k \in \mathcal{R}^2$ is the coordinates of a keypoint.

Matching Keypoints. We search the total memory database $\mathcal{M}$ for the nearest neighbors of each $\mathbf{d}_k \in \mathcal{P}_{\mathrm{D}}$ based on the Euclidean distance. We define the matching confidence of $\mathbf{d}_k$ and its nearest neighbor[1] $\mathbf{d}_k^{1\mathcal{N}}$ as the cosine similarity $C(\mathbf{d}_k, \mathbf{d}_k^{1\mathcal{N}})$ between two descriptors. The candidate matching keypoint can be found if the matching confidence is larger than a predefined threshold, denoted as θ_{T} for the matching of target points and θ_{B} for that of background points. The different threshold settings are used to control the different recalls for the matching of foreground and background keypoints. In practice, the precision of the matching of target points is more important since they are used to estimate the current state of the target. Therefore, it is important to reject outliers during target point matching as discussed in [126, 148]. To further reject outliers, we employ the distinct non-parametric nearest neighbor classifier [126] by computing the ratio of the distances:

$$r(\mathbf{d}_k) = \frac{d(\mathbf{d}_k, \mathbf{d}_k^{1\mathcal{N}})}{d(\mathbf{d}_k, \mathbf{d}_k^{2\mathcal{N}})}, \tag{5.6}$$

where $\mathbf{d}_k^{2\mathcal{N}}$ is the second nearest neighbor of $\mathbf{d}_k$. The matched point $\mathbf{d}_k^{1\mathcal{N}}$ is classified as an inlier if $r(\mathbf{d}_k)$ is smaller than a threshold θ_r. Finally, the detected feature points $(\mathbf{d}_k, \mathbf{p}_k) \in \mathcal{P}_{\mathrm{D}}$ are classified into one of the three following sets: the matched target keypoints $\mathcal{P}_{\mathrm{T}}$, the matched background keypoints $\mathcal{P}_{\mathrm{B}}$ and

[1]For simplicity, we denote the nearest and the second nearest neighbors of $\mathbf{d}_k$ in $\mathcal{M}$ by $\mathbf{d}_k^{1\mathcal{N}}$ and $\mathbf{d}_k^{2\mathcal{N}}$, respectively.

unmatched keypoints $\mathcal{P}_\mathrm{N}$, according to the formula below:

$$\begin{cases} \mathbf{p}_k \in \mathcal{P}_\mathrm{T}, & \mathbf{d}_k^{1\mathrm{N}} \in \mathcal{T},\, C(\mathbf{d}_k, \mathbf{d}_k^{1\mathrm{N}}) > \theta_\mathrm{T},\, r(\mathbf{d}_k) < \theta_r \\ \mathbf{p}_k \in \mathcal{P}_\mathrm{B}, & \mathbf{d}_k^{1\mathrm{N}} \in \mathcal{B},\, C(\mathbf{d}_k, \mathbf{d}_k^{1\mathrm{N}}) > \theta_\mathrm{B} \\ \mathbf{p}_k \in \mathcal{P}_\mathrm{N}, & \text{otherwise .} \end{cases} \tag{5.7}$$

Once the matched target keypoints $(\mathbf{d}_k, \mathbf{p}_k) \in \mathcal{P}_\mathrm{T}$ are determined, we further find their corresponding coordinates $\mathbf{p}_k^o$ in the original template, which are then added into $\mathcal{P}_\mathrm{T}$ so as to get the final complete version, i.e., $\mathcal{P}_\mathrm{T} = \{(\mathbf{d}_k, \mathbf{p}_k^o, \mathbf{p}_k)\}_{k=1}^{N_m}$.

Forward-Backward Tracking. In addition to matching keypoints and inspired by [137], we also maintain an active set of keypoints $\mathcal{P}_\mathrm{A}^{t-1} = \{(\mathbf{p}_i^o, \mathbf{p}_i^{t-1})\}_{i=1}^{N_A}$, where $\mathbf{p}_i^{t-1}$ is the coordinates of the point in the $t-1$ frame and $\mathbf{p}_i^o$ is the corresponding coordinates of $\mathbf{p}_i^{t-1}$ in the original template. This active set can be also regarded as the short-term memory of MUSTer, and it provides additional information for long short-term processing. To obtain the coordinates of $\mathbf{p}_i^{t-1}$ in frame I_t, the optical flow can be computed using the Lucas-Kanade (LK) method [127]. To improve robustness, we employ a Forward-Backward (FB) tracker [88] to obtain a set of keypoints with reliable tracking results. In the FB tracker, the forward optical flow from $\mathbf{p}_i^{t-1}$ to $\mathbf{p}_i^t$ and the backward optical flow from $\mathbf{p}_i^t$ to $\mathbf{p'}_i^{t-1}$ are computed using two consecutive frames: I_{t-1} and I_t. The displacement $d(\mathbf{p}_i^{t-1}, \mathbf{p'}_i^{t-1})$ between $\mathbf{p}_i^{t-1}$ and $\mathbf{p'}_i^{t-1}$ is then used to identify any tracking failures. Ideally, $d(\mathbf{p}_i^{t-1}, \mathbf{p'}_i^{t-1})$ should be small if the tracking is successful. Therefore, a threshold θ_{fb} is defined and a failure is detected if $d(\mathbf{p}_i^{t-1}, \mathbf{p'}_i^{t-1}) > \theta_{fb}$. Finally, we can obtain a set of remaining active keypoints $\mathcal{P}_\mathrm{A}^t = \{(\mathbf{p}_i^o, \mathbf{p}_i^t)\}_{i=1}^{N'_A}$ that contain the keypoints successfully tracked by the FB tracker.

RANSAC Estimation. Once the matched and tracked keypoints are obtained, a candidate set $\mathcal{P}_\mathrm{C}$ is formed consisting of the keypoints in $\mathcal{P}_\mathrm{T}$ and $\mathcal{P}_\mathrm{A}^t$, which is used to estimate the target state. Similar to [137, 148], we only consider the similarity transformation, which is more reliable than homography transformation for tracking generic objects, especially in cases where the planarity assumption does not hold [137]. In the case of the similarity transformation, the state of the target can be defined as $\mathbf{s}_t = \{x_t, y_t, s_t, \beta_t\}$, which are the parameters of translations, scale, and rotation angle, respectively. To predict the target state $\mathbf{s}_t$, a transformation $F_{\mathbf{s}_t}$ letting $F_{\mathbf{s}_t}(\mathbf{p}_i^o) \rightarrow \mathbf{p}_i$ can be estimated using $\mathcal{P}_{\mathrm{C}} = \{(\mathbf{p}_i^o, \mathbf{p}_i)\}_{i=1}^{N_C}$. However, mistakes in matching and tracking keypoints cannot always be avoided, especially when the background is cluttered or rapidly changing. Therefore, we employ the RANSAC estimator to compute $F_{\mathbf{s}_t}$ by randomly proposing putative solutions and identifying inliers and outliers. In our implementation, we use the robust MLESAC estimator [70, 168], which works by maximizing the likelihood rather than just the number of inliers.

