5-2021

Deep Learning and Optimization in Visual Target Tracking

Mohammadreza Javanmardi
Utah State University

DEEP LEARNING AND OPTIMIZATION IN VISUAL TARGET TRACKING

by

Mohammadreza Javanmardi

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Computer Science

Approved:

Xiaojun Qi, Ph.D. Major Professor	Vladimir Kulyukin , Ph.D. Committee Member
Haitao Wang, Ph.D. Committee Member	John Edwards, Ph.D. Committee Member
Ziqi Song, Ph.D. Committee Member	D. Richard Cutler, Ph.D. Interim Vice Provost for Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2021

ABSTRACT

Deep Learning and Optimization In Visual Target Tracking

by

Mohammadreza Javanmardi, Doctor of Philosophy

Utah State University, 2021

Major Professor: Xiaojun Qi, Ph.D.
Department: Computer Science

Visual tracking is the process of estimating states of a moving object in a dynamic frame sequence. It has been considered as one of the most paramount and challenging topics in computer vision. Although numerous tracking methods have been introduced, developing a robust algorithm that can handle different challenges still remains unsolved. In this dissertation, we introduce four different trackers and evaluate their performance in terms of tracking accuracy on challenging frame sequences. Each of these trackers aims to address the drawbacks of their peers. The first developed method is called a structured multi-task multi-view tracking (SMTMVT) method, which exploits the sparse appearance model in the particle filter framework to track targets under different challenges. Specifically, we extract features of the target candidates from different views and sparsely represent them by a linear combination of templates of different views. Unlike the conventional sparse trackers, SMTMVT not only jointly considers the relationship between different tasks and different views but also retains the structures among different views in a robust multi-task multi-view formulation. The second developed method is called a structured group local sparse tracker (SGLST), which exploits local patches inside target candidates in the particle filter framework. Unlike the conventional local sparse trackers, the proposed optimization model in SGLST not only adopts local and spatial information of the target candidates

but also attains the spatial layout structure among them by employing a group-sparsity regularization term. To solve the optimization model, we propose an efficient numerical algorithm consisting of two subproblems with closed-form solutions. The third developed tracker is called a robust structured tracker using local deep features (STLDF). This tracker exploits the deep features of local patches inside target candidates and sparsely represents them by a set of templates in the particle filter framework. The proposed STLDF utilizes a new optimization model, which employs a group-sparsity regularization term to adopt local and spatial information of the target candidates and attain the spatial layout structure among them. To solve the optimization model, we adopt the alternating direction method of multiplier (ADMM) to design a fast and parallel numerical algorithm by deriving the augmented Lagrangian of the optimization model into two closed-form solution problems: the quadratic problem and the Euclidean norm projection onto probability simplex constraints problem. The fourth developed tracker is called an appearance variation adaptation (AVA) tracker, which aligns the feature distributions of target regions over time by learning an adaptation mask in an adversarial network. The proposed adversarial network consists of a generator and a discriminator network that compete with each other over optimizing a discriminator loss in a mini-max optimization problem. Specifically, the discriminator network aims to distinguish recent target regions from earlier ones by minimizing the discriminator loss, while the generator network aims to produce an adaptation mask to maximize the discriminator loss. We incorporate a gradient reverse layer in the adversarial network to solve the aforementioned mini-max optimization in an end-to-end manner. We compare the performance of the proposed four trackers with the most recent state-of-the-art trackers by doing extensive experiments on publicly available frame sequences, including OTB50, OTB100, VOT2016, and VOT2018 tracking benchmarks.

(99 pages)

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my supervisor Dr. Xiaojun Qi for her endless support, professional advice, and immense knowledge during the course of my doctoral studies. I am deeply indebted to her for teaching me invaluable skills that are directly applicable to every aspects of my personal and professional life. I would also like to extend my gratitude to my committee members Dr. Vladimir Kulyukin, Dr. Haitao Wang, Dr. John Edwards, and Dr. Ziqi Song for providing me insightful suggestions, invaluable feedback, and relentless supports.

I cannot be thankful enough to my family and friends, especially my parents, my aunt, and my sisters, whose endless love and unwavering belief made me the person I am today. Above all, I would like to thank my wife, Reem, for her constant love and encouragement throughout this journey.

Mohammadreza Javanmardi

CONTENTS

Page

ABSTRACT iii

ACKNOWLEDGMENTS v

LIST OF TABLES viii

LIST OF FIGURES ix

1 INTRODUCTION 1

2 Structured Multi-task Multi-view Tracking 7
2.1 Introduction 7
2.2 SMTMVT Optimization Model 8
2.3 SMTMVT Numerical Algorithm 10

3 Structured Group Local Sparse Tracker 12
3.1 Introduction 12
3.2 SGLST Optimization Model 13
3.3 SGLST Numerical Algorithm 16

4 Structured Tracker Using Local Deep Features 19
4.1 Introduction 19
4.2 STLDF Local Deep Feature Extraction 20
4.3 STLDF Optimization Model and Numerical Algorithm 21
4.4 STLDF Template Update 23
4.5 STLDF Summary 23

5 Appearance Variation Adaptation TrackerUsing Adversarial Network 25
5.1 Introduction 25
5.2 AVA Model 28
5.3 Network Architecture 31
5.4 Online AVA Tracking Settings 33

6 Experimental Results 35
6.1 Datasets 35
6.2 SMTMVT Results 36
6.2.1 Experimental Results on Publicly Available Sequences 37
6.2.2 Experimental Results on OTB50 40
6.2.3 Experimental Results on OTB100 42
6.3 SGLST Results 42
6.3.1 Experimental Results on Publicly Available Sequences 43
6.3.2 Experimental Results on OTB50 46

6.3.3 Experimental Results on OTB100 50
6.4 STLDF Results 53
6.4.1 Experimental Results on OTB50 54
6.4.2 Experimental Results on OTB100 57
6.5 AVA Results 61
6.5.1 Experimental Results on OTB50 61
6.5.2 Experimental Results on OTB100 63
6.5.3 Experimental Results on VOT2016 65
6.5.4 Experimental Results on VOT2018 66
6.5.5 Comparison and Discussion 67
6.6 Discussion 70

7 Conclusions 76

REFERENCES 79

CURRICULUM VITAE 87

LIST OF TABLES

Table Page

6.1 The average overlap score of the proposed SMTMVT method and eight compared methods for 15 sequences. Bold number in blue indicates the best performance, while red indicates the second best. 39

6.2 Summary of the average overlap scores of 11 state-of-the-art trackers and the two variants of the proposed SGLST on 16 sequences. The bold numbers in blue indicate the best performance, while the numbers in red indicate the second best. 45

6.3 Summary of the tracking performance of the proposed tracker, its two variants, and nine representative sparse trackers on OTB50. Bold numbers indicate the highest AUC and precision scores (i.e., the best tracking performance). 56

6.4 Comparison of the state-of-the-art trackers in terms of EAO, failure rate, and accuracy on the VOT2016 dataset. 66

6.5 Comparison of the state-of-the-art trackers in terms of EAO, failure rate, and accuracy on the VOT2018 dataset. 67

6.6 Comparison of the proposed AVA tracker with 11 state-of-the-art trackers on OTB50, OTB100, and VOT2016 challenging tracking benchmarks. Numbers in red, blue, green indicate the best, the second best, and the third best performance, respectively. The dash line (-) indicates no reported result. . . . 68

6.7 Comparison of the proposed trackers in terms of AUC scores on OTB50 and OTB100 tracking benchmarks 71

LIST OF FIGURES

Figure Page

1.1 Illustration of four sample tracking results for three videos containing a person (top row), a car (middle row), and a bike (bottom row). 1

1.2 Visual tracking challenges. 2

3.1 Illustration of the sparse representation of two sample local patches of the j^{th} target candidate in: (a) Conventional local sparse trackers [1, 2]. One local patch of the j^{th} target candidate, shown in the red bounding box, is represented by its corresponding patch in the first and the tenth templates, while another local patch of this candidate, shown in the blue bounding box, is represented by its corresponding patch in two different templates (e.g., the second and the ninth templates). (b) The proposed SGLST. Both local patches of the j^{th} target candidate, shown in red and blue bounding boxes, are represented by their corresponding patches in the same templates (e.g., the first and the tenth templates). 13

4.1 The overview of the proposed STLDF method. 24

5.1 Comparison of the proposed AVA tracker with CNN-SVM [3], MDNet [4], and VITAL [5] on three OTB100 sequences including *Jump* (Top), *CarScale* (Middle), and *Skating1* (Bottom). 27

5.2 The architecture of the proposed AVA tracker. The adversarial network consists of two networks: Generator and Discriminator. These two networks work hand-in-hand to learn an adaptation mask in a mini-max optimization problem to align the feature distributions of recent and earlier target regions up to the current frame. The classification network, including the feature extractor and three fully connected layers, is the same as the MDNet tracker. [4]. 31

6.1 Tracking results of different methods. The frame index is shown at the upper left corner of each frame. The frame sequences are 38

6.2 OTB50 overall success plot and the OTB50 success plot of the proposed SMTMVT and 31 trackers for each of 11 challenge subsets. Top 10 trackers are plotted. 40

6.3 OTB100 overall success plot and the OTB100 success plot of the proposed SMTMVT and 31 trackers for each of 11 challenge subsets. Top 10 trackers are plotted. 41

6.4 Comparison of the tracking results of 11 state-of-the-art trackers and the two variants of the proposed SGLST on *boy*, *faceocc1*, *faceocc2*, *girl*, *kitesurf*, and *surfer* image sequences. frame index is shown at the upper left corner of each representative frame. Results are best viewed on high-resolution displays. The tracking results of the top four trackers (i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines. 43

6.5 Comparison of the tracking results of 11 state-of-the-art trackers and the two variants of the proposed SGLST on *jogging1*, *crossing*, *human5*, *human7*, *walking2*, and *doll* image sequences. The frame index is shown at the upper left corner of each representative frame. Results are best viewed on high-resolution displays. The tracking results of the top four trackers (i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines. . 44

6.6 Comparison of the tracking results of 11 state-of-the-art trackers and the two variants of the proposed SGLST on *board*, *box*, *car4*, and *car2* image sequences. The frame index is shown at the upper left corner of each representative frame. Results are best viewed on high-resolution displays. The tracking results of the top four trackers (i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines. 44

6.7 OTB50 overall OPE success plots and the OPE success plots of top 10 trackers among 43 trackers on BC, DEF, FM, IPR, and OPR challenge subsets. The value shown in the title is the number of sequences in the specific subset. The value shown in the legend is the AUC score.The results of the other trackers can be found in [6]. 47

6.8 OTB50 OPE success plots of top 10 trackers among 43 trackers on 48

6.9 OTB100 overall OPE success plots and the OPE success plots of top 10 trackers among 37 trackers on BC, DEF, FM, IPR, and OPR challenge subsets. The value shown in the title is the number of sequences in the specific subset. The value shown in the legend is the AUC score. The results of the other trackers can be found in [7]. 51

6.10 OTB100 OPE success plots of top 10 trackers among 37 trackers on 52

6.11 The overall OPE plots of top 20 trackers among the proposed STLDF, its two variants, and 46 compared trackers for 50 frame sequences in OTB50. 55

6.12 The overall OPE plots blackof top 20 trackers among the proposed STLDF, its two variants, and 44 compared trackers for 100 frame sequences in OTB100. 58

6.13 The OPE success plots of top 20 trackers among the proposed STLDF, its two variants, and 44 compared trackers for LR, OPR, IV, OV, BC, SV, MB, OCC, and FM subsets in OTB100. 59

6.14 The OPE precision plots of top 20 trackers among the proposed STLDF, its two variants, and 44 compared trackers for LR, OPR, IV, OV, BC, SV, MB, OCC, and FM subsets in OTB100. 60

6.15 The overall OPE plots of top 10 trackers among the proposed AVA and 54 compared trackers for OTB50 benchmark [6]. 62

6.16 The overall OPE plots of top 10 trackers among the proposed AVA and 52 compared trackers for OTB100 benchmark [7]. 63

6.17 OTB100 OPE success plot of top 10 trackers among the proposed AVA and 53 compared trackers for DEF, IPR, OPR, OCC, LR, SV, MB, and OV challenge subsets. 64

6.18 OTB100 OPE success plot of our trackers for OCC, SV, DEF, FM, IPR, and MB challenge subsets. 72

6.19 Comparison of the tracking results of our proposed trackers on *bird1*, *bolt1*, *blurOwl*, and *jump* frame sequences. Results are best viewed on high-resolution displays. 73

6.20 Comparison of the tracking results of our proposed trackers on *coupon*, *carScale*, *dragonBaby*, and *human8* frame sequences. Results are best viewed on high-resolution displays. 74

CHAPTER 1

INTRODUCTION

Visual tracking aims to estimate states of a moving object or multiple objects in frame sequences under different conditions. Specifically, given the initial location of a target in a video (a frame sequence), visual trackers aim to estimate various properties (e.g., position, appearance, and shape) of the target to identify its location in each frame. Figure 1.1 shows four sample tracking results for three videos containing a person, a car, and a bike, respectively.

Visual tracking has been considered as one of the most active and challenging computer vision topics with a large array of applications in autonomous driving, video content analysis and understanding, surveillance, etc. Although some improvements have been achieved in several tracking methods [8–13], computer vision researchers still aim to develop more robust algorithms capable of handling various challenges including occlusion, illumination variations, in-plane and out-plane rotation, background clutter, deformation, and low resolution. Figure

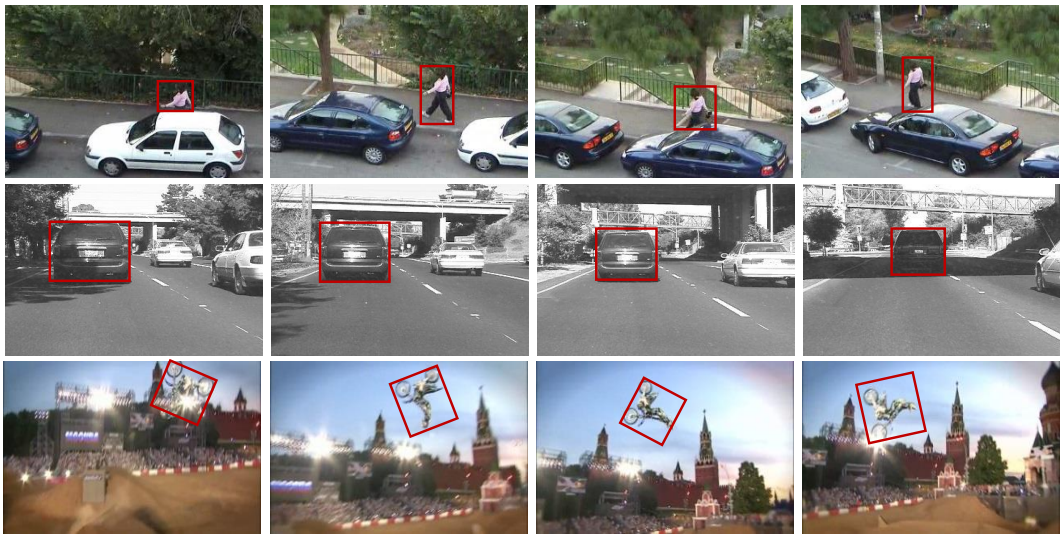
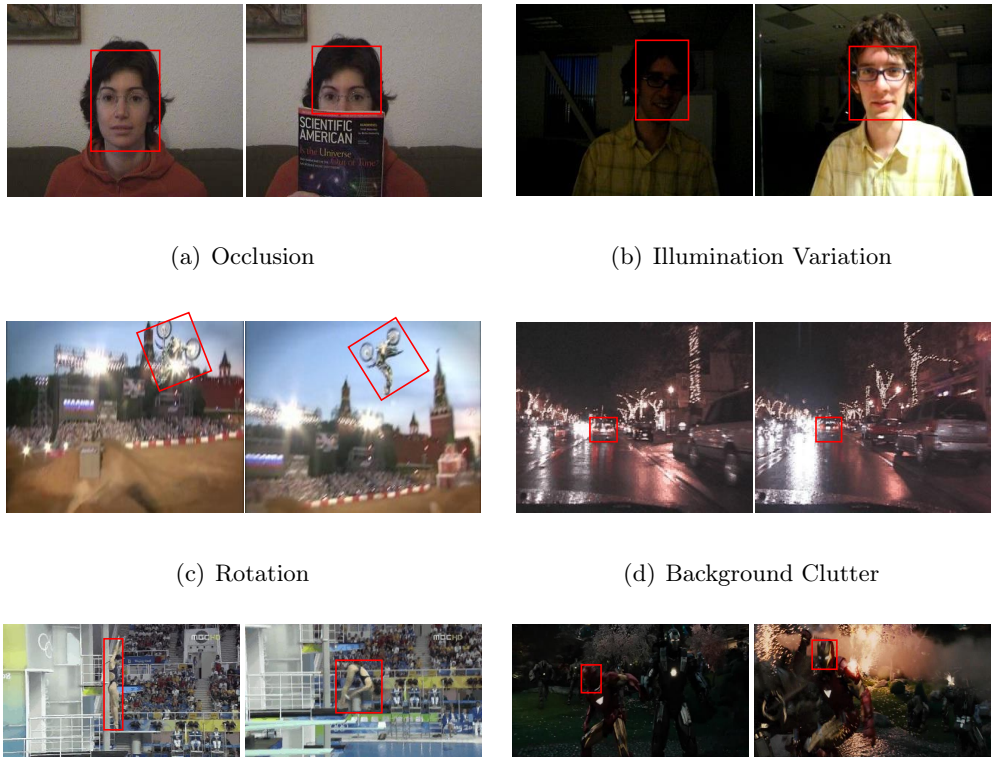


Fig. 1.1: Illustration of four sample tracking results for three videos containing a person (top row), a car (middle row), and a bike (bottom row).



(a) Occlusion (b) Illumination Variation

(c) Rotation (d) Background Clutter

(e) Deformation (f) Low resolution

Fig. 1.2: Visual tracking challenges.

1.2 presents these challenges in different videos. Figure 1.2(a) shows an example of occlusion where the book blocks the lower part of the person's face. Figure 1.2(b) shows an example of illumination variations where the person's face goes through a transition from in shadow to out of shadow. Figure 1.2(c) presents a scenario when the cyclist goes through different in-plane rotations. Figure 1.2(d) shows an example of a cluttered background under different lighting conditions. Figure 1.2(e) presents a scenario when the diver has a completely different pose (i.e., deformation) after jumping from the platform to do various stunts. Figure 1.2(f) shows an example of a low resolution video, where the objects and background are kind of blurry.

Visual tracking algorithms are classified to two different major categories: 1) hand-crafted feature based trackers; and 2) deep learning based trackers. For many years, hand-crafted feature based trackers have demonstrated favorable performance in visual target tracking.

One of the advantages of hand-crafted feature based trackers is that they do not require any additional learning steps to model the target regions with respect to background. This makes them a suitable choice, when a large number of data is not available. On the other hand, deep learning-based trackers have recently attracted researchers' attention to automatically extract target features as the increasing availability of data and computation powers becomes a reality.

Hand-crafted feature based trackers model the representation of the target using traditional features such as gray-level intensities, histogram of oriented gradients (HOG), and local binary pattern (LBP). One category of these trackers is discriminative, which utilizes these features to model the problem as binary classification and formulate a decision boundary to separate the target from backgrounds. Representative examples include the ensemble tracker [14], the online boosting [15, 16], the multiple instance learning [17], the PN-learning [18], and correlation filter-based trackers [19–21]. In contrast, another category of these trackers is generative, which adopts a model to represent the target and cast the tracking as a searching procedure to find the most similar region to the target model. Representative examples include eigen-tacking [22], mean-shift [23], Frag-Track [24], incremental learning [25], visual tracking decomposition [26], adaptive color tracking [27] and sparse-trackers [2, 28, 29].

Recently, convolutional neural network (CNN) based trackers [4, 5, 30–33] have shown state-of-the-art performance in terms of accuracy and robustness. One category of these trackers is deep feature trackers and utilizes pre-trained neural networks on a large-scale classification dataset [34] to extract deep features of target candidates. These features are then separately can be integrated in correlation filter-based trackers and sparse trackers [30, 35, 36]. One major advantage of these deep learning based trackers is obtaining generic feature representation of target regions without requiring an additional training step. However, due to the difference between tracking and classification tasks, these methods do not fully exploit the advantages of deep learning. Another category of trackers [4, 5] directly uses external videos to pretrain CNN models, which classify the target candidates to target

and background classes. As one of the representative works, Nam and Han [4] introduce the MDNet tracker, which pretrains a discriminative CNN using auxiliary sequences with tracking ground truths to obtain a generic object representation. Various trackers have then been proposed to improve the performance of MDNet by using a tree structure to manage multiple target appearance models [37], using adversarial learning to identify the mask that maintains the most robust features of the target objects over a long temporal span [5], and using reciprocative learning to exploit visual attention for training deep classifiers [31].

In this dissertation, we introduce four trackers, which have been developed during the course of my Ph.D. journey. Two of these trackers are categorized as hand-crafted feature based and the other two trackers are considered as deep learning based trackers. Each of the proposed trackers aim to address the drawback of their peers and improve the performance of their peers in terms of tracking accuracy.

We name the two proposed hand-crafted feature based tracking methods as a structured multi-task multi-view tracking (SMTMVT) method [38] and a structured group local sparse tracker (SGLST) [39]. The SMTMVT extracts features of the target candidates from different views such as color intensity, color histogram, HOG, LBP and sparsely represents them by a linear combination of templates of different views. Unlike the conventional sparse trackers, SMTMVT not only jointly considers the relationship between different tasks and different views but also retains the structures among different views in a robust multi-task multi-view formulation. We introduce a numerical algorithm based on the proximal gradient method to quickly and effectively find the sparsity by dividing the optimization problem into two subproblems with the closed-form solutions. The SGLST exploits local patches inside a target candidate and represents them in a novel convex optimization model. The proposed optimization model not only adopts local and spatial information of the target candidates but also attains the spatial layout structure among them by employing a group-sparsity regularization term. To solve the optimization model, we propose a fast and parallel numerical algorithm based on the alternating direction method of multiplier (ADMM), which consists of two subproblems with the closed-form solutions.

We name the two deep learning based trackers as a structured tracker using local deep features (STLDF) [40] and an appearance variation adaptation (AVA) tracker [41]. The STLDF tracker exploits the deep features of local patches inside target candidates and sparsely represents them by a set of templates in the particle filter framework. It utilizes a new optimization model, which employs a group-sparsity regularization term to adopt local and spatial information of the target candidates and attain the spatial layout structure among them. To solve the optimization model, we propose an efficient and fast numerical algorithm that consists of two subproblems with the closed-form solutions. The AVA tracker aligns the feature distributions of target regions over time by learning an adaptation mask in an adversarial network. The proposed adversarial network consists of a generator and a discriminator network that compete with each other over optimizing a discriminator loss in a mini-max optimization problem. Specifically, the discriminator network aims to distinguish recent target regions from earlier ones by minimizing the discriminator loss, while the generator network aims to produce an adaptation mask to maximize the discriminator loss. We incorporate a gradient reverse layer in the adversarial network to solve the aforementioned mini-max optimization in an end-to-end manner.

Throughout this dissertation, we use italic lowercase, bold lowercase, and bold uppercase letters to denote scalars, vectors, and matrices, respectively. For a column vector $\mathbf{x}$, we use $\mathbf{x}_i$ to represent the i^{th} element of $\mathbf{x}$ and $\text{diag}(\mathbf{x})$ to represent a diagonal matrix formed by the elements of $\mathbf{x}$. For a matrix $\mathbf{X}$, we use $\mathbf{X}_{i,j}$ to denote the element at the i^{th} row and j^{th} column. Two sets of numbers $\{1, 2, \ldots, n\}$ and $\{1, 2, \ldots, K\}$ are respectively denoted by $\mathcal{N}$ and $\mathcal{K}$. Vector $\mathbf{1}$ is a column vector of all ones of an appropriate dimension. For a given matrix $\mathbf{Y} \in \mathbb{R}^{n \times m}$, we denote its Frobenius norm, nuclear norm, L_1 norm, and the ℓ_p norm of ℓ_q norm of the rows in $\mathbf{Y}$ by $\|\mathbf{Y}\|_F$, $\|\mathbf{Y}\|_*$, $\|\mathbf{Y}\|_1$, and $\|\mathbf{Y}\|_{p,q}$, respectively. The *soft-thresholding operator* is defined as $S_\rho(\mathbf{Y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \text{sign}(\mathbf{Y}_{ij}) \max(|\mathbf{Y}_{ij}| - \rho, 0)$. For a set $\mathcal{Y}$, the indicator function $\delta_{\mathcal{Y}}(\mathbf{Y})$ returns $+\infty$ when $\mathbf{Y} \notin \mathcal{Y}$ and returns 0 when $\mathbf{Y} \in \mathcal{Y}$. The *proximal operator* is defined as $\text{Prox}_\sigma^f(\mathbf{Y}) = \text{argmin}_{\mathbf{Z}} f(\mathbf{Z}) + \frac{1}{2\sigma}\|\mathbf{Z} - \mathbf{Y}\|_F^2$, where $\sigma > 0$ and $f(\cdot)$ is a given function. In addition, we use $\text{tr}(\cdot)$ as the trace operator, $\mathbf{X} \otimes \mathbf{Y}$ as the

Kronecker product on two matrices $\mathbf{X}$ and $\mathbf{Y}$, $\mathbf{1}_l$ as a column vector of all ones, and $\mathbf{I}_k$ as a $k \times k$ identity matrix. It should be noted there is no relation between the variables in two trackers even with the same name.

The remainder of this dissertation is organized as follows: Chapters 2, 3, 4, and 5 introduce our proposed trackers, namely, SMTMVT, SGLST, STLDF, and AVA, respectively. Chapter 6 presents the experimental results of the proposed trackers on tracking benchmarks, compares their performance with the state-of-the-art trackers using appropriate evaluation metrics, and demonstrates several use cases of the proposed trackers. Chapter 7 draws a conclusion.

CHAPTER 2

Structured Multi-task Multi-view Tracking

2.1 Introduction

Exploiting multi-view information (e.g., intensity, edge, and histogram of target candidates) has been used in visual tracking. Hong et al. [28] propose a tracker that considers the underlying relationship among different views and particles in terms of the least square (LS) criteria. To handle the data contaminated by outliers and noise, Mei et al. [29] use the least absolute deviation (LAD) in their optimization model. Both approaches cannot retain the underlying layout structure among different views. In other words, different views of a target candidate may be reconstructed by activating the same templates in the dictionary set, whose representation coefficients do not resemble the similar combination of activated templates.

To address these issues, we propose a novel structured multi-task multi-view tracking (SMTMVT) method to track objects under different challenges. Similar to [28], SMTMVT exploits multi-view information such as intensity, edge, and histogram of target candidates and jointly represents them using templates. The proposed SMTMVT aims to address the aforementioned drawback of Hong's tracker [28] by proposing a new optimization model to attain the underlying layout structure among different views and reduce the error corresponding to outlier target candidates. The main contributions of the proposed work are summarized as:

- Designing a novel optimization model to effectively utilize a nuclear norm of the sparsity for multi-task multi-view sparse trackers.
- Representing a particular view of a target candidate as an individual task and simultaneously retaining the underlying layout structure among different views.

- Incorporating an outlier minimization term in the optimization model to efficiently reduce the error of outlier target candidates.
- Adopting the proximal gradient (PG) method to quickly and effectively solve the optimization problem.

In sections 2.2 and 2.3, we introduce the proposed SMTMVT optimization model and the numerical algorithm to solve the model, respectively.

