GOOD FEATURES TO CORRELATE FOR VISUAL TRACKING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

ERHAN GÜNDOĞDU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2017

Approval of the thesis:

GOOD FEATURES TO CORRELATE FOR VISUAL TRACKING

submitted by **ERHAN GÜNDOĞDU** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Tolga Çiloğlu
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. A. Aydın Alatan
Supervisor, **Electrical and Electronics Eng. Dept., METU**

Examining Committee Members:

Prof. Dr. Murat Tekalp
Electrical and Electronics Eng. Dept., Koç University

Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. Nazlı İkizler Cinbiş
Computer Engineering Dept., Hacettepe University

Assist. Prof. Dr. Elif Vural
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Fatih Kamışlı
Electrical and Electronics Engineering Dept., METU

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: ERHAN GÜNDOĞDU

Signature :

ABSTRACT

GOOD FEATURES TO CORRELATE FOR VISUAL TRACKING

Gündoğdu, Erhan
Ph.D., Department of Electrical and Electronics Engineering
Supervisor : Prof. Dr. A. Aydın Alatan

September 2017, 148 pages

Estimating object motion is one of the key components of video processing and the first step in applications which require video representation. Visual object tracking is one way of extracting this component, and it is one of the major problems in the field of computer vision. Numerous discriminative and generative machine learning approaches have been employed to solve this problem. Recently, correlation filter based (CFB) approaches have been popular due to their computational efficiency and notable performances on benchmark datasets. The ultimate goal of CFB approaches is to find a filter (*i.e.*, template) which can produce high correlation outputs around the actual object location and low correlation outputs around the locations that are far from the object. Nevertheless, CFB visual tracking methods suffer from many challenges, such as occlusion, abrupt appearance changes, fast motion and object deformation. The main reasons of these sufferings are forgetting the past poses of the objects due to the simple update stages of CFB methods, non-optimal model update rate and features that are not invariant to appearance changes of the target object.

In order to address the aforementioned disadvantages of CFB visual tracking methods, this thesis includes three major contributions. First, a spatial window learning method is proposed to improve the correlation quality. For this purpose, a window that is to be element-wise multiplied by the object observation (or the correlation filter) is learned by a novel gradient descent procedure. The learned window is capable of suppressing/highlighting the necessary regions of the object, and can improve the tracking performance in the case of occlusions and object deformation. As the second contribution, an ensemble of trackers algorithm is proposed to handle the issues of non-optimal learning rate and forgetting the past poses of the object. The trackers in the ensemble are organized in a binary tree, which stores individual expert trackers at its nodes. During the course of tracking, the relevant expert trackers to the most recent object appearance are activated and utilized in the localization and update stages. The proposed ensemble method significantly improves the tracking accuracy, especially when the expert trackers are selected as the CFB trackers utilizing the proposed window learning method. The final contribution of the thesis addresses the feature learning problem specifically focused on the CFB visual tracking loss function. For this loss function, a novel backpropagation algorithm is developed to train any fully deep convolutional neural network. The proposed gradient calculation, which is required for backpropagation, is performed efficiently in both frequency and image domain, and has a linear complexity with the number of feature maps. The training of the network model is fulfilled on carefully curated datasets including well-known difficulties of visual tracking, *e.g.*, occlusion, object deformation and fast motion. When the learned features are integrated to the state-of-the-art CFB visual trackers, favorable tracking performance is obtained on benchmark datasets against the CFB methods that employ hand-crafted features or deep features extracted from the pre-trained classification models.

Keywords: visual object tracking, mixture of experts, correlation filters, convolutional neural networks

ÖZ

KORELASYON İLE GÖRSEL TAKİP İÇİN İYİ ÖZNİTELİKLER

Gündoğdu, Erhan
Doktora, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. A. Aydın Alatan

Eylül 2017 , 148 sayfa

Nesne hareketi tahmin etme, video işlemenin temel bileşenlerinden biridir ve video temsiline ihtiyaç duyan uygulamalardaki ilk adımdır. Görsel nesne takibi, bu bileşenin çıkarılma yollarından birisi olup, bilgisayarla görme alanındaki önemli bir problemdir. Bu sorunu çözmek için geçmişte birçok ayırımcı ve üretken makine öğrenme yaklaşımları kullanılmıştır. Son zamanlarda, korelasyon süzgeci tabanlı (KST) yaklaşımlar, hesaplama verimliliği ve karşılaştırma amaçlı kullanılan veri kümeleri üzerinde dikkate değer performansları nedeniyle popüler olmuştur. KST yaklaşımlarının nihai amacı, gerçek nesne konumu etrafında yüksek korelasyon çıktıları üretebilen ve nesneden uzaktaki yerler çevresinde düşük korelasyon çıktıları üretebilen bir süzgeci (diğer bir deyişle şablon) hesaplamaktır. Bununla birlikte, KST görsel takip yöntemleri; kapanma, ani görünüm değişiklikleri, hızlı hareket ve nesne deformasyonu gibi birçok durumda zorlanmaktadır. KST yöntemlerinin basit güncelleme aşamaları, en iyi olmayan model güncelleme oranı ve hedef nesnenin görünüm değişikliklerine karşı sağlam olamaması KST yöntemlerinin takip ve konumlandırma performanslarındaki azalmaların sebepleri olarak gösterilebilir.

KST görsel takip yöntemlerinin yukarıda belirtilen dezavantajlarını gidermek için bu tez üç önemli katkı içermektedir. İlk olarak, korelasyon kalitesini arttırmak için mekânsal pencere öğrenme yöntemi önerilmiştir. Bu amaçla, nesne görüntüsü (veya korelasyon filtresi) ile çarpılacak bir pencere, yeni bir gradyan iniş prosedürüyle öğrenilir. Öğrenilen pencere, nesnenin gerekli bölgelerini bastırma/vurgulama yeteneğine sahiptir ve kapanma ve nesne deformasyonu durumunda takip performansını artırabilir. İkinci bir katkı olarak, en iyi olmayan öğrenme hızı ve nesnenin geçmiş pozlarını unutma zorlukları ile baş edebilmek için birden çok takipçiyi (hedef takip yöntemi) içeren bir takip grubu yöntemi önerilmiştir. Gruptaki takipçiler, ikili bir ağaçta düzenlenir, ve her takipçi ağacın düğümlerinde saklanır. Takip sırasında, en son nesne görünümüne ilişkin uzman takipçiler etkinleştirilir ve konumlandırma ve güncelleme aşamalarında kullanılır. Önerilen takipçiler grubu yönteminin, uzman takipçilerin bu tezde önerilen pencere öğrenme yöntemiyle birleştirilmesi ile konumlandırma doğruluğunu önemli ölçüde geliştirdiği gözlenmiştir. Tezin son katkısı, KST görsel takip kayıp fonksiyonu üzerine odaklanan öznitelik öğrenme problemini ele alır. Bu kayıp fonksiyonu için, tamamen evrişimsel derin sinir ağını eğitmek için yeni bir geri yayılım algoritması geliştirilmiştir. Geri yayılım için gerekli olan gradyan hesaplaması, frekans ve görüntü uzaylarında etkin bir şekilde gerçekleştirilir ve öznitelik haritalarının sayısı ile doğrusal bir karmaşıklığa sahiptir. Ağ modelinin eğitimi, görsel izlemenin iyi bilinen zorluklarını (örneğin kapanma, nesne deformasyonu ve hızlı hareket) da dâhil ederek hazırlanmış veri kümeleri üzerinde gerçekleştirilir. Öğrenilen öznitelikler, en gelişmiş KST görsel takipçilere entegre edildiğinde - manuel olarak tasarlanmış öznitelikleri veya önceden eğitim görmüş sınıflandırma modellerinden çıkarılan derin öznitelikleri kullanan KST yöntemlerine kıyasla - karşılaştırma veri kümelerinde olumlu takip performansı sağlamıştır.