Confidence of Matching. Once the resulting transformation $F_{\mathbf{s}_t}$ is estimated, a target bounding box $\mathbf{B}_l$ defined by the current state can be computed. Meanwhile, a set of inlier keypoints $\mathcal{P}_{\mathrm{I}} = \{\mathbf{p}_i\}_{i=1}^{N_I}$ can also be obtained, where the number of inliers N_I is an important evidence predicting tracking success; in general, the more inliers included in $F_{\mathbf{s}_t}$, the more confident the result. Therefore, we can define a binary variable G_C that indicates the tracking success and set it as:

$$G_C = \begin{cases} True, & N_I > \theta_I \\ False, & \text{otherwise} \end{cases}, \tag{5.8}$$

where θ_I is a predefined threshold controlling recall strictness.

Occlusion Handling. In the proposed framework, the long-term memory is progressively updated on the fly. Therefore, it is important to consider cases of occlusion, which have been discussed previously [96, 148]. In particular, we consider the set $\mathcal{P}(\mathbf{B}_l)$, which consists of the keypoints inside target bounding box $\mathbf{B}_l$, and the set $\mathcal{P}_{\mathrm{O}}$ of *occluding keypoints* is defined as the matched background keypoints inside $\mathbf{B}_l$, i.e., $\mathcal{P}_{\mathrm{O}} = \mathcal{P}(\mathbf{B}_l) \cap \mathcal{P}_{\mathrm{B}}$. Intuitively, if there is no occlusion, the number of occluding points $N_O = |\mathcal{P}_{\mathrm{O}}|$ should be small and close to zero. In contrast, when the target is occluded, the number of occluding keypoints is likely to be high and close to the number of keypoints belonging to the target. Therefore, we can consider the ratio of N_O and N_G and define a binary variable G_O that indicates occlusion occurrence:

$$G_O = \begin{cases} True, & N_O/N_G > \theta_o \\ False, & \text{otherwise} \end{cases}, \tag{5.9}$$

where $N_G = |\mathcal{P}_{\mathrm{G}}|$ is the number of keypoints in $\mathcal{P}_{\mathrm{G}}$, and $\mathcal{P}_{\mathrm{G}} = \mathcal{P}(\mathbf{B}_l) \cap \mathcal{P}_{\mathrm{T}}$ is the matched target keypoints inside the region of $\mathbf{B}_l$. In our implementation, we empirically set $\theta_o = 0.5$.

Update the Active Set. The active set $\mathcal{P}^t_{\mathrm{A}}$ that stores the short-term memory is updated at each frame. However, we should not track the keypoints if an occlusion is detected since the keypoints may gradually lock onto occluding objects or the background. Therefore, we set $\mathcal{P}^t_{\mathrm{A}} = \emptyset$ if $G_O = True$. Otherwise, we set $\mathcal{P}^t_{\mathrm{A}} = \mathcal{P}_{\mathrm{I}}$, which are the inliers found by the RANSAC estimator. However, when the target is stable or moving slowly, most of the keypoints in $\mathcal{P}^t_{\mathrm{A}}$ would be successfully tracked and identified as inliers, and meanwhile matched target keypoints would continually be added to $\mathcal{P}^t_{\mathrm{A}}$. This might lead to a very large $\mathcal{P}^t_{\mathrm{A}}$ and thus computational inefficiency. Therefore, we design a strategy to find the redundant points in $\mathcal{P}^t_{\mathrm{A}}$ and let the candidate set of RANSAC $\mathcal{P}_{\mathrm{C}} = \mathcal{P}_{\mathrm{T}} \cup (\mathcal{P}^t_{\mathrm{A}} \backslash \mathcal{P}_{\mathrm{R}})$, where $\mathcal{P}_{\mathrm{R}}$ denotes the set of redundant keypoints. This approach is based on the assumption that the matched keypoints are more reliable and, therefore, have higher priority. The redundant points are found using the quantization IDs. Specifically, a virtual grid is built upon the original target template, and each target keypoint can be assigned a quantization ID according to its corresponding coordinates $\mathbf{p}^o_i$ in the original template. Finally, the redundant points in $\mathcal{P}_{\mathrm{A}}$ are found by searching the repetitive quantization IDs in $\mathcal{P}_{\mathrm{T}}$.

5.3.3 Long-term Memory Updates

The use of keypoints as the appearance model allows natural handling of in-plane rotation. For instance, the keypoint database is not updated in [137] but performance is still reasonable. However, it is still crucial to update on the fly to generalize the appearance model to handle out-of-plane rotation, severe scale changes, and appearance variations of the target [148].

To maintain relatively reliable memory of the target appearance, the memory database $\mathcal{M}$ is updated conservatively only when the short-term processing is confident about the result ($G_C = True$) and claims there is no occlusion ($G_O = False$). Both the target keypoint database $\mathcal{T}$ and the background keypoint database $\mathcal{B}$ need to be updated. In particular, we consider the unmatched points that are important for capturing any changes in visual structure. During the update, we add the unmatched keypoints inside the $\mathbf{B}_l$ to $\mathcal{T}$, and those outside $\mathbf{B}_l$ to $\mathcal{B}$, as follows:

$$\begin{aligned} \mathcal{T} &= \mathcal{T} \cup (\mathcal{P}_\mathrm{N} \cap \mathcal{P}(\mathbf{B}_l)) \\ \mathcal{B} &= \mathcal{B} \cup (\mathcal{P}_\mathrm{N} \cap (\mathcal{P}_\mathrm{D} \setminus \mathcal{P}(\mathbf{B}_l)) \ . \end{aligned} \tag{5.10}$$

While the human brain is good at remembering, as well as forgetting. The remembering-forgetting interplay helps us effectively manage the valuable and finite memory capacity when handling massive quantities of input signals each day. To avoid the unbounded growth of the memory database $\mathcal{M}$, a certain capacity should be set for $\mathcal{T}$ and $\mathcal{B}$, and features should be forgotten over time. To model remembering-forgetting, we employ the famous *forgetting curve* [51] to maintain $\mathcal{M}$ and forget unimportant features according to the retention of features. The forgetting curve hypothesizes a decline in memory retention and shows how information is lost over time when there is no attempt to retain it. The memory retention r over time can typically be modeled using an exponential function:

$$r = \exp(-\frac{\tau}{\Gamma h}) \ , \tag{5.11}$$

where h is the relative strength of the memory, τ is the relative period of forgetting, and Γ is a constant controlling the scale of the timespan. In (5.11), the speed of decline in memory retention is decided by the relative strength h. For information with high relative strength h, the decline in memory retention becomes slow and it is more possible to be remembered over time. The relative strength of information can be increased using certain methods such as repetitive retrieving. In MUSTer, each feature in $\mathbf{d}_i \in \mathcal{M}$ is assigned a set of memory variables (r_i, τ_i, h_i), where the memory retention r_i of features can be updated according to their corresponding τ_i and h_i. To model the process of forgetting, during each update term, all relative periods τ_i of $\mathbf{d}_i \in \mathcal{M}$ are increased by 1. Moreover, for the retrieved background features $\mathbf{d}_k^{1\mathcal{N}} \in \mathcal{B}$ where $\mathbf{d}_k \in \mathcal{P}_\mathrm{B}$, and foreground features $\mathbf{d}_k^{1\mathcal{N}} \in \mathcal{T}$ where $\mathbf{d}_k \in (\mathcal{P}_\mathrm{T} \cap \mathcal{P}_\mathrm{I})$, the corresponding relative strength h_i is increased by 1 and the relative period τ_i is set to 0. In this way, the recalled features are renewed and strengthened in memory, while frequently recalled features obtain a high relative strength and become hard to forget. Once the numbers of features in the respective databases exceed a predefined memory capacity, $N_{\mathcal{T}} > \Theta_{\mathcal{T}}$ or $N_{\mathcal{B}} > \Theta_{\mathcal{B}}$, the features with low retention are removed and completely forgotten by the model.