2.2 SMTMVT Optimization Model

The proposed SMTMVT method utilizes the sparse appearance model to exploit multi-task multi-view information in a new optimization model, attain the underlying layout structure among different views, and reduce the error of outlier target candidates. At time t, we consider n particles with their corresponding image observations (target candidates). Using the state of the i-th particle, its observation is obtained by cropping the region of interest around the target. Each observation is considered to have K different views. For the k-th view, d_k dimensional feature vectors of all particles, $\{\mathbf{x}_i^k\}_{i\in\mathcal{N}}$, are combined to form the matrix $\mathbf{X}^k = [\mathbf{x}_1^k, \ldots, \mathbf{x}_n^k] \in \mathbb{R}^{d_k \times n}$ and N target templates are used to create its target dictionary $\mathbf{D}^k \in \mathbb{R}^{d_k \times N}$. Following the same notations in [28], we use the k-th dictionary $\mathbf{D}^k$ to represent the k-th feature matrix $\mathbf{X}^k$ and learn the sparsity $\mathbf{C}^k \in \mathbb{R}^{N \times n}$. In addition, We divide the reconstruction errors of the k-th view into two components as follows:

$$\mathbf{X}^k - \mathbf{D}^k\mathbf{C}^k = \mathbf{G}^k + \mathbf{E}^k \tag{2.1}$$

The first error component $\mathbf{G}^k \in \mathbb{R}^{d_k \times n}$ corresponds to the minor reconstruction errors resulted from the representation of good target candidates. The second error component $\mathbf{E}^k \in \mathbb{R}^{d_k \times n}$ corresponds to the significant reconstruction errors resulted from the representation of outlier target candidates. We use the Frobenius norm factor minimization of $\mathbf{G}^k$ error to minimize the square root of the sum of the absolute squares of its elements and adopt the ℓ_1 norm

minimization of $\mathbf{E}^k$ error to minimize the maximum column-sum of its elements. This assures the reconstruction errors for both good and bad target candidates are minimized.

To boost the performance, we maintain the underlying layout structure between K different views. For the i-th particle, we not only represent all its feature vectors $\{\mathbf{x}_i^k\}_{k\in\mathcal{K}}$ by activating the same subset of target templates in the target dictionaries (i.e., $\{\mathbf{D}^k\}_{k\in\mathcal{K}}$), but also equalize the representation coefficients of activated templates for all K views. In other words, we aim to resemble the ith columns of $\mathbf{C}^k$s, for $k \in \mathcal{K}$, to have a similar representation structure in terms of the activated templates and similar coefficients in terms of the activated values. To do so, we concatenate $\mathbf{C}^k$s to form $\mathbf{C} \in \mathbb{R}^{N\times(nK)}$, which is the sparsity matrix corresponding to the representation of K views in n observations. We then minimize the nuclear norm of the matrix $\Pi_{i,n}(\mathbf{C})$, which is a good surrogate of the rank minimization, to ensure the columns to be similar or linearly dependent of each other. Here, $\Pi_{i,n}(\mathbf{C})$ selects a sub-matrix of columns of $\mathbf{C}$, whose index belongs to the set $\{(l-1)n+i\}_{l\in\mathcal{K}}$. The selected columns are the simultaneous columns of the i-th target candidate in different views.

We formulate the SMTMVT sparse appearance model as the following optimization problem by jointly evaluating its K view matrices $\{\mathbf{X}_i^k\}_{k\in\mathcal{K}}$ with n different particles (tasks):

$$\underset{\mathbf{C},\mathbf{E},\{\mathbf{G}_k\}}{\text{minimize}} \quad \sum_{k\in\mathcal{K}}\left\|\mathbf{G}^k\right\|_F^2+\lambda\sum_{i\in\mathcal{N}}\|\Pi_{i,n}(\mathbf{C})\|_*+\gamma(\|\mathbf{C}\|_1+\|\mathbf{E}\|_1) \tag{2.2a}$$

$$\text{subject to} \quad \mathbf{X}^k = \mathbf{D}^k\mathbf{C}^k + \mathbf{G}^k + \mathbf{E}^k, \quad k \in \mathcal{K} \tag{2.2b}$$

$$\mathbf{C} \geq 0 \tag{2.2c}$$

where $\mathbf{E}^k$'s are vertically stacked to form the bigger matrix $\mathbf{E} \in \mathbb{R}^{(\sum_{k\in\mathcal{K}} d_k)\times n}$, parameter λ regularizes the nuclear norm of $\mathbf{C}$, and parameter γ controls the sparsity of $\mathbf{C}$ and $\mathbf{E}$.

Due to the equality constraint (2.2b), we eliminate $\mathbf{G}^k$ from (2.2a) and equivalently solve the following problem:

$$\begin{aligned} \underset{\mathbf{C},\mathbf{E}}{\text{minimize}} \quad & \sum_{k\in\mathcal{K}} \left\| \mathbf{X}^k - \mathbf{D}^k\mathbf{C}^k - \mathbf{E}^k \right\|_F^2 + \lambda \sum_{i\in\mathcal{N}} \|\Pi_{i,n}(\mathbf{C})\|_* \\ & + \gamma(\mathbf{1}^\top\mathbf{C}\mathbf{1} + \|\mathbf{E}\|_1) \qquad (2.3a) \\ \text{subject to} \quad & \mathbf{C} \geq 0 \qquad (2.3b) \end{aligned}$$

Finally, we compute the likelihood of the i-th candidate as follows:

$$p_i = \exp(-\alpha \sum_{k\in\mathcal{K}} \left\| \mathbf{D}^k\mathbf{c}_i^k - \mathbf{x}_i^k \right\|^2) \qquad (2.4)$$

where $\mathbf{c}_i^k$ is the sparse coefficients of the i-th candidate corresponding to the target templates in the k-th view and α is a constant value. We select the candidate with the highest likelihood value as the tracking result at frame t. Similar to [42], we update the target templates to handle the appearance changes of the target throughout the frame sequences.

2.3 SMTMVT Numerical Algorithm

Since the convex problem in (2.3) can be split into differentiable and non-differentiable subproblems, we adopt the proximal gradient method [43] to develop a numerical solution to the proposed model. To do so, we cast the differentiable subproblem as follows:

$$\begin{aligned} \mathcal{L}(\mathbf{C},\mathbf{E}) = & \sum_{k\in\mathcal{K}} \left\| \mathbf{X}^k - \mathbf{D}^k\mathbf{C}^k - \mathbf{E}^k \right\|_F^2 + \lambda \sum_{i\in\mathcal{N}} \|\Pi_{i,n}(\mathbf{C})\|_* \\ & + \gamma\,(\mathbf{1}^\top\mathbf{C}\mathbf{1}) \end{aligned} \qquad (2.5)$$

This equation is sub-differentiable with respect to $\mathbf{C}$ and differentiable with respect to $\mathbf{E}$. Hence, two variables $\mathbf{C}$ and $\mathbf{E}$ can be updated at time $t+1$ by the following equations:

$$\mathbf{E}^{(t+1)} := \mathrm{Prox}_\sigma^{f_\mathbf{E}}(\mathbf{E}^{(t)} - \sigma\,\nabla\mathcal{L}(\mathbf{C}^{(t)},\mathbf{E}^{(t)})), \qquad (2.6a)$$

$$\mathbf{C}^{(t+1)} := \mathrm{Prox}_\sigma^{f_\mathbf{C}}(\mathbf{C}^{(t)} - \sigma\,\nabla\mathcal{L}(\mathbf{C}^{(t)},\mathbf{E}^{(t)})), \qquad (2.6b)$$

where the step-size σ controls the convergence rate, $\nabla\mathcal{L}(\cdot,\cdot)$ is the sub-gradient operator, $f_{\mathbf{E}} = \|\mathbf{E}\|_1$, and $f_{\mathbf{C}} = \delta_{\mathcal{C}}$. We adopt the computationally efficient PG algorithm to iteratively update $\mathbf{E}$ and $\mathbf{C}$, which are initially set as 0's, until they converge to the constant matrices. Both subproblems (2.6a) and (2.6b) can be easily solved via the existing methods. Specifically, (2.6a) is a ℓ_1-minimization problem with an analytic solution, which can be obtained using soft thresholding, i.e., $S_{(\sigma\gamma)}(\cdot)$. Moreover, (2.6b) is an Euclidean norm projection onto the nonnegative orthant, which enjoys the closed-form solution. It should be emphasized that the convergence rate of this numerical algorithm can be further improved by the acceleration techniques presented in [44].

CHAPTER 3

Structured Group Local Sparse Tracker

3.1 Introduction

To further improve the tracking performance, recent sparse trackers consider both global and local information of all target candidates in their optimization models. Zhang et al. [12] represent local patches inside all target candidates along with the global information using $\ell_{1,2}$ norm regularization on the sparse representative matrix. They assume that the same local patches of all target candidates are similar. However, this assumption does not hold in practice due to outlier candidates and occlusion in tracking. To address this shortcoming, Zhang et al. [30] take into account both factors to design an optimal target region searching method. These recent sparse trackers achieve improved performance. However, considering the relationship of all target candidates degrades the performance when drifting occurs. In addition, using $\ell_{1,2}$ norm regularization in the optimization model to integrate both local and global information of target candidates lessens the tracking accuracy in the cases of heavy occlusions.

We propose a structured group local sparse tracker (SGLST), which exploits local patches inside a target candidate and represents them in a novel convex optimization model. The proposed optimization model not only adopts local and spatial information of the target candidates but also attains the spatial layout structure among them by employing a group-sparsity regularization term. The main contributions of the proposed work are summarized as follows:

- Proposing a local sparse tracker, which employs local and spatial information of a target candidate and attains the spatial structure among different local patches inside a target candidate.

- Developing a convex optimization model, which introduces a group-sparsity regularization term to motivate the tracker to select the corresponding local patches of the same small subset of templates to represent the local patches of each target candidate.

- Designing a fast and parallel numerical algorithm based on the alternating direction method of multiplier (ADMM), which consists of two subproblems with the closed-form solutions.

In sections 3.2 and 3.3, we introduce the proposed SGLST optimization model and the numerical algorithm to solve the model, respectively.

3.2 SGLST Optimization Model

Conventional local sparse trackers [1, 2] individually represent local patches without considering their spatial layout structure. For instance, local patches in [1] are separately represented by solving the Lasso problem. As a consequence, local patches inside the j^{th} target candidate may be sparsely represented by the corresponding local patches inside *different* dictionary templates, as illustrated in Figure 3.1, where two local patches of the

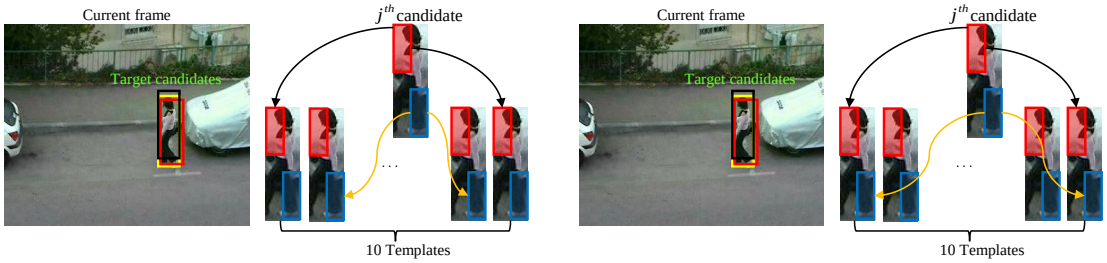


Fig. 3.1: Illustration of the sparse representation of two sample local patches of the j^{th} target candidate in: (a) Conventional local sparse trackers [1, 2]. One local patch of the j^{th} target candidate, shown in the red bounding box, is represented by its corresponding patch in the first and the tenth templates, while another local patch of this candidate, shown in the blue bounding box, is represented by its corresponding patch in two different templates (e.g., the second and the ninth templates). (b) The proposed SGLST. Both local patches of the j^{th} target candidate, shown in red and blue bounding boxes, are represented by their corresponding patches in the same templates (e.g., the first and the tenth templates).

j^{th} target candidate, shown in the red and blue bounding boxes, may be represented by the corresponding local patches in different dictionary templates.

We propose a novel SGLST that adopts both local and spatial information of the target candidates for tracking. It employs a new optimization model in the particle filter framework to address aforementioned issues associated with conventional local sparse trackers [1, 2] by attaining the spatial layout structure among different local patches inside a target candidate. Specifically, SGLST formulates an optimization problem to impose a structure on the achieved sparse vectors for different local patches inside each target candidate and attain the spatial layout structure among the local patches. To solve the proposed model, we develop an efficient numerical algorithm consisting of two subproblems with closed-form solutions by adopting the alternating direction method of multiplier (ADMM) within each target candidate in the optimization function.

To maintain the spatial layout structure among local patches, we jointly represent all the local patches of a target candidate in a new convex optimization model. In other words, if the r^{th} local patch of the j^{th} target candidate is best represented by the r^{th} local patch of the q^{th} template, the s^{th} local patch of the j^{th} target candidate should also be best represented by the s^{th} local patch of the q^{th} template. As shown in Figure 3.1, we aim to represent both local patches of the j^{th} target candidate, shown in the red and blue bounding boxes, by their corresponding patches in the *same* dictionary templates (e.g., the first and the tenth templates).

To do so, we first use k target templates and extract l overlapping d dimensional local patches inside each template to construct the dictionary $\mathbf{D}$. Such a representation generates the local dictionary matrix $\mathbf{D} = [\mathbf{D}_1, \ldots, \mathbf{D}_k] \in \mathbb{R}^{d\times(lk)}$, where $\mathbf{D}_i \in \mathbb{R}^{d\times l}$. Then, we construct a matrix $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_n] \in \mathbb{R}^{d\times(ln)}$, which contains the local patches of all the target candidates, where n is the number of particles. Next, we define the sparse matrix coefficients $\mathbf{C}$ corresponding to the j^{th} target candidate as $\mathbf{C} \triangleq \begin{bmatrix} \mathbf{C}_1 & \cdots & \mathbf{C}_k \end{bmatrix}^\top \in \mathbb{R}^{(lk)\times l}$, where $\{\mathbf{C}_q\}_{q=1}^k$ is a $l \times l$ matrix indicating the group sparse representation of l local patches of the j^{th} target candidate using l local patches of the q^{th} template. Finally, we formulate

the following convex model:

$$\underset{\mathbf{C}\in\mathbb{R}^{(lk)\times l}}{\text{minimize}} \quad \|\mathbf{X}_j-\mathbf{D}\mathbf{C}\|_{\mathrm{F}}^2+\lambda\left\|\left[\mathbf{C}_1(:)\dots\mathbf{C}_k(:)\right]^\top\right\|_{1,\infty} \tag{3.1a}$$

$$\text{subject to} \quad \mathbf{C}\geq 0, \tag{3.1b}$$

$$\mathbf{1}_{lk}^\top\mathbf{C}=\mathbf{1}_l^\top, \tag{3.1c}$$

The first term in (3.1a) represents the similarity measurement between the feature matrix $\mathbf{X}_j$ and its representation using the dictionary $\mathbf{D}$. The second term is a group-sparsity regularization term, which is a penalization term in the objective function to select dictionary words (templates). This term also establishes the $\|\cdot\|_{1,\infty}$ minimization on matrix $\left[\mathbf{C}_1(:)\dots\mathbf{C}_k(:)\right]^\top$, which leads to imposing local features inside a target candidate to choose similar few dictionary words (templates). It should be noted that each group ($\mathbf{C}_q$ of $l\times l$) is vectorized via $\mathbf{C}_q(:)$ and is represented by a column vector. The sum of the maximum absolute values per group is minimized by imposing $\|\cdot\|_{1,\infty}$. Therefore, the l_1 norm minimization selects few dictionary words for representation by imposing the rows of $\left[\mathbf{C}_1(:)\dots\mathbf{C}_k(:)\right]^\top$ to be sparse. The l_∞ norm minimization on the columns of $\left[\mathbf{C}_1(:)\dots\mathbf{C}_k(:)\right]^\top$ motivates the group of local patches to jointly select similar few templates. The parameter $\lambda>0$ is a trade-off between the first and the second terms. The constraint (3.1b) ensures sparse coefficients to be non-negative since a tracking target can be represented by target templates dominated by non-negative coefficients [42]. The constraint (3.1c) ensures that each local feature vector in $\mathbf{X}_j$ is expressed by at least one selected local feature vector of the dictionary $\mathbf{D}$.

For each target candidate, we find the sparse matrix $\mathbf{C}$ using the numerical algorithm presented in subsection 3.3. We then perform an averaging process along with an alignment pooling strategy [1] to find a representative vector. Finally, we calculate the summation of this representative vector as the likelihood value. The candidate with the highest likelihood value is selected as the tracking result. We also update the templates throughout the sequence using the same strategy as proposed in [1] to handle the appearance variations of the target region.

3.3 SGLST Numerical Algorithm

This section presents a numerical algorithm based on the ADMM [45] to efficiently solve the proposed model (3.1). The idea of the ADMM is to utilize auxiliary variables to convert a complicated convex problem to smaller sub-problems, where each one is efficiently solvable via an explicit formula. The ADMM iteratively solves the sub-problems until convergence. To do so, we first define vector $\mathbf{m} \in \mathbb{R}^k$ such that $\mathbf{m}_i = |\,\mathbf{C}_i(:)|$ and rewrite (3.1) as:

$$\begin{aligned}
\underset{\substack{\mathbf{C}\in\mathbb{R}^{(lk)\times l}\\ \mathbf{m}\in\mathbb{R}^k}}{\text{minimize}} \quad & \|\mathbf{X}_j - \mathbf{D}\mathbf{C}\|_{\mathrm{F}}^2 + \lambda \mathbf{1}_k^\top \mathbf{m} && \text{(3.2a)}\\
\text{subject to} \quad & \mathbf{C} \geq 0, && \text{(3.2b)}\\
& \mathbf{1}_{(lk)}^\top \mathbf{C} = \mathbf{1}_l^\top, && \text{(3.2c)}\\
& \mathbf{m} \otimes \mathbf{1}_l \mathbf{1}_l^\top \geq \mathbf{C}. && \text{(3.2d)}
\end{aligned}$$

It should be noted that constraint (3.2d) is imposed in the above reformulation to ensure the equivalence between (3.1) and (3.2). This inequality constraint can be transformed into an equality one by introducing a non-negative slack matrix $\mathbf{U} \in \mathbb{R}^{(lk)\times l}$, which compensates the difference between $\mathbf{m} \otimes \mathbf{1}_l \mathbf{1}_l^\top$ and $\mathbf{C}$. Using the resultant equality constraint, $\mathbf{1}_k^\top \mathbf{m}$ can be equivalently written as $\frac{1}{l^2}\mathbf{1}_{(lk)}^\top(\mathbf{C}+\mathbf{U})\mathbf{1}_l$. Moreover, this equality constraint implies that the columns of $\mathbf{C}+\mathbf{U}$ are regulated to be identical. Hence, one can simply replace it by a linear constraint independent of $\mathbf{m}$ as presented in (3.3d). Therefore, we rewrite (3.2) independent of $\mathbf{m}$ as:

$$\begin{aligned}
\underset{\mathbf{C},\mathbf{U}\in\mathbb{R}^{(lk)\times l}}{\text{minimize}} \quad & \|\mathbf{X}_j - \mathbf{D}\mathbf{C}\|_{\mathrm{F}}^2 + \frac{\lambda}{l^2}\mathbf{1}_{(lk)}^\top(\mathbf{C}+\mathbf{U})\mathbf{1}_l && \text{(3.3a)}\\
\text{subject to} \quad & \mathbf{C} \geq 0, && \text{(3.3b)}\\
& \mathbf{1}_{(lk)}^\top \mathbf{C} = \mathbf{1}_l^\top, && \text{(3.3c)}\\
& \mathbf{E}(\mathbf{C}+\mathbf{U}) = \frac{\mathbf{I}_k \otimes \mathbf{1}_l\mathbf{1}_l^\top}{l}(\mathbf{C}+\mathbf{U}), && \text{(3.3d)}\\
& \mathbf{U} \geq 0, && \text{(3.3e)}
\end{aligned}$$

where matrix $\mathbf{E}$ serves as the right circular shift operator on the rows of $\mathbf{C}+\mathbf{U}$. To construct the ADMM formulation, whose subproblems possess closed-form solutions, we define auxiliary variables $\hat{\mathbf{C}}, \hat{\mathbf{U}} \in \mathbb{R}^{(lk)\times l}$ and reformulate (3.3) as:

$$
\begin{aligned}
\underset{\mathbf{C},\hat{\mathbf{C}},\mathbf{U},\hat{\mathbf{U}}\in\mathbb{R}^{(lk)\times l}}{\text{minimize}} \quad & \|\mathbf{X}_j - \mathbf{D}\mathbf{C}\|_{\mathrm{F}}^2 + \frac{\lambda}{l^2}\mathbf{1}_{(lk)}^{\top}(\mathbf{C}+\mathbf{U})\mathbf{1}_l \\
& + \frac{\mu_1}{2}\left\|\mathbf{C}-\hat{\mathbf{C}}\right\|_{\mathrm{F}}^2 + \frac{\mu_2}{2}\left\|\mathbf{U}-\hat{\mathbf{U}}\right\|_{\mathrm{F}}^2 && \text{(3.4a)} \\
\text{subject to} \quad & \hat{\mathbf{C}} \geq 0, && \text{(3.4b)} \\
& \mathbf{1}_{(lk)}^{\top}\hat{\mathbf{C}} = \mathbf{1}_l^{\top}\ , && \text{(3.4c)} \\
& \mathbf{E}(\mathbf{C}+\mathbf{U}) = \frac{\mathbf{I}_k \otimes \mathbf{1}_l\mathbf{1}_l^{\top}}{l}(\mathbf{C}+\mathbf{U}), && \text{(3.4d)} \\
& \hat{\mathbf{U}} \geq 0, && \text{(3.4e)} \\
& \mathbf{C} = \hat{\mathbf{C}}, \quad \mathbf{U} = \hat{\mathbf{U}}. && \text{(3.4f)}
\end{aligned}
$$

where $\mu_1, \mu_2 > 0$ are the augmented Lagrangian parameters. Without loss of generality, we assume $\mu = \mu_1 = \mu_2$ [45]. The last two terms in the objective function (3.4a) are then vanished for any feasible solutions, which implies (3.3) and (3.4) are equivalent. We further form the augmented Lagrangian function to solve (3.4) as follows:

$$
\begin{aligned}
\mathcal{L}_{\mu}(\mathbf{C},\mathbf{U},\hat{\mathbf{C}},\hat{\mathbf{U}},\mathbf{\Lambda}_1,\mathbf{\Lambda}_2) = \ & \|\mathbf{X}_j - \mathbf{D}\mathbf{C}\|_{\mathrm{F}}^2 + \frac{\lambda}{l^2}\mathbf{1}_{(lk)}^{\top}(\mathbf{C}+\mathbf{U})\mathbf{1}_l \\
& + \frac{\mu}{2}\left\|\mathbf{C}-\hat{\mathbf{C}}+\frac{\mathbf{\Lambda}_1}{\mu}\right\|_{\mathrm{F}}^2 + \frac{\mu}{2}\left\|\mathbf{U}-\hat{\mathbf{U}}+\frac{\mathbf{\Lambda}_2}{\mu}\right\|_{\mathrm{F}}^2
\end{aligned}
\tag{3.5}
$$

where $\mathbf{\Lambda}_1, \mathbf{\Lambda}_2 \in \mathbb{R}^{(lk)\times l}$ are the Lagrangian multipliers corresponding to the equations in (3.4f).

Given initialization for $\hat{\mathbf{C}}$, $\hat{\mathbf{U}}$, $\mathbf{\Lambda}_1$, and $\mathbf{\Lambda}_2$ at time $t = 0$ (e.g., $\hat{\mathbf{C}}^0, \hat{\mathbf{U}}^0, \mathbf{\Lambda}_1^0, \mathbf{\Lambda}_2^0$), (3.5) is solved through the ADMM iterations. At the next iteration, $\mathbf{C}$ and $\mathbf{U}$ are updated by minimizing (3.5) under the constraint (3.4d). To do so, we first define $\{\mathbf{z}_i\}_{i=1}^{lk}$, where $\mathbf{z}_i \in \mathbb{R}^{2l}$ is obtained by stacking the i^{th} rows of $\mathbf{C}$ and $\mathbf{U}$. We then divide this minimization problem into lk equality constrained quadratic programs, where each program has its analytical solution. Using the updated $\mathbf{C}$ and $\mathbf{U}$, we compute $\hat{\mathbf{C}}$ and $\hat{\mathbf{U}}$ by minimizing (3.5) with the constraints

(3.4b), (3.4c), (3.4e). To this end, we split the problem into two separate subproblems with closed-form solutions over $\hat{\mathbf{C}}$ and $\hat{\mathbf{U}}$, where the first subproblem consists of l independent Euclidean norm projections onto the probability simplex constraints and the second subproblem consists of l independent Euclidean norm projections onto the non-negative orthant. Finally, we update $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ by performing l parallel updates over their respective columns. All these iterative updates can be quickly performed due to the closed-form solutions.

CHAPTER 4

Structured Tracker Using Local Deep Features

4.1 Introduction

Deep features extracted from convolutional neural networks have been recently utilized in visual tracking to obtain a generic and semantic representation of target candidates. We propose a robust structured tracker using local deep features (STLDF). This tracker exploits the deep features of local patches inside target candidates and sparsely represents them by a set of templates in the particle filter framework. Unlike the conventional local sparse trackers [2], the proposed optimization model in STLDF employs a group-sparsity regularization term to adopt local and spatial information of the target candidates and attain the spatial layout structure among them. The major contributions of the proposed work are summarized as follows:

- Proposing a deep features-based structured local sparse tracker, which employs CNN deep features of the local patches within a target candidate and keeps the relative spatial structure among the local deep features of a target candidate.
- Developing a convex optimization model, which combines nine local features of each target candidate with a group-sparsity regularization term to encourage the tracker to sparsely select appropriate local patches of the same subset of templates.
- Designing a fast and parallel numerical algorithm by deriving the augmented Lagrangian of the optimization model into two close-form problems: the quadratic problem and the Euclidean norm projection onto probability simplex constraints problem by adopting the alternating direction method of multiplier (ADMM).
- Utilizing the accelerated proximal gradient (APG) method to update the CNN deep feature-based template by casting it as a Lasso problem.

The preliminary results of this work are presented in SGLST, which is based on hand-crafted features. We made a number of improvements in the proposed method: (i) STLDF automatically extracts representative local deep features of target candidates using the pre-trained CNN. (ii) STLDF efficiently derives the augmented Lagrangian of the optimization model into two close-form problems: the quadratic problem and the Euclidean norm projection onto probability simplex constraints problem. (iii) STLDF updates the CNN deep feature-based template by casting it as a Lasso problem and numerically solving it using the accelerated proximal gradient (APG) method.

In sections 4.2, 4.3, 4.4 and 4.5, we introduce local deep feature extraction, the optimization model along with its numerical method, template update, and the summary of the proposed tracker.

4.2 STLDF Local Deep Feature Extraction

Inspired by [39], we select l overlapping local patches in k target templates and extract a d-dimensional feature vector for each patch. Unlike [39], which utilized hand crafted features such as color intensity and HOG features, we extract local deep features using a pre-trained CNN model. To this end, we set the size of each target candidate to 64×64 pixels to contain sufficient object-level information with decent resolution. Each target candidate is passed to the pre-trained VGG19 [34] network on the large-scale ImageNet dataset [46] to automatically extract their representative features. This network has been proven to achieve better tracking performance than other CNNs such as AlexNet since its strengthened semantic with deeper architecture is more insensitive to significant appearance change. Its default input size of $224 \times 224 \times 3$ has also been used in other VGG19-based trackers [47], [30] to achieve good tracking results. To ensure fair comparison with other VGG19-based trackers, we resize each target candidate to this default input size before forward propagation. We utilize the output of the *Conv5-4* layer as the feature map of the target candidate since the fifth layer is proven to be effective in discriminating the targets even with dramatic background changes [47]. The generated feature map has a size of $7 \times 7 \times 512$, which is not large enough to provide spatial information of target candidates. As a result, we use the bilinear interpolation technique

introduced in [47] to perform a two-layer upsampling operation to increase the feature map from $7 \times 7 \times 512$ to $14 \times 14 \times 512$ then to $28 \times 28 \times 512$. The final upsampled feature map is of sufficient spatial resolution to extract overlapping local patches of size $14 \times 14 \times 512$, which has been shown to be effective in discriminating the target [47], to provide more detailed local information. To this end, we employ the concept of shared features [48] to extract l local deep features inside the upsampled feature map. To do so, we divide the upsampled feature map into $l = 9$ overlapping $14 \times 14 \times 512$ features maps with the stride of 7. The feature map of each of 9 overlapping patches is vectorized as a feature vector with the size of 1×100352. Finally, we apply principal component analysis (PCA) on the feature vector of each patch to attain the top 1120 principal components for each local feature vector (e.g., $d = 1120$) to speed up the process to find the best target candidate by the proposed optimization model. We choose 1120 principal components since at least 95% of variance is retained.