Anahtar Kelimeler: görsel nesne takibi, uzmanların karışımı, korelasyon süzgeci, evrişimsel sinir ağları

to my wife

ACKNOWLEDGMENTS

I would like to express my gratitude and deep appreciation to my supervisor Prof. Dr. A. Aydın Alatan, with whom I have spent almost one third of my life, for his guidance, positive suggestions and also for the great research environment he had provided. I would like to thank him for believing in my ideas during this dissertation process.

I thank the members of my thesis progress committee, Assist. Prof. Dr. Elif Vural and Assist. Prof. Dr. Nazlı İkizler Cinbiş, for their positive attitudes and encouraging comments which lead me to finalize my PhD studies. Moreover, I would like to thank the jury members of the thesis defense, Prof. Dr. A. Murat Tekalp and Assoc. Prof. Dr. Fatih Kamışlı, for their useful and constructive comments.

I also thank all of my friends who always encourage me in my progress, especially Yeti Ziya Gürbüz, Veysel Yücesoy, Sami Özarık and Mert Yakut.

I thank ASELSAN Research Center and all of my colleagues who were always indulgent, especially my manager, Dr. Aykut Koç.

Without the support of my family, I would not become so eager to proceed my PhD studies. I would like to thank my mother, Semra, my father Hasan and my sister, Dr. Elçin Gündoğdu Aktürk, who shares my concerns, worries and happiness.

Finally, I thank my wife and my best friend, Ceren, for her infinite support, understanding and patience that make me succeed in writing this thesis.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xvi

LIST OF FIGURES xviii

CHAPTERS

1 INTRODUCTION 1

1.1 The General Flow of Visual Tracking 2

1.2 The Challenges of the Visual Object Tracking 4

1.3 Benchmark Datasets 6

1.4 Short Summary of the State of the Art 7

1.5 Problem Statement and Contributions of the Thesis 8

1.6 Outline of the Thesis 10

2 LITERATURE SURVEY ON VISUAL OBJECT TRACKING 13

2.1 Discriminative Methods 16

2.2 Generative Tracking Methods 20

2.3 Hybrid Methods 23

2.3.1 Correlation Filter Based Visual Tracking Methods . 24

2.3.1.1 Fundamental Correlation Filters 25

2.3.1.2 Discriminative Correlation Filters . . . 26

2.3.1.3 Correlation Filters with Multiple Channels 27

2.3.1.4 Scale Adaptability 28

2.3.1.5 Other Extensions of Correlation Filters 28

2.3.1.6 Learned Representations for Correlation Filters 30

2.3.2 Methods based on neural networks 30

2.4 Ensemble of Trackers 31

3 CORRELATION FILTER BASED TRACKING 33

3.1 Single Channel Correlation Filters for Visual Tracking 33

3.1.1 Correlation Filter Computation 34

3.1.2 Object Localization 35

3.1.3 Model Update 36

3.2 Discriminative Scale Space Tracker (DSST) 36

3.3 Continuous Convolution Operators for Visual Tracking . . . 39

4 LEARNING SPATIAL WINDOWS FOR CORRELATION FILTER BASED TRACKING 43

4.1 Spatial Windowing Applied to the Object Observation 46

4.2 Object Localization Algorithm for Windowing on the Object Observation 48

4.3 Spatial Windowing Applied to the Filter 49

4.4 Object Localization Algorithm for Windowing on the Filter . 51

4.5 Implementation Details 51

4.6 Computational Complexity of the Window Learning 52

4.7 Experimental Analysis on the Learned Spatial Windows . . . 52

4.8 Discussions 55

5 TREE-STRUCTURED ENSEMBLE OF TRACKERS 59

5.1 Proposed Tree-structured Ensemble Tracking Framework . . 60

5.2 Combination of the Expert Tracker Outputs 63

5.2.1 Tree Balancing 66

5.2.2 Quality Metric of Expert Trackers 67

5.2.3 Calculation of the Weights w_k's 67

5.3 Performance Evaluation of the Proposed Spatial Windowing and Ensemble of Trackers 68

5.3.1 Implementation Details and Computational Complexity 68

5.3.2 Experimental Results of the Proposed Ensemble Tracker on VOT2014 Dataset 69

5.3.2.1 Quantitative Results 69

5.3.2.2 Qualitative Results 76

5.3.3 Experimental Results of the Proposed Ensemble Tracker and Spatial Windowing Based Tracker on VOT2015 Dataset 78

5.3.3.1 Performance Evaluation 79

5.3.3.2 Evaluation with respect to the baseline 79

5.3.3.3 Comparisons to the-state-of-the-art trackers 83

5.3.4 Performance Evaluation on OTB-2015 Dataset . . 84

6 LEARNING DEEP CONVOLUTION FEATURES FOR CORRELATION FILTERS 87

6.1 Convolutional Layers 88

6.2 Convolutional Features for Correlation Filters 89

6.3 Backpropagation of the Loss for a Single Feature Map 90

6.4 Backpropagation of the Loss for Multiple Feature Maps . . . 95

6.5 Dataset Generation from VOT2015 Dataset 97

6.6 Proposed Custom architecture and Implementation Details . . 98

6.7 Tracking Application and Experimental Results on Benchmark Datasets 100

6.7.1 Experimental Results on OTB-2013 Benchmark in Comparison to the Hand-Crafted Features 105

6.7.2 Comparison with Respect to the State-of-the-Art Deep Features 109

6.7.3 Fine-tuning the Pre-trained Networks 111

6.7.4 VOT2016 Performance Analysis of the Fine-tuned VGG-M 120

7 CONCLUSION AND FUTURE WORK 123

7.1 Summary of the Thesis 123

7.2 Conclusions 125

7.3 Future Work 127

REFERENCES 129

APPENDICES

A THE EFFECT OF A LAYER ON THE CORRELATION QUALITY OF THE PREVIOUS LAYER 141

CURRICULUM VITAE 145

LIST OF TABLES

TABLES

Table 1.1 The VOT challenge winners and their categories. Disc. and Gen. means discriminative track-by-classification and generative methods, respectively. Comb. represents the combination of trackers. CFB stands for correlation filter based trackers. These categories and the remaining ones are explained in Chapter 2. 8