5.3.4 Output and Short-term Memory Refreshing

Once the different processing procedures are performed in the short-term store, the results of filtering $\mathbf{B}_s$ and the result of the long short-term processing $\mathbf{B}_l$ together with the state variables G_C and G_O can be obtained by a controller. As mentioned earlier, short-term filtering is generally accurate in relatively stable scenarios. Therefore, if the results $\mathbf{B}_s$ and $\mathbf{B}_l$ are to some extent consistent (indicating that the long-term memory agrees with the output of the short-term tracker), or the long-term component is not confident about the result ($G_C = False$), or an occlusion is detected ($G_C = True$), the tracking result $\mathbf{B}_o$ is output as $\mathbf{B}_o = \mathbf{B}_s$, and the short-term filters are updated using $\mathbf{B}_o$ and the predefined learning rates $\gamma = \gamma_o$ and $\mu = \mu_o$. Otherwise, the tracking result is output as $\mathbf{B}_o = R(\mathbf{B}_l)$, where $R(\cdot)$ is a function to rotate a bounding box along its center and in the same orientation as $\mathbf{B}_s$. Moreover, the short-term memory should be refreshed. We use $\mathbf{B}_o$ to update the short-term filters and set the learning rates γ and μ as 1 for the update, which cleans up all the previous short-term memory stored in the filters. In particular, the inconsistency of $\mathbf{B}_s$ and $\mathbf{B}_l$ is detected by the Intersection Over Union (IOU) metric $U(\mathbf{B}_s, \mathbf{B}_l) = \frac{|\mathbf{B}_s \cap \mathbf{B}_l|}{|\mathbf{B}_s \cup \mathbf{B}_l|}$ with a threshold θ_U.

5.4 Experiments

The proposed tracker[1] was implemented using Matlab & C++ (OpenCV). The average time cost on OOTB is 0.287s/frame on a cluster node (3.4GHz, 8 cores, 32GB RAM) The parameters mentioned in Section 5.3 are specified as follows: $\theta_U = 0$, the learning rate γ_o and μ_o for the ICF are set to 0.02 and 0.01 respectively, and the padding size p in KCF is set to 1. The thresholds for matching

[1]The code to reproduce the experiments is available in `https://sites.google.com/site/multistoretrackermuster/`

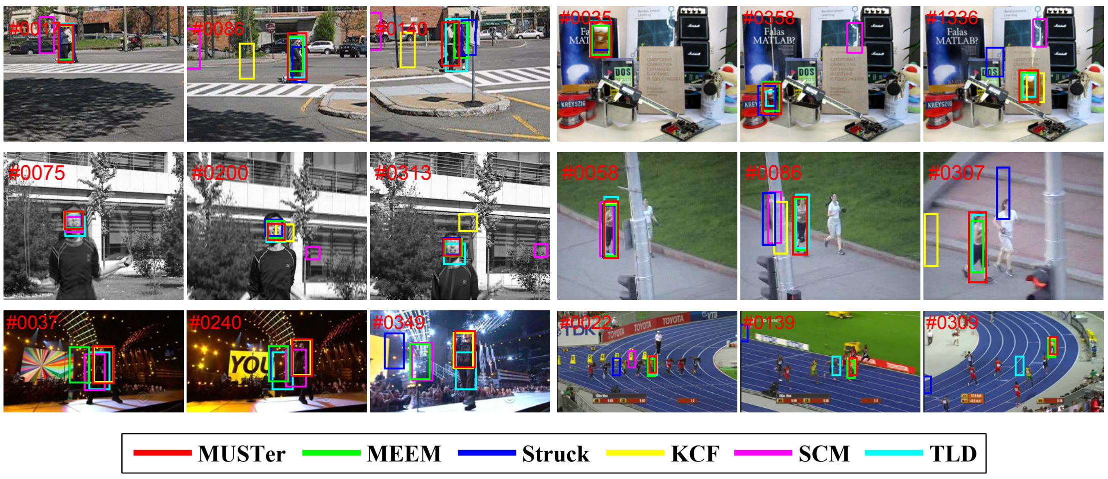


Figure 5.3: Tracking results of selected algorithms in representative frames. Frame indexes are shown in the top left of each figure. The showing examples are from sequences *Couple*, *Lemming*, *Jumping*, *Jogging*, *Singer2*, *Bolt*, respectively.

keypoints are set as $\theta_T = 0.8$, $\theta_B = 0.9$, $\theta_r = 0.85$, and $\theta_I = 8$. The threshold θ_{fb} for FB Tracker is set to 4. The Γ in the forgetting curve model is set to 10, while the memory capacities $\Theta_{\mathcal{T}}$ and $\Theta_{\mathcal{B}}$ of keypoint databases are both set to 2000. Note that the protocols proposed in [181] were strictly followed and all parameters were fixed for all video sequences in the following evaluations. For each sequences, we simply use the ground truth of the first frame given by the dataset for initialization.

5.4.1 Evaluation on CVPR2013 OOTB

In this section, we report the evalution on the CVPR2013 Online Object Tracking Benchmark (OOTB) [181] and compare MUSTer with a number of state-of-the-art trackers. OOTB is a popular comprehensive benchmark specifically designed for evaluating performance. OOTB contains 50 fully annotated sequences which extensively used by previous work. In [181], the evaluation for the robustness of trackers is based on two different metrics: the precision plot and success plot. The precision plot shows the percentage of successfully tracked frames on which the Center Location Error (CLE) of a tracker is within a given threshold T_C,

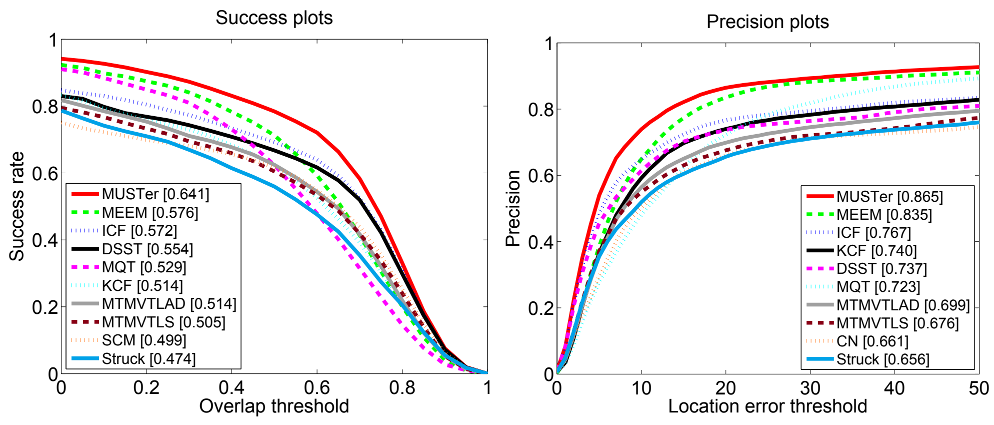


Figure 5.4: Quantitative comparison on CVPR2013 OOTB. The performance score for each tracker is shown in the legend. For each figure, only the top 10 trackers are presented.

and a representative precision score at $T_C = 20$ is used for ranking the trackers. The success plot also counts the percentage of successfully tracked frames, by measuring the Intersection Over Union (IOU) metrics for trackers on each frame. In success plot, the threshold of IOU is varied from 0 to 1, and the ranking of trackers is based on the Area Under Curve (AUC) score. For more details about OOTB and the adopted metrics, we refer readers to [181].