4.3 STLDF Optimization Model and Numerical Algorithm

The extracted deep features are used to build the dictionary $\mathbf{D} = [\mathbf{D}_1, \ldots, \mathbf{D}_k] \in \mathbb{R}^{d \times (lk)}$, where $\mathbf{D}_i \in \mathbb{R}^{d \times l}$. Using n number of particles (target candidates), we build a matrix $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_n] \in \mathbb{R}^{d \times (ln)}$ to include local deep features of target candidates. We denote the sparse coefficient matrix as $\mathbf{C} \triangleq \begin{bmatrix} \mathbf{C}_1 & \ldots & \mathbf{C}_k \end{bmatrix}^\top \in \mathbb{R}^{(lk) \times l}$, where $\{\mathbf{C}_q\}_{q=1}^k$ is a $l \times l$ matrix indicating the group representation of l local deep features of the j^{th} target candidate using l local features of the q^{th} template. We use the model in (3.1) to represent the $\mathbf{X}_j$, the deep feature matrix of j^{th} target candidate, using the deep feature dictionary $\mathbf{D}$. Similar to (3.5), we write the Lagrangian using the Lagrangian multipliers $\mathbf{\Lambda}_1, \mathbf{\Lambda}_2 \in \mathbb{R}^{(lk) \times l}$. Given initialization for $\hat{\mathbf{C}}$, $\hat{\mathbf{U}}$, $\mathbf{\Lambda}_1$, and $\mathbf{\Lambda}_2$ at time $t = 0$ (e.g., $\hat{\mathbf{C}}^0, \hat{\mathbf{U}}^0, \mathbf{\Lambda}_1^0, \mathbf{\Lambda}_2^0$), the Lagrangian is solved through the ADMM iterations described below:

$$
\begin{aligned}
(\mathbf{C}^{t+1}, \mathbf{U}^{t+1}) := & \underset{\mathbf{C}, \mathbf{U} \in \mathbb{R}^{(lk) \times l}}{\arg\min} \quad \mathcal{L}_\mu(\mathbf{C}, \mathbf{U}, \hat{\mathbf{C}}^t, \hat{\mathbf{U}}^t, \mathbf{\Lambda}_1^t, \mathbf{\Lambda}_2^t) \\
& \text{subject to} \quad \text{(3.4d)}
\end{aligned}
\tag{4.1}
$$

$$
\begin{aligned}
(\hat{\mathbf{C}}^{t+1}, \hat{\mathbf{U}}^{t+1}) := \underset{\mathbf{C},\mathbf{U}\in\mathbb{R}^{(lk)\times l}}{\arg\min} \; & \mathcal{L}_{\mu}(\mathbf{C}^{t+1}, \mathbf{U}^{t+1}, \hat{\mathbf{C}}, \hat{\mathbf{U}}, \mathbf{\Lambda}_1^t, \mathbf{\Lambda}_2^t) \\
\text{subject to} \quad & \text{(3.4b)}, \ \text{(3.4c)}, \ \text{(3.4e)}.
\end{aligned}
\tag{4.2}
$$

$$
\begin{aligned}
\mathbf{\Lambda}_1^{t+1} &= \mathbf{\Lambda}_1^t + \mu(\mathbf{C}^{t+1} - \hat{\mathbf{C}}^{t+1}) \\
\mathbf{\Lambda}_2^{t+1} &= \mathbf{\Lambda}_2^t + \mu(\mathbf{U}^{t+1} - \hat{\mathbf{U}}^{t+1})
\end{aligned}
\tag{4.3}
$$

By considering the quadratic and linear terms of $\mathbf{C}$ and $\mathbf{U}$ in (3.5), we first define $\{\mathbf{z}_i\}_{i=1}^{lk}$, where $\mathbf{z}_i \in \mathbb{R}^{2l}$ is obtained by stacking the i^{th} rows of $\mathbf{C}$ and $\mathbf{U}$. We then divide (4.1) into lk equality constrained quadratic programs as follows:

$$
\begin{aligned}
\underset{\mathbf{z}_i\in\mathbb{R}^{2l}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{z}_i^{\top}\mathbf{Q}\mathbf{z}_i + \mathbf{z}_i^{\top}\mathbf{q}_i && \text{(4.4a)} \\
\text{subject to} \quad & \mathbf{A}\mathbf{z}_i = \mathbf{0} && \text{(4.4b)}
\end{aligned}
$$

where $\mathbf{Q} \in \mathbb{R}^{l\times l}$ is a block diagonal positive semi-definite matrix and $\mathbf{A}$ is a sparse matrix constructed based on the constraint (3.4d). Each of the above quadratic programs has its analytical solution by writing the KKT conditions.

Similarly, we split (4.2) into two separate sub-problems with close-form solutions over $\hat{\mathbf{C}}$ and $\hat{\mathbf{U}}$ as follows:

$$
\begin{aligned}
\underset{\mathbf{z}_i\in\mathbb{R}^{2l}}{\text{minimize}} \quad & \left\| \hat{\mathbf{C}} - (\mathbf{C} + \frac{\mathbf{\Lambda}_1}{\mu}) \right\|_{\mathrm{F}}^2 && \text{(4.5a)} \\
\text{subject to} \quad & \hat{\mathbf{C}} \geq 0, && \text{(4.5b)} \\
& \mathbf{1}_{(lk)}^{\top}\hat{\mathbf{C}} = \mathbf{1}_l^{\top} && \text{(4.5c)}
\end{aligned}
$$

$$
\begin{aligned}
\underset{\mathbf{z}_i\in\mathbb{R}^{2l}}{\text{minimize}} \quad & \left\| \hat{\mathbf{U}} - (\mathbf{U} + \frac{\mathbf{\Lambda}_2}{\mu}) \right\|_{\mathrm{F}}^2 && \text{(4.6a)} \\
\text{subject to} \quad & \hat{\mathbf{U}} \geq 0 && \text{(4.6b)}
\end{aligned}
$$

where sub-problems (4.5) and (4.6) consist of l independent Euclidean norm projections onto the probability simplex constraints and the non-negative orthant, respectively. Both sub-problems have analytical solutions. Finally, we solve the two sub-problems over $\mathbf{\Lambda}_1$ and $\mathbf{\Lambda}_2$ in (4.3) by performing l parallel updates over their respective columns. The closed form solutions lead to quick updates in each iteration.

4.4 STLDF Template Update

We adopt the same strategy as used in [1] to update templates. We generate a cumulative probability sequence and a random number according to uniform distribution on the unit interval $[0, 1]$. We then choose the template to be replaced based on the section that the random number lies in. This ensures that the old templates are slowly updated and the new ones are quickly updated. As a result, the drifting issues are alleviated.

We replace the selected template by using the information of the tracking result in the current frame. To do so, we represent the tracking result by a dictionary in a Lasso problem. This dictionary contains trivial templates (identity matrix) [42] and PCA basis vectors, which are calculated from the templates $\mathbf{D}$. We numerically solve the Lasso problem using the accelerated proximal gradient (APG) method. To further improve the computational time, we consider the structure of the identity matrix in our Lasso numerical solver to quickly perform the matrix multiplications and find the descend direction faster in each iteration.

4.5 STLDF Summary

The tracking steps of the proposed STLDF for two consecutive frames (i.e., frame #1 and frame #2) are summarized in Figure 4.1. In the first step, local deep features of k target templates are extracted using the initial target location in the frame #1. Their top principal components are selected using PCA. The dictionary $\mathbf{D}$ consisting of these local deep features is then constructed. In the second step, local deep features of target candidates are extracted to construct $\mathbf{X}$ in the frame #2. In the third step, local deep features of each target candidate, $\mathbf{X}_j$, is represented by the dictionary matrix in the optimization model in (3.1). Finally, the optimization model is iteratively solved to obtain $\mathbf{C}$ for each target candidate

using (4.1), (4.2), and (4.3). The best target candidate with the minimum reconstruction error is then selected as the target candidate. The tracking continues for the next frame using the previously estimated target location and templates are updated as explained in 4.4 until all the frames are processed.

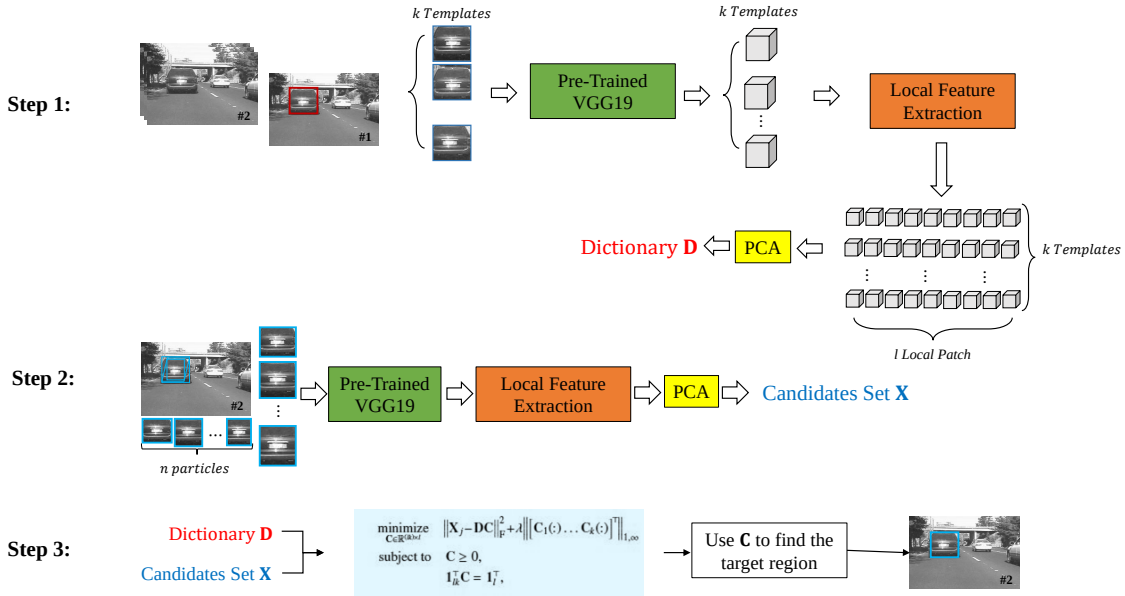


Fig. 4.1: The overview of the proposed STLDF method.

CHAPTER 5

Appearance Variation Adaptation TrackerUsing Adversarial Network

5.1 Introduction

Despite the favorable performance of deep feature based trackers such as [30, 39, 47] against hand crafted feature in obtaining generic feature representations, their effectiveness in terms of tracking is limited due to the inconsistency between classification and tracking problems, i.e., predicting object class labels versus locating targets of arbitrary classes. One major challenge of exploiting CNN is learning a unified representation using large variation frame sequences. It is mainly because, individual sequences involve different types of targets whose class labels, moving patterns, and appearances are different, and tracking algorithms suffer from sequence-specific challenges including occlusion, deformation, lighting condition change, motion blur, etc. Training CNNs is even more difficult since the same kind of objects can be considered as a target in a sequence and as a background object in another. To address these issues, Nam and Han [4] introduced the multi-domain network (MDNet) tracker, which pretrains a discriminative CNN using auxiliary sequences to learn the shared representation of targets from multiple video sequence with ground-truth labels. MDNet has three convolution layers followed by three fully connected layers. The first two fully connected layers are shared for all frame sequences, while the last fully connected layer is different for each frame sequence. The last fully connected layer is a binary classification and shares the common information captured from all sequences in the preceding layers for generic representation learning. The last fully connected layer for each frame sequence (domain) in MDNet is trained separately and iteratively while the shared layers are updated in every iteration. Various trackers have then been proposed to improve the performance of MDNet by using a tree structure to manage multiple target appearance models [37], using adversarial learning to identify the mask that maintains the most robust features of the

target objects over a long temporal span [5], and using reciprocative learning to exploit visual attention for training deep classifiers [31].

A major drawback of CNN-based trackers is lack of temporal generalization of their model, which is caused by over-fitting on the overlapped initial target regions. Therefore, CNN-based trackers such as MDNet [4] fail to maintain the similarities between the discriminative features of targets over time. To increase the generalization of the classification network, VITAL tracker [5] utilizes adversarial learning to generate a mask that dropouts convolutional features of target candidates. In each training iteration, VITAL prepares 9 random masks, where each mask covers one of 9 locations in the 3×3 feature map, to learn the optimal mask in a least square optimization problem. Therefore, this optimal mask is updated to cover only one part of local features in each iteration. This can lead to the loss of the informative local features during training.

We propose an appearance variation adaptation (AVA) tracker to not only improve the model generalization but also maintain the informative local features. The AVA tracker aligns the feature distributions of target regions over time by learning an adaptation mask in an adversarial network. This adversarial network works with the classification network to learn robust discriminative features of targets. The proposed adversarial network consists of a generator and a discriminator network that compete with each other over optimizing a discriminator loss in a mini-max optimization problem. Specifically, the discriminator network aims to distinguish recent target regions from earlier ones by minimizing the discriminator loss, while the generator network aims to produce an adaptation mask to maximize the discriminator loss. This leads to alignment of informative features of recent and earlier target regions during tracking, while maintaining accuracy of the classification network to distinguish targets and backgrounds. We incorporate a gradient reverse layer [49] in the adversarial network to solve the aforementioned mini-max optimization in an end-to-end manner. Unlike the VITAL tracker, the learned mask in AVA incorporates a weighted combination of multiple parts of target features in each training iteration. The proposed AVA tracker is evaluated on multiple tracking benchmarks [6, 7, 50] and achieves a favorable

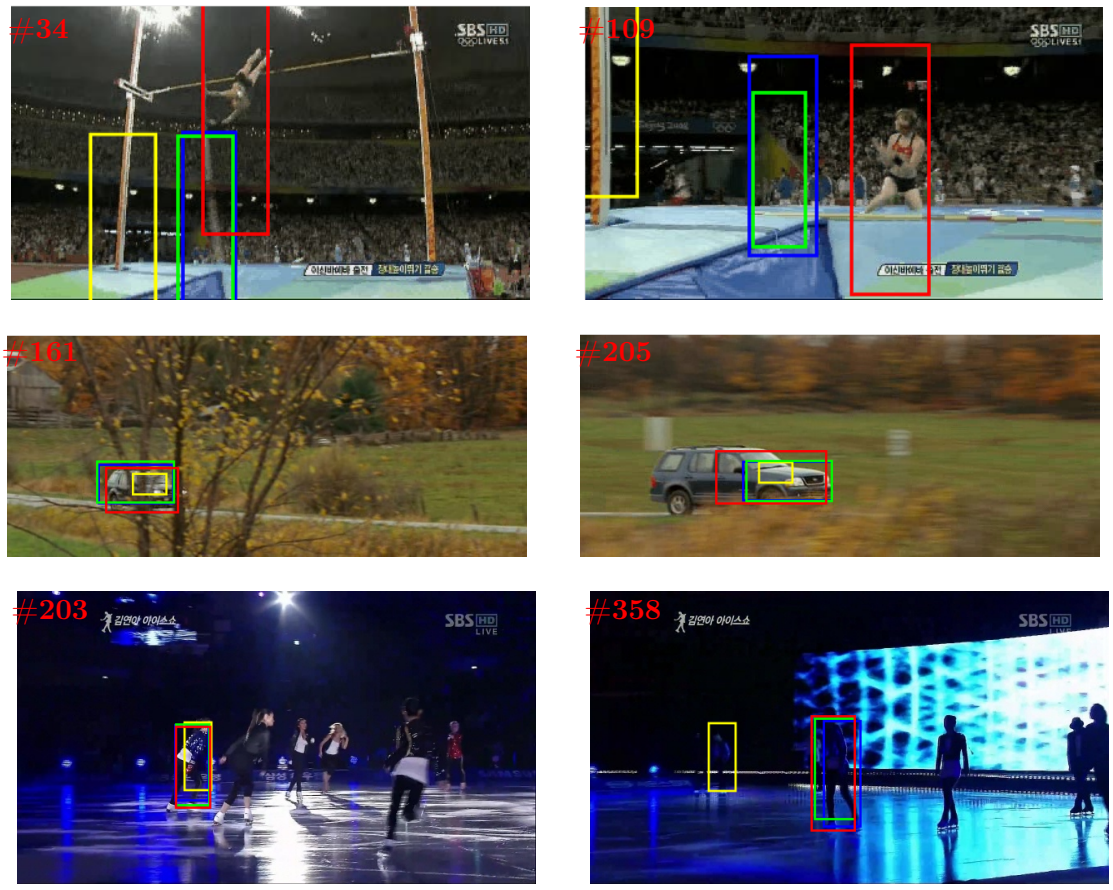


Fig. 5.1: Comparison of the proposed AVA tracker with CNN-SVM [3], MDNet [4], and VITAL [5] on three OTB100 sequences including *Jump* (Top), *CarScale* (Middle), and *Skating1* (Bottom).

performance against state-of-art trackers. Sample qualitative results are shown in Figure 5.1. The major contributions of the proposed tracker are:

- Employing adversarial learning to improve the model generalization and learn more robust discriminative features of target regions over a long time span.
- Designing an adversarial network including both generator and discriminator networks that compete with each other to learn an adaptation mask, which aligns feature distributions of target regions that may undergo various changes.
- Incorporating a gradient reverse layer in the adversarial network to solve the mini-max optimization problem in an end-to-end manner.

- Performing extensive experiments on challenging benchmarks to evaluate the performance of the AVA tracker against state-of-the-art trackers.

In sections 5.2, 5.3, and 5.4, we introduce the AVA tracker model, the AVA network architecture, and online AVA tracking settings, respectively.

5.2 AVA Model

Taking advantage of adversarial learning [51] and domain adaptation [49], we adversarially learn an adaptation mask to align features of recent and earlier target regions to make the AVA tracker model more generalized. In domain adaptation using an adversarial network [49], the training set consists of two domains (subsets) coming from different distributions. Images in one domain have classification labels, while images in the other domain may belong to different classes without labels. Existing domain adaptation methods cast the convolutional layers as a generator network and utilize a discriminator network to adapt the feature distributions of both domains. The same domain adaptation concept cannot be directly adopted in visual tracking due to the following reasons: 1) The training set exclusively contains object candidates and does not include two domains coming from different distributions. 2) The convolutional layers are expensive to learn during online tracking.

In visual tracking, a target may undergo various changes in a frame sequence due to in-plane and out-plane rotations, deformation, partial occlusion, and scale variation. However, its identity remains unchanged even with various appearance changes. To this end, we can safely assume that recent and earlier target regions in a sequence come from two domains with different distributions. Therefore, we propose to utilize adversarial learning, which adapts the feature distributions of recent and earlier target regions to maintain the similarities between the discriminative features of targets over time. Instead of learning convolutional layers directly during online tracking, we propose to learn an adaptation mask in a generator network and apply it to the feature map coming from convolution layers to produce a more robust feature representation of target candidates over time.

Here we formulate the proposed AVA tracker mathematically. Suppose that the training set of target regions up to the current frame is $\{\mathbf{x}_j\}_{j=1}^{N_t}$, where $\mathbf{x}_j \in \mathbb{R}^{1\times m}$ is the convolutional features of the j^{th} target region, N_t is the number of target samples up to the current frame, and $m = w \times h \times d$ with w, h, and d respectively being the width, the height, and the depth of the feature map in the last convolutional layer. Suppose $\mathbf{x}_i$ is a recent target region with a distribution of $\mathcal{S}(x)$ and $\mathbf{x}_k$ is an earlier target region with a distribution of $\mathcal{O}(x)$. The goal of the proposed tracker is to align the feature distributions corresponding to $\mathbf{x}_i$ and $\mathbf{x}_k$ to increase the network generalization. The alignment is performed by an adaptation mask that is adversarially learned in the proposed adversarial network, which consists of a generator network G_f and a discriminator network G_d. It should be emphasized that the adaptation mask, which is included in G_f, responds to the input features since the proposed AVA tracker model seamlessly integrates both G_f and G_d. These two networks (G_f and G_d) compete with each other over the optimization of the proposed discriminator loss $\mathcal{L}_d$ between $\mathbf{x}_i$ and $\mathbf{x}_k$. Specifically, the discriminator network G_d tries to adjust its parameters $\mathbf{w}_d$ to minimize $\mathcal{L}_d$, while the generator network G_f attempts to learn its parameters $\mathbf{w}_f$ to deceive G_d and maximize $\mathcal{L}_d$. The aligned feature map pair $(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_k)$ is then fed to the classification network G_c with its parameters $\mathbf{w}_c$ to minimize the classification loss $\mathcal{L}_c$. Finally, the proposed AVA tracker model is formulated as below:

$$\begin{aligned}\mathcal{L}(\mathbf{w}_c, \mathbf{w}_f, \mathbf{w}_d) &= \mathcal{L}_c(\cdot) - \lambda\mathcal{L}_d(\cdot) \\ &= \sum_{l\in\{1:N\}} \mathcal{L}_c(G_c(G_f(\mathbf{z}_l)), y_l) \\ &\quad - \lambda \sum_{j\in\{1:N_t\}} \mathcal{L}_d(G_d(G_f(\mathbf{x}_j)), v_j) \end{aligned} \tag{5.1}$$

where $\mathcal{L}_c(\cdot)$ is the cross-entropy classification loss between target and background [4], $\mathcal{L}_d(\cdot)$ is a cross-entropy loss between the feature maps of recent and earlier target regions, and λ is a hyper-parameter to control the balance between $\mathcal{L}_c$ and $\mathcal{L}_d$. In both loss terms, the operator $G(\cdot)$ generates the output feature map of network G. In $\mathcal{L}_c$, $\mathbf{z}_l$ is the feature map of a target

or background candidate, N is the total number of target and background samples up to the current frame, and y_l is the class label (e.g., target and background). In $\mathcal{L}_d$, N_t is the number of target samples up to the current frame, and v_j is the binary label of the j^{th} target region defined as follows:

$$v_j = \begin{cases} 0 & x_j \sim \mathcal{S}(x) \\ 1 & x_j \sim \mathcal{O}(x) \end{cases} \tag{5.2}$$

The optimal parameters ($\hat{\mathbf{w}}_c$, $\hat{\mathbf{w}}_f$, $\hat{\mathbf{w}}_d$) for the proposed model in equation (5.1) are learned in the following mini-max optimization problem [49]:

$$(\hat{\mathbf{w}}_c, \hat{\mathbf{w}}_f) = \underset{\mathbf{w}_c, \mathbf{w}_f}{\arg\min} \quad \mathcal{L}(\mathbf{w}_c, \mathbf{w}_f, \hat{\mathbf{w}}_d) \tag{5.3}$$

$$(\hat{\mathbf{w}}_d) = \underset{\mathbf{w}_d}{\arg\max} \quad \mathcal{L}(\hat{\mathbf{w}}_c, \hat{\mathbf{w}}_f, \mathbf{w}_d) \tag{5.4}$$

The solution to (5.3) and (5.4) can be found by using the following stochastic updates:

$$\mathbf{w}_c^{(i+1)} = \mathbf{w}_c^{(i)} - \mu \nabla \mathcal{L}_c(\mathbf{w}_c^{(i)}) \tag{5.5}$$

$$\mathbf{w}_f^{(i+1)} = \mathbf{w}_f^{(i)} - \mu\big(\nabla \mathcal{L}_c(\mathbf{w}_f^{(i)}) - \lambda \nabla \mathcal{L}_d(\mathbf{w}_f^{(i)})\big) \tag{5.6}$$

$$\mathbf{w}_d^{(i+1)} = \mathbf{w}_d^{(i)} - \mu \nabla \mathcal{L}_d(\mathbf{w}_d^{(i)}) \tag{5.7}$$

where $\nabla f(x)$ is the gradient of f with respect to x, μ is the learning rate, and (i) is the iteration number. The only difference between the update (5.5)-(5.7) and stochastic gradient descent (SGD) update is the $-\lambda$ factor in (5.6). Without this factor, SGD aims to make features of target regions over time dissimilar in order to minimize the discriminator loss. Therefore, $-\lambda$ factor is important to adjust the weight of generator network, $\mathbf{w}_f$, for producing similar features for recent and earlier target regions. On the other hand, the weights of the

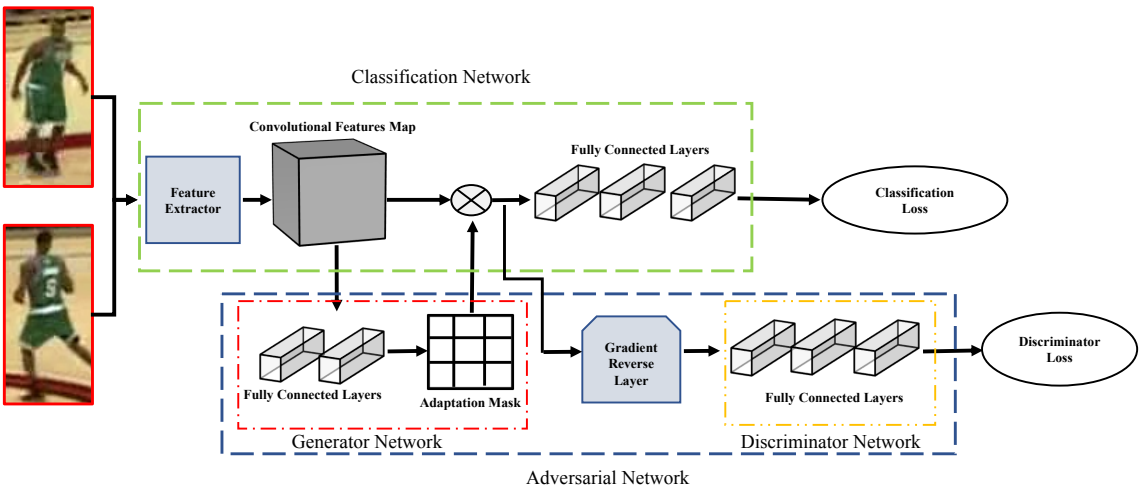


Fig. 5.2: The architecture of the proposed AVA tracker. The adversarial network consists of two networks: Generator and Discriminator. These two networks work hand-in-hand to learn an adaptation mask in a mini-max optimization problem to align the feature distributions of recent and earlier target regions up to the current frame. The classification network, including the feature extractor and three fully connected layers, is the same as the MDNet tracker. [4].

discriminator network, $\mathbf{w}_d$, are adjusted in (5.7) to discriminate between the features of recent and earlier target regions. This competition results in achieving the solution of the mini-max optimization problem in (5.3) and (5.4). We incorporate a gradient reverse layer [49] in the adversarial network to make the updates in (5.5)-(5.7) in align with the updates of the SGD method and find the solution in an end-to-end manner. In the feed-forward, the gradient reverse layer is an identity transform, while in the back-propagation, the gradient reverse layer multiplies the gradient by $-\lambda$ and passes it to the preceding layer. As shown in Figure 5.2, the gradient reverse layer is inserted between G_f and G_d to make them competing with each other over optimization of $\mathcal{L}_d$.

5.3 Network Architecture

As mentioned in subsection 5.2, the proposed AVA tracker consists of three networks, whose architectures are presented in Figure 5.2. In this subsection, we provide detailed information about the dimension of input and output features of each network layer.