Table 5.1 Average accuracy results per sequence. *, ** and *** indicate first, second and the third rankings of the methods. Red means the proposed method has an accuracy better than or equal to the baseline tracker DSST. . 70

Table 5.2 Comparison of our proposed weighting strategy and naive weighting. Gray cells indicate better performance. 73

Table 5.3 Average accuracy ranking results calculated using per attribute. Red, blue and green indicate the best, second and third ranking. 75

Table 5.4 Average accuracy of the proposed and compared correlation filter based methods over successfully tracked frames. 76

Table 5.5 VOT2015 [35] per sequence experimental results for our proposed tracker configurations and DSST [26]. Acc., Rob. and Rank. mean accuracy, robustness and ranking, respectively. The definitions of these metrics are included in Section 5.3.2.1. 80

Table 5.6 VOT2015 [35] per attribute experimental results for our proposed tracker configurations and DSST [26]. 80

Table 5.7 VOT2015 [35] accuracy results for our proposed tracker configurations and the compared trackers. Red, blue and green indicate first, second and third rankings, respectively. 81

Table 5.8 VOT2015 [35] robustness results for our proposed tracker configurations and the compared trackers. Red, blue and green indicate first, second and third rankings, respectively. 82

Table 5.9 OTB-2015 results for different configurations of the proposed method. Red, blue and green indicate 1^{st}, 2^{nd} and 3^{rd} rankings, respectively. 85

Table 6.1 Mean distance precision and mean overlap precision comparison for 40 sequences from OTB [111] for different features. 109

Table 6.2 AUC values for 11 attributes of the 40 test sequences [111]. FM means feature maps. 110

Table 6.3 VOT2016 performance results 121

LIST OF FIGURES

FIGURES

Figure 1.1 The general flow of a tracking loop. 3

Figure 2.1 A family of visual tracking taxonomy. 14

Figure 2.2 A detailed categorization of visual appearance models in visual tracking. 15

Figure 2.3 A representative illustration of a discriminative visual tracking framework. Any of the building blocks of this framework might change as well as their order according to the algorithm design. 16

Figure 2.4 A representative illustration of a generative visual tracking framework, which is drawn by getting an inspiration from the framework of L1APG [5] tracker. Many other designs might differentiate from the depicted figure. However, the general indication of a presence of the object is the closeness of the reconstruction error to zero. 21

Figure 2.5 A representative illustration of a correlation filter based visual tracking framework. The aim is to find a correlation filter such that the resulting correlation response of this filter and the unlocalized object observation will have a peak around the true center of the object. 25

Figure 2.6 Illustrating the effectiveness of using a desired correlation response. 26

Figure 4.1 *Ball sequence,* frame $\#\,87$. **Top-left:** The object patch. **Top-right:** Correlation result using the conventional Hanning window. **Bottom-left** The estimated window by the proposed method. **Bottom-right:** Correlation result after the proposed method. (Dark values indicate low values for window function and the correlation results.) The correlation score is much sharper and located at a better position (*i.e.* non-translation) 45

Figure 4.2 Comparison of Peak to Sidelobe Ratio and localization error (Euclidean dist. btw. ground truth and the predicted locations) between the proposed and Hanning windowing. (*Skating* sequence) 53

Figure 4.3 Visual examples from *Tiger* sequence. The first and the third rows present the observations, and the second and the fourth rows illustrate the calculated spatial windows. 54

Figure 4.4 Visual examples from the sequences *Face Occluded 1, Car, Jogging, Torus*). 1^{st} **row**: spatial windows extracted by the proposed method. 2^{nd} **row**: corresponding objects. 55

Figure 5.1 Visualization of the complete (no-pruning) tree with depth 2. At each frame; an object appearance x_t is processed through the tree by applying the separator function η_λ^S at each visited node S. Each row of the image matrix shows the object patches dropped to the leaves of the tree ($\{R_1, R_2, R_3, R_4\}$). Note that object poses with similar lightening conditions and appearances are clustered at the same nodes as desired. (*Skating* sequence from VOT2014 dataset [59]). 61

Figure 5.2 All possible partitions $\{P_1, ..., P_5\}$ are illustrated for a depth-2 tree in a 2-dimensional space. The union of the blue regions is the complete space. 63

Figure 5.3 Example frames showing the tracking of the correlation filter based algorithms. Red: Ground truth, White: DSST [26], Black: SAMF [69], Blue: KCF [48], Green: Proposed Method. The name of each sequence is on the bottom left corner of the example patches. 72

Figure 5.4 Average success rate plot of the proposed and the compared correlation filter based method. **X-axis** shows the accuracy threshold and **Y-axis** indicates the ratio of # of successfully tracked frames over the total # of frames. 77

Figure 5.5 For green circles, **y-axis**: the average accuracy difference between the proposed method and the baseline tracker and **x-axis**: the length of the sequences on VOT2014 dataset. Blue and red markers indicate the performance of the baseline tracker and the proposed method respectively. Horizontal red line separates the sequences as below or above 50% of average accuracy. 78

Figure 5.6 Accuracy-robustness ranking plot for the-state-of-the-art comparison. 84

Figure 5.7 Success curves of the proposed method with the tree depth 4 and the compared trackers for OTB-2015 dataset. 86

Figure 6.1 Block diagram of the proposed architecture. $\circledast$ indicates the circular correlation. 91

Figure 6.2 Architecture dimensions. 98

Figure 6.3 Visual illustrations from our trained network for single feature map. MSE scores of the correlation filter of the feature map extracted by our network after the first epoch, 400^{th} epoch and the correlation filter of the gray-level image intensities. 99

Figure 6.4 Cost function decrease during the training. 100

Figure 6.5 Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features. 101

Figure 6.6 Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features. 102

Figure 6.7 Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features. 103

Figure 6.8 Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features. 104

Figure 6.9 Overall results on 40 sequences from [111]. MCFCF_IS is MCFCF with an intelligent search algorithm that performs an object search in a region with a size 4 times larger than the object size, when the tracking quality (Peak-to-Sidelobe-Ratio) is below 10. MCFCF_SW is MCFCF with windowing algorithm of Chapter 4. 106