We run the One-Pass Evaluation (OPE) on the benchmark using the proposed MUSTer and use the online available software[1] provided by [181] to compute the evaluation plots. For the competitor trackers, we first consider those 29 popular approaches whose results are available in OOTB, such as Struck [68], TLD [88], SCM [198], VTD [98], VTS [99], and ASLA [84]. And on top of these, we include recent correlation filter-based trackers CN [43], KCF [73], DSST [42], which is the best performing tracker in [95], and further include very recent state-of-the-art trackers MEEM [192], the LGT [170] based on coupled-layer visual model, and MTMVTLS, MTMVTLAD [130] and MQT proposed in previous chatpers. Last but not least, we also compare quantitatively with the short-term ICF presented in Section 5.3.1, so as to demonstrate the importance of the long-term component in our proposed tracker.

[1]http://visual-tracking.net/

Figure 5.3 shows a qualitative comparison with selected trackers on several representative videos/frames. To quantitatively compare all 37 trackers, we show the precision plot and success plot in Figure 5.4, which indicates that MUSTer achieves overall the best performance using both the metrics and significantly outperforms the second best tracker MEEM with 11% performance gain using the metric of AUC score. As discussed in [181], the AUC score that measures the overall performance in success plot is more accurate than the precision score at one threshold of precision plot. Therefore, the experimental result clearly demonstrates the superior performance of the proposed tracker. In addition, MUSTer also significantly outperforms its baseline short-term component ICF, other correlation filter-based trackers DSST and KCF, and the well-known long-term tracker TLD, which validates the important role of the long-term component in MUSTer and the effectiveness of the proposed tracking framework based on ASMM.

5.4.2 Evaluation on ALOV++ Dataset

To further validate the robustness of MUSTer, we conducted the second evaluation on a larger dataset [160], namely ALOV++ (Amsterdam Library of Ordinary Videos), which is recently developed by Smeulders *et al.*. It consists of 14 challenge subsets, totally 315 sequences and focuses on systematically and experimentally evaluating trackers' robustnesses in a large variety of situations including light changes, low contrast, occlusion, etc. In [160], survival curves based on F-score were proposed to evaluate trackers' robustnesses and demonstrated its effectiveness. A survival curve shows the performance of a tracker on all videos in the dataset. The videos are sorted according to the F-score. By sorting the videos, the graph gives a bird's eye view in cumulative rendition of the quality of the tracker on the whole dataset. We refer the reader to the original paper [160] and the author's website[1] for details about the dataset and the evaluation tools.

To evaluate MUSTer on ALOV++ dataset, we ran MUSTer on all the 315 sequences using the ground truth of the first frame as initialization and the same parameters as the previous evaluation shown in Section 5.4.1. We compare our

[1]`http://imagelab.ing.unimore.it/dsm/`

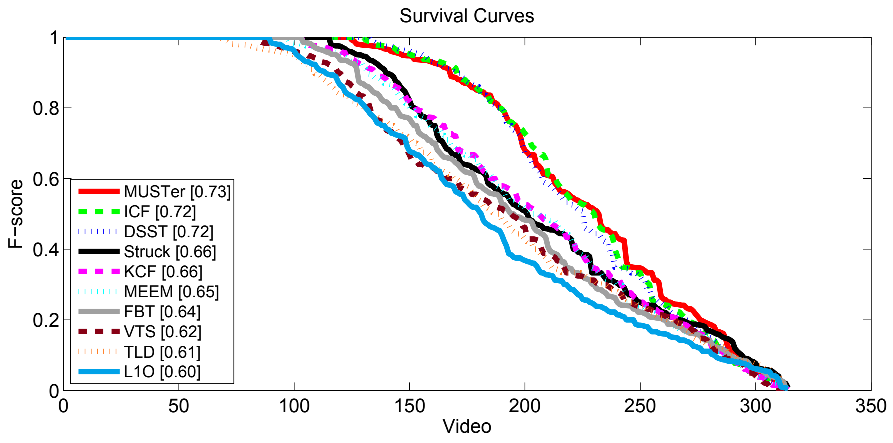


Figure 5.5: Survival curves for top ten trackers on AlOV++ dataset. The average F-scores over all sequences are specified in the legend.

tracker with 19 popular trackers[1] that were evaluated in [160]. In addition, we also ran MEEM, ICF, DSST, KCF on ALOV++, which rank on the top five trackers in the previous evaluation. The survival curves of the top ten trackers and the average F-scores over all sequences are shown in Figure 5.5, which demonstrates that MUSTer achieves the best overall performance over 24 compared trackers in this comparison.

5.5 Conclusion

In this chapter, we propose the MUlti-Store Tracker (MUSTer) based on the Atkinson-Shiffrin Memory Model to handle tracking memory problems. MUSTer consists of two important but relatively independent components and exploits them to process the image input according to the short- and long-term memories of the target being tracked. In the short-term store, an Integrated Correlation Filter (ICF), which stores the short-term memory and depends on spatiotemporal consistency, is employed to provide an instant response via two-stage filtering. In addition, a complementary component based on keypoint matching-tracking and RANSAC estimation is integrated, which is able to interact with the key-

[1]We refer the reader to [160] and its references for the details about the compared trackers. The evaluation results of these 19 trackers were obtained from the author of [160].

point feature database in the long-term store and control the final output and the short-term memory states. To maintain a reasonable keypoint feature database size, the well-known forgetting curve is employed to model the remembering-forgetting loop and retain the most useful features. The experimental results on two large datasets demonstrate that the proposed tracker is capable of taking advantage of both the short-term and long-term systems and boosting the tracking performance.

Chapter 6

Conclusions

Visual tracking plays a key role in many computer vision systems, such as robotics, video surveillance, automatic control, vehicle navigation, and human computer interaction (HCI). In this thesis, we tackled challenges that are present in practical tracking scenarios in videos and developed robust online visual trackers that achieve superior performance over existing trackers, by taking advantage of advanced techniques in machine learning including distance metric learning [124], sparse representation [179], multi-view learning [184], multi-task learning [36], conditional random field [72].

In particular, we developed a novel dual-force distance metric which is elaborately designed for distracter-resistant tracking. The proposed metric systematically includes the normalized margin maximization, the similarity propagation and the reconstruction error constraint, in which the normalized margin maximization gives a force to separate positive samples and negative ones while the similarity propagation gives another force to drag negative samples into negative space. We seamlessly integrate our metric with the L1 minimization framework and takes advantage perfectly of both the descriptive power of L1 minimization with occlusion robustness and the discriminative power of our metric with distracter resistance. We tested our tracker on several challenging sequences and compared it with five other popular trackers including original BPR-L1 tracker to validate its superiority.