The classification network, G_c, is the main network used in trackers such as MDNet [4], DAT [31], and VITAL [5]. This network has a simple architecture that is suitable for visual tracking and has shown to achieve superior tracking performance in multiple trackers. It has three convolutional layers as shown in the feature extractor block in Figure 5.2 followed by three fully connected layers. The input image to the feature extractor block is resized to the dimension of $107 \times 107 \times 3$ and the output feature map of this block has the dimension of $3 \times 3 \times 512$. This feature map is vectorized to a 4608-dimensional array and passed to the fully connected layers. The output of the first, second, and the third fully connected layers is 512, 512, and 2, respectively. The last 2 output values correspond to the background and target scores. More information can be found in [4].

The generator network, G_f, in the proposed AVA tracker aims to learn an adaptation mask for feature alignment by maximizing the discriminator loss $\mathcal{L}_d$. This network consists of two fully connected layers combined with dropout and activation functions. Specifically, after applying a dropout operation with a dropout rate of 0.5, the first fully connected layer takes the input feature vector with the dimension of 4608 and outputs a 256-dimensional feature vector. The resultant feature vector is then activated using the ReLU activation function. A similar dropout operation is performed before the second fully connected layer to produce a 3×3 adaptation mask, which is activated using the sigmoid activation function. The dimension of the adaptation mask is the same as the spatial dimension of the output from the feature extractor block in G_c. This mask is multiplied with the feature map generated from the feature extractor block to highlight the representation of regions with higher feature similarity.

The discriminator network, G_d, in the proposed AVA tracker aims to distinguish recent target regions from earlier target regions by minimizing the discriminator loss $\mathcal{L}_d$. This network has three fully connected layers combined with dropout and activation functions. The input for the first fully connected layer is the vectorized aligned features with the dimension of 4608. The output of the first, second, and third fully connected layers are 512, 512, and 1, respectively. The output of the first and second fully connected layers are

ReLU activated and the output of the third fully connected layer is sigmoid activated. The output value from the last fully connected layer is a binary prediction value for recent and earlier target regions, which is further used to optimize the cross-entropy loss (i.e., $\mathcal{L}_d$) in the aforementioned mini-max optimization problem.

5.4 Online AVA Tracking Settings

In this subsection, we present detailed information regarding initializing the tracker, obtaining the tracking result for each frame, updating the model, and setting parameters to run the AVA tracker on a sequence.

Model Initialization: Following the initial parameters set in the MDNet tracker [4], we pre-train the classification network G_c on auxiliary frame sequences. At the first frame of each testing sequence, we load the pre-trained model of G_c, freeze the convolutional layer parameters, and fine-tune the parameters of the fully connected layers [4]. Particularly, we randomly draw target and background samples around the initial location of the target region and re-train the fully connected layers of G_c to be adapted to the current frame sequence. However, we do not perform any training on the adversarial network for the first frame in the sequence.

Tracking: We produce n_s number of samples around the target identified in the previous frame. These candidates are passed through the classification network G_c. The candidate that yields the highest target score is considered as the tracking result in the current frame. The adversarial network (i.e., G_f and G_d) is disabled in this testing step [5, 52].

Model Update: The model is automatically updated every other 10 frames. In order to capture the latest appearance variations of targets over time, we update the classification network G_c, the generator network G_f, and the discriminator network G_d in the adversarial network in an end-to-end manner. The adversarial network attempts to align feature distributions of both recent and earlier target regions. When a tracking result has a negative target score, we update the classification network G_c using the tracking results up to the current frame. It should be noted that the adversarial network is not updated to avoid error propagation.

Parameters Setup: The following parameter settings are used to run the proposed AVA tracker on each frame sequence. n_s target candidates are generated similar to MDNet tracker, where $n_s = 256$. We use the same parameters for MDNet to pre-train G_c since they have shown to be effective to achieve good tracking performance. The parameters for the adversarial network are empirically determined to be optimal to achieve good tracking performance and fast convergence. They are summarized as follows: The learning rate of G_f is 0.1 and the learning rate of G_d is 0.0001. These learning rates are empirically determined for fast convergence. The number of training iterations for the adversarial network is set to be 50. The momentum and weight decay for all three networks G_c, G_f, and G_d are set to be 0.9 and 0.0005, respectively. The number of target samples in each mini-batch is 64, where 32 of them are randomly selected from recent target subset and 32 of them are randomly selected from earlier target subset. The recent target subset is defined as the second half of the target samples up to the current frame. The earlier target subset is the first half of the target samples up to the current frame. The number of background samples in each mini-batch is 96. We set $\lambda = 0.01$ for the discriminator loss.

CHAPTER 6

Experimental Results

In this chapter, we first introduce four major datasets in visual target tracking (i.e., OTB50 [6], OTB100 [7], VOT2016 [50], and VOT2018 [53]) in subsection 6.1. We then provide the experimental setup and experimental results of the proposed SMTMVT, SGLST, STLDF, and AVA trackers in subsections 6.2, 6.3, 6.4, and 6.5, respectively. Each tracker is compared to its peers and various state-of-the-art trackers. We finally compare the performance of the proposed trackers and discuss their success and failure cases in subsection 6.6.

6.1 Datasets

Various tracking benchmarks have been introduced in recent years including OTB50 [6], OTB100 [7], VOT2016 [50], UAV123 [54], and Temple Color [55]. Among these benchmarks, OTB50, OTB100, and VOT tracking benchmarks are considered as the most popular and challenging ones and the majority tracking frameworks provide their experimental results on these three benchmarks. In this subsection, we provide more detailed information on the OTB and VOT tracking benchmarks, respectively.

OTB Benchmark: This benchmark contains two different datasets: 1) OTB50 [6] and 2) OTB100 [7]. The OTB50 benchmark consists of 50 annotated sequences, where 49 sequences have one annotated target and one sequence (*jogging*) has two annotated targets. Each sequence is also labeled with attributes specifying the presence of different challenges including illumination 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 clutter (BC), and low resolution (LR). The sequences are categorized based on the attributes and 11 challenge subsets are generated. These subsets are utilized to evaluate the performance of trackers in different challenge categories. OTB100 [7] extends OTB50 [6] by adding 48 additional annotated sequences. Two sequences,

jogging and *Skating*, have two annotated targets. The rest of the sequences have one annotated target. Each of 100 sequences is also labeled with attributes specifying the presence of the aforementioned challenges.

For OTB50 and OTB100 datasets, we perform one pass evaluation (OPE) experiments and display success and precision plots. OPE is conventionally used to evaluate trackers by initializing them using the ground truth location in the first frame. Success plots display success rates at different overlap thresholds for the bounding box overlap ratio. Precision plots display precision rates at different error thresholds for the center location error. It is worthy of mentioning that the overlap score between the tracked bounding box r_t and the ground truth bounding box r_g is defined as $S = \frac{|r_t \cap r_g|}{|r_t \cup r_g|}$, where $|\cdot|$ is the number of pixels in the bounding box, $\cap$ represents the intersection of the two bounding boxes, and $\cup$ represents the union of the two bounding boxes. To rank trackers using success plots, we calculate the area under curve (AUC) score for each compared tracker on all image sequences. To rank trackers using precision plots, we calculate the average precision score for each compared tracker on all image sequences at the location error threshold of 20 pixels [6, 7].

VOT Benchmark: This benchmark consists of two datasets: VOT2016 [50] and VOT2018 [53]. Both datasets contain 60 frame sequences. Out of these sequences, 50 of them are the same for both datasets. It should be noted that the 10 different frame sequences in VOT2018 contain more sequences with small objects such as ants. For the VOT benchmark, we use accuracy, failure rate, and expected average overlap (EAO) to evaluate the tracker's performance. Accuracy is the average of overlap ratios between ground-truth and detected bounding boxes. Failure rate is a robustness measure and computed as the average of the number of times that trackers fail. EAO measures the expected no-reset overlap of a tracker run on a short-term sequence and combines the raw values of accuracy and failure per frame in a principled manner.

6.2 SMTMVT Results

In this section, we evaluate the performance of the proposed SMTMVT method on 15 publicly available frame sequences, OTB50, and OTB100 tracking benchmarks [6, 7].

To ensure fair comparison, we employ four popular features as used in [28, 29] in the proposed SMTMVT method. These features are intensity, color histogram, histogram of oriented gradients (HOGs) [56], and local binary patterns (LBPs) [57]. In addition, we employ a simple but effective illumination normalization method [58] before feature extraction to eliminate the effect of illumination and improve the quality and discriminative power of the features. Following the same settings in [28, 29], we set the size of intensity template to be one third of the size of the initial target or the half size of the initial target when its shorter length is less than 20 pixels. For all the experiments, we set $\lambda=0.1$, $\gamma=0.25$, $\alpha=30$, the number of particles $n=400$, and the number of target templates $N=10$.

6.2.1 Experimental Results on Publicly Available Sequences

We extensively conduct experiments on 15 challenging frame sequences and follow the same settings as in [28, 29] to resize all frames to 320×240. We compare the proposed SMTMVT method with eight state-of-the-art tracking methods, namely, L1 tracker [42], multi-task tracking (MTT) [59], Struck [60], tracking with multiple instance learning (MIL) [61], incremental learning for visual tracking (IVT) [25], visual tracking decomposition (VTD) [26], multi-task multi-view tracking least square (MTMVTLS) [28], and multi-task multi-view tracking least absolute deviation (MTMVTLAD) [29]. We use the publicly available source code or binary code provided by the authors to produce the tracking results. We use the default parameters for initialization.

Figure 6.1 demonstrates the tracking results of all compared methods on two representative frames for each of the 15 sequences. In the *david1*, *david2*, *girl*, *faceocc2*, *fleetface*, and *jumping* sequences, the task is to track human faces under occlusion and scale variations. Let us take the *girl* sequence as an example. IVT drifts from the target because of appearance changes. MIL and VTD are prone to drifts due to scale changes and occlusion, respectively. Struck successfully tracks the target in most frames. MTMVTLS and MTMVTLAD achieve better performance than L1T and MTT due to use of different features. SMTMVT achieves the best performance in handling the occlusion and scale variations because it retains the structure among different views.

Fig. 6.1: Tracking results of different methods. The frame index is shown at the upper left corner of each frame. The frame sequences are
david1, *david2*, *girl*, *faceOcc2*, *fleetFace*, *jumping*, *basketball*, *walking*, *subway*, *football*, *singer2*, *crossing*, *doll*,*dog1*, *carDark*.
(—L1T, —MTT, —Struck, —MIL, —IVT, —VTD, —MTMVTLS, —MTMVTLAD, —SMTMVT)

In the *basketball*, *walking*, *subway*, *football*, *singer2*, and *crossing* sequences in Figure 6.1, the task is to track multiple human bodies under fast motion, rapid pose changes, and illumination variations. For instance, in the *singer2* sequence, the algorithms aim to track a target with illumination variation, deformation, and rotations. IVT, L1T, MTT, and Struck quickly drift from the target mainly because of illumination changes. VTD gradually drifts from the target and loses it completely after some deformation and rotations. MIL is able to only track a part of the target without losing it. MTMVTLS and MTMVTLAD achieve good overall performance. SMTMVT achieves the best performance due to use of different views and structured representation of them.

In the *doll*, *dog*, and *carDark* sequences in Figure 6.1, the task is to track various objects

Table 6.1: The average overlap score of the proposed SMTMVT method and eight compared methods for 15 sequences. Bold number in blue indicates the best performance, while red indicates the second best.

	L1T	MTT	Struck	MIL	IVT	VTD	MTMVTLS	MTMVTLAD	SMTMVT
basketball	0.31	0.17	0.31	0.22	0.20	**0.71**	0.61	0.64	0.66
david1	0.54	0.29	0.24	0.42	0.65	0.55	0.70	0.67	**0.71**
david2	0.79	0.82	**0.86**	0.47	0.71	0.68	0.69	0.67	0.85
girl	0.70	0.67	0.70	0.39	0.21	0.55	0.72	0.70	**0.75**
subway	0.16	0.06	0.63	0.63	0.16	0.15	0.62	0.64	**0.73**
singer2	0.04	0.05	0.04	0.52	0.03	0.42	0.70	0.71	**0.76**
doll	0.44	0.39	0.54	0.46	0.43	0.64	**0.73**	0.67	0.71
dog1	0.70	0.68	0.54	0.53	**0.74**	0.59	0.72	0.69	0.70
faceocc2	0.68	0.74	**0.78**	0.67	0.71	0.73	0.70	0.71	0.72
fleetface	0.45	0.50	0.60	0.49	0.46	0.62	0.64	0.62	**0.68**
football	0.55	0.57	0.53	0.58	0.55	0.56	0.35	0.65	**0.69**
carDark	0.83	0.81	**0.89**	0.22	0.64	0.53	0.57	0.74	0.78
crossing	0.21	0.25	0.67	0.71	0.29	0.31	0.76	0.73	**0.78**
jumping	0.15	0.09	0.61	0.52	0.15	0.12	**0.71**	0.70	0.67
walking	**0.76**	0.64	0.56	0.54	0.70	0.60	0.58	0.60	0.74
Average	0.48	0.45	0.57	0.49	0.44	0.51	0.65	0.68	**0.73**

under different challenges. For instance, in the *doll* sequence, the algorithms aim to track a doll with various rotations and background clutters. MTT loses the target due to the background clutter and IVT fails when the target undergoes pose changes. L1T, MIL, and Struck include much of background in the results. However, they don't lose the target since they track a part of the target throughout the frames. VTD, MTMVTLS, and MTMVTLAD achieve better performance comparing with five other methods due to incorporation of multiple features. SMTMVT produces more accurate tracking results specially when the target undergoes in-plane and out-of-plane rotations.

For quantitative comparison, we compute the average overlap score across all frames of each image sequence for each compared method. Table 6.1 summarizes the average overlap scores across all frames of each of 15 sequences for the nine compared methods. It is clear that the proposed SMTMVT method achieves the best overall performance for the tested sequences. It improves the second best method (i.e., MTMVTLAD) by 7.35% in terms of the average overlap score for all 15 sequences. It ranks the best on seven sequences (e.g., *david1*, *girl*, *subway*, *singer2*, *fleetface*, *football*, and *crossing*) and ranks the second best on four sequences (e.g., *basketball*, *david2*, *doll*, and *walking*).

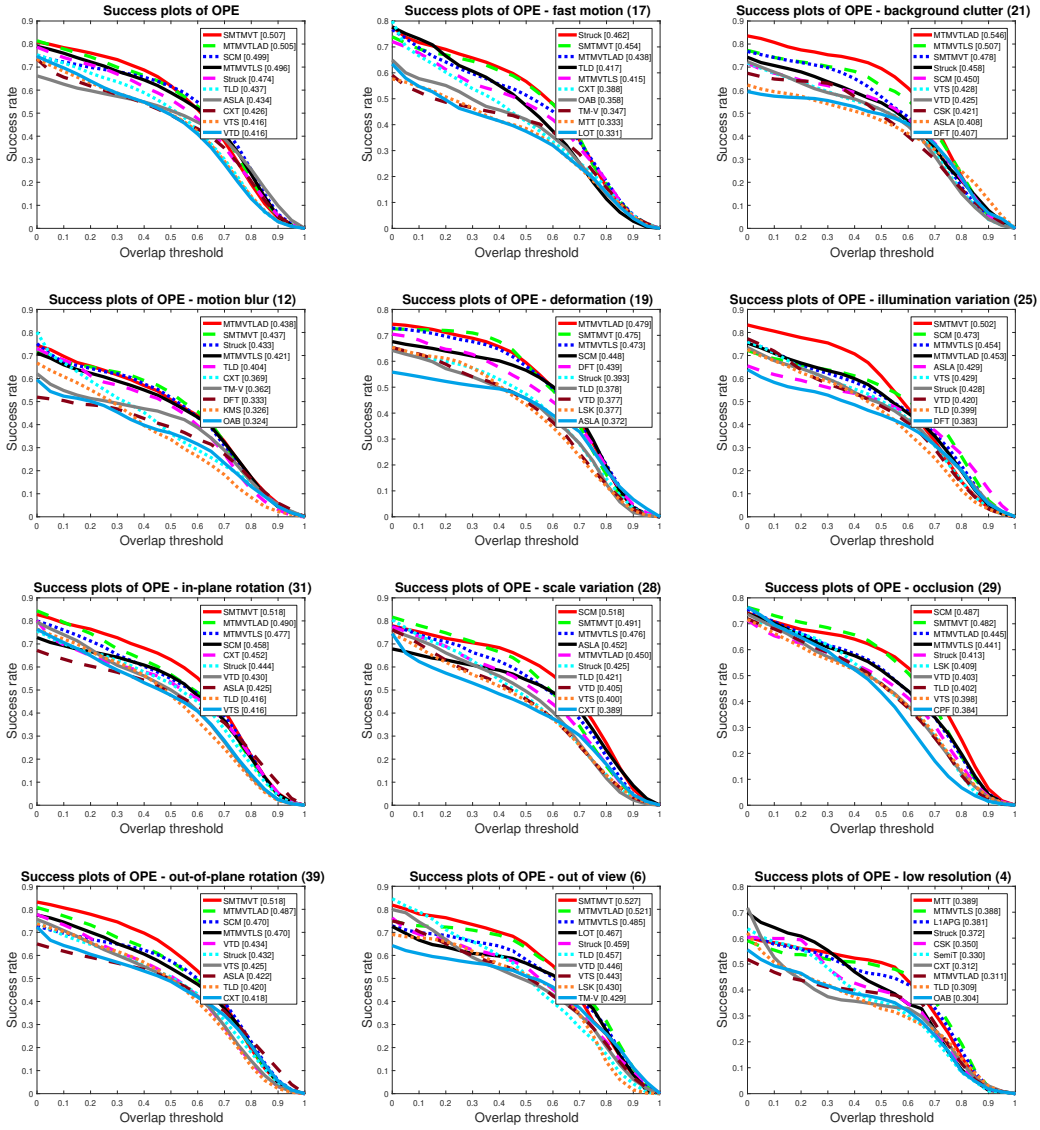


Fig. 6.2: OTB50 overall success plot and the OTB50 success plot of the proposed SMTMVT and 31 trackers for each of 11 challenge subsets. Top 10 trackers are plotted.

6.2.2 Experimental Results on OTB50

We conduct the experiments on the OTB50 tracking benchmark [6] to evaluate the performance of SMTMVT under different challenges. For this benchmark dataset, there are online available tracking results for 29 trackers. In addition, we include the results of MTMVTLS and MTMVTLAD provided by the authors. Following the protocol in [6], we use the same parameters for all the sequences to produce the results for SMTMVT. We run

SMTMVT to obtain the OPE results and compare them with the OPE results of the other 31 trackers. We present the overall OPE success plot and the OPE success plot for each of 11 challenge subsets in Figure 6.2. Here, we include the top 10 of 32 trackers in each plot for clarity. The values shown in the parenthesis alongside the legends are the AUC scores. The values shown in the parenthesis alongside the titles for 11 challenge subsets are the number of video sequences in the respective subset. It is clear that SMTMVT achieves the best overall

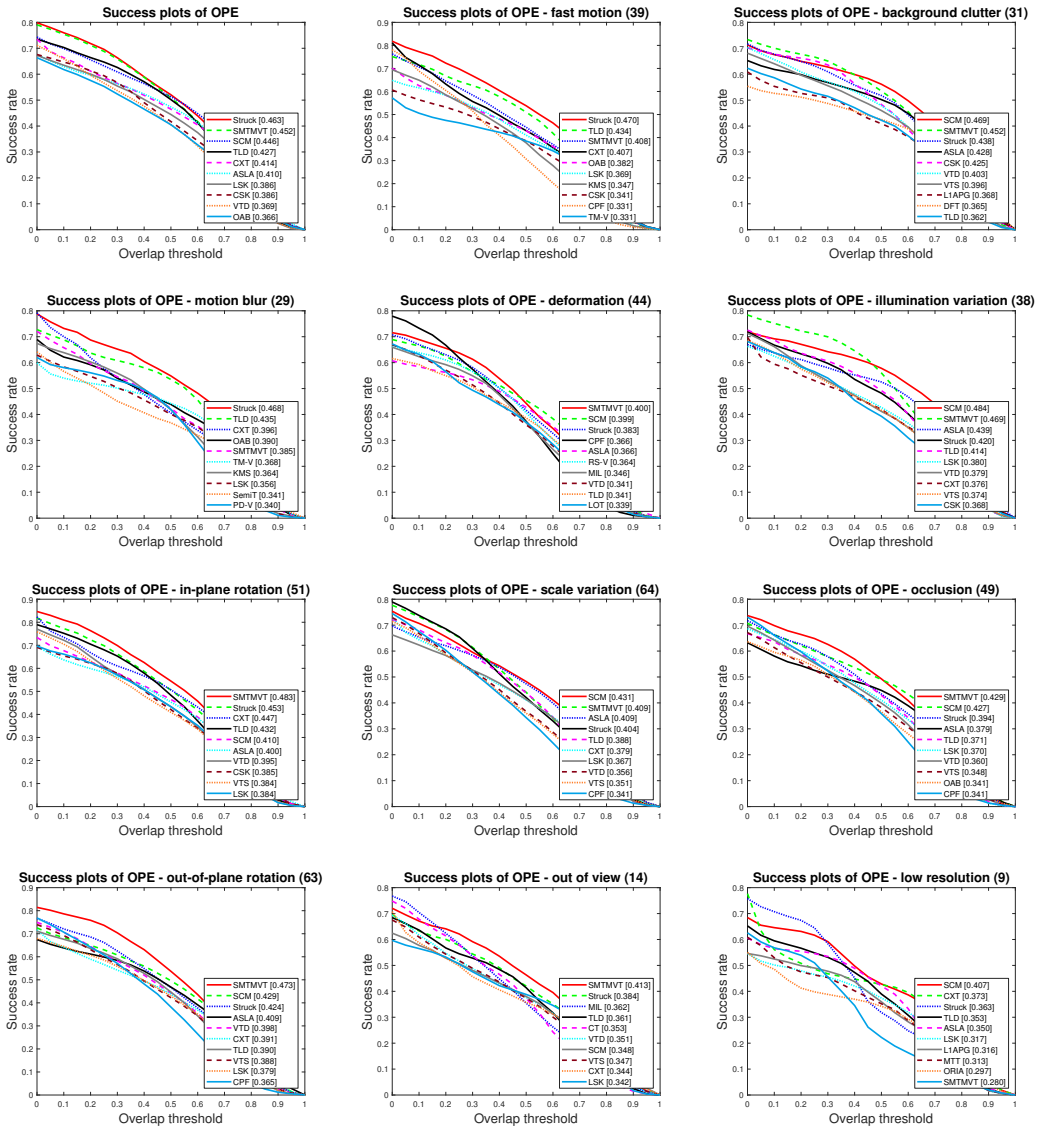


Fig. 6.3: OTB100 overall success plot and the OTB100 success plot of the proposed SMTMVT and 31 trackers for each of 11 challenge subsets. Top 10 trackers are plotted.

performance since it has the largest AUC score of 0.507. Also, SMTMVT ranks the best for four challenge subsets. It achieves the highest AUC score of 0.502 for IV, 0.518 for IPR, 0.518 for OPR, and 0.527 for OV. It achieves the second best for five challenge subsets (e.g., FM, MB, DEF, SV, and OCC), and the third best for BC.

6.2.3 Experimental Results on OTB100

We conduct the experiments on the OTB100 tracking benchmark [7] to evaluate the overall performance of the proposed SMTMVT under different challenges. We evaluate the proposed SMTMVT against 29 baseline trackers used in [7], PCOM [62], and KCF [35]. MTMVTLS and MTMVTLAD compared in the OTB50 benchmark do not provide their results on the OTB100 benchmark. Therefore, they are excluded in this experiment. We present the overall OPE success plot and the OPE success plot for each of 11 challenge subsets in Figure 6.3. Here, we include the top 10 of 32 trackers in each plot for clarity. The values shown in the parenthesis alongside the legends are the AUC scores. The values shown in the parenthesis alongside the titles for 11 challenge subsets are the number of video sequences in the respective subset. It is clear that SMTMVT achieves the second best overall performance. Also, SMTMVT ranks the best for five challenge subsets. It achieves the highest AUC score of 0.400 for DEF, 0.483 for IPR, 0.429 for OCC, 0.473 for OPR, and 0.413 for OV. It achieves the second best AUC scores for three challenge subsets (e.g., BC, IV, and SV), and the third best AUC score for FM.

6.3 SGLST Results

In this section, we evaluate the performance of the proposed SGLST on 16 publicly available frame sequences, the OTB50 [6], and the OTB100 [7] tracking benchmarks.

We resize each target region to 32×32 pixels and extract overlapping local patches of 16×16 pixels inside the target region using the step size of 8 pixels. This leads to $l = 9$ local patches. For each local patch, we extract two sets of features, namely, gray-level intensity features and histogram of oriented gradients (HOG) features, to represent its characteristics from two perspectives. Both features have shown promising tracking results in different trackers and

HOG features [56] have demonstrated significant improvement in visual tracking [29, 30, 63]. The proposed SGLST therefore has two variants: SGLST_Color and SGLST_HOG. For the HOG features, we resize the target candidates to 64×64 pixels and exploit 196 dimensional HOG features for each of the 32×32 local patches to capture relatively high-resolution edge information. For all the experiments, we set $\lambda = 0.1$, $\mu_1 = \mu_2 = \mu = 0.1$, the number of particles $n = 400$, and the number of target templates $k = 10$. We adopt the same setting as used in [1] to update templates.