Figure 6.10 Success plots of 40 sequences from [111] for 6 attributes. 107

Figure 6.11 Precision plots of 40 sequences from [111] for 6 attributes. 108

Figure 6.12 One-Pass-Evaluation (OPE) curves. **Left:** Overlap precision (OP) and **right**: center localization error (CLE). Avg. OPs are ordered according to Area-Under-Curve (AUC) while avg. CLEs are in the order of 20 pixel threshold. 110

Figure 6.13 Average precision and success plots of 51 sequences in [111]. . . . 112

Figure 6.14 Precision and success plots of 51 sequences in [111] for 4 attributes: fast motion, background clutter, motion blur and deformation. . . 113

Figure 6.15 Precision and success plots of 51 sequences in [111] for 4 attributes: illumination variation, in-plane rotation, low resolution and occlusion. 114

Figure 6.16 Precision and success plots of 51 sequences in [111] for 3 attributes: out-of-plane rotation, out-of-view and scale variation. 115

Figure 6.17 Average precision and success plots of 100 sequences in [112]. . . 116

Figure 6.18 Precision and success plots of 100 sequences in [112] for 4 attributes: fast motion, background clutter, motion blur and deformation. . . 117

Figure 6.19 Precision and success plots of 100 sequences in [112] for 4 attributes: illumination variation, in-plane rotation, low resolution and occlusion. 118

Figure 6.20 Precision and success plots of 100 sequences in [112] for 3 attributes: out-of-plane rotation, out-of-view and scale variation. 119

Figure 6.21 Speed comparison between CCOT_VGG (baseline) and the proposed tracker CCOT_CFCF on OTB-2015 sequences. 120

Figure 6.22 Accuracy-Robustness Ranking plot. Closeness to the top right indicates good tracking performance. 122

CHAPTER 1

INTRODUCTION

Modern visual surveillance and weapon-oriented defense systems require automated visual analysis and such systems aim to interpret the outputs of electro-optic imaging sensors. There exist numerous sub-tasks of these systems, such as motion-based recognition, anomaly action detection, human-computer interaction, traffic monitoring, vehicle navigation and high-level video representation [117]. Most of these tasks should typically be fed by the output which is mostly provided by a visual tracking subsystem due to numerous reasons; for instance, the trajectory information extracted from the tracked objects can be exploited to analyze the state of a junction in a traffic monitoring system. Furthermore, individual motions of the human body parts might help to identify the type of the performed action. Activity recognition methods may utilize the trajectories of the subjects involved in the activity. Hence, visual tracking modules are inevitable parts of such systems.

Due to the requirements of the aforementioned systems, visual object tracking is a popular problem in computer vision and it has been studied diligently. The problem of visual object tracking mainly aims to estimate the state of a target object throughout the frames of a video sequence by the help of the visual appearance model of the object. In this thesis, the state of the object carries the information about the center location and the size of the object patch in a video frame, and the visual appearance model is obtained by utilizing a cost function that considers the available training image patches of the target object as well as their locations. The previous state of the object is assumed to be available, estimated by the visual tracking method itself or provided by an oracle or a previous task, such as automatic object detection. The state estimation is fulfilled either in an offline or online manner. Moreover, the properties of the visual systems, such as the number of tracked objects and the type of the object state, play a critical role in the problem definition. For instance, the requirement of a multiple object tracking task significantly differs from the single object visual tracking scenario. This thesis considers the latter one and addresses the main problems of visual tracking for only a *single object* in such scenes.

During the last two decades, visual tracking of objects have become popular and differentiated from the radar target tracking concept [6, 79, 44], which utilizes the observations (*i.e.*, raw measurements) provided by the radar or any other appropriate sensor for the target environment. However, in visual tracking, there is no information related to the object observation except for the raw pixels of the video frames. Hence, the major difficulty of visual tracking is obtaining robust and relevant candidates for the state of the object by only utilizing the available visual information.

In the literature, the state-of-the-art computer vision and machine learning tools are both employed to obtain good candidates for the object state. The mostly preferred input for these tools is obviously the image region or the region of interest for the target object. Concretely, the pixel values of the rectangular image patches surrounding or containing the objects are treated as the raw patterns. Then, computer vision algorithms try to represent these patterns in a high-dimensional feature space, and machine learning algorithms can select relevant patterns to obtain the "best" candidate for the object state during the course of tracking. In this thesis, single object tracking problem is the major consideration; in the following section, the generic flow of a single object tracking framework is explained.

1.1 The General Flow of Visual Tracking

The generic flow of a tracking methodology is demonstrated in Figure 1.1 with a representative loop repeating over the frames of a video sequence. As described above, the initial state is utilized to extract a representation (*i.e.*, initial visual model) of the object. Following this, an appearance model or a classifier is trained. In Figure 1.1, it is represented by the model $f_\theta(.)$ with model parameters θ. When the next frame arrives, a set of candidate states are evaluated around the region of interest and the best candidate is claimed to be the state of the object. In its simplest form, the Cartesian coordinates of the object is a frequently utilized state vector. Once the object is localized, an instantaneous object representation is extracted to update the model, if necessary. The green part in the figure illustrates this loop, which will continuously repeat during the time steps. Although the structure of the flow diagram is representative and might vary according to the complexity of a particular algorithm, most of such techniques can be reduced to this flow.

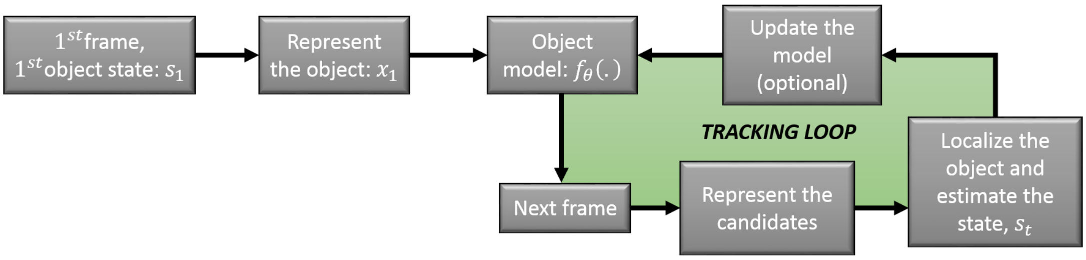

1^{st}frame, 1^{st}object state: s_1
Represent the object: x_1
Object model: $f_\theta(.)$
Update the model (optional)
TRACKING LOOP
Localize the object and estimate the state, s_t
Next frame
Represent the candidates

Figure 1.1: The general flow of a tracking loop.