We also developed a LAD-based robust multi-task multi-view sparse learning method for particle filter-based tracking. By appropriately introducing the $l_{1,2}$

norm regularization, the method not only exploits the underlying relationship shared by different views and different particles, but also captures the frequently emerging outlier tasks which have been previously ignored. The proposed regularized LAD problem is effectively approximated by the Nesterov's smoothing method and efficiently solved by the APG. We implemented our method using four types of complementary features, i.e. intensity, color histogram, HOG and LBP, and extensively tested it on numerous challenging sequences including publicly available sequences, synthetic noisy sequences, real-world noisy sequences and two comprehensive tracking datasets. The experimental results demonstrate that the proposed method is capable of taking advantage of multi-view data and correctly handling the outlier tasks. Compared to several popular trackers, our tracker demonstrates superior performance.

For non-rigid object tracking, we proposed a tracking method based on a hierarchical appearance representation using multilevel quantization. The different levels of the representation are incorporated into a Conditional Random Field model using a coherent framework. By exploiting all the quantization levels, the method utilizes and integrates the information contained at each representation level by explicitly modeling the interactions and constraints between them; this results in significantly improved performance compared to other state-of-the-art tracking methods based on a single quantization. Moreover, Online Random Forests are used to update the appearance model in different levels of the tracker, in order to capture changes in object appearance over time. The experimental results demonstrate that the proposed method is capable of taking advantage of multilevel information and significantly boosting tracking performance.

To handle tracking memory problems, we proposed the MUlti-Store Tracker (MUSTer) based on the Atkinson-Shiffrin Memory Model. MUSTer consists of two important but relatively independent components and exploits them to process the image input according to the short- and long-term memories of the target being tracked. In the short-term store, an Integrated Correlation Filter (ICF), which stores the short-term memory and depends on spatiotemporal consistency, is employed to provide an instant response via two-stage filtering. In addition, a complementary component based on keypoint matching-tracking and RANSAC estimation is integrated, which is able to interact with the keypoint feature database in the long-term store and control the final output and the short-term memory states. To maintain a reasonable keypoint feature database size, the well-known forgetting curve is employed to model the remembering-forgetting loop and retain the most useful features. The experimental results on two large datasets demonstrate that the proposed tracker is capable of taking advantage of both the short-term and long-term systems and very promising performance.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012. 84, 89, 92

[2] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 798–805. 35, 82

[3] C. Aeschliman, J. Park, and A. C. Kak, "A probabilistic framework for joint segmentation and tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1371–1378. 82

[4] R. C. Atkinson and R. M. Shiffrin, "Human memory: A proposed system and its control processes," *Psychology of Learning and Motivation*, vol. 2, pp. 89–195, 1968. 99

[5] S. Avidan, "Ensemble tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 494–501. 22

[6] ——, "Ensemble tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261–271, 2007. 12, 82, 87, 91, 102

[7] ——, "Support vector tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1064–1072, 2004. 11, 12, 102

[8] B. Babenko, M. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proceedings of the IEEE Conference on Computer*

Vision and Pattern Recognition, 2009, pp. 983–990. 2, 4, 13, 19, 22, 34, 35, 38

[9] ——, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011. xiv, xv, 44, 45, 62, 76, 82, 83, 87, 92, 97, 102

[10] V. Badrinarayanan, P. Perez, F. Le Clerc, and L. Oisel, "Probabilistic color and adaptive multi-feature tracking with dynamically switched priority between cues," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007, pp. 1–8. 42

[11] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004. 11, 12

[12] A. O. Balan and M. J. Black, "An adaptive appearance model approach for model-based articulated object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 758–765. 22

[13] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l1 tracker using accelerated proximal gradient approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1830–1837. 8, 9, 44, 70

[14] I. Barrodale and F. D. Roberts, "An improved algorithm for discrete l_1 linear approximation," *SIAM Journal on Numerical Analysis*, vol. 10, no. 5, pp. 839–848, 1973. 47

[15] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009. 74

[16] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003. 21, 27

[17] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Toulminet, “A cooperative approach to vision-based vehicle detection,” in *Proceedings of Intelligent Transportation Systems*, 2001, pp. 207–212. 2

[18] J. Berclaz, F. Fleuret, and P. Fua, “Robust people tracking with global trajectory optimization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 744–750. 2, 3

[19] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, “Multiple object tracking using k-shortest paths optimization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1806–1819, 2011. 2

[20] W. Bian and D. Tao, “Biased discriminant euclidean embedding for content-based image retrieval,” *IEEE Transactions on Image Processing*, vol. 19, no. 2, pp. 545–554, 2010. 27

[21] S. Birchfield, “Elliptical head tracking using intensity gradients and color histograms,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1998, pp. 232–237. 46

[22] C. M. Bishop, “Training with noise is equivalent to Tikhonov regularization,” *Neural computation*, vol. 7, no. 1, pp. 108–116, 1995. 67

[23] M. J. Black and A. D. Jepson, “Eigentracking: Robust matching and tracking of articulated objects using a view-based representation,” *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998. 5

[24] Y. Bogomolov, G. Dror, S. Lapchev, E. Rivlin, and M. Rudzsky, “Classification of moving targets based on motion and appearance,” in *Proceedings of the British Machine Vision Conference*, 2003, pp. 44.1–44.10. 23

[25] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, “Visual object tracking using adaptive correlation filters,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550. 102

[26] A. Bosch, A. Zisserman, and X. Muoz, "Image classification using random forests and ferns," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007, pp. 1–8. 88

[27] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient nd image segmentation," *International Journal of Computer Vision*, vol. 70, no. 2, pp. 109–131, 2006. 86

[28] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004. 87

[29] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," in *Proceedings of the European Conference on Computer Vision*, 2002, pp. 707–720. 5

[30] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001. 13, 83

[31] K. Briechle and U. D. Hanebeck, "Template matching using fast normalized cross correlation," in *Proc. SPIE Optical Pattern Recognit. XII*, vol. 4387, 2001, pp. 95–102. xiv, 76

[32] R. Brunelli, *Template matching techniques in computer vision: theory and practice*. John Wiley & Sons, 2009. 85

[33] S. D. Buluswar and B. A. Draper, "Color machine vision for autonomous vehicles," *Engineering Applications of Artificial Intelligence*, vol. 11, no. 2, pp. 245–256, 1998. 2

[34] K. Cannons, "A review of visual tracking," *Department of Computer Science and Engineering, York University, Toronto, Canada, Technical Report. CSE-2008-07*, 2008. 4

[35] K. J. Cannons, J. M. Gryn, and R. P. Wildes, "Visual tracking using a pixelwise spatiotemporal oriented energy representation," in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 511–524. 23

[36] R. Caruana, “Multitask learning,” *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997. 14, 43, 118

[37] X. Chen, W. Pan, J. Kwok, and J. Carbonell, “Accelerated gradient method for multi-task sparse learning problem,” in *Proceedings of the IEEE International Conference on Data Mining*, 2009, pp. 746–751. 9, 33, 46, 51, 55, 56

[38] P. Chockalingam, N. Pradeep, and S. Birchfield, “Adaptive fragments-based tracking of non-rigid objects using level sets,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 1530–1537. 82

[39] R. T. Collins, Y. Liu, and M. Leordeanu, “Online selection of discriminative tracking features.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–43, 2005. xv, 23, 42, 44, 97

[40] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003. 9, 10, 45

[41] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 886–893. 61

[42] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, “Accurate scale estimation for robust visual tracking,” in *Proceedings of the British Machine Vision Conference*, 2014. 3, 102, 104, 106, 114

[43] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer, “Adaptive color attributes for real-time visual tracking,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1090–1097. 102, 105, 114