6.3.1 Experimental Results on Publicly Available Sequences

We conduct extensive experiments on 16 publicly available challenging frame sequences, which are used to evaluate trackers including IVT [25], VTD [26], L1T [42], and RSST [30]. These sequences contain various challenges such as fast motion, occlusion, deformation, and scale variation and therefore are commonly used to evaluate the qualitative performance of different trackers. We compare SGLST_Color and SGLST_HOG with 11 state-of-the-art trackers, namely, L1T [42], Struck [64], IVT [25], MTT [59], MIL [61], VTD [26], Frag [24], ASLA [1], KCF [35], MEEM [65], and RSST_HOG [30]. To ensure fair comparison, we

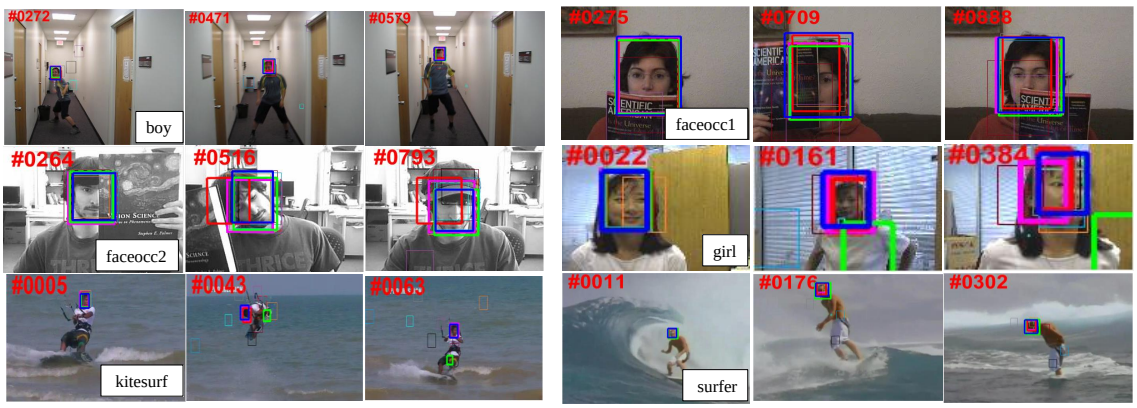


Fig. 6.4: Comparison of the tracking results of 11 state-of-the-art trackers and the two variants of the proposed SGLST on *boy*, *faceocc1*, *faceocc2*, *girl*, *kitesurf*, and *surfer* image sequences. frame index is shown at the upper left corner of each representative frame. Results are best viewed on high-resolution displays. The tracking results of the top four trackers (i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines. (—L1T, - - -Struck, —IVT, —MTT, —MIL, —VTD, —Frag, —ASLA, - - -KCF, —MEEM, —RSST_HOG, —SGLST_Color, —SGLST_HOG)

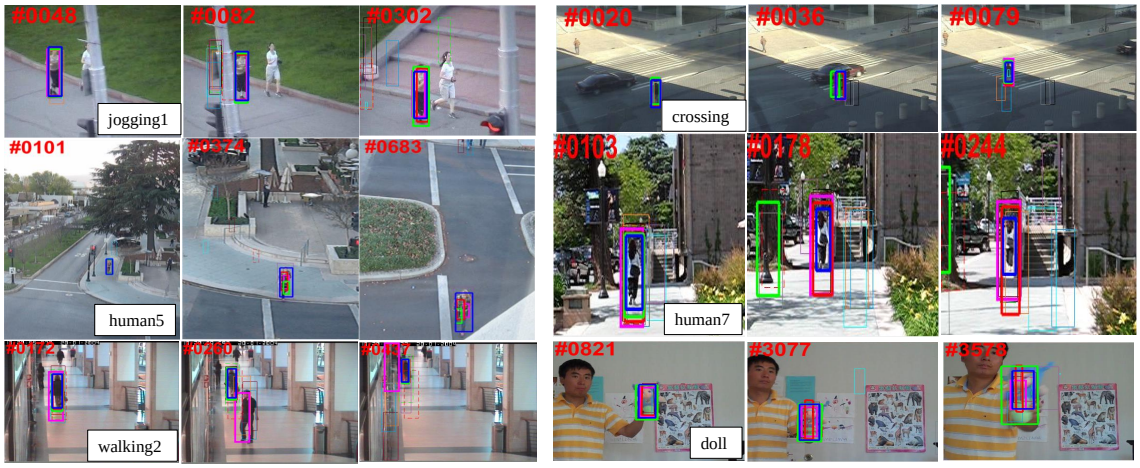


Fig. 6.5: Comparison of the tracking results of 11 state-of-the-art trackers and the two variants of the proposed SGLST on *jogging1*, *crossing*, *human5*, *human7*, *walking2*, and *doll* image sequences. The frame index is shown at the upper left corner of each representative frame. Results are best viewed on high-resolution displays. The tracking results of the top four trackers (i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines.
(—L1T, - - -Struck, —IVT, —MTT, —MIL, —VTD, —Frag, —ASLA, - - -KCF, —MEEM, —RSST_HOG, —SGLST_Color, —SGLST_HOG)

use the available source code or the binary code together with the optimal parameters provided by the respective authors to produce the tracking results. Figures 6.4, 6.5, and 6.6 demonstrate the tracking results of the 13 aforementioned compared methods on three representative frames of each of the 16 sequences. The tracking results of the top four trackers

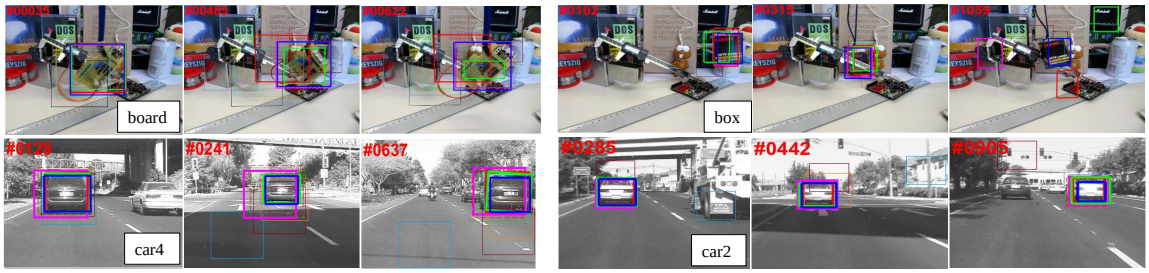


Fig. 6.6: Comparison of the tracking results of 11 state-of-the-art trackers and the two variants of the proposed SGLST on *board*, *box*, *car4*, and *car2* image sequences. The frame index is shown at the upper left corner of each representative frame. Results are best viewed on high-resolution displays. The tracking results of the top four trackers (i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines.
(—L1T, - - -Struck, —IVT, —MTT, —MIL, —VTD, —Frag, —ASLA, - - -KCF, —MEEM, —RSST_HOG, —SGLST_Color, —SGLST_HOG)

(i.e., SGLST_HOG, RSST_HOG, SLGST_Color, and MEEM) are highlighted by thicker lines.

Here, we briefly analyze the tracking performance of each compared tracker under different challenging scenarios. The L1T tracker fails when the target undergoes fast motion and rotation as shown in *Kitesurf* and *surfer* sequences, occlusion as shown in the *jogging1* sequence, or scale variation as shown in the *board* sequence. Struck cannot track the target when occlusion (*jogging1* and *box*) or fast motion (*surfer*) occurs. IVT drifts from the target in the frame sequences containing the out-of-view challenge (*girl* and *jogging*), fast motion (*boy*), or scale variation (*human5*). MTT loses the target having large motions between consecutive frames (*board* and *crossing*). MIL fails to track the target when scale variation (*car4* and *car2*) or occlusion (*walking2*) happens. VTD and Frag lead to the drift of the target under fast motion and deformation circumstances as shown in *crossing* and *human7* sequences. In addition, they cannot adequately handle scale variation as shown in the *box* sequence. ASLA does not yield good performance in the cases of heavy occlusions (*faceocc1*, *jogging*, and *walking2*). KCF is incapable of dealing with scale variation (*car4* and *walking2*), occlusion *(jogging1)*, or out-of-view challenges (*box*). MEEM achieves good overall performance. However, it drifts from the target when scale varies (*car4*) and does not sufficiently address the challenge of partial occlusion (*walking2* and *box*). RSST_HOG performs well in most sequences, but it drifts away in the sequences with scale variations (*doll*, *board*, and *box*). SGLST_Color

Seq	L1T	Struck	IVT	MTT	MIL	VTD	Frag	ASLA	KCF	MEEM	RSST_HOG	SGLST_Color	SGLST_HOG
boy	0.73	0.76	0.26	0.49	0.49	0.62	0.38	0.36	0.77	0.79	0.76	**0.81**	0.78
faceocc1	0.74	0.73	0.72	0.70	0.60	0.69	0.82	0.41	0.76	0.75	0.70	0.74	**0.78**
faceocc2	0.68	0.76	0.72	0.74	0.67	0.71	0.65	0.65	0.74	**0.78**	0.67	0.75	0.72
girl	0.73	0.74	0.16	0.66	0.39	0.60	0.43	0.72	0.58	0.69	**0.75**	0.21	0.70
kitesurf	0.25	0.64	0.30	0.35	0.31	0.15	0.19	0.25	0.48	0.67	**0.71**	0.22	0.47
surfer	0.04	0.41	0.06	0.10	0.25	0.31	0.22	0.41	0.47	0.51	0.61	**0.71**	0.65
jogging1	0.14	0.17	0.17	0.17	0.18	0.21	0.52	0.22	0.25	0.67	0.71	**0.78**	0.74
crossing	0.24	0.67	0.29	0.19	0.72	0.31	0.31	0.77	0.71	0.71	0.76	0.72	**0.79**
human5	0.38	0.35	0.18	0.45	0.21	0.28	0.03	0.68	0.21	0.28	0.51	0.35	**0.72**
human7	0.51	0.48	0.23	0.28	0.48	0.28	0.27	0.29	0.29	0.48	0.58	0.40	**0.83**
walking2	0.75	0.51	0.79	0.78	0.29	0.40	0.35	0.37	0.39	0.31	0.76	0.80	**0.83**
doll	0.46	0.55	0.43	0.39	0.42	0.66	0.61	0.78	0.59	0.60	0.48	**0.84**	0.75
board	0.13	0.66	0.19	0.19	0.40	0.28	0.52	0.30	0.65	0.68	0.42	0.56	**0.75**
box	0.55	0.21	0.51	0.21	0.27	0.42	0.46	0.34	0.35	0.31	0.33	0.21	**0.62**
car4	0.72	0.48	0.82	0.75	0.25	0.36	0.19	0.75	0.48	0.45	**0.87**	0.82	0.85
car2	0.86	0.68	0.89	0.87	0.16	0.80	0.25	0.85	0.68	0.68	**0.91**	0.81	0.88
Average	0.49	0.55	0.42	0.46	0.38	0.44	0.39	0.51	0.53	0.59	0.66	0.61	**0.74**

Table 6.2: Summary of the average overlap scores of 11 state-of-the-art trackers and the two variants of the proposed SGLST on 16 sequences. The bold numbers in blue indicate the best performance, while the numbers in red indicate the second best.

also demonstrates favorable performance in most of the sequences. However, it encounters problems when illumination changes happen (*kitesurf* and *box*). Among all the compared methods, SGLST_HOG performs well in tracking human faces, human bodies, objects, and vehicles in the 16 challenging sequences. The favorable performance of the proposed SGLST reflects the advantages of adopting local patches within the target and keeping the spatial structure among local patches. In addition, using HOG features in SGLST helps to improve the tracking performance yielded by using intensity features.

For quantitative comparison, we compute the average overlap score across all frames of each image sequence for each compared method. Table 6.2 summarizes the average overlap scores across all frames of each of 16 sequences for compared methods. It is clear that the two proposed trackers, SGLST_Color and SGLST_HOG, achieve overall favorable tracking performance for the tested sequences. On average, SGLST_Color drastically improves the average overlap scores of L1T, IVT, MTT, MIL, VTD, and Frag by 24.49%, 45.24%, 32.61%, 60.53%, 38.64%, and 56.41%, respectively. It also outperforms Struck, ASLA, KCF, and MEEM by improving their average overlap scores by 10.91%, 19.61%, 15.09%, and 3.39%, respectively. RSST_HOG is the only tracker that outperforms SGLST_Color by 8.2% mainly due to the use of HOG features. The proposed SGLST_HOG achieves the best average overlap score and significantly outperforms SGLST_Color and RSST_HOG by 21.31% and 12.12%, respectively. In summary, the qualitative results shown in Figures 6.4, 6.5, and 6.6 and the quantitative results shown in Table 6.2 demonstrate that SGLST_HOG achieves the best tracking performance and SGLST_Color achieves the third best tracking performance, inferior to RSST_HOG that uses HOG features instead of intensity features. Both variants of the proposed SGLST can successfully track the targets in a majority of frames in all 16 tested sequences with different challenging conditions such as fast motion, rotation and scale variations, occlusions, and illumination changes.

6.3.2 Experimental Results on OTB50

We conduct the experiments on the OTB50 tracking benchmark [6] to evaluate the overall performance of the proposed SGLST_Color and SGLST_HOG under different challenges.

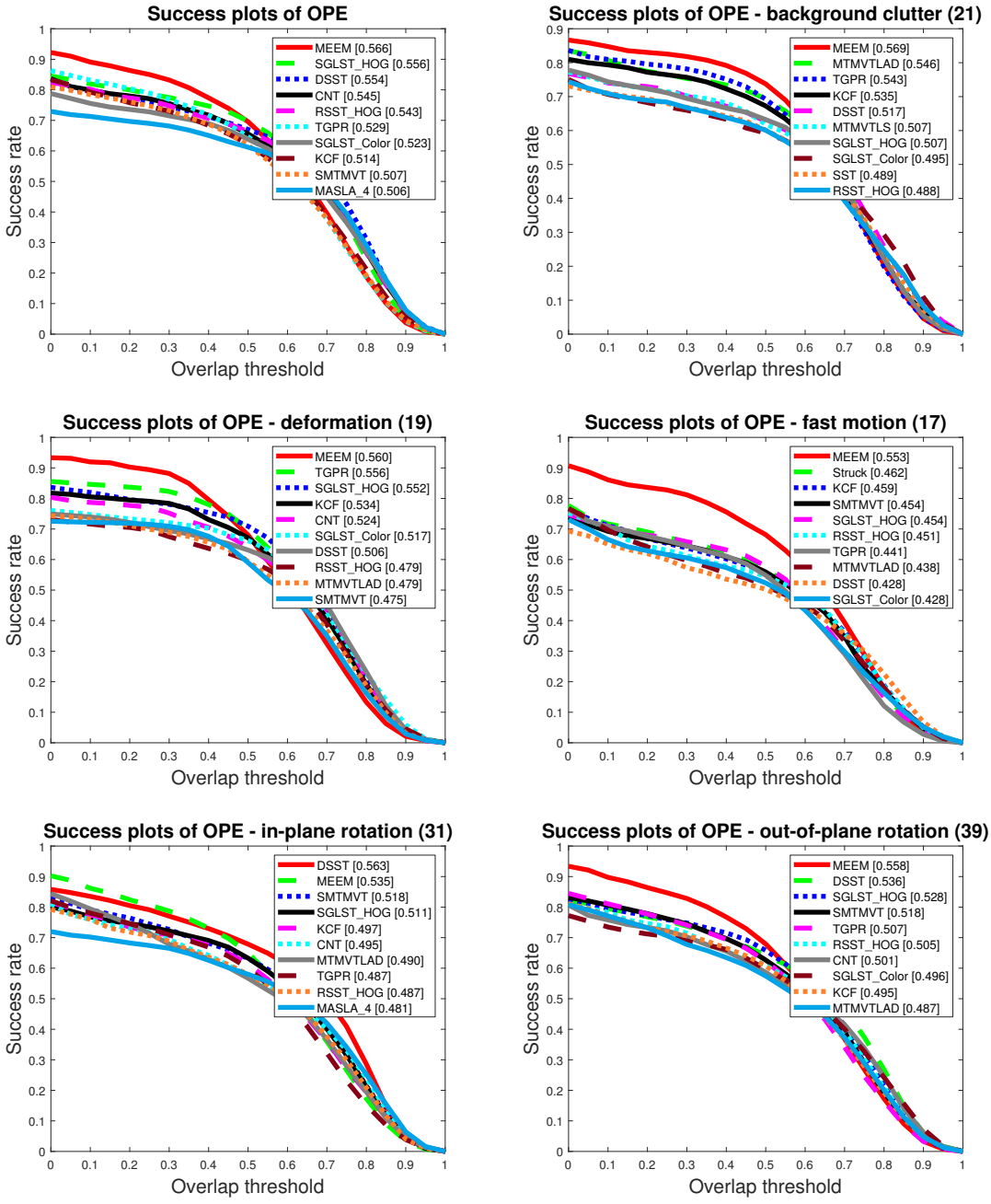


Fig. 6.7: OTB50 overall OPE success plots and the OPE success plots of top 10 trackers among 43 trackers on BC, DEF, FM, IPR, and OPR challenge subsets. The value shown in the title is the number of sequences in the specific subset. The value shown in the legend is the AUC score.The results of the other trackers can be found in [6].

For this benchmark data set, there are online available tracking results for 29 trackers [6]. In addition, we include the tracking results of additional 12 recent trackers, namely,

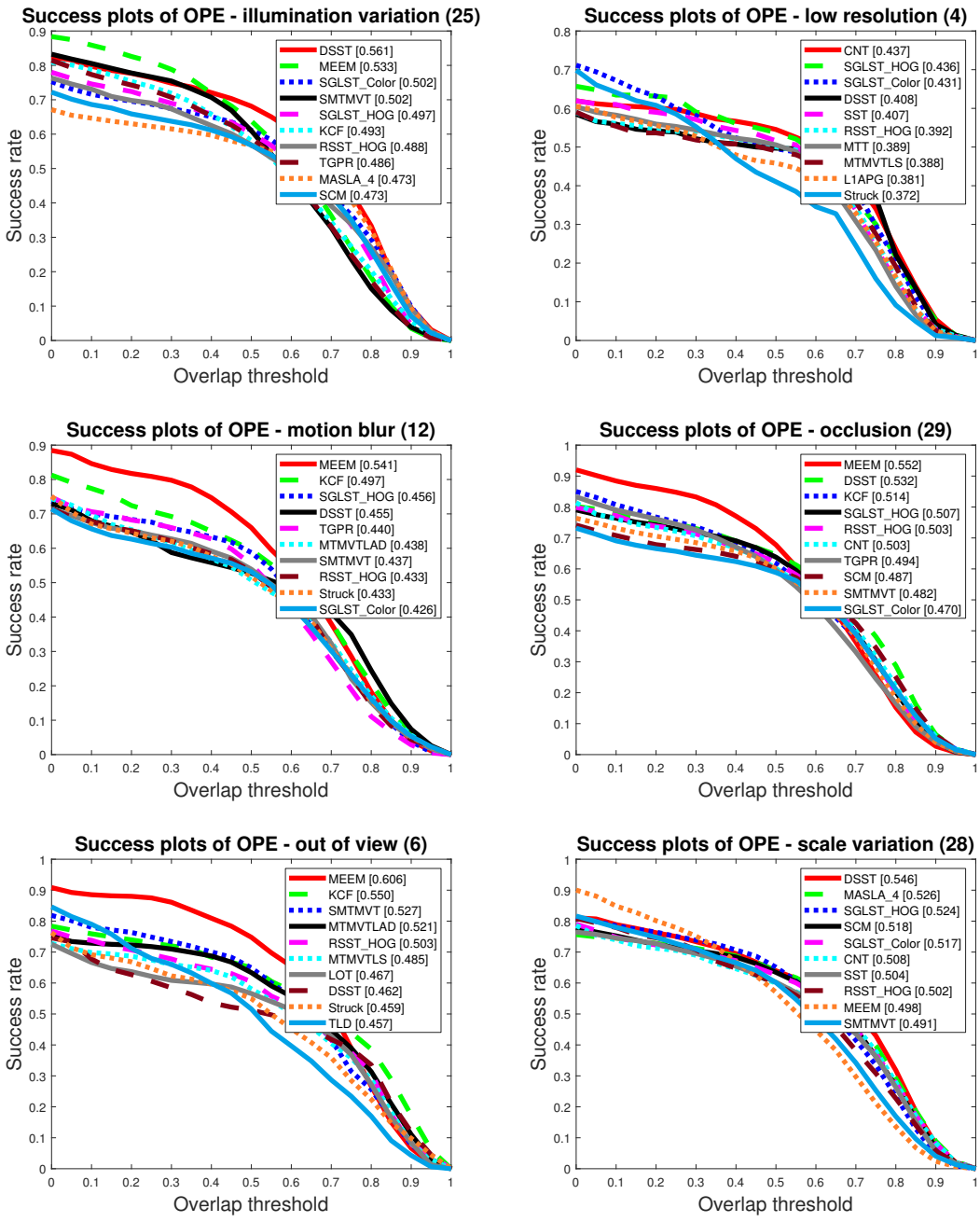


Fig. 6.8: OTB50 OPE success plots of top 10 trackers among 43 trackers on IV, LR, MB, OCC, OV, and SV challenge subsets. The value shown in the title is the number of sequences in the specific subset. The value shown in the legend is the AUC score. The results of the other trackers can be found in [6].

MTMVTLS [28], MTMVTLAD [29], MSLA-4 [2] (the recent version of ASLA [1]), SST [12], SMTMVT [38], CNT [66], TGPR [67], DSST [19], PCOM [62], KCF [35], MEEM [65],

and RSST [30]. Following the protocol proposed in [6], we use the same parameters for SGLST_Color and SGLST_HOG on all the sequences to obtain the OPE results, which are conventionally used to evaluate trackers by initializing them using the ground truth location in the first frame. We present the overall OPE success plot and the OPE success plots for BC, DEF, FM, IPR, and OPR challenge subsets in Figure 6.7 and the OPE success plots for IV, LR, MB, OCC, OV, and SV challenge subsets in Figure 6.8. For fair comparison, we use AUC score of each success plot to rank the trackers. For convenience of the reader, we only include top 10 of the 43 compared trackers in each plot. The values in the parenthesis alongside the legends are AUC scores. The values in the parenthesis alongside the titles for 11 challenge subsets are the number of video sequences in the respective subset.

It is clear from the overall success plot in Figure 6.7 that SGLST_HOG (i.e., incorporating HOG features in SGLST) improves the tracking performance (i.e., the AUC score) of SGLST_Color (i.e., incorporating intensity features in SGLST) by 6.31% due to the incorporation of the HOG features instead of the intensity features. The similar improvement trends are also observed in [30, 63]. Among the 29 baseline trackers employed in [6], SCM achieves the most favorable performance. SGLST_HOG outperforms SCM by 11.42% in terms of the AUC score. Compared with the 12 additional recent trackers, SGLST_HOG outperforms MSLA-4, SMTMVT, KCF, TGPR, RSST_HOG, CNT, and DSST by 9.88%, 9.66%, 8.17%, 5.10%, 2.39%, 2.02%, and 0.36%, respectively. It achieves an overall performance that is shy of 1.80% when comparing with the performance of the best tracker, MEEM. It should be mentioned that the two variants of SGLST (i.e., SGLST_Color and SGLST_HOG) slightly outperform the two variants of RSST (i.e., RSST_Color and RSST_HOG) by 0.53% and 2.39%, respectively. These slight improvements indicate that the proposed optimization model is better than its counterpart in RSST due to its employment of a group-sparsity regularization term to adopt local and spatial information of the target candidates and attain the spatial layout structure among them.

The proposed SGLST_HOG performs significantly better than traditional sparse trackers such as L1APG [68], LRST [69], ASLA [1], MTT [59], and MTMVTLS [28]. It outperforms

most recent sparse trackers such as MTMVTLAD [29], SST [12], MSLA-4 [2], SMTMVT [38], and RSST_HOG [30]. SGLST_HOG, which yields the AUC score of 0.556, also achieves better performance than some correlation filter (CF) based methods such as KCF (AUC score of 0.514) and DSST (AUC score of 0.554). Moreover, it outperforms some deep learning-based methods such as CNT (AUC score of 0.545) and GOTURN (AUC score of 0.444) [70]. However, the proposed SGLST_HOG yields lower performance than some deep learning-based methods such as FCNT [71] (AUC score of 0.599), DLSSVM [72] (AUC score of 0.589), and RSST_Deep [30] (AUC score of 0.590). We believe that SGLST can be further improved by incorporating the deep features, as the similar improvement trends are clearly shown in RSST [30].

We further evaluate the performance of SGLST on 11 challenge subsets. As demonstrated in Figure 6.7 and Figure 6.8, SGLST_HOG ranks as one of the top three trackers in 5 subsets with DEF, OPR, LR, MB, and SV challenges and SGLST_Color ranks as one of the top three trackers in 2 subsets with IV and LR challenges. SGLST_HOG achieves the fourth rank on 2 subsets with IPR and OCC challenges and the fifth rank on 2 subsets with FM and IV challenges. SGLST_Color achieves the fifth rank on one subset with the SV challenge. However, SGLST is not in the list of the top 10 trackers for the subset with the OV challenge. In overall, the proposed SGLST ranks as one of the top 5 trackers on 9 out of 11 subsets (e.g., 22 out of 50 image sequences) with DEF, OPR, LR, MB, SV, IV, IPR, OCC, and SV challenges.

6.3.3 Experimental Results on OTB100

We conduct the experiments on the OTB100 tracking benchmark [7] to evaluate the overall performance of the proposed SGLST_Color and SGLST_HOG under different challenges. We evaluate the proposed SGLST against 29 baseline trackers used in [7] and six recent trackers including DSST [19], PCOM [62], KCF [35], MEEM [65], TGPR [67], and RSST [30]. The other 6 trackers compared in the OTB50 benchmark do not provide their results on the OTB100 benchmark. Therefore, they are excluded in this experiment.

Figure 6.9 presents the overall OPE success plot and the OPE success plots for BC,

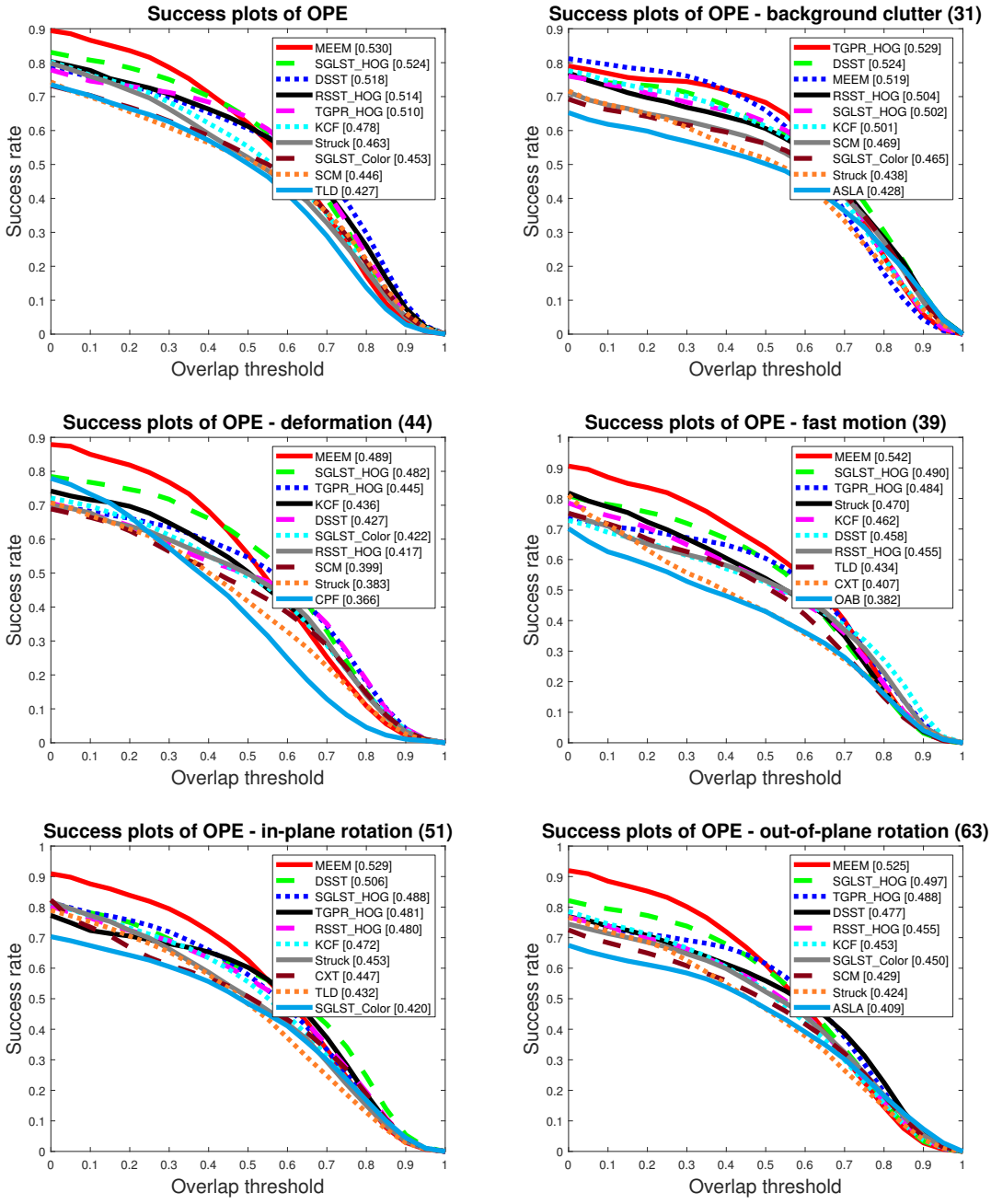


Fig. 6.9: OTB100 overall OPE success plots and the OPE success plots of top 10 trackers among 37 trackers on BC, DEF, FM, IPR, and OPR challenge subsets. The value shown in the title is the number of sequences in the specific subset. The value shown in the legend is the AUC score. The results of the other trackers can be found in [7].