It is possible to analyze in detail any block of the framework in Figure 1.1 and the overall algorithm could be improved by proposing novel object initializations, robust object representations, smart policies for model update and so forth. Among all these enhancements, object representation is an anchor point to discuss, since the performances of the tracking algorithms are significantly affected by the quality of this representation. The desired property of the object representation is its invariance to appearance changes of the objects due to various challenges, *e.g.*, illumination change, deformation etc. Moreover, this representation should disentangle subtle differences between the target object and the background. Numerous efforts have been devoted for feature selection process to serve for improving the representation quality. Recently, the advances in machine learning research have evolved to exploit deep networks for learning to recognize [60], detect [74] and segment [94] objects. The main reason of this evolution is due to the fact that deep learning techniques provide hierarchical and task or domain specific representation learning either in a supervised or unsupervised manner; hence, closing the gap between representation design and the object model.

The other important component of visual tracking is the state estimation stage, which can also be denoted as *localization*. Discriminative and generative models are adopted in visual tracking to differentiate the object from the background. On the other hand, deep learning based approaches are also proposed for tracking problem [9, 65, 84]. The important nuance between the utilization of these two models for visual object tracking in video sequences (and other computer vision tasks related to still images) is as follows: An online version is required in the discriminative techniques; in other words, an online extension of the adopted method is necessary due to the appearance changes of the object, although the model might also be trained offline in the beginning before tracking starts. This necessity arises from the fact that the designed representation model is not invariant to all kinds of appearance changes of the object. For instance, a classifier is trained in discriminative tracking methods; as the time goes on, the classifier should be updated with the newly observed object appearances to become robust to any challenge. Thus, online classification methods should be developed to update the object models. For example, for visual tracking, online structured-output Support Vector Machines (SVM) are proposed in [45], whereas online AdaBoost is studied in [3]. More recent methods and their detailed taxonomies are discussed in Chapter 2 .

1.2 The Challenges of the Visual Object Tracking

It is notable that many challenges should be considered while designing a visual tracking algorithm. *Appearance change*, which is mentioned above, is one of the major challenges. It can be defined as any difference of the object appearance between a test frame and a reference frame. Object deformation and scale variation are examples of this challenge. The desirable property of a tracking method is maintaining the object tracking accurately in such conditions. If the object is moving towards the camera, it is quite likely that the search range will not be adequate to represent the object, whereas the visual information is significantly lost as the object moves away from the camera, and it deteriorates the object representation quality. Thus, the tracker should be able to operate in multiple scales in order to estimate the size of the object as well.

In some tracking algorithms that make use of hand-crafted features, object is assumed to be rigid. Yet, many objects vary their form in such a way that the object parts are displaced due to the motion type. This scenario can be named as object deformation, which is indeed an example of appearance change, yet requires a special attention. For instance, the fingers of a hand change their relative location. Hence, a part-based model can be employed or features that are invariant to particular deformations could be utilized.

Another crucial problem of visual tracking is the *abrupt motion* of the object or the camera, resulting in a tracking failure, if the search region of the object in the next frame is limited. If the localization is so erroneous that the update of the model is performed with the non-localized object examples, an undesirable *drift* from the true location occurs. If this drift insists, then the failure of the tracker becomes inevitable. For this reason, model update mechanisms should be carefully designed to prevent such a drift. A potential reason of this undesirable update also occurs in the case of full or partial *occlusions* of the object. If the model is updated with an occluded object, the model could localize the object at the undesired region of the occlusion even after the object moves away from there. Following this challenge, the search range and the update rate should be carefully designed. The size of the search range is a compromise, which suffers from the large search range due to the jump of the tracker to a similar object, whereas it should not be small to handle fast motion of the object relative to the image coordinates.

Motion blur is another critical factor in visual tracking, since a non-stationary camera could output blurred frames due to the fast motion of the object or the focusing problems of the camera in some time intervals of the tracking. In these cases, the appearance of the object might vary quite significantly such that the localization performance degrades. Similar to the motion blur problem, the object can be exposed to *illumination change* due to the hardware limitations or the nature of the environment. In most cases, the illumination condition changes suddenly, *e.g.*, the object could enter a shaded area. Thus, the utilized features should be designed to handle illumination changes.

The challenges discussed above are the major issues of the state-of-the-art tracking algorithms that are evaluated in the state-of-the-art benchmarks. Although there might be some other critical issues to be addressed in some particular tracking scenarios, the main aim of most of the published visual tracking research efforts is to enhance the capabilities of the trackers in terms of handling the aforementioned problems. To this end, these challenges have been addressed in this thesis by proposing three major contributions: (1) improving the robustness of a correlation filter-based tracking algorithm against occlusions and deformations, (2) proposing an ensemble of trackers method to mitigate the tracking failures resulting from appearance changes of the target objects, and (3) learning features that are robust to various appearances of the same object and designed for ‘good’ correlation operation.

1.3 Benchmark Datasets

Every year, hundreds of visual tracking studies are reported and published in various organizations, including conferences, journals and symposiums. About a decade ago, the proposed studies were assessed in terms of the manually designed performance metrics in the videos recorded by the authors of the study or in a subset of the publicly available videos. Hence, it was not easy to fairly compare the methods and to show the validity of a tracking approach.

In order to build a fair comparison basis and avoid subjective evaluations, visual tracking benchmarks, such as OTB-2013 [111], OTB-2015 [112], VOT2013 [58], VOT2014 [59], VOT2015 [35], VOT2016 [36], have been introduced. OTB-2013 and OTB-2015 are the most frequently employed datasets to assess the performance of a visual tracking method. These datasets include video sequences with the aforementioned challenges in Section 1.2. Moreover, Visual Object Tracking (VOT) Challenge Workshops [58, 59, 35, 36] have also been organized since 2013 to evaluate the novel tracking methodologies. These efforts make it possible to compare and contrast the tracking algorithms in terms of the performance metrics peculiar to the individual datasets. For instance, VOT challenge [59] proposes to measure the performance of the trackers from many different aspects that are complementary to each other. *Accuracy* is the fundamental measure that deals with how accurately the trackers estimate the state of the object. On the other hand, *robustness* measures the number of failures which is defined as the case, when the accuracy of the tracker is below a fixed threshold. In order to gather a statistical meaning from these datasets, VOT challenges report the average ranking of the competing trackers in terms of these two metrics by statistical significance analysis.

In this thesis, in order to assess the tracking performance of the proposed methods, OTB-2013 [111], OTB-2015 [112], VOT2015 [35] and VOT2016 [36] datasets are employed as the most recent state-of-the-art studies have been evaluated in those datasets. The detailed information about the performance evaluation criteria is presented in the experimental sections of the proposed methods in the manuscript.

1.4 Short Summary of the State of the Art

Visual tracking has long been studied by many researchers. Eventually, various methods are proposed with a variety of approaches from different disciplines. The type of the approach within a particular era is significantly determined by the popular trend of computer vision and machine learning during that time interval. Today, deep learning is by far the most popular concept in the computer vision and machine learning communities. Thus, top performing trackers are expected to be based on deep neural networks. Although such an expectation is acceptable to a limited extend, the contribution of the correlation filter based trackers are too good to be underestimated for the success of the state-of-the-art trackers. Notably, methods that employ a correlation filter based approach along with the features extracted from a deep convolutional neural network perform favorably against their counterparts in the benchmark datasets [76, 27, 30].