[44] C. Ding, C. Xu, and D. Tao, “Multi-task pose-invariant face recognition,” *IEEE Transactions on Image Processing*, 2015. 46

[45] T. B. Dinh, N. Vo, and G. Medioni, "Context tracker: Exploring supporters and distracters in unconstrained environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1177–1184. xiii, xv, 4, 23, 72, 96, 102

[46] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2012. 1, 2

[47] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006. 7

[48] A. Doucet, N. De Freitas, N. Gordon *et al.*, *Sequential Monte Carlo methods in practice*. Springer, 2001. 4, 25

[49] W. Du and J. Piater, "A probabilistic approach to integrating multiple cues in visual tracking," in *Proceedings of the European Conference on Computer Vision*, 2008, pp. 225–238. 42

[50] S. Duffner and C. Garcia, "Pixeltrack: a fast adaptive algorithm for tracking non-rigid objects," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2480–2487. 81, 82, 92, 93

[51] H. Ebbinghaus, *Memory: A contribution to experimental psychology*. Teachers college, Columbia university, 1913, no. 3. 111

[52] D. Eck and J. Schmidhuber, "A first look at music composition using lstm recurrent neural networks," *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 2002. 103

[53] A. Elgammal, R. Duraiswami, and L. S. Davis, "Probabilistic tracking in joint feature-spatial spaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2003, pp. 781–788. 23

[54] J. Fan, Y. Wu, and S. Dai, "Discriminative spatial attention for robust tracking," in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 480–493. 23

[55] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010. 1, 105

[56] D. Freedman and M. W. Turek, "Illumination-invariant tracking via graph cuts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 10–17. 3

[57] J. Gai and R. L. Stevenson, "Studentized dynamical system for robust object tracking," *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 186–199, 2011. 6

[58] J. Gall and V. Lempitsky, "Class-specific hough forests for object detection," in *Decision Forests for Computer Vision and Medical Image Analysis*, 2013, pp. 143–157. 13

[59] M. Godec, P. M. Roth, and H. Bischof, "Hough-based tracking of non-rigid objects," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 81–88. 13, 61, 82, 92, 93

[60] ——, "Hough-based tracking of non-rigid objects," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1245–1256, 2013. 4

[61] P. Gong, J. Ye, and C. Zhang, "Robust multi-task feature learning," in *Proceedings of the IEEE International Conference on Data Mining*, 2012, pp. 895–903. 43, 48, 49, 55

[62] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 260–267. xv, 83, 97

[63] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting." in *Proceedings of the British Machine Vision Conference*, vol. 1, no. 5, 2006, p. 6. 11, 12

[64] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proceedings of the European Conference on Computer Vision*, 2008, pp. 234–247. 13

[65] H. Grabner, J. Matas, L. Van Gool, and P. Cattin, "Tracking the invisible: Learning where the object might be," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1285–1292. 106

[66] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2013, pp. 6645–6649. 103

[67] N. Guan, D. Tao, Z. Luo, and J. Shawe-Taylor, "Mahnmf: Manhattan non-negative matrix factorization," *arXiv:1207.3438[stat.ML]*, 2012. 42, 47, 51, 53

[68] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 263–270. xiii, xiv, xv, 2, 12, 44, 62, 72, 75, 76, 85, 87, 92, 96, 101, 102, 114

[69] H. L. Harter, "The method of least squares and some alternatives: Part I," *International Statistical Review*, vol. 42, no. 2, pp. 147–174, 1974. 42, 46, 51

[70] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision.* Cambridge university press, 2003. 109

[71] X. He, D. Cai, S. Yan, and H. J. Zhang, "Neighborhood preserving embedding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2005, pp. 1208–1213. 32

[72] X. He, R. S. Zemel, and M. Á. Carreira-Perpiñán, "Multiscale conditional random fields for image labeling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004, pp. 695–702. 14, 83, 118

[73] J. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 583–596, 2015. 101, 102, 104, 105, 114

[74] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 702–715. xiii, xv, 72, 82, 85, 92, 96, 102

[75] J. Ho, K. C. Lee, M. H. Yang, and D. Kriegman, "Visual tracking using learned linear subspaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2004, pp. 782–789. 5, 22

[76] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 103

[77] Z. Hong, X. Mei, D. Prokhorov, and D. Tao, "Tracking via robust multi-task multi-view joint sparse representation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 649–656. 45, 51, 62, 87, 101

[78] Z. Hong, X. Mei, and D. Tao, "Dual-force metric learning for robust distracter-resistant tracker," in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 513–527. 46, 83, 102

[79] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Tracking using multilevel quantizations," in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 155–171. 44, 101

[80] C. Huang, B. Wu, and R. Nevatia, "Robust object tracking by hierarchical association of detection responses," in *Proceedings of the European Conference on Computer Vision*. Springer, 2008, pp. 788–801. 2, 3

[81] Q. Huang, M. Han, B. Wu, and S. Ioffe, "A hierarchical conditional random field model for labeling and segmenting images of street scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1953–1960. 83

[82] M. Isard and A. Blake, “Condensation-conditional density propagation for visual tracking,” *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998. 4, 46

[83] 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*, vol. 25, no. 10, pp. 1296–1311, 2003. 23, 42

[84] X. Jia, H. Lu, and M.-H. Yang, “Visual tracking via adaptive structural local sparse appearance model,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1822–1829. xiii, xv, 44, 72, 96, 114

[85] L. W. W. Y. Jiang, N., “Adaptive and discriminative metric differential tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1161–1168. 23, 34

[86] Z. Kalal, J. Matas, and K. Mikolajczyk, “Pn learning: Bootstrapping binary classifiers by structural constraints,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 49–56. xiii, xiv, 11, 14, 45, 72, 76

[87] Z. Kalal, K. Mikolajczyk, and J. Matas, “Forward-backward error: Automatic detection of tracking failures,” in *Proceedings of the International Conference on Pattern Recognition*, 2010, pp. 2756–2759. 85, 89

[88] ——, “Tracking-learning-detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012. xv, 4, 14, 83, 85, 90, 92, 93, 96, 101, 102, 108, 114

[89] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960. 4

[90] O. J. Karst, “Linear curve fitting using least deviations,” *Journal of the American Statistical Association*, vol. 53, no. 281, pp. 118–132, 1958. 46

[91] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale l 1-regularized least squares," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606–617, 2007. 33

[92] P. Kohli, J. Rihan, M. Bray, and P. H. Torr, "Simultaneous segmentation and pose estimation of humans using dynamic graph cuts," *International Journal of Computer Vision*, vol. 79, no. 3, pp. 285–298, 2008. 83, 87

[93] P. Kohli and P. H. Torr, "Dynamic graph cuts for efficient inference in markov random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2079–2088, 2007. 81, 87, 89

[94] R. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," in *Proceedings of the International Conference on Machine Learning*, 2002. 29

[95] M. Kristan and R. Pflugfelder et al., "The visual object tracking VOT2014 challenge results," in *Proceedings of the European Conference on Computer Vision Workshop*, 2014, pp. 1–27. 101, 104, 114

[96] S. Kwak, W. Nam, B. Han, and J. H. Han, "Learning occlusion with likelihoods for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011. 3, 109

[97] J. Kwon and K. M. Lee, "Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1208–1215. 82

[98] ——, "Visual tracking decomposition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1269–1276. xiii, xv, 19, 35, 42, 46, 62, 72, 96, 114