DEF, FM, IPR, and OPR challenge subsets and Figure 6.10 provides the OPE success plots for IV, LR, MB, OCC, OV, and SV challenge subsets. Top 10 trackers are included in

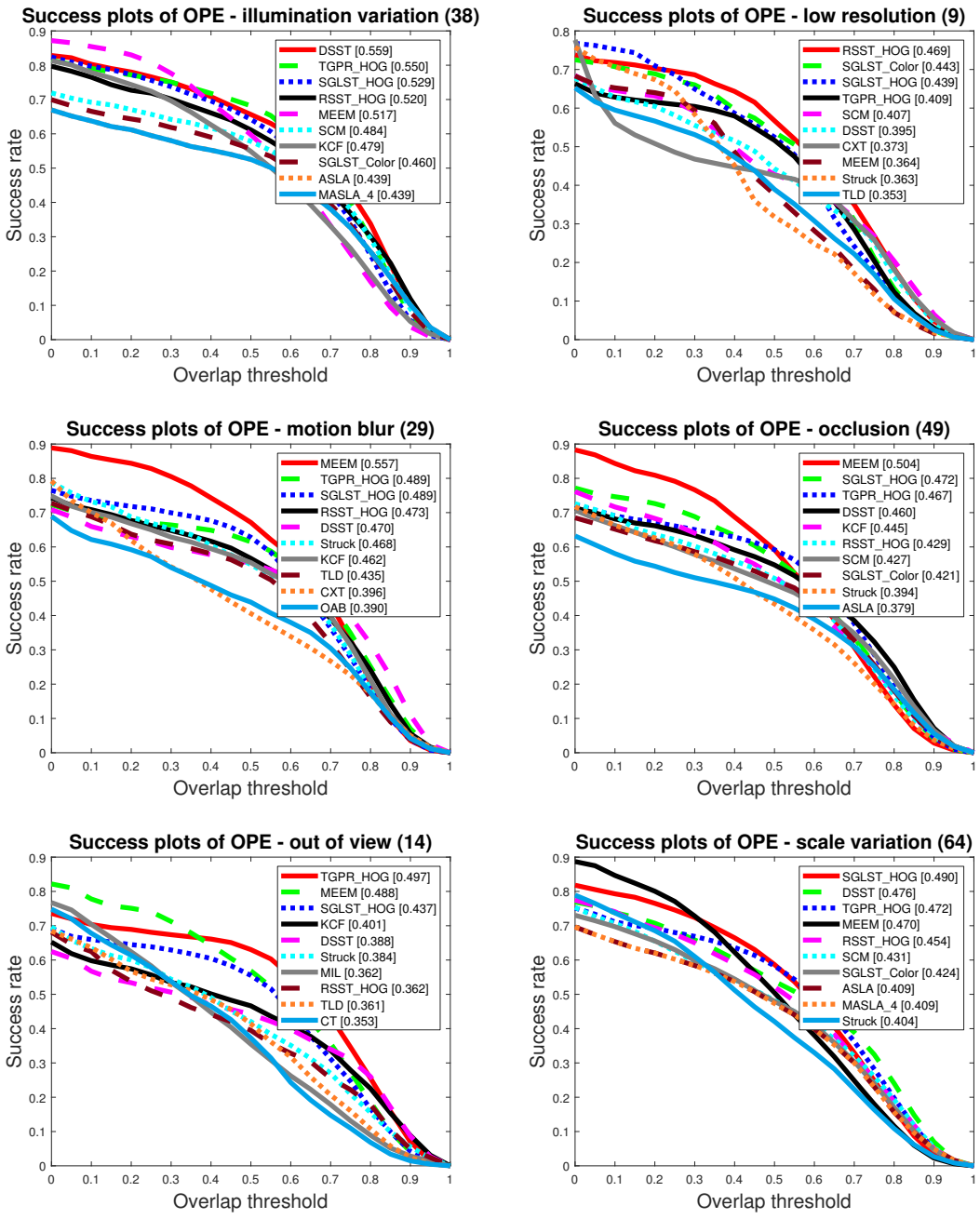


Fig. 6.10: OTB100 OPE success plots of top 10 trackers among 37 trackers on IV, LR, MB, OCC, OV, and SV challenge subsets. The value shown in the title is the number of sequences in the specific subset. The value shown in the legend is the AUC score. The results of the other trackers can be found in [7].

each plot. The overall success plot in Figure 6.9 clearly demonstrates that the best tracker MEEM, a multi-expert tracker employing an online linear SVM and an explicit feature

mapping method, has a slightly better AUC score than the second best tracker, the proposed SGLST_HOG. The difference in terms of the AUC score is only 0.006. SGLST_HOG improves its variant, SGLST_Color, by 15.67% due to the use of HOG features over intensity features. It also improves the fourth-ranked tracker RSST_HOG, the most recent sparse tracker, by 1.95% due to its novel optimization model. Compared to the third-ranked tracker DSST, a discriminative CF-based tracker, it improves the AUC score of DSST by 1.16%.

Similar to the tracking results obtained on the OTB50 tracking benchmark, the proposed SGLST_HOG performs significantly better than traditional sparse trackers such as L1APG [68], LRST [69], ASLA [1], and MTT [59]. It also outperforms RSST_HOG [30], one of the most recent sparse trackers that provides the results on the OTB100 tracking benchmark. SGLST_HOG, which yields the AUC score of 0.524, also achieves better performance than some CF and deep learning based methods such as KCF (AUC score of 0.478), DSST (AUC score of 0.518), and GOTURN (AUC score of 0.427) [70]. However, it yields lower performance than some deep learning-based methods such as CNN-SVM (AUC of 0.554), CF2 (AUC of 0.562) [47], and RSST_Deep (AUC of 0.583). We believe that incorporating the deep features in SGLST can further improve its tracking performance to be more comparable with the other deep-learning-based trackers.

We further evaluate the performance of SGLST on 11 challenge subsets in the OTB100 benchmark. As demonstrated in Figure 6.9 and Figure 6.10, SGLST_HOG ranks as one of the top three trackers in all 11 subsets except one subset with the BC challenge and SGLST_Color ranks as one of the top three trackers in one subset with the LR challenge. SGLST_HOG achieves the fifth rank on the subset with the BC challenges. It achieves better performance than the best tracker, MEEM, in three subsets with IV, LR, and SV challenges. Overall, the proposed SGLST ranks as the top 3 trackers on 10 out of 11 subsets (e.g., 69 out of 100 image sequences) with DEF, FM, IPR, OPR, IV, SV, LR, MB, OCC, and OV challenges.

6.4 STLDF Results

In this section, we evaluate the performance of the proposed STLDF and its two variants,

namely, structured tracker using local color features (STLCF) and structured tracker using local HOG features (STLHF), on the object tracking benchmark (OTB), which contains fully annotated videos with substantial variations. We evaluate these three trackers on both OTB50 [6] and OTB100 [7] benchmarks for fair comparison since not all the trackers provide the results on both benchmarks.

The two variants are similar to the proposed tracker except that STLCF uses gray-level intensity features and STLHF uses histogram of oriented gradients (HOG) features to represent each local patch. We implement these two variants since both gray-level intensity and HOG features have shown promising tracking results in different trackers [1, 29, 30, 63]. To extract intensity features, we resize each target region to 32×32 pixels and extract $l = 9$ overlapping local patches of 16×16 pixels inside the target region using the stride of 8 pixels. As a result, we use $d = 256$ dimensional gray-level intensity features to represent local patches. To extract HOG features, we resize the target candidates to 64×64 pixels to contain sufficient edge-level information with decent resolution. We then exploit $d = 196$ dimensional HOG features [56] for $l = 9$ overlapping local patches of 32×32 inside the target region using the stride of 16 pixels. As a result, we use $d = 196$ dimensional HOG features to capture relatively high-resolution edge information to represent local patches.

For all the experiments, we set $\lambda = 0.1$, $\mu = \mu_1 = \mu_2 = 0.1$, the number of particles $n = 400$, and the number of templates $k = 10$. We initially set the variances of affine parameters for particle filter resampling as (8,8,0.01,0.001,0.005, 0.0001) and adaptively update the resampling variances based on the tracking results. We use the maximum of the initial variance and the variance of the affine parameters of the most recent five tracking results to update the standard deviation of the affine parameters. We implement the proposed STLDF in MATLAB with the MatConvNet toolbox [73] on a machine with a 3.60 GHz CPU, 32 GB RAM, and a 1080Ti 11 GB Nvidia GPU. The GPU is utilized for CNN forward propagation to extract deep features of 9 local patches for each target candidate.

6.4.1 Experimental Results on OTB50

We evaluate the overall performance of the proposed STLDF and its two variants (i.e.,

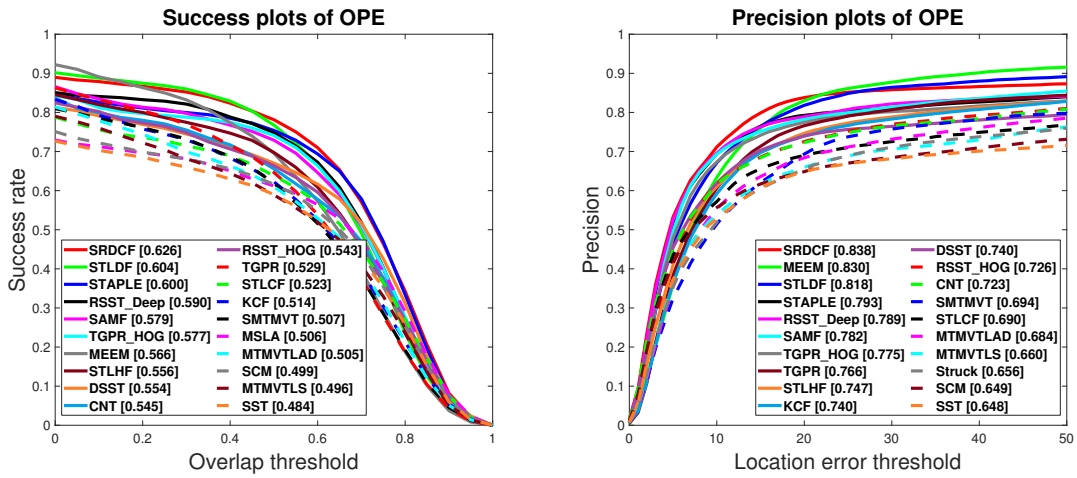


Fig. 6.11: The overall OPE plots of top 20 trackers among the proposed STLDF, its two variants, and 46 compared trackers for 50 frame sequences in OTB50.

STLCF and STLHF) against 29 baseline trackers in [6] and 17 recent trackers including MTMVTLS [28], MTMVTLAD [29], MSLA [2] (the recent version of ASLA [1]), SST [12], SMTMVT [38], CNT [66], two variants of TGPR (i.e., TGPR_Color and TGPR_HOG) [67], DSST [19], PCOM [62], KCF [35], MEEM [65], SAMF [74], SRDCF [75], STAPLE [76], and two variants of RSST (i.e., RSST_HOG and RSST_Deep) [30]. We present the overall OPE success and precision plots in Figure 6.11. We only include top 20 of the 49 compared trackers in each plot to avoid clutter and increase the readability. The value within the parenthesis alongside each legend of the success plots is the AUC score for the corresponding tracker. Similarly, the value within the parenthesis alongside each legend of the precision plots is the precision score for the corresponding tracker.

It is clear from Figure 6.11 that incorporating deep features improves the tracking performance as the proposed STLDF achieves better AUC and precision scores than its two variants, i.e., STLHF and STLCF. The similar improvement trends are also observed in [30], where RSST_Deep achieves better performance than RSST_HOG. Among the 29 baseline trackers employed in [6], SCM [77] and Struck [64] achieve the most favorable performance. STLDF significantly improves both baseline trackers. Specifically, it outperforms SCM and Struck by 21.04% and 24.73% in terms of the AUC scores, respectively. It outperforms

Table 6.3: Summary of the tracking performance of the proposed tracker, its two variants, and nine representative sparse trackers on OTB50. Bold numbers indicate the highest AUC and precision scores (i.e., the best tracking performance).

Score	L1APG	ASLA	MTT	MTMVTLAD	MSLA	SST	RSST			Proposed Method		
							Color	HOG	Deep	STLCF	STLHF	STLDF
AUC	0.380	0.434	0.376	0.505	0.506	0.484	0.520	0.543	0.590	0.523	0.556	**0.604**
Precision	0.485	0.532	0.479	0.684	0.631	0.648	0.691	0.726	0.789	0.690	0.747	**0.818**

SCM and Struck by 26.04% and 24.70% in terms of the precision scores, respectively. When comparing with the 17 additional recent trackers, STLDF achieves higher AUC scores than 16 of these trackers and a comparable score as SRDCF (the best tracker in comparison). Specifically, it improves the AUC scores of MTMVTLAD, MSLA, KCF, CNT, DSST, MEEM, TGPR_HOG, SAMF, RSST_Deep, and STAPLE by 19.60%, 19.37%, 17.51%, 10.83%, 9.03%, 6.71%, 4.68%, 4.32%, 2.37%, and 0.67%, respectively. SRDCF shows slightly better performance than STLDF (0.626 AUC score for SRDCF vs. 0.604 AUC score for STLDF). STLDF also achieves higher precision scores than 15 out of 17 additional recent trackers. Specifically, it outperforms MTMVTLAD, SMTMVT, CNT, DSST, KCF, TGPR_HOG, SAMF, RSST_Deep, and STAPLE by 19.59%, 17.87%, 13.14%, 10.54%, 10.54%, 5.55%, 4.60%, 3.68%, and 3.15%, respectively. It attains a comparable precision score as MEEM and SRDCF. All three trackers yields the precision scores above 0.81.

To demonstrate the effectiveness of the proposed optimization model, we compare the proposed tracker and its two variants (i.e., STLDF, STLHF, and STLCF) with representative traditional and recent sparse trackers in terms of the two evaluation metrics in Table 6.3. It is clear that STLDF achieves the highest overall AUC and precision scores among all the compared sparse trackers. It improves RSST_Deep, one of the most recent sparse trackers that incorporates the deep features, by 2.37% in the AUC score and 3.68% in the precision score. Its two variants (STLHF, and STLCF) also outperforms RSST's counterparts (RSST_HOG, and RSST_ Color) in terms of two evaluation metrics except that STLCF achieves the similar precision score as RSST_Color (0.690 vs. 0.691). In addition, STLCF achieves higher AUC and precision scores than other sparse trackers that utilize intensity features such as L1APG [68], ASLA [1], MTT [59], MSLA [2], and SST [12]. STLHF also

achieves higher AUC and precision scores than the sparse trackers that utilize HOG features such as MTMVTLAD [29] and RSST_HOG [30]. It is worthy of mentioning that the proposed method attains significant improvements over conventional local sparse trackers (ASLA and MSLA) by preserving the spatial layout structures among different local patch features inside a target candidate. The robust tracking performance of the proposed method demonstrate the effectiveness of the proposed optimization model that employs a group-sparsity regularization term to adopt local and spatial information of the target candidates and attain the spatial layout structure among them.

In addition to sparse trackers, the proposed STLDF achieves a better or comparable AUC score (0.604) than some correlation filter (CF) based trackers including KCF (0.514) [35], DSST (0.556) [19], LCT (0.612) [20], HDT (0.603) [36], CF2 (0.605) [47], and ACFN (0.607) [78]. It also achieves a better or comparable precision score (0.818) than the following CF-based trackers: KCF (0.740), DSST (0.740), LCT (0.848), HDT (0.889), CF2 (0.891), and ACFN (0.860).

Comparing with deep learning-based trackers, the proposed STLDF outperforms or achieves a comparable AUC score than CNT (0.545) [66], GOTURN (0.444) [70], CNN-SVM (0.597) [3], FCNT (0.599) [71], DLSSVM (0.589) [72], and SiamFC (0.608) [79]. Moreover, it outperforms or achieves comparable precision score than CNT (0.723), GOTURN (0.620), CNN-SVM (0.852), FCNT (0.856), DLSSVM (0.829), and SiamFC (0.815).

6.4.2 Experimental Results on OTB100

We evaluate the proposed STLDF and its two variants (STLHF and STLCF) against 29 baseline trackers in [7], and 15 recent trackers including DSST, PCOM, KCF, TGPR_HOG, MEEM, SAMF, SRDCF, LCT, STAPLE, CF2, CNN-SVM, DLSSVM, HDT, and two variants of RSST (i.e., RSST_HOG and RSST_Deep). Some trackers used in the experiments of OTB50 are excluded from this experiment since they do not publish their results on OTB100. Similar to the experiments on OTB50, we follow standard protocols proposed in [6, 7] and use the same parameters on all sequences to obtain the OPE results. We present the overall OPE success and precision plots of the top 20 trackers out of 47 compared trackers in Figure

6.12.

It is clear from Figure 6.12 that the proposed STLDF achieves higher AUC and precision scores than its two variants for 100 sequences in OTB100 due to its utility of local deep features. It also achieves higher AUC and precision scores than RSST_Deep due to its novel optimization model. Similar to the tracking results obtained on OTB50, SCM and Struck are the top two trackers among the 29 baseline trackers on OTB100. STLDF improves the AUC scores of SCM and Struck by 31.39% and 26.57% and the precision scores of SCM and Struck by 39.30% and 24.06%, respectively. Compared to the 15 recent trackers, STLDF achieves comparable AUC scores as SRDCF (0.586 vs. 0.598) and improves the AUC scores of the remaining 14 trackers. Specifically, it improves the AUC scores of the top 13 trackers, namely, KCF, TGPR_HOG, RSST_HOG, DSST, MEEM, DLSSVM, SAMF, CNN-SVM, LCT, CF2, HDT, RSST_Deep, and STAPLE by 22.59%, 14.90%, 14.01%, 13.13%, 10.57%, 8.72%, 5.97%, 5.59%, 4.27%, 4.27%, 3.72%, ,0.87%, and 0.52%, respectively. It also significantly improves the precision scores of six of these 15 trackers including SST, TGPR_HOG, KCF, SAMF, LCT, and DLSSVM by 15.57%, 13.92%, 13.75%, 5.59%, 4.20%, and 4.06%, respectively. In addition, it achieves a little bit improvement over four trackers including MEEM, STAPLE, RSST_Deep, and SRDCF. It is inferior to three trackers such as HDT, CF2, and CNN-SVM

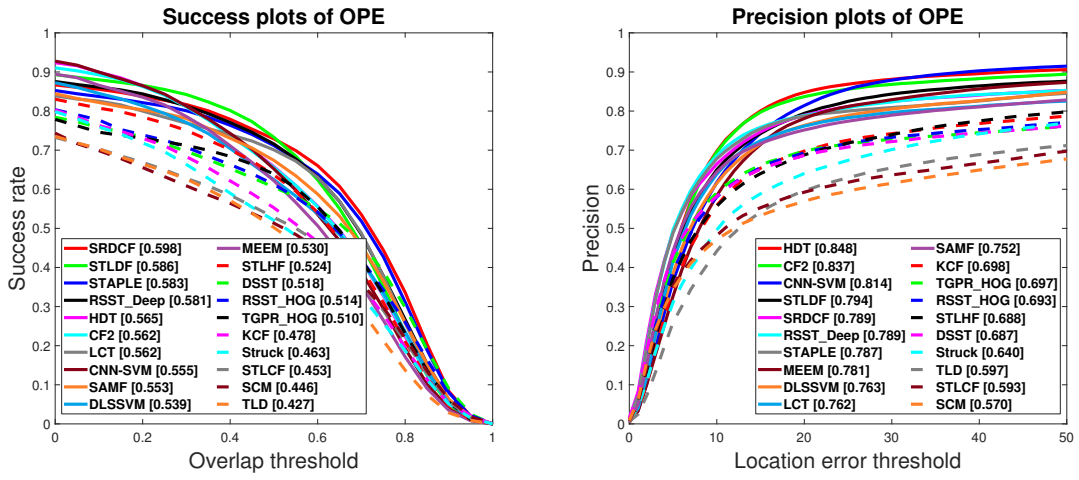


Fig. 6.12: The overall OPE plots blackof top 20 trackers among the proposed STLDF, its two variants, and 44 compared trackers for 100 frame sequences in OTB100.

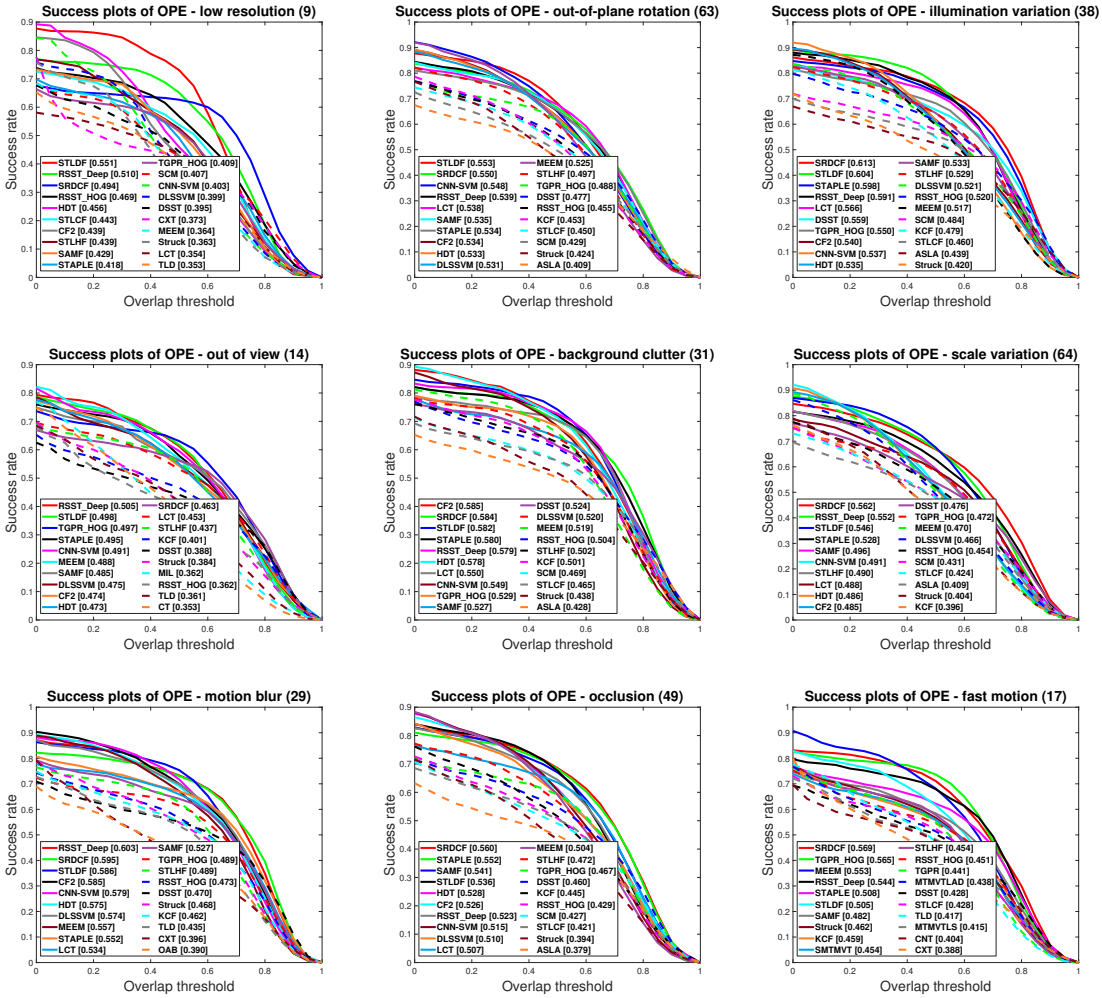


Fig. 6.13: The OPE success plots of top 20 trackers among the proposed STLDF, its two variants, and 44 compared trackers for LR, OPR, IV, OV, BC, SV, MB, OCC, and FM subsets in OTB100.

by a small margin.

The proposed STLDF significantly outperforms conventional sparse trackers such as L1APG [68], LRST [69], ASLA [1], and MTT [59] and improves both AUC and precision scores of RSST_Deep [30], one of the most recent sparse trackers, by 0.87% and 0.64%, respectively. STLDF with the achieved AUC score of 0.586 outperforms some CF-based trackers such as KCF (0.478), DSST (0.518), LCT (0.562), CF2 (0.562), and HDT (0.565) and some deep learning-based trackers such as GOTURN (0.427), CNN-SVM (0.555), and DLSSVM (0.539). These OTB100 tracking results follow the similar trends in OTB50

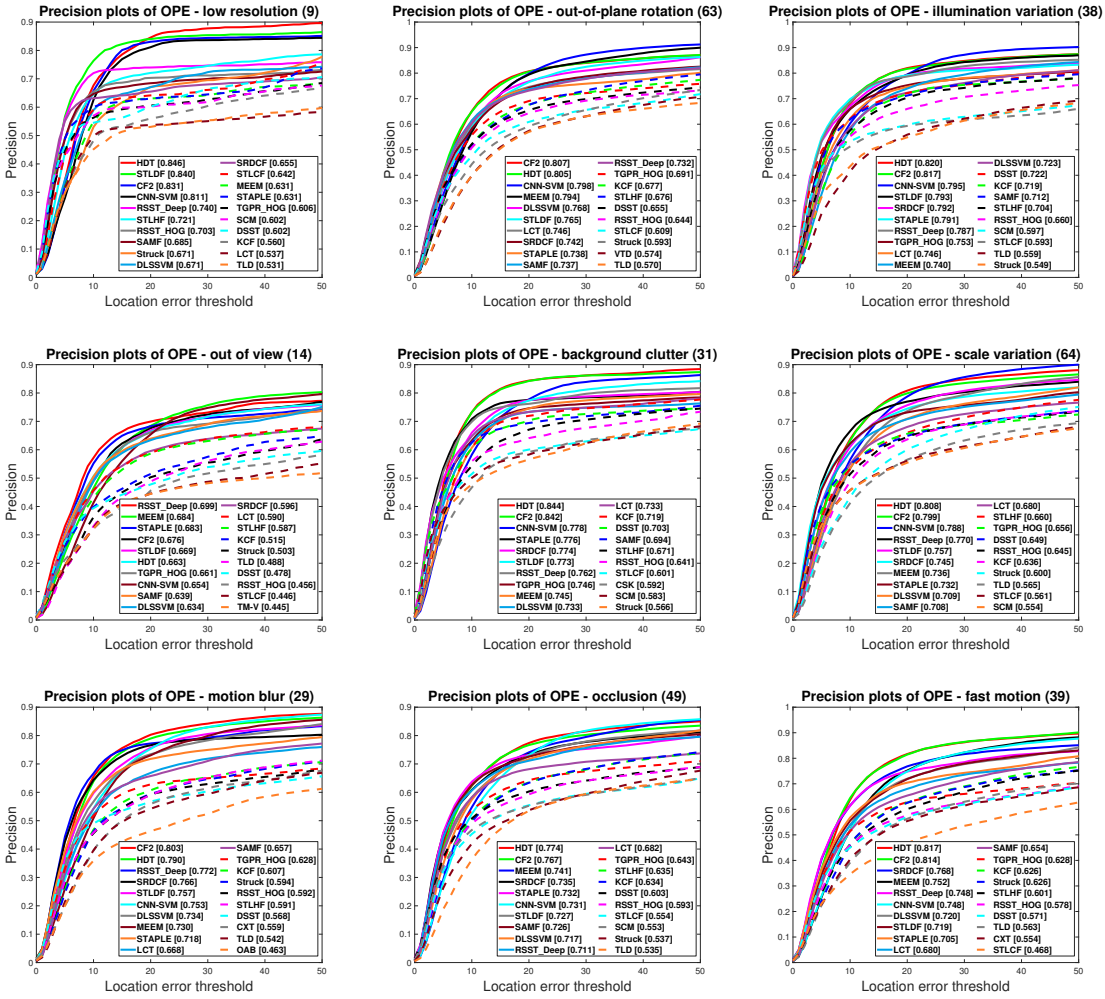


Fig. 6.14: The OPE precision plots of top 20 trackers among the proposed STLDF, its two variants, and 44 compared trackers for LR, OPR, IV, OV, BC, SV, MB, OCC, and FM subsets in OTB100.

tracking results and demonstrate the effectiveness of the proposed optimization model and the integration of local deep features.