In Table 1.1, top three winners of VOT challenges are demonstrated from 2013 to 2016. As it is pointed out, deep learning based discriminative methods, such as MD-Net [84] and TCNN [83], are effective for 2016 challenge. Nevertheless, correlation filter based methods have a dominant effect for the top trackers over the last three years. For instance, VOT2014 winners belong to this family of trackers. Moreover, the 2016 winner is also a correlation filter based approach. Although the top performing methods might not be the unique indicator for the success of tracking research direction, the computation speeds of correlation filter based methods are favorable against the discriminative track-by-classification trackers due to the efficient dense

Table1.1: The VOT challenge winners and their categories. Disc. and Gen. means discriminative track-by-classification and generative methods, respectively. Comb. represents the combination of trackers. CFB stands for correlation filter based trackers. These categories and the remaining ones are explained in Chapter 2.

Ranking	VOT2013 [58]	VOT2014 [59]	VOT2015 [35]	V0T2016 [36]
1^{st}	PLT [58] (Disc.)	DSST [26] (CFB)	MDNet [84] (Disc.)	CCOT [30] (CFB)
2^{nd}	FoT [107] (Comb.)	SAMF [69] (CFB)	DeepSRDCF [27] (CFB)	T-CNN (Disc. + Comb.) [83]
3^{rd}	EDFT [39] (Gen.)	KCF [48] (CFB)	EBT [129] (Disc.)	SSAT [36] (Disc.)

matching advantage in the discrete frequency domain. Moreover, the correlation concept is much more simpler than many others due to the simple update equations, efficient formulations in the frequency domain and visual interpretation opportunities. Hence, the major concentration of the thesis is devoted to the correlation filter based trackers. On the other hand, influence of deep neural networks are also exploited in the proposed feature learning framework.

1.5 Problem Statement and Contributions of the Thesis

In this thesis, single object visual tracking problem is embraced. The major concern of this problem is estimating the state of a target object throughout the frames of a video sequence, when the first state information is provided by an oracle.

In the utilized framework, the state of the object is defined as the four-dimensional vector containing the horizontal and vertical Cartesian coordinates of the object as well as its width and height, since the benchmark datasets and state-of-the-art trackers favor this type of state. Moreover, the problem includes tracking of arbitrary objects without any constraints, such as rigid body assumption or color constancy. The designed trackers should be able to track a single object without the help of any secondary object or environment as opposed to the multiple object tracking problem.

Specifically, the dealt family of trackers in this study is correlation filter based visual tracking methods, since correlation filters have shown spectacular performances in the discussed benchmark datasets. Thus, it is aimed to increase the accuracy of the visual tracking performance by proposing novel methods designed for correlation filter based trackers.

In order to improve the state-of-the-art correlation filter based trackers, three major contributions have been achieved:

1) A novel window learning methodology is proposed to enhance the quality of the correlation and to increase the localization accuracy. For this purpose, a window that is to be element-wise multiplied by the object observation (or the correlation filter) is learned by a gradient descent procedure which has an efficient formulation. The resulting tracking algorithm is able to suppress or highlight the (ir)relevant regions for the correlation operation in the image domain.

2) An ensemble of trackers algorithm is proposed to handle the appearance changes of the object by a tree-structured ensemble tracker. The presented tree stores different models at the nodes for a variety of appearance types. The designed framework is capable of providing the optimum combination decision, if the correlation quality (or the tracking quality) is assumed to be the ground truth quality of the decision made by the individual nodes of the tree. Moreover, extensive experiments in benchmark datasets with different performance metrics validate the effectiveness of these two proposed methods. When the proposed window learning algorithm is adopted in the tree-structured ensemble method, a further increase in the tracking performance is obtained. Thus, these approaches carry complementary and beneficial properties for visual tracking task.

3) The final contribution of this thesis is a feature learning framework to obtain robust features for correlation task. In other words, a deep and fully convolutional neural network model is learned with respect to the proposed loss function. Moreover, an efficient backpropagation procedure is formulated in the frequency domain to handle the nonlinear relationships between the correlation filter and the object observations. The learned features by the presented model is shown to outperform the hand-crafted features as well as the pre-trained networks that are obtained by the help of object recognition loss functions.

1.6 Outline of the Thesis

In Chapter 2, state-of-the-art tracking approaches are discussed in detail by a novel taxonomy which is appropriate for the recently proposed tracking methods when compared to the previous visual tracking surveys, such as [117] and [97]. The key difference between the presented taxonomy and the previous ones is the exhaustive categorization of correlation filter based tracking methods while explaining the remaining family of trackers as recent and comprehensive as the past surveys.

In Chapter 3, the utilized correlation filter based tracking approaches are summarized. The main focus of this thesis is to provide proposals specifically customized for correlation filter based trackers. Thus, the single channel linear correlation filter formulation of [14] is explained. Then, the method in [26], the multi-channel extension of [14] with a multi-scale search support, is described, since this tracking approach, which is also the winner of VOT2014 challenge [59], is employed as the baseline in the development of the proposed window learning approach and the tree-structured ensemble method. In order to obtain top tracking success, the winner of VOT2016 challenge [36] is exploited in the proposed feature learning methodology. Hence, this approach is expounded in the subsequent section of Chapter 3.

Proposed spatial window learning methodology is formulated in Chapter 4 with two different algorithm options. The first one selects to learn a spatial window for the object observation while the second one opts to learn the window for the correlation filter. Both of the windows are learned to multiply the particular signal in the spatial domain element-wise. The experimental results with deep analysis are provided in the following chapter.

In Chapter 5, the tree-structured ensemble approach is introduced with extensive experiments in the final sections. Moreover, the experimental results section covers the performance evaluation of the spatial windowing method. Remarkably, the integration of the spatial windowing approach to the ensemble tracker is also tested and reported in this section.

The feature learning framework for correlation filters is explained in detail in Chapter 6. Firstly, the preliminaries to clarify the utilized signal processing and computer vision tools are presented. Then, the single channel feature learning methodology is proposed. Finally, the multiple channel extension is formulated for the efficient backpropagation formulation. In the last section, the experimental results are reported for the evaluated benchmarks as it is fulfilled in the previous chapters for the spatial windowing and ensemble tracking proposals.

In Chapter 7, an overall qualitative assessment of the thesis is provided with a brief summary of the thesis. Moreover, the conclusive and remarkable points are emphasized. Since the goal of the thesis is to take place among the state-of-the-art visual tracking techniques, the proposed approaches contain particular novelties that should be explored and exploited further more. Hence, promising feature directions are discussed.