[99] ——, "Tracking by sampling trackers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 1195–1202. xiii, xiv, xv, 72, 75, 76, 96, 114

[100] ——, “Tracking by sampling and integrating multiple trackers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1428–1441, 2014. 46

[101] J. Kwon, J. Roh, K. M. Lee, and L. Van Gool, “Robust visual tracking with double bounding box model,” in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 377–392. 101

[102] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr, “Associative hierarchical crfs for object class image segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 739–746. 83

[103] L. Ladickỳ, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr, “What, where and how many? combining object detectors and crfs,” in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 424–437. 83

[104] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, “Convergence properties of the nelder–mead simplex method in low dimensions,” *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998. 87, 89

[105] E. Learned-Miller and L. Sevilla-Lara, “Distribution fields for tracking,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1910–1917. xiii, xv, 73, 97

[106] K. Lebeda, S. Hadfield, J. Matas, and R. Bowden, “Long-term tracking through failure cases,” in *Proceedings of the IEEE International Conference on Computer Vision Workshop*, 2013, pp. 153–160. 101, 102

[107] I. Leichter, M. Lindenbaum, and E. Rivlin, “Visual tracking by affine kernel fitting using color and object boundary,” in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2007, pp. 1–6. 10

[108] ——, “Tracking by affine kernel transformations using color and boundary cues,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 1, pp. 164–171, 2009. 10

[109] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1465–1479, 2006. 88

[110] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29–44, 2001. 88

[111] A. Levinshtein, C. Sminchisescu, and S. Dickinson, "Optimal contour closure by superpixel grouping," in *Proceedings of the European Conference on Computer Vision*. Springer, 2010, pp. 480–493. 84

[112] G. Li, L. Qin, Q. Huang, J. Pang, and S. Jiang, "Treat samples differently: Object tracking with semi-supervised online covboost," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 627–634. 13

[113] H. Li, C. Shen, and Q. Shi, "Real-time visual tracking using compressive sensing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1305–1312. 8, 9, 23, 26, 44, 46, 83

[114] X. Li, A. Dick, H. Wang, C. Shen, and A. van den Hengel, "Graph mode-based contextual kernels for robust svm tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 1156–1163. 12

[115] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, "A survey of appearance models in visual object tracking," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 4, pp. 58:1–58:48, 2013. 4, 42, 45, 80, 82, 101

[116] X. Li, W. Hu, Z. Zhang, X. Zhang, and G. Luo, "Robust visual tracking based on incremental tensor subspace learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007, pp. 1–8. 6

[117] X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, and J. Cheng, "Visual tracking via incremental log-euclidean riemannian subspace learning," in *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8. 6

[118] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Proceedings of the European Conference on Computer Vision Workshop*, 2014. 102, 104

[119] B. Liu, J. Huang, L. Yang, and C. Kulikowsk, "Robust tracking using local sparse appearance model and k-selection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1313–1320. xiii, xv, 46, 73, 96

[120] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski, "Robust and fast collaborative tracking with two stage sparse optimization," in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 624–637. 23

[121] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 978–994, 2011. 106

[122] J. Liu, S. Ji, and J. Ye, "Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2009, pp. 339–348. 56

[123] W. Liu and D. Tao, "Multiview hessian regularization for image annotation," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2676–2687, 2013. 42

[124] W. Liu, X. Tian, D. Tao, and J. Liu, "Constrained metric learning via distance gap maximization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010, pp. 518–524. 14, 29, 30, 118

[125] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157. 106

[126] ——, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. 106, 107

[127] B. D. Lucas, T. Kanade *et al.*, “An iterative image registration technique with an application to stereo vision.” in *Proceedings of the International Joint Conferences on Artificial Intelligence*, 1981, pp. 674–679. 108

[128] G. Marola, “Using symmetry for detecting and locating objects in a picture,” *Computer Vision, Graphics, and Image Processing*, vol. 46, no. 2, pp. 179–195, 1989. 2

[129] I. Matthews, T. Ishikawa, and S. Baker, “The template update problem,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 810–815, 2004. 4

[130] X. Mei, Z. Hong, D. Prokhorov, and D. Tao, “Robust multitask multiview tracking in videos,” *IEEE Transactions on Neural Networks and Learning Systems*, 2015. 114

[131] X. Mei and H. Ling, “Robust visual tracking using ℓ_1 minimization,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009. xi, 2, 4, 5, 7, 8, 9, 19, 23, 24, 33, 34, 35, 38

[132] ——, “Robust visual tracking and vehicle classification via sparse representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2259–2272, 2011. 44

[133] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, “Minimum error bounded efficient ℓ_1 tracker with occlusion detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1257–1264. xiv, 8, 23, 25, 35, 42, 43, 44, 46, 47, 48, 53, 59, 62, 76, 82, 83, 85, 101, 102

[134] X. Mei, S. K. Zhou, and F. Porikli, “Probabilistic visual tracking via robust template matching and incremental subspace update,” in *Proceedings of the*

IEEE International Conference on Multimedia and Expo, 2007, pp. 1818–1821. 42

[135] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Efficient minimum error bounded particle resampling l1 tracker with occlusion detection," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2661–2675, 2013. xiv, 46, 76, 83

[136] F. Moreno-Noguer, A. Sanfeliu, and D. Samaras, "Dependent multiple cue integration for robust tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 670–685, 2008. 46

[137] G. Nebehay and R. Pflugfelder, "Consensus-based matching and tracking of keypoints for object tracking," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2014, pp. 862–869. 106, 108, 110

[138] Y. Nesterov, "Gradient methods for minimizing composite objective function," *Center for Operations Research and Econometrics (CORE), Catholic Univ. Louvain, Louvain-la-Neuve, Belgium, CORE Discussion Paper 2007/76*, 2007. 42, 44, 55

[139] ——, "Smooth minimization of non-smooth functions," *Mathematical Programming*, vol. 103, no. 1, pp. 127–152, 2005. 43, 45, 52, 53, 54

[140] A. Ng and M. Jordan, "On discriminative vs. generative classifier: a comparison of logistic regression and naive bayes," in *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2001, pp. 841–848. 44

[141] H. T. Nguyen and A. W. Smeulders, "Robust tracking using foreground-background texture discrimination," *International Journal of Computer Vision*, vol. 69, no. 3, pp. 277–293, 2006. xiv, 76

[142] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, 2002. 61, 88

[143] S. Oron, A. Bar-Hille, and S. Avidan, "Extended lucas-kanade tracking," in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 142–156. 101

[144] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1940–1947. xv, 60, 96

[145] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 448–461, 2010. 13

[146] Y. Pang and H. Ling, "Finding the best from the second bests-inhibiting subjective bias in evaluation of visual tracking algorithms," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2784–2791. 12, 102

[147] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proceedings of the European Conference on Computer Vision*, 2002, pp. 661–675. xiii, xv, 73, 97

[148] F. Pernici and A. Del Bimbo, "Object tracking by oversampling local features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2538–2551, 2014. 101, 102, 103, 106, 107, 108, 109, 110

[149] N. Quadrianto and C. H. Lampert, "Learning multi-view neighborhood preserving projections," in *Proceedings of the International Conference on Machine Learning*, 2011, pp. 425–432. 42

[150] D. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1, pp. 125–141, 2008. xiv, 5, 6, 22, 35, 45, 62, 76, 81, 82, 85, 87, 101, 102