We further evaluate the performance of STLDF in terms of AUC and precision scores on nine challenge subsets including LR, OPR, IV, OV, BC, SV, MB, OCC, and FM. Figures 6.13 and 6.14 show the success and precision plots of top 20 trackers for these 9 challenge subsets, respectively. The value within the parenthesis on the title line of each plot is the number of video sequences in the specific subset. The value within the parenthesis alongside each legend of the success plot is the AUC score for the corresponding tracker and the value

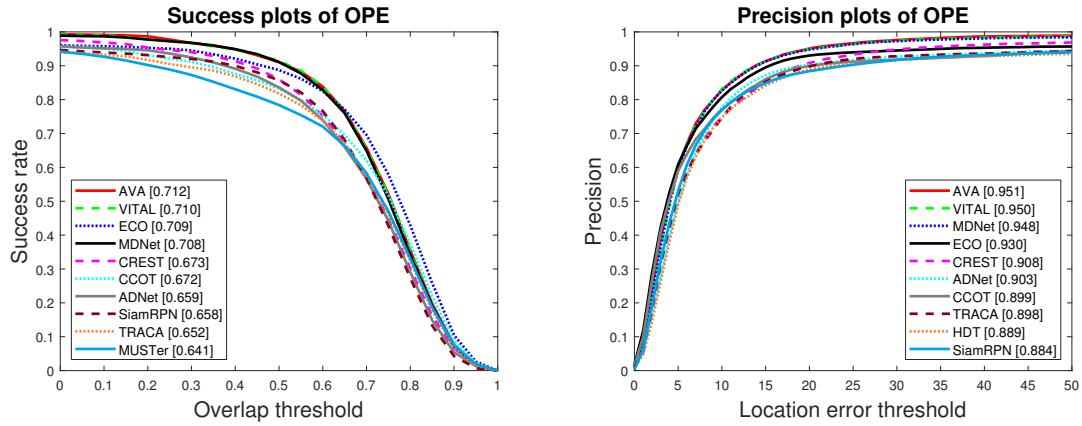
within the parenthesis alongside each legend of the precision plots is the precision score for the corresponding tracker. It is clear that STLDF achieves significantly better performance than its two variants (STLHF and STLCF) due to its integration of local deep features. As it is shown in Figure 6.13, STLDF ranks the best for two subsets with LR and OPR challenges, the second for two subsets with IV and OV challenges, the third for three subsets with BC, SV, and MB challenges, the fourth for the OCC subset, and the top sixth tracker for FM challenge in terms of ACU score. As it is demonstrated in Figure 6.14, STLDF ranks as one of the top five trackers for five subsets with LR, IV, OV, SV, and MB, the sixth best trackers in OPR and BC, and the top eight trackers for two subsets with OCC and FM challenges in terms of the precision scores. The DEF and IPR challenge subsets are not included in Figures 6.13 and 6.14 due to lack of space. STLDF obtains the AUC and precision scores of 0.529 (6th rank) and 0.727 (7th rank) for the DEF subset, respectively. STLDF yields the AUC and precision scores of 0.543 (8th rank) and 0.742 (10th rank) for the IPR challenge subset, respectively.

6.5 AVA Results

We perform extensive experiments to evaluate the performance of the proposed AVA tracker in terms of accuracy and robustness (for the source code and datasets see [80]). We compare the AVA tracker with state-of-the-art trackers on OTB50 [6], OTB100 [7], VOT2016 [50], and VOT2018 [53] tracking benchmarks. For OTB experiments, we pre-train the network using 58 VOT2016 sequences, which do not include the common sequences in the OTB100 dataset. For VOT experiments, we pre-train the network using 89 OTB100 sequences, which do not include the common sequences in the VOT dataset. We implement the AVA tracker in Python with PyTorch deep learning framework on a machine with a 3.60 GHz CPU, 32 GB RAM, and a 1080Ti 11GB Nvidia GPU. The source of AVA is available at: https://gitlab.com/mamrez7/ava-tracker.

6.5.1 Experimental Results on OTB50

We compare AVA with 29 baseline trackers in [6], and 25 recent trackers including DSST



(a) Success plot of OPE (b) Precision plot of OPE

Fig. 6.15: The overall OPE plots of top 10 trackers among the proposed AVA and 54 compared trackers for OTB50 benchmark [6].

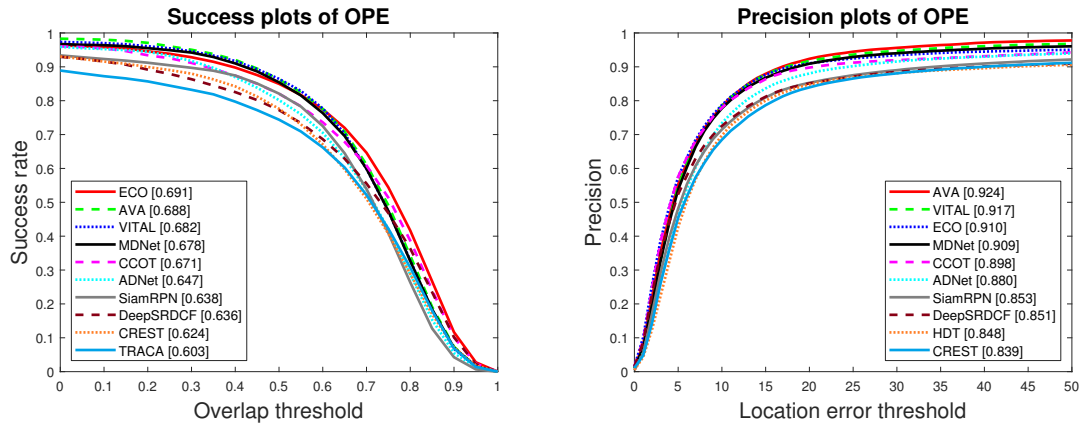
[19], KCF [35], TGPR [67], MEEM [65], MUSTer [81], LCT [20], RSST [30], SRDCF [75], DeepSRDCF [82], SiamFC [79], ADNet [83], CFNet [84], SGLST [39],SCT [85], CNN-SVM [3], CCOT [86], ECO [87], MDNet [4], VITAL [5], CREST [88], TRACA [89], SiamRPN [90], STAPLE [76], CNT [66], and HDT [36]. Adopting the protocol proposed in [6], we use the same parameters for all sequences to obtain OPE results.

We present the overall OPE success and precision plots in Figure 6.15. We include the top 10 of the 55 compared trackers in each plot to avoid clutter and increase the readability. The value within the parenthesis alongside each legend of success plots is the AUC score for its corresponding tracker. Similarly, the value within the parenthesis alongside each legend of precision plots is the precision score for its corresponding tracker. Figure 6.15 clearly demonstrates that the proposed AVA tracker achieves the best tracking performance with the highest AUC score of 0.712 and the highest precision score of 0.951 when comparing with 55 state-of-the-art trackers. Among the 29 baseline trackers employed in [6], SCM [77] achieves the best performance with an AUC score of 0.499 and a precision score of 0.649. The proposed AVA tracker significantly outperforms SCM by 42.69% and 46.53% in terms of AUC and precision scores, respectively. It also improves AUC scores of the top 9 trackers among the 25 additional recent trackers, namely, MUSTer, TRACA, SiamRPN, ADNet,

CCOT, CREST, MDNet, ECO, and VITAL by 11.08%, 9.20%, 8.21%, 8.04%, 5.95%, 5.79%, 0.56%, 0.42%, 0.28%, respectively. It outperforms precision scores of the top 9 trackers among the 25 additional recent trackers, namely, SiamRPN, HDT, TRACA, CCOT, ADNet, CREST, ECO, MDNet, and VITAL by 7.58 %, 6.97%, 5.90%, 5.78%, 5.32%, 4.74%, 2.26%, 0.32%, and 0.11%, respectively.

6.5.2 Experimental Results on OTB100

We evaluate the performance of the proposed AVA tracker against the same state-of-the-arts trackers presented in the OTB50 experiment. The MUSTer and CNT tracker are excluded from this experiment since they do not have any published results on OTB100. Similar to the experiments on OTB50, we follow the protocol proposed in [6, 7] and use the same parameters on all the sequences to obtain OPE results. To avoid clutter and increase the readability, we present the overall OPE success and precision plots for the top 10 of the 53 compared trackers in Figure 6.16. Each tracker's AUC and precision scores are shown inside their corresponding parenthesis in the success and precision plots, respectively. It clearly shows that the proposed AVA tracker achieves a favorable performance against state-of-the-art trackers in terms of both AUC and precision scores. Among the 29 baseline



(a) Success plot of OPE (b) Precision plot of OPE

Fig. 6.16: The overall OPE plots of top 10 trackers among the proposed AVA and 52 compared trackers for OTB100 benchmark [7].

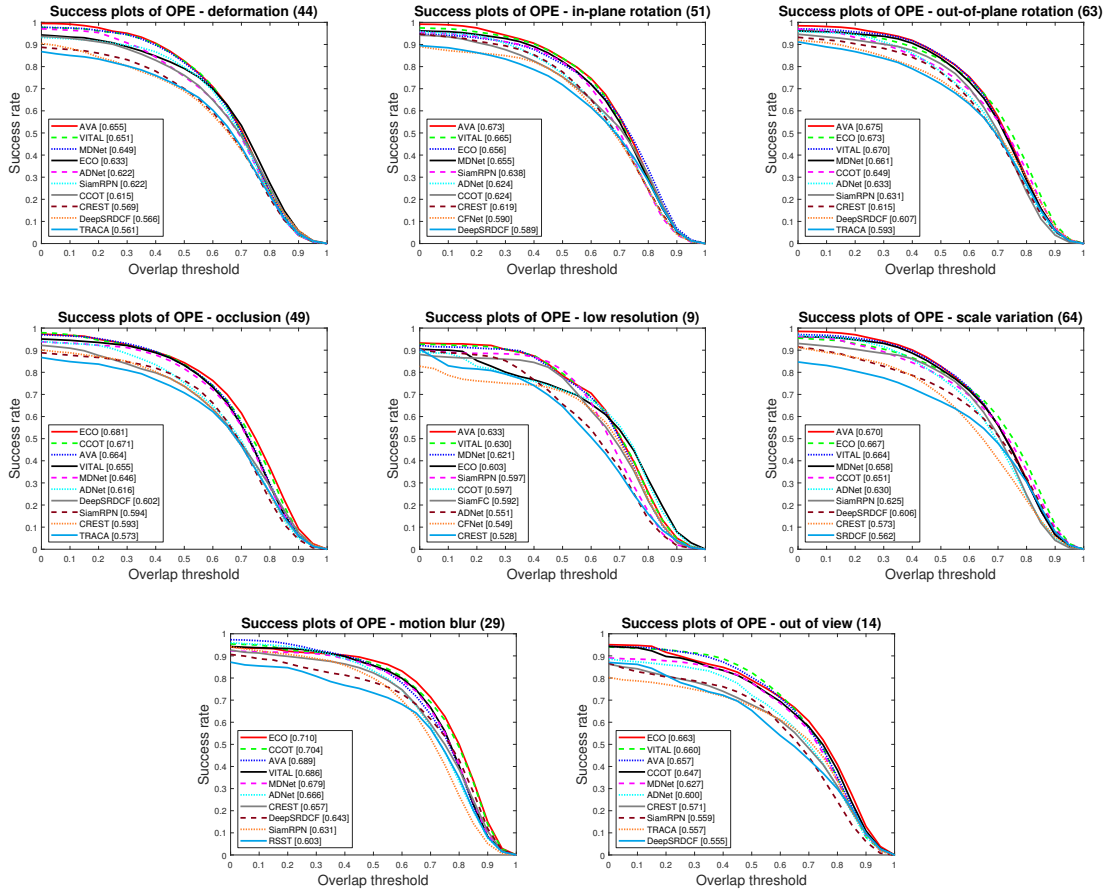


Fig. 6.17: OTB100 OPE success plot of top 10 trackers among the proposed AVA and 53 compared trackers for DEF, IPR, OPR, OCC, LR, SV, MB, and OV challenge subsets.

trackers, Struck [64] is the best tracker yielding an AUC score of 0.463 and a precision score of 0.640. The proposed AVA tracker outperforms Struck by 48.60% in AUC score and 44.38% in precision score. When comparing with the 23 additional recent trackers, the proposed AVA tracker achieves the second highest AUC score of 0.688 and the highest precision score of 0.924. ECO achieves the best AUC score of 0.691, which improves the AUC score of AVA by 0.44%. Specifically, AVA improves the AUC scores of TRACA, CREST, DeepSRDCF, SiamRPN, ADNet, CCOT, MDNet, and VITAL by 14.10%, 10.26%, 8.18%, 7.84%, 6.34%, 2.53%, 1.47%, and 0.88%, respectively. It also outperforms CREST, HDT, DeepSRDCF, SiamRPN, ADNet, CCOT, MDNet, ECO, and VITAL in terms of precision score by 10.13%, 8.96%, 8.58%, 8.32%, 5.00%, 2.90%, 1.65%, 1.54%, and 0.76%, respectively.

In Figure 6.17, we present success plots of the top 10 trackers for 8 challenge subsets containing large appearance changes of target regions. The number of sequences in each specific subset is shown in the parenthesis at the top of its plot. The AUC scores are shown in the parenthesis alongside the legend of the tracker. The results of the other trackers can be found in [7]. It is clear from Figure 6.17 that the proposed AVA tracker successfully handles significant appearance variations of targets due to deformation, scale variation, in-plate rotations, out-plane rotations and occlusions. It achieves the best performance in 5 of these 8 challenge subsets such as DEF, IPR, OPR, LR, and SV and achieves the third best performance in the remaining 3 challenge subsets. Compared to the base model MDNet, AVA achieves the best AUC scores for all the aforementioned challenge subsets. This mainly due to the adaptation mask learned in the generator and discriminator network. This adaptation mask highlights different variations of target over time. In addition, it aligns the discriminative features of target candidates to increase their similarities during frame sequences, while simultaneously maintaining their distinctive properties from the background. Compared to the improved base model VITAL, AVA tracker achieves better performance in all challenge subsets except for the OV subset. This is mainly due to the incorporation of a weighted combination of different parts of target features in each training iteration. Such an incorporation increases the temporal generalization capability of the model and therefore avoids the loss of informative local features over a long temporal span.

6.5.3 Experimental Results on VOT2016

We conduct evaluation on 60 frame sequences in VOT2016 [50]. Based on the VOT challenge protocol, the target is re-initialized using the ground-truth whenever a tracker fails. A tracker is considered as failed in a frame, when the overlap ratio of the tracking result and the ground-truth is zero. The re-initialization happens 5 frames after the failure and the performance is re-evaluated after 10 frames to avoid the bias.

Table 6.4 compares the proposed AVA tracker with the baseline tracker Staple and the top 4 trackers (ECO, VITAL, MDNet, and CCOT) in OTB100 in terms of accuracy, failure rate, and EAO. Values in red indicate the best performance and values in blue

indicate the second best performance. It shows that the proposed AVA tracker obtains a comparable EAO value (e.g., 0.366) with ECO and stands as the second best tracker in terms of EAO. It achieves the best robustness performance and yields the lowest failure rate of 0.68 among the compared trackers. It achieves the second best accuracy of 0.53, which is comparable to the best accuracy of 0.54 tied by ECO, Staple, and VITAL. It is interesting to observe that trackers (e.g., Staple, MDNet, and VITAL) with a higher failure rate (i.e., more re-initialization) still attain better accuracy despite the reduction of the re-initialization bias for accuracy calculation. As a result, the EAO measure, which simultaneously considers both accuracy and failure rate, is considered as the best evaluation metric for the VOT2016 benchmark. The VOT2016 report [50] states that trackers with the EAO value exceeding a limit of 0.251 are considered as state-of-the-art. Table 6.4 clearly demonstrates that the proposed AVA tracker outperforms its peers in terms of its EAO score. It improves the EAO score of MDNet by 42.41 % and the EAO score of VITAL by 13.66 %. This improvement is mainly due to integration of the adversarial network and alignment of the discriminative features of target candidates over time.

	CCOT	ECO	Staple	MDNet	VITAL	AVA
EAO	0.329	0.374	0.294	0.257	0.322	0.366
Failure Rate	0.85	0.72	1.35	1.20	0.98	0.68
Accuracy	0.52	0.54	0.54	0.53	0.54	0.53

Table 6.4: Comparison of the state-of-the-art trackers in terms of EAO, failure rate, and accuracy on the VOT2016 dataset.

6.5.4 Experimental Results on VOT2018

We conduct evaluation on 60 frame sequences in VOT2018 [53]. Table 6.5 summarizes the VOT2018 tracking results of AVA, the baseline tracker Staple, and the top 4 trackers (ECO, VITAL, MDNet, and CCOT). MDNet and VITAL are AVA's peer trackers and ECO and CCOT are two top performing trackers for both OTB100 and VOT2016 datasets. We use red to indicate the best performance and blue to indicate the second best performance. It is

clear that AVA achieves the second best accuracy score of 0.521, with a slightly less accuracy than the MDNet. It improves the accuracy of its adversarial peer tracker, VITAL, by a small margin of 0.7%. It also significantly improves its two peer trackers in both robustness and EAO measurements. Specifically, it improves the robustness of VITAL and MDNet by 21.24% and 9.66%, and the EAO score of VITAL and MDNet by 21.39% and 3.65%, respectively. CCOT achieves a better EAO score and a lower failure rate than our AVA tracker on the VOT2018 dataset. It is due to its success in maintaining lower failure rate in some frame sequences such as *ants1*, *ants3*, and *conduction*. However, the proposed AVA tracker maintains the second best tracking accuracy for both VOT2016 and VOT2018 datasets while ECO, Staple, and VITAL trackers achieve better accuracy than AVA on the VOT2016 dataset and worse accuracy than AVA on the VOT2018 dataset. The stable accuracy of AVA on both datasets is mainly attributed to the integration of the adversarial network and the alignment of the discriminative features of target candidates over time.

	CCOT	ECO	Staple	MDNet	VITAL	AVA
EAO	0.267	0.277	0.164	0.216	0.187	0.227
Failure Rate	1.315	1.117	2.507	1.718	1.996	1.552
Accuracy	0.491	0.482	0.519	0.530	0.514	0.521

Table 6.5: Comparison of the state-of-the-art trackers in terms of EAO, failure rate, and accuracy on the VOT2018 dataset.

6.5.5 Comparison and Discussion

We provide comprehensive comparison of the proposed AVA tracker, eight additional state-of-the-art trackers published in 2018 or 2019, and the top three trackers (ECO, MDNet, and VITAL) in previous subsections on three tracking benchmarks. The eight additional trackers include DAT [31], SiamRPN [90], StructSiam [91], TriSiam [92], TADT [93], UDT [94], LDES [95], and SiamRPN+ [33]. Table 6.6 summarizes the performance of these 12 compared trackers in terms of the AUC score on OTB50 and OTB100 benchmarks and in terms of the EAO score on the VOT2016 benchmark. The AUC and EAO scores of the 11 compared

trackers are directly copied from the researchers' published work. We also include the year and the publication venue that each compared tracker was published. To facilitate comparison, we list the trackers in the chronological order. Table 6.6 clearly demonstrates that the proposed AVA tracker achieves the best AUC score of 0.712 on OTB50, which improves the second best tracker VITAL by 0.28% and the third best tracker ECO by 0.42%. AVA achieves the second best AUC score of 0.688 on OTB100 with a 0.44% decrease when compared to the best tracker ECO and a 0.88% improvement when compared to the third best tracker VITAL. AVA achieves the third best EAO score of 0.366 on VOT2016 with a 3.82% decrease when compared to the best tracker SiamRPN+ and a 2.19% decrease when compared to the second best tracker ECO. It is clear that none of these state-of-the art trackers consistently performs the best on three tracking benchmarks. AVA and ECO are the only two trackers that rank as the top 3 trackers on three tracking benchmarks.

We also summarize the performance comparison of the proposed AVA tracker, its model-based peer tracker MDNet [4], and its adversarial learning-based peer tracker VITAL [5] on OTB50, OTB100, and VOT2016 challenging tracking benchmarks. For the OTB50

Table 6.6: Comparison of the proposed AVA tracker with 11 state-of-the-art trackers on OTB50, OTB100, and VOT2016 challenging tracking benchmarks. Numbers in red, blue, green indicate the best, the second best, and the third best performance, respectively. The dash line (-) indicates no reported result.

Trackers	Year	Publisher	OTB50 (AUC)	OTB100 (AUC)	VOT2016 (EAO)
MDNet	2017	CVPR	0.708	0.678	0.257
ECO	2017	CVPR	0.709	0.691	0.374
DAT	2018	NIPS	0.704	0.673	0.320
SiamRPN	2018	CVPR	-	0.640	0.340
StructSiam	2018	ECCV	0.640	0.620	0.260
TriSiam	2018	ECCV	0.62	0.59	-
VITAL	2018	CVPR	0.710	0.682	0.322
TADT	2019	CVPR	0.680	0.660	0.299
UTD	2019	CVPR	-	0.632	0.301
LDES	2019	AAAI	0.677	0.643	-
SiamRPN+	2019	CVPR	0.670	0.670	0.380
AVA (ours)	2019	JNN	0.712	0.688	0.366

benchmark (Figure 6.15), AVA outperforms MDNet by 0.56% and VITAL by 0.28% in the AUC score and outperforms MDNet by 0.32% and VITAL by 0.11% in the precision score. For the OTB100 benchmark (Figure 6.16), AVA improves the AUC and precision scores of MDNet by 1.47% and 1.65%, respectively. It improves VITAL by 0.88% in terms of the AUC score and by 0.76% in terms of the precision score. For the VOT2016 benchmark (Table 6.4), AVA attains comparable accuracy with both MDNet and VITAL. However, it drastically improves the failure rate of both MDNet and VITAL. This results in an EAO score improvement of 42.41% and 13.66% over MDNet and VITAL, respectively. For eight challenge subsets containing large appearance changes of target regions (Figure 6.17), AVA achieves better AUC scores than both MDNet and VITAL when a target undergoes deformation, in-plane rotations, out-plane rotations, occlusions, low resolution, scale variation, and motion blur. It achieves a better AUC score than MDNet and a comparable AUC score as VITAL when target is out of view. Overall, the proposed AVA tracker uses the model of MDNet as a base network. Unlike MDNet, AVA aligns the feature distributions of target regions over time by learning an adaptation mask adversarially. This adaptation mask increases the model generalization by highlighting the informative features of target regions over time and dropping out some non-informative features. Therefore, the more generalized model tends to attain the similarity between the features distributions of recent and earlier target regions while maintaining distinctive properties from the background. The VITAL tracker also learns a mask during tracking. However, it prepares 9 random masks where each mask covers one of 9 locations in the 3×3 feature map in each training iteration and learns an optimal mask in a least square optimization problem. Therefore, this optimal mask is updated to cover only one part of local features in each iteration, which leads to the loss of the informative local features during training. Unlike the optimal mask learned in VITAL, the adaptation mask learned in AVA increases the temporal generalization capability and avoids the loss of informative local features over time by incorporating a weighted combination of multiple parts of target features in each training iteration via a gradient reverse layer.

All CNN-based trackers aim to construct a model to classify the candidates in each

frame as a target or a background. They update the model during tracking to keep track of the latest changes of target regions. However, one major shortcoming is that the model may overfit to the initial target appearances, which leads to the failure to discriminate the similarity of the current target with its tracked targets in earlier frames when a target appearance has a drastic change. The proposed AVA tracker aims to address this shortcoming using adversarial learning. Our extensive experimental results show that the AVA tracker outperforms most state-of-the-art trackers in various challenges (e.g., occlusion, fast motion, scale variation, rotations, etc) for OTB and VOT benchmarks. However, like all other trackers, its performance decreases for the most challenging sequences such as *soccer, bird1, fenando, rabbit* where a target suffers from heavy occlusion, *skiing, trans, motorRolling, matrix* where a target's scale changes with a fast rate, and *matrix, iroman, gymnastics* where a target's motion rate is high. To the best of our knowledge, none of the existing trackers is able to handle all the severe challenges that lead to significant appearance changes of the tracked targets. Designing a tracker that is able to produce a model to discriminate target regions from background in all frames of challenging sequences is still an active and open computer vision task.

6.6 Discussion

As discussed in this dissertation, SMTMVT and SGLST are categorized as hand-crafted feature-based trackers and STLDF and AVA are considered as deep learning-based trackers. Each of the proposed trackers aims to address the drawbacks of their peers and improve the performance of their peers in terms of AUC scores and other evaluation measures (e.g., EAO scores and failure rates) if applicable. Despite improvements over their compared methods, the four proposed trackers have different applicability and performance under certain conditions. In this section, we thoroughly compare our trackers (i.e., SMTMVT, SGLST, STLDF, and AVA) in terms of their applications, performance, and shortcomings.

Table 6.7 presents the overall AUC scores of our proposed trackers, namely, SMTMVT, two variants of SGLST (i.e., SGLST_Color and SGLST_HOG), STLDF, and AVA, on the OTB50 and OTB100 tracking benchmarks. The AVA tracker achieves the highest AUC scores

Table 6.7: Comparison of the proposed trackers in terms of AUC scores on OTB50 and OTB100 tracking benchmarks

	SMTMVT	SGLST_Color	SGLST_HOG	STLDF	AVA
OTB50 (AUC)	0.507	0.523	0.556	0.604	0.712
OTB100 (AUC)	0.452	0.453	0.524	0.586	0.688

on both benchmarks. Specifically, it significantly outperforms SMTMVT, SGLST_Color, SGLST_HOG, and STLDF by 40.43%, 36.14%, 28.06%, and 17.88% on OTB50 benchmark, respectively. It improves the AUC scores of SMTMVT, SGLST_Color, SGLST_HOG, and STLDF by 52.21%, 51.88%, 31.30%, and 17.41% on OTB100 benchmark, respectively. Incorporating deep features to represent each candidate improves the tracking performance as STLDF achieves better AUC scores than the other hand-crafted feature-based sparse trackers (e.g., SMTMVT, SGLST_Color, and SGLST_HOG) on OTB50 and OTB100 tracking benchmarks.

To provide more insight on the performance of proposed trackers on tracking challenges, we present the OPE success plots on 6 challenges (OCC, SV, DEF, FM, IPR, and MB) of the OTB100 benchmark in Figure 6.18. It shows that AVA outperforms all the trackers in these challenge subsets due to its full utilization of the power of deep learning. Moreover, incorporating deep features in STLDF results in better performance in tracking challenges compared to other hand-crafted feature-based sparse trackers including SMTMVT, SGLST_Color, and SGLST_HOG.

To visually show the success and failure cases of the proposed trackers, we select eight frame sequences titled *bird1*, *bolt1*, *dragonBaby*, *jump*, *coupon*, *carScale*, *blurOwl*, and *human8*. The sequences are selected since they contain variety of full and partial occlusion, fast motion, deformation, scale variation, motion blur, and low resolution. Figure 6.19 demonstrates the tracking results of our proposed trackers on four selected frames of the four sequences, where the proposed AVA obtains better tracking results than the other three trackers. Figure 6.20 demonstrates the tracking results of our proposed trackers on four selected frames of another four sequences, where at least one of the other trackers achieves comparable or better performance than AVA.