CHAPTER 2

LITERATURE SURVEY ON VISUAL OBJECT TRACKING

In this chapter, state-of-the-art visual tracking methods are discussed in detail. Although this problem has been the subject of numerous studies for single or multiple targets, the aim of this research is limited to single object visual tracking, since this study targets at designing algorithms to mitigate the weaknesses of existing methods for tracking a single object. Due to the variety of the requirements of the systems, visual tracking algorithms could specialize for a specific scenario, such as traffic systems or an electro-optic imaging system with an intelligent gimbal platform led by the tracking algorithm. The variety of systems and developed tracking methods cause the researchers to classify them based on different aspects [117, 111]. In [117], Yilmaz *et.al.* classify the trackers into the categories based on different anchor tools, such as point detectors, segmentation, background modeling, supervised classifiers. On the other hand, evaluation studies [97] and tracking benchmarks [111] attempt to categorize the methods regarding some other aspects including the use of template matching and/or classifiers (which are also called as track-by-classification). Appearance models are investigated and analyzed in the context of single object tracking problem in [68], while [19] only deals with a specific family of trackers, *i.e.*, correlation filter based trackers, which is our main concentration in this study.

As it can be deduced from the past surveys of visual object tracking, it is possible to categorize visual tracking methods according to more than one aspect. A family of four categorizations has been demonstrated in Figure 2.1, where the trackers are categorized in various ways. Although the focused problem of the thesis is single object tracking, the tracking problem is divided into two as: (1) single target and (2)

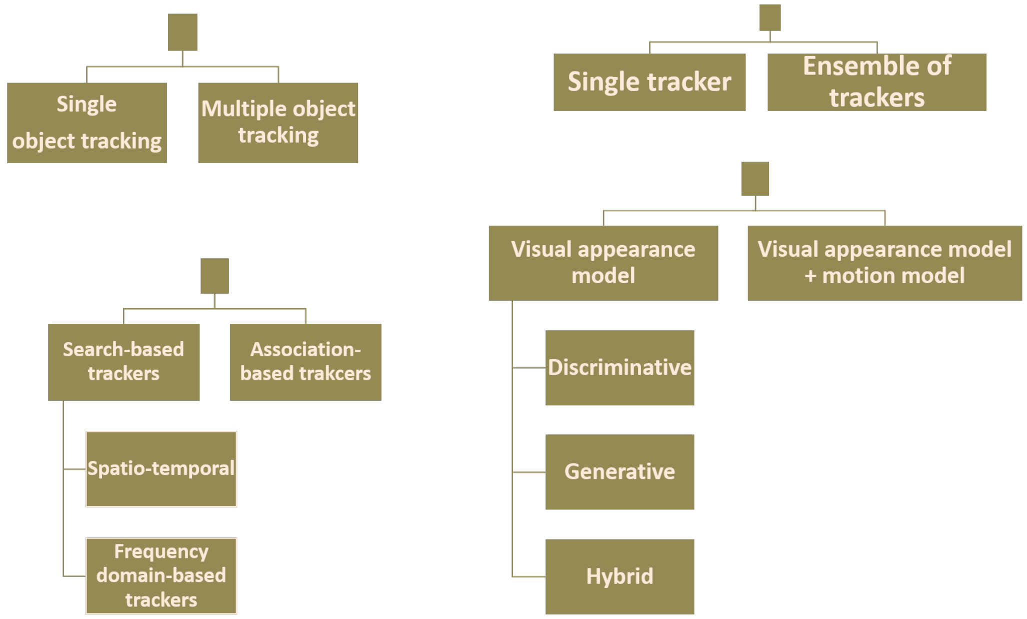


Figure 2.1: A family of visual tracking taxonomy.

multiple target tracking. Moreover, some tracking methods combine the results of multiple trackers while most methods prefer to rely on a single visual tracker. It is also possible to divide the tracking approaches into two classes as: (1) search-based trackers and (2) association-based trackers. Concretely, the search-based trackers are based on a search algorithm that aims at finding the object within the current frame by using template matching, optical flow or correlation filter based approaches. On the other hand, association based trackers generally gather candidate object patches and associate them with the visual appearance model of the object. Finally, it is also possible to classify a visual tracker according to the usage of a motion model. For instance, most of the correlation filter based methods do not exploit any motion model. However, a considerable amount of discriminative trackers in the literature utilize a motion model, such as particle filtering.

Visual object tracking is a rapidly developing research topic. For instance, the best performing tracker of two years ago does not even rank among the current top-thirty trackers. Since visual appearance modeling is an important aspect of visual tracking methods and one of the key factors to increase the tracking accuracy, in this thesis, the methods in the literature are categorized according to the way they design visual appearance model. For this purpose, the single object tracking methods are divided

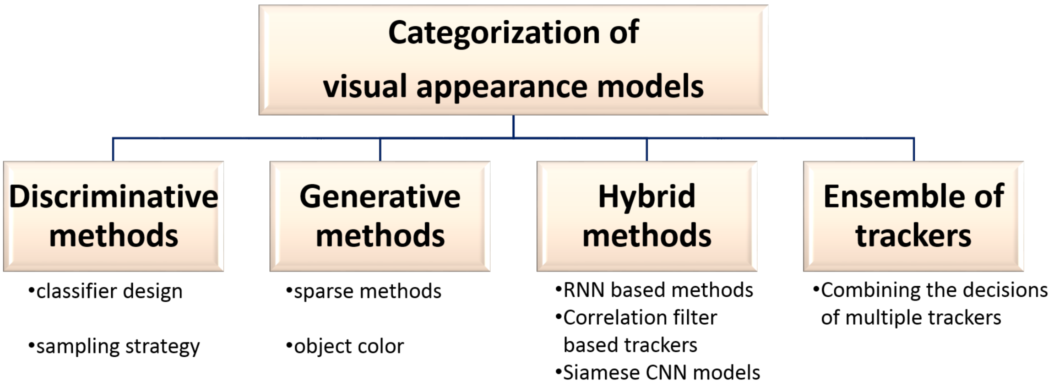


Figure 2.2: A detailed categorization of visual appearance models in visual tracking.

into four categories: (1) Discriminative Methods, (2) Generative Methods, (3) Hybrid Methods and (4) Ensemble of Visual Trackers. This categorization is obtained as a result of the recent advances in the tracking literature, where correlation filter based methods perform favorably against its contemporaries, and it is demonstrated in Figure 2.2. Moreover, there exist custom algorithms specifically designed for tracking, such as the ones based on Recurrent Neural Networks or Siamese Networks. Complementary advantages of the tracking methods have also been studied and multiple trackers are concurrently run and merged in particular studies.