[151] C. Rother, V. Kolmogorov, and A. Blake, “Grabcut: Interactive foreground extraction using iterated graph cuts,” *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004. 14, 89

[152] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 22, pp. 232–2326, 2000. 21

[153] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, “On-line random forests,” in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2009, pp. 1393–1400. 13, 81, 83, 87, 89

[154] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, “Prost: Parallel robust online simple tracking,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 723–730. 83

[155] R. E. Schapire, “The strength of weak learnability,” *Machine learning*, vol. 5, no. 2, pp. 197–227, 1990. 12

[156] E. Schlossmacher, “An iterative technique for absolute deviations curve fitting,” *Journal of the American Statistical Association*, vol. 68, no. 344, pp. 857–859, 1973. 47

[157] C. Shen, M. J. Brooks, and A. Van Den Hengel, “Fast global kernel density mode seeking: Applications to localization and tracking,” *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1457–1469, 2007. 10

[158] J. Shi and C. Tomasi, “Good features to track,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 94, 1994, pp. 593–600. 23

[159] J. Shrager, T. Hogg, and B. Huberman, “Observation of phase transitions in spreading activation networks,” vol. 236, no. 4805, p. 1092, 1987. 29

[160] A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, “Visual tracking: An experimental survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2014. 4, 12, 42, 61, 74, 75, 101, 102, 115, 116

[161] B. Stenger, T. Woodley, and R. Cipolla, “Learning to track with multiple observers,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2647–2654. 46

[162] Z. Sun, G. Bebis, and R. Miller, “On-road vehicle detection: A review,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 694–711, 2006. 1

[163] J. S. Supancic III and D. Ramanan, “Self-paced learning for long-term tracking,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2379–2386. 101, 102, 103

[164] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” in *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2013, pp. 2553–2561. 1, 2

[165] X. Tan and B. Triggs, “Enhanced local texture feature sets for face recognition under difficult lighting conditions,” *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1635–1650, 2010. 61

[166] F. Tang, S. Brennan, Q. Zhao, and H. Tao, “Co-tracking using semi-supervised support vector machines,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2007, pp. 1–8. 11, 12

[167] D. Tao, X. Tang, X. Li, and Y. Rui, “Direct kernel biased discriminant analysis: a new content-based image retrieval relevance feedback algorithm,” *IEEE Transactions on Multimedia*, vol. 8, no. 4, pp. 716–727, 2006. 27

[168] P. H. Torr and A. Zisserman, “Mlesac: A new robust estimator with application to estimating image geometry,” *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000. 109

[169] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus, “Learning color names for real-world applications,” *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, 2009. 105

[170] L. Čehovin, M. Kristan, and A. Leonardis, “Robust visual tracking using an adaptive coupled-layer visual model,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 941–953, 2013. 114

[171] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. I–511. 12

[172] C. Wang, M. de La Gorce, and N. Paragios, “Segmentation, ordering and multi-object tracking using graphical models,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 747–754. 66, 83, 86

[173] C. Wang, N. Komodakis, and N. Paragios, “Markov random field modeling, inference & learning in computer vision & image understanding: A survey,” *Computer Vision and Image Understanding*, vol. 117, no. 11, pp. 1610–1627, 2013. 83

[174] H. Wang and N. Ahuja, “Rank-r approximation of tensors using image-as-matrix representation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 346–353. 6

[175] L. Wang, M. D. Gordon, and J. Zhu, “Regularized least absolute deviations regression and an efficient algorithm for parameter tuning,” in *Proceedings of the IEEE International Conference on Data Mining*, 2006, pp. 690–700. 47

[176] S. Wang, H. Lu, F. Yang, and M.-H. Yang, “Superpixel tracking,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 1323–1330. 81, 82, 88, 91, 92, 93

[177] X. Wang, G. Hua, and T. X. Han, “Discriminative tracking by metric learning,” in *Proceedings of the European Conference on Computer Vision*, 2010, pp. 200–214. 23

[178] C. Wojek and B. Schiele, "A dynamic conditional random field model for joint labeling of object and scene classes," in *Proceedings of the European Conference on Computer Vision*, 2008, pp. 733–747. 83

[179] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009. 7, 14, 118

[180] Y. Wu, J. Cheng, J. Wang, and H. Lu, "Real-time visual tracking via incremental covariance tensor learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 1631–1638. 6

[181] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2411–2418. xiii, 3, 12, 42, 61, 69, 70, 71, 72, 73, 80, 82, 92, 94, 95, 101, 102, 113, 114, 115

[182] Y. Wu, H. Ling, J. Yu, F. Li, X. Mei, and E. Cheng, "Blurred target tracking by blur-driven tracker," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 1100–1107. 3

[183] T. Xia, D. Tao, T. Mei, and Y. Zhang, "Multiview spectral embedding," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, no. 6, pp. 1438–1446, 2010. 42

[184] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *arXiv:1304.5634[cs.LG]*, 2013. 14, 45, 47, 118

[185] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, 2011. 3, 4, 45, 80, 82, 101

[186] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006. 1, 4

[187] A. Yilmaz, "Object tracking by asymmetric kernel mean shift with automatic scale and orientation selection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–6. 10

[188] J. Yoon, D. Kim, and K.-J. Yoon, “Visual tracking via adaptive tracker selection with multiple features,” in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 28–41. 46

[189] Q. Yu, T. Dinh, and G. Medioni, “Online tracking and reacquisition using co-trained generative and discriminative trackers,” in *Proceedings of the European Conference on Computer Vision*, 2008, pp. 678–691. 22

[190] X.-T. Yuan and S. Yan, “Visual classification with multi-task joint sparse representation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3493–3500. 46, 47

[191] C. Zhang and Z. Zhang, “A survey of recent advances in face detection,” Tech. rep., Microsoft Research, Tech. Rep., 2010. 1

[192] J. Zhang, S. Ma, and S. Sclaroff, “Meem: Robust tracking via multiple experts using entropy minimization,” in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 188–203. 114

[193] K. Zhang, L. Zhang, and M.-H. Yang, “Real-time compressive tracking,” in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 864–877. 60

[194] ——, “Real-time object tracking via online discriminative feature selections,” *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4664–4677, 2013. 44

[195] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, “Robust visual tracking via multi-task sparse learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2042–2049. 9, 42, 43, 44, 46, 47, 52, 53, 59, 62, 72, 83, 101

[196] ——, “Low-rank sparse learning for robust visual tracking,” in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 470–484. 9, 46

[197] T. Zhang, S. Liu, N. Ahuja, M.-H. Yang, and B. Ghanem, “Robust visual tracking via consistent low-rank sparse learning,” *International Journal of Computer Vision*, pp. 1–20, 2014. xiii, 71, 72

[198] W. Zhong, H. Lu, and M.-H. Yang, “Robust object tracking via sparsity-based collaborative model,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1838–1845. xiii, xv, 44, 72, 83, 96, 114

[199] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, “Learning with local and global consistency,” in *Proceedings of the Annual Conference on Neural Information Processing Systems*, vol. 16, 2004, pp. 321–328. 29

[200] X. Zhu and A. B. Goldberg, “Introduction to semi-supervised learning,” *Synthesis lectures on artificial intelligence and machine learning*, vol. 3, no. 1, pp. 1–130, 2009. 3

[201] H. Zou and T. Hastie, “Regularization and variable selection via the elastic net,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005. 74