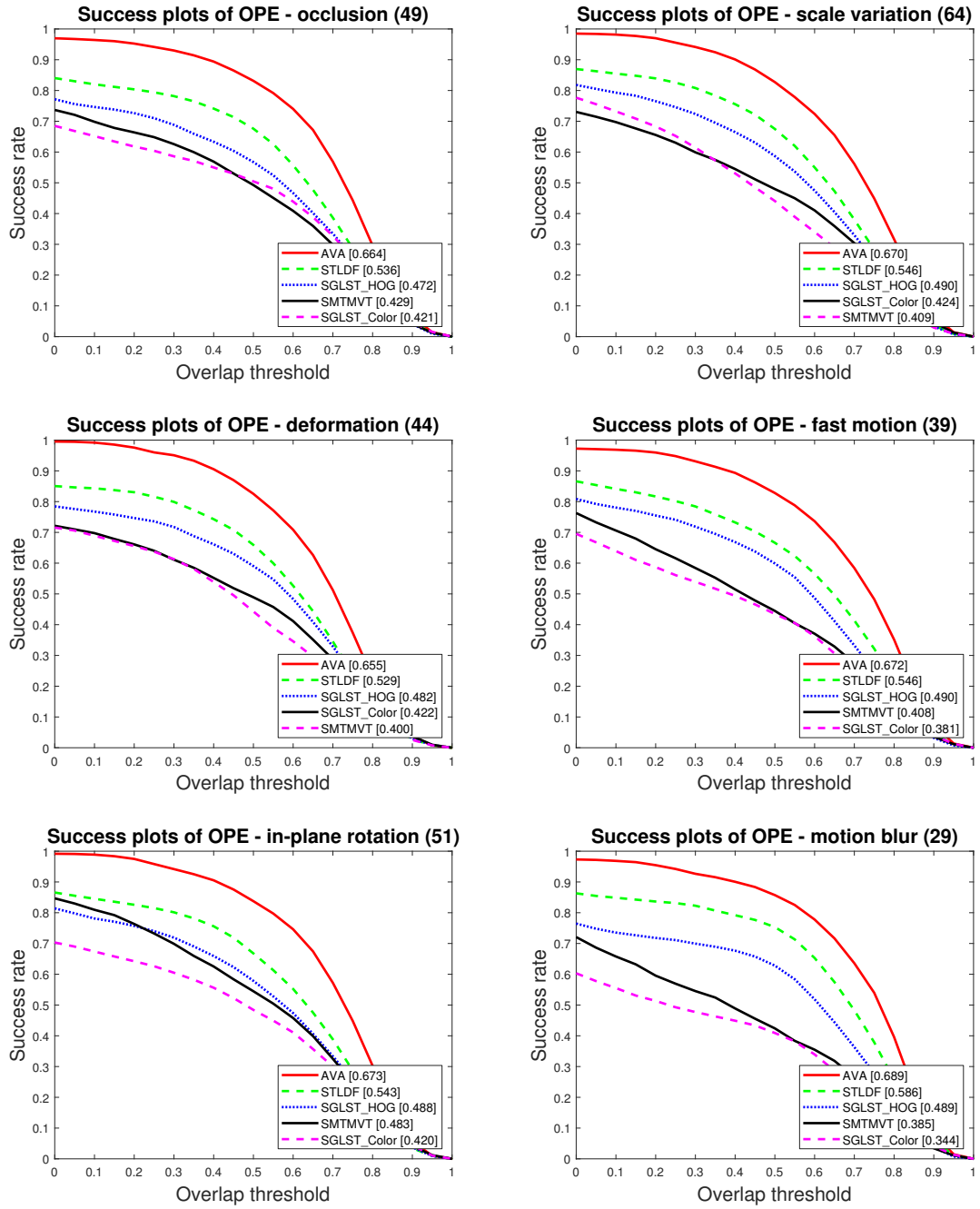


Fig. 6.18: OTB100 OPE success plot of our trackers for OCC, SV, DEF, FM, IPR, and MB challenge subsets.

We first discuss the tracking results of four proposed trackers on the four sequences shown in Figure 6.19. For the *bird1* sequence (top row), the bird undergoes a full occlusion. Once the occlusion happens, the trackers try to expand the search area to find the bird. Since

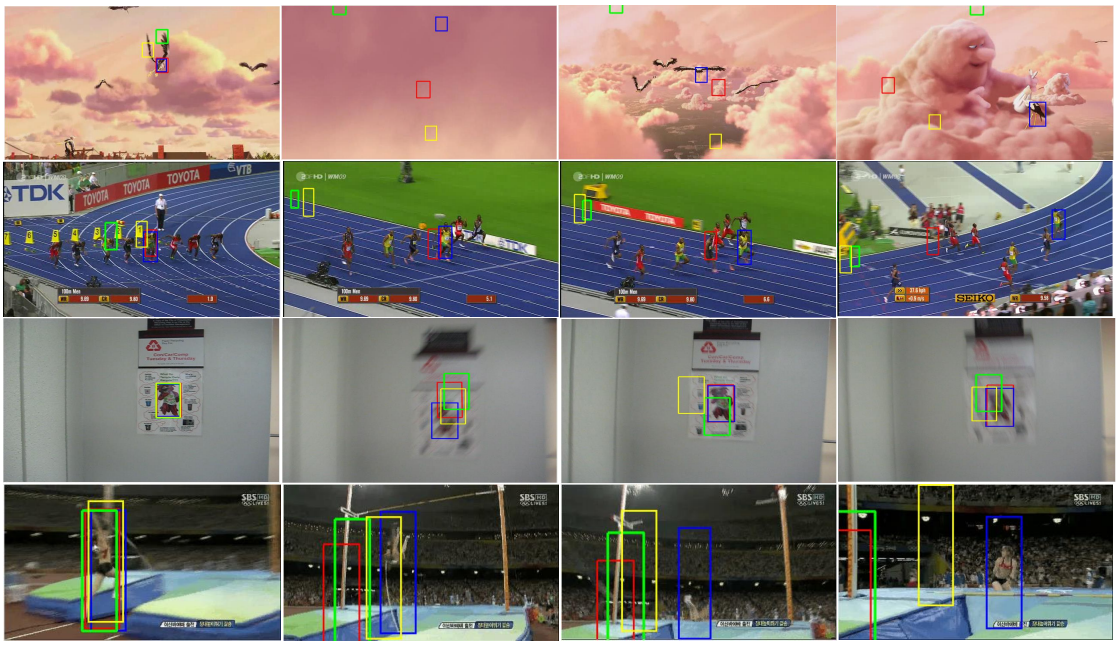


Fig. 6.19: Comparison of the tracking results of our proposed trackers on *bird1*, *bolt1*, *blurOwl*, and *jump* frame sequences. Results are best viewed on high-resolution displays. (—AVA, —STLDF, —SGLST_HOG, —SMTMVT)

the occlusion continues for a long time, the dictionaries used in the proposed SMTMVT, SGLST_HOG, and STLDF trackers include wrong target templates, which lead to their failure to find the bird when it appears again. In contrast, AVA successfully finds the bird after occlusion. For the *bolt1* sequence (second row), hand-crafted feature-based trackers (SMTMVT and SGLST_HOG) fail to track Usain Bolt from the beginning of sequence. STLDF mistakenly considers another runner as Bolt in the middle of sequence. It is mainly because Bolt moves very fast between two consecutive frames and STLDF tries to generate tracking candidates around the location of Bolt in the previous frame. AVA successfully keeps track of Bolt throughout the entire sequence. For the *blurOwl* sequence (third row), the camera moves drastically in various directions, which results in blurred frames. SMTMVT and SGLST_HOG cannot locate the owl due to the use of hand-crafted features. In contrast, STLDF and AVA track the owl in this frame sequence prosperously. For the *jump* sequence (bottom row), the sparse trackers (i.e., SMTMVT, SGLST_HOG and STLDF) fail. AVA successfully tracks the jumper. However, it cannot fully handle jumper's scale variations. All

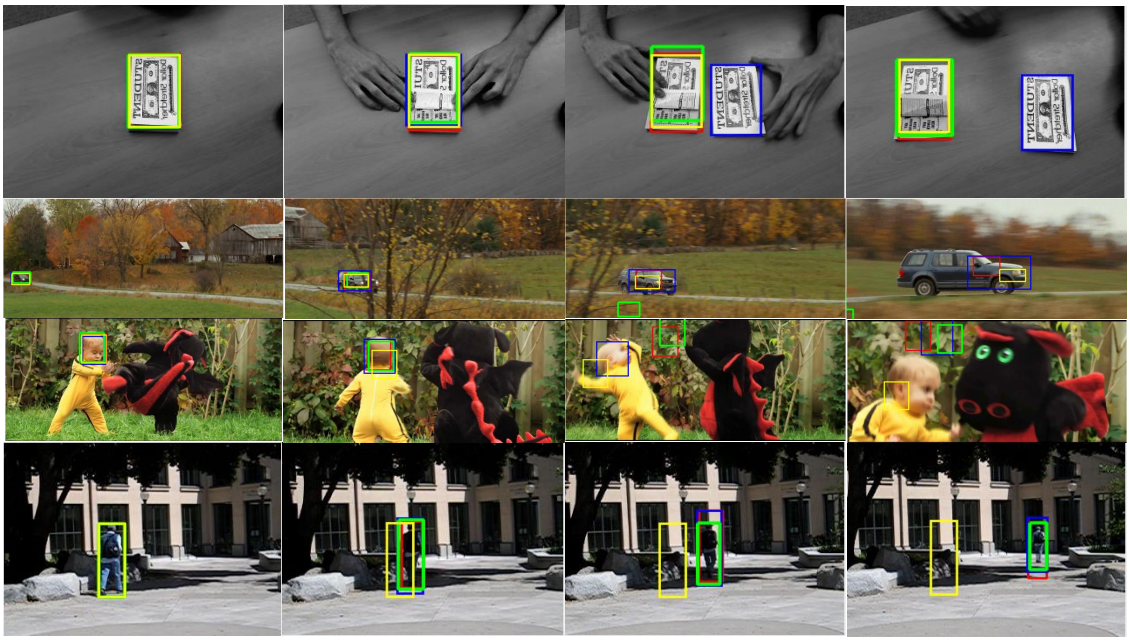


Fig. 6.20: Comparison of the tracking results of our proposed trackers on *coupon*, *carScale*, *dragonBaby*, and *human8* frame sequences. Results are best viewed on high-resolution displays.
(—AVA, —STLDF, —SGLST_HOG, —SMTMVT)

the successful tracking results obtained by AVA are attributed to its utilization of adversarial learning to minimize the gap between target distributions over time.

We then discuss the tracking results of four proposed trackers on the four sequences shown in Figure 6.20. For the *coupon* sequence (top row), AVA fails to track the coupon due to the similarities between the two coupons in the frames while STLDF, SGLST_HOG, and SMTMVT successfully track the coupon. For the *carScale* sequence (second row), the car scale varies throughout the frame sequence and the car gets occluded by a tree near the road in the middle of sequence. When occlusion occurs, SGLST_HOG fails and the other three trackers can follow the car until the end of the sequence. However, they cannot handle the scale of the car nicely with AVA attaining more accurate tracking results than STLDF and SMTMVT. For the *dragonBaby* sequence (third row), the baby moves and rotates rapidly. Towards the end of the sequence, the baby appears significantly bigger in size since the camera zooms in quickly. All four trackers can successfully handle various rotations in

this sequence. However, the motion blur makes SMTMVT and SGLST_HOG fail to locate the baby due to their use of lower-level features. When the camera zooms in quickly, all four trackers try to locate the baby in a similar size around his location in the previous frame and therefore fail to find the baby whose size enlarges immensely. For the *human8* sequence (bottom row), all the frames have low resolutions. SMTMVT fails to track the person. All the other three trackers can track the person. However, SLGST_HOG can handle the scale variation more accurately than STLDF and AVA.

We evaluate the average tracking times of AVA, STLDF, SGLST, and SMTMVT on the OBT100 dataset to compare their tracking speed in terms of average frame per second (fps). Average tracking times are 1.2 fps for AVA, 0.45 fps for STLDF, 0.55 fps for SGLST, and 0.52 fps for SMTMVT. AVA achieves the best performance in terms of not only AUC scores (as discussed) but also tracking speed. SGLST, STLDF and SMTMVT are slower to track the object. It is mainly due to the involvement of solving optimization model in each frame.

In general, SMTMVT and SGLST trackers utilize hand-crafted features such as intensity, HOG, LBP, and histogram and they do not require any additional learning step. In contrast, STLDF and AVA trackers require a pre-trained model before tracking to automatically extract deep features. As a result, SMTMVT and SGLST are more applicable for the situations when loading a pre-trained model is not possible due to costs and low-memory sensors and devices. SMTMVT, SGLST, and STLDF are sparse trackers, which are computational expensive. Since these sparse trackers have to solve an optimization model in each frame to find the target among a number of candidates. This limits their utility in real-time applications when compared to AVA.

CHAPTER 7

Conclusions

In this dissertation, we introduced 4 different trackers and compared their performance with the most recent trackers on challenging tracking benchmarks. Each of the proposed trackers aims to address some of the drawbacks of their peers. Specifically, we summarize the strategy and performance of each tracker as follows:

- We propose a robust SMTMVT method that uses sparse representation in the particle filter framework to track objects in frame sequences. By introducing the nuclear norm regularization, we represent all views of a target candidate using the same subset of templates in the target dictionaries. We further equalize the representation coefficients of activated templates for all views. The proposed model is efficiently solved by a numerical algorithm based on the PG method. The results on 15 publicly frame sequences and OTB50 and OTB100 tracking benchmarks demonstrate that the SMTMVT method outperforms various state-of-the-art trackers.

- We propose a novel tracker, called structured group local sparse tracker (SGLST), which exploits local patches within target candidates in the particle filter framework. Unlike conventional local sparse trackers, SGLST employs a new convex optimization model to preserve spatial layout structure among the local patches. To solve the proposed optimization model, we develop an efficient numerical algorithm consisting of two subproblems with closed-form solutions based on ADMM. We test the performance of the proposed tracker with two types of features including gray-level intensity features and HOG features. The qualitative and quantitative results on 16 publicly available frame sequences demonstrate that SGLST_HOG outperforms all compared state-of-the-art trackers in terms of the overlap score. The experimental results on OTB50 and OTB100 tracking benchmarks demonstrate that SGLST_HOG outperforms all

compared state-of-the-art trackers except the MEEM tracker in terms of the average AUC score.

- We propose a structured tracker using local deep features (STLDF), which exploits CNN deep features of local patches within target candidates and represents them in a novel optimization problem. The proposed optimization model combines the CNN deep features of local patches of each target candidate with a group-sparsity regularization term to encourage the tracker to sparsely select appropriate local patches of the same subset of templates. We design a fast and parallel numerical algorithm by deriving the augmented Lagrangian of the optimization model into two close-form problems: the quadratic problem and the Euclidean norm projection onto probability simplex constraints problem. STLDF outperforms existing sparse trackers by incorporating local deep features of target candidates and maintaining the spatial relation between them. The extensive experimental results on OTB50 and OTB100 demonstrate that STLDF outperforms various state-of-the-art methods including its two variant trackers, representative conventional and recent sparse trackers, correlation filter-based trackers, and convolutional neural network based trackers in terms of AUC and precision scores.

- We propose an appearance variation adaptation (AVA) tracker that is capable of handling the significant appearance variations of targets. Specifically, we align feature distributions of target regions over a long time span by adversarially learning an adaptation mask. This adaptation mask is applied on the discriminative features of target regions to increase the generalization of the classification network. We design an adversarial network, which consists of a generator network and a discriminator network competing with each other over optimization of a discriminator loss between recent and earlier target regions. The discriminator network aims to distinguish recent target regions from earlier ones by minimizing the discriminator loss, while the generator network aims to produce an adaptation mask to maximize the discriminator loss. We incorporate the adversarial network with the classification network to align informative features of recent and earlier target regions during tracking, while maintaining the

network classification accuracy to distinguish targets and backgrounds. We add a gradient reverse layer to solve the aforementioned mini-max optimization in an end-to-end manner. Our extensive experiments on OTB and VOT challenge benchmarks show that the proposed AVA tracker achieves favorable performance against state-of-the-arts trackers.

REFERENCES

[1] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1822–1829.

[2] ——, "Visual tracking via coarse and fine structural local sparse appearance models," *IEEE Transactions on Image processing*, vol. 25, no. 10, pp. 4555–4564, 2016.

[3] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *International Conference on Machine Learning*, 2015, pp. 597–606.

[4] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4293–4302.

[5] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang, "Vital: Visual tracking via adversarial learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8990–8999.

[6] 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.

[7] ——, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.

[8] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[9] S. Salti, A. Cavallaro, and L. Di Stefano, "Adaptive appearance modeling for video tracking: Survey and evaluation," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4334–4348, 2012.

[10] 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.

[11] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder, "The visual object tracking vot2015 challenge results," in *Proceedings of the IEEE international conference on computer vision workshops*, 2015, pp. 1–23.

[12] T. Zhang, S. Liu, C. Xu, S. Yan, B. Ghanem, N. Ahuja, and M.-H. Yang, "Structural sparse tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 150–158.

[13] T. Zhang, A. Bibi, and B. Ghanem, "In defense of sparse tracking: Circulant sparse tracker," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3880–3888.

[14] S. Avidan, "Ensemble tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 2, 2007.

[15] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting." in *Bmvc*, vol. 1, no. 5, 2006, p. 6.

[16] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *European conference on computer vision*. Springer, 2008, pp. 234–247.

[17] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 983–990.

[18] Z. Kalal, J. Matas, and K. Mikolajczyk, "Pn learning: Bootstrapping binary classifiers by structural constraints," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 49–56.

[19] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.

[20] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5388–5396.

[21] Q. Hu, Y. Guo, Y. Chen, and J. Xiao, "Correlation filter tracking: Beyond an open-loop system," in *British Machine Vision Conference (BMVC)*, 2017.

[22] 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.

[23] 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.

[24] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 798–805.

[25] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International journal of computer vision*, vol. 77, no. 1-3, pp. 125–141, 2008.

[26] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1269–1276.

[27] G. Roffo and S. Melzi, "Online feature selection for visual tracking," 2016.

[28] 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.

[29] X. Mei, Z. Hong, D. Prokhorov, and D. Tao, "Robust multitask multiview tracking in videos," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 11, pp. 2874–2890, 2015.

[30] T. Zhang, C. Xu, and M.-H. Yang, "Robust structural sparse tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[31] S. Pu, Y. Song, C. Ma, H. Zhang, and M.-H. Yang, "Deep attentive tracking via reciprocative learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 1931–1941.

[32] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.

[33] Z. Zhang and H. Peng, "Deeper and wider siamese networks for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4591–4600.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[35] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[36] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang, "Hedged deep tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4303–4311.

[37] H. Nam, M. Baek, and B. Han, "Modeling and propagating cnns in a tree structure for visual tracking," *arXiv preprint arXiv:1608.07242*, 2016.

[38] M. Javanmardi and X. Qi, "Robust structured multi-task multi-view sparse tracking," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*, July 2018, pp. 1–6.

[39] ——, "Structured group local sparse tracker," *IET Image Processing*, vol. 13, no. 8, pp. 1391–1399, 2019.

[40] M. Javanmardi, A. H. Farzaneh, and X. Qi, "A robust structured tracker using local deep features," *Electronics*, vol. 9, no. 5, p. 846, 2020.

[41] M. Javanmardi and X. Qi, "Appearance variation adaptation tracker using adversarial network," *Neural Networks*, vol. 129, pp. 334 – 343, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608020302264

[42] X. Mei and H. Ling, "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.

[43] N. Parikh, S. Boyd *et al.*, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[44] Y. Nesterov, "Smooth minimization of non-smooth functions," *Math. Prog.*, vol. 103, no. 1, pp. 127–152, 2005.

[45] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[47] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3074–3082.

[48] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[49] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," *arXiv preprint arXiv:1409.7495*, 2014.

[50] M. Kristan and et al, "The visual object tracking vot2016 challenge results," in *Computer Vision – ECCV 2016 Workshops*. Cham: Springer International Publishing, 2016, pp. 777–823.

[51] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[52] Z. Cao, L. Ma, M. Long, and J. Wang, "Partial adversarial domain adaptation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 135–150.

[53] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Č. Zajc, T. Vojír, G. Bhat, A. Lukežič, A. Eldesokey, G. Fernández, Á. García-Martín, Á. Iglesias-Arias, A. A. Alatan, A. González-García, A. Petrosino, A. Memarmoghadam, A. Vedaldi, A. Muhič, A. He, A. Smeulders, A. G. Perera, B. Li, B. Chen, C. Kim, C. Xu, C. Xiong, C. Tian, C. Luo, C. Sun, C. Hao, D. Kim, D. Mishra, D. Chen, D. Wang, D. Wee, E. Gavves, E. Gundogdu, E. Velasco-Salido, F. S. Khan, F. Yang, F. Zhao, F. Li, F. Battistone, G. De Ath, G. R. K. S. Subrahmanyam, G. Bastos, H. Ling, H. K. Galoogahi, H. Lee, H. Li, H. Zhao, H. Fan, H. Zhang, H. Possegger, H. Li, H. Lu, H. Zhi, H. Li, H. Lee, H. J. Chang, I. Drummond, J. Valmadre, J. S. Martin, J. Chahl, J. Y. Choi, J. Li, J. Wang, J. Qi, J. Sung, J. Johnander, J. Henriques, J. Choi, J. van de Weijer,

J. R. Herranz, J. M. Martínez, J. Kittler, J. Zhuang, J. Gao, K. Grm, L. Zhang, L. Wang, L. Yang, L. Rout, L. Si, L. Bertinetto, L. Chu, M. Che, M. E. Maresca, M. Danelljan, M.-H. Yang, M. Abdelpakey, M. Shehata, M. Kang, N. Lee, N. Wang, O. Miksik, P. Moallem, P. Vicente-Moñivar, P. Senna, P. Li, P. Torr, P. M. Raju, Q. Ruihe, Q. Wang, Q. Zhou, Q. Guo, R. Martín-Nieto, R. K. Gorthi, R. Tao, R. Bowden, R. Everson, R. Wang, S. Yun, S. Choi, S. Vivas, S. Bai, S. Huang, S. Wu, S. Hadfield, S. Wang, S. Golodetz, T. Ming, T. Xu, T. Zhang, T. Fischer, V. Santopietro, V. Štruc, W. Wei, W. Zuo, W. Feng, W. Wu, W. Zou, W. Hu, W. Zhou, W. Zeng, X. Zhang, X. Wu, X.-J. Wu, X. Tian, Y. Li, Y. Lu, Y. W. Law, Y. Wu, Y. Demiris, Y. Yang, Y. Jiao, Y. Li, Y. Zhang, Y. Sun, Z. Zhang, Z. Zhu, Z.-H. Feng, Z. Wang, and Z. He, "The sixth visual object tracking vot2018 challenge results," in *Computer Vision – ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Cham: Springer International Publishing, 2019, pp. 3–53.

[54] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *European conference on computer vision*. Springer, 2016, pp. 445–461.

[55] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: Algorithms and benchmark," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5630–5644, 2015.

[56] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[57] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 7, pp. 971–987, 2002.

[58] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1635–1650, 2010.

[59] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*. IEEE, 2012, pp. 2042–2049.

[60] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *ICCV*, 2011.

[61] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.

[62] D. Wang and H. Lu, "Visual tracking via probability continuous outlier model," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 3478–3485.

[63] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *European conference on computer vision*. Springer, 2012, pp. 702–715.

[64] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. Torr, "Struck: Structured output tracking with kernels," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 10, pp. 2096–2109, 2016.

[65] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: robust tracking via multiple experts using entropy minimization," in *Proc. of the European Conference on Computer Vision (ECCV)*, 2014.

[66] K. Zhang, Q. Liu, Y. Wu, and M.-H. Yang, "Robust visual tracking via convolutional networks without training," *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1779–1792, 2016.

[67] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with gaussian processes regression," in *European Conference on Computer Vision*. Springer, 2014, pp. 188–203.

[68] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l1 tracker using accelerated proximal gradient approach," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1830–1837.

[69] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Low-rank sparse learning for robust visual tracking," in *European conference on computer vision*. Springer, 2012, pp. 470–484.

[70] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 749–765.

[71] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3119–3127.

[72] J. Ning, J. Yang, S. Jiang, L. Zhang, and M.-H. Yang, "Object tracking via dual linear structured svm and explicit feature map," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4266–4274.

[73] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 689–692.

[74] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *European conference on computer vision*. Springer, 2014, pp. 254–265.

[75] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4310–4318.

[76] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr, "Staple: Complementary learners for real-time tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1401–1409.

[77] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1838–1845.

[78] J. Choi, H. J. Chang, S. Yun, T. Fischer, Y. Demiris, J. Y. Choi *et al.*, "Attentional correlation filter network for adaptive visual tracking." in *CVPR*, vol. 2, no. 6, 2017, p. 7.

[79] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*. Springer, 2016, pp. 850–865.

[80] M. Javanmardi, "Source code for appearance variation adaptation tracker using adversarial network," *Accessed 15 December 2020*. [Online]. Available: https://gitlab.com/mamrez7/ava-tracker

[81] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 749–758.

[82] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 58–66.

[83] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2711–2720.

[84] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2805–2813.

[85] J. Choi, H. Jin Chang, J. Jeong, Y. Demiris, and J. Young Choi, "Visual tracking using attention-modulated disintegration and integration," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4321–4330.

[86] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 472–488.

[87] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6638–6646.

[88] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. Lau, and M.-H. Yang, "Crest: Convolutional residual learning for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2555–2564.

[89] J. Choi, H. Jin Chang, T. Fischer, S. Yun, K. Lee, J. Jeong, Y. Demiris, and J. Young Choi, "Context-aware deep feature compression for high-speed visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 479–488.

[90] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8971–8980.

[91] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu, "Structured siamese network for real-time visual tracking," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 351–366.

[92] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 459–474.

[93] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang, "Target-aware deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1369–1378.

[94] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li, "Unsupervised deep tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1308–1317.

[95] Y. Li, J. Zhu, S. C. Hoi, W. Song, Z. Wang, and H. Liu, "Robust estimation of similarity transformation for visual object tracking," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8666–8673.

CURRICULUM VITAE

Mohammadreza Javanmardi

Published Articles

- **Mohammadreza Javanmardi**, Ziqi Song, Xiaojun Qi. *"A Fusion Approach to Detect Traffic Signs Using Registered Color Images and Noisy Airborne LiDAR Data,"* Under final review in *MDPI Applied Science,* , 2020. **Impact Factor: 2.474, h5-index: 31**

- Brendan Robeson, **Mohammadreza Javanmardi**, Xiaojun Qi. *"Object Tracking Using Temporally Matching Filters"*, IET Image Processing 2020. **Impact Factor: 1.995, h5-index: 29**

- **Mohammadreza Javanmardi**, Xiaojun Qi. *"Appearance Variation Adaptation Tracker Using Adversarial Network"*, Journal of Neural Network, 2020. **Impact Factor: 5.535, h5-index: 64**

- **Mohammadreza Javanmardi**, Amir Hossein Farzaneh, and Xiaojun Qi. *"A Robust Structured Tracker Using Local Deep Features"*, MDPI Electronics, Deep Neural Networks and Their Applications, 2020. **Impact Factor: 2.412, h5-index: 31**

- **Mohammadreza Javanmardi**, Xiaojun Qi. *" Structured Group Local Sparse Tracker,"* in *IET Image Processing*, 2019. **Impact Factor: 1.995, h5-index: 29**

- **Mohammadreza Javanmardi**, Ziqi Song, Xiaojun Qi. *"Automated Traffic Sign and Light Pole Detection in Mobile Laser Scanning Data,"* In *IET Intelligent Transport Systems*, 2019. **Impact Factor: 2.480, h5-index: 26**

- **Mohammadreza Javanmardi**, Xiaojun Qi. *"Robust Structured Multi-Task Multi-View Spars Tracking"* In *IEEE International Conference on Multimedia and Expo (ICME)*, 2018. **h5-index: 30, Qualis rank: A1, ERA rank: B**

- **Mohammadreza Javanmardi**, Xiaojun Qi. *"Visual Tracking of Resident Space Objects via a RFS-Based Multi- Bernoulli Track-Before-Detect Method,"* In *Journal of Machine Vision and Applications*, 2018. **Impact Factor: 1.6, h5-index: 26**

- **Mohammadreza Javanmardi**, Mehran Yazdi, and Mohammadali Masnadi-Shirazi. *"Fast and Robust L0-tracker Using Compressive Sensing,"* In *IEEE International Conference on Pattern Recognition and Image Analysis (IPRIA)*, 2015.