As it has already been defined in Chapter 1, visual tracking of objects is described as estimating the position of the object in the consecutive frames of a video, when the state of the object is predefined in the first frame. For the sake of efficiency and simplicity on representing the objects from a particular video frame, its state is mostly defined by a rectangular region of interest, although there exist a few methods which utilize more fine-grained state information, such as the pixels or super pixels of the target object. Once the state information is provided to the visual tracking algorithm, a tracker is responsible for tracking the object (*i.e.*, estimating the object state) throughout the video frames without obtaining further information. The methods differentiate from each other in terms of the way they distinguish the target object from the background. In the following, the tracking methodologies are discussed according to the way their object appearance being discriminated from the background.

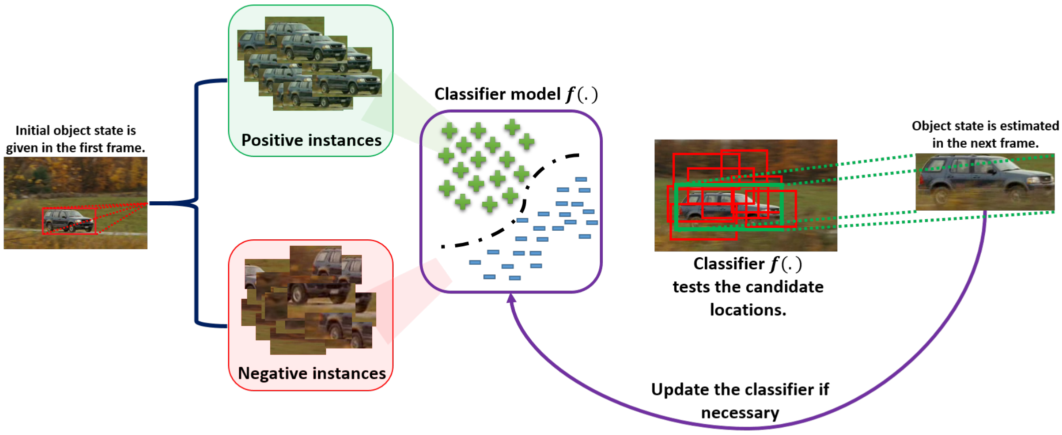

Initial object state is given in the first frame.
Positive instances
Negative instances
Classifier model $f(.)$
Classifier $f(.)$ tests the candidate locations.
Object state is estimated in the next frame.
Update the classifier if necessary

Figure 2.3: A representative illustration of a discriminative visual tracking framework. Any of the building blocks of this framework might change as well as their order according to the algorithm design.

2.1 Discriminative Methods

Discriminative methods aim to estimate the conditional probability density of a candidate sample to be the object given the representation of its appearance. In order to achieve this, they generally utilize a classifier model, which is responsible for the classification of a visual sample as either the object or background. The confidence of the classifier for a particular instance is assumed to be the conditional probability of the described discrimination above. In order to fulfill the classification task, an appearance description is required. Concretely, the object of interest is represented in the high dimensional feature space. Features are extracted from the object patches. Histogram of Oriented Gradients (HOG) [24] and Scale Invariant Feature Transform (SIFT) [75] are typical examples from the hand-crafted feature types. Model training is performed by collecting positive, hard negative and negative samples (extracted features) from the region of interest that is provided at the beginning of the tracking by the oracle. The object localization is generally performed by looking for the candidate location with the highest objectness score. Figure 2.3 presents a generic discriminative track-by-classification algorithm.

Pioneering studies are Online AdaBoost Tracker [43] and ensemble tracking [2]. In [2], Avidan proposes an ensemble method consisting of weak classifiers to label each pixel in the region of interest as object or background. In that work, the object position is determined by the mean shift algorithm, and the K best weak classifiers are selected out of T and updated throughout the frames while the remaining weak classifiers are trained in the next frame. Such a methodology provides an online update process which is explicitly required by any tracking algorithm. A similar online learning strategy is employed in [43], where an updated strong classifier is available at each frame and these classifiers are operated on the specific dimensions of the feature describing the samples. In order to achieve real-time computational efficiency, [2] utilizes HOG features and color channels obtained for each pixel. On the other hand, [43] exploits HOG features, Local Binary Patterns [86] and Haar-like features [106], all of which can be efficiently calculated using integral images.

After these two pioneering efforts, many variations and improvements over them have been proposed. For instance, another similar approach [72] extends ensemble tracking to the semi-supervised training to make use of more training samples including unlabeled ones. In [4], the weight vector which combines the weak classifiers is treated as a random vector and the posterior distribution of this vector is computed in a Bayesian framework. A scale adaptive version of ensemble tracking [2] is proposed in [63] to become robust to scale variation of the target objects.

It should be noted that the inaccurate labeling of the training samples is a potential reason for the tracking failure which gradually occurs as the number of wrong labels grows. This issue is called *drift* in the tracking literature. The tracking algorithms discussed above and even some of the recently proposed trackers accept the region surrounding the object as the positive instance, while the remaining ones as negative or background. This kind of labeling might trigger a tracking drift due to the insufficient number of positive examples. Hence, multiple instance learning (MIL) framework is adopted in visual tracking [3], where the main idea is to utilize a bag of positive samples containing at least one positive instance. Further extensions to this idea have been considered in [128] and [62]. This kind of learning strategy helps the tracker to remain robust with respect to the slight inaccuracies; thus, results in better localization. Unlike the use of bags in the MIL framework [3], Online Discriminative Feature Selection (ODFS) [122] directly selects features on the instance level by using a supervised method which is superior over MIL [3], and it keeps the computational efficiency significantly below the MIL-based method in [3]. In order to increase the efficiency, Fast Compressive Tracker (FCT) [123] employs compressive sensing theory to project the high dimensional features to a randomly chosen low dimensional space. This enables efficient computation without sacrificing much from performance. The studies MIL, ODFS and FCT rely on Haar-like features, since they can be computed efficiently and robust to appearance changes, such as rotation and motion blur up to a certain degree. The pre-trained deep features, which are recently involved in many computer vision problems, could also be exploited in the feature selection frameworks, *i.e.*, MIL, due to their robust nature in comparison to the hand-crafted features (*i.e.* Haar-like, HOG etc.). Nevertheless, no comprehensive research has been performed yet for this purpose.

Although the combination of weak classifiers, their proper update and looking for the most confident classifier response sound appropriate for any tracking task, the location information provided at the beginning of the tracking might be further utilized in the context of structured-output learning. Moreover, a model with a unique strong classifier could be beneficial, since it could avoid selecting the so-called best weak classifiers. Margin maximizing classifiers, such as Support Vector Machines (SVM) [22], are good candidates for this kind of task, since it has already been proven to be effective in many computer vision problems including object detection and visual object classification. In [45], the structured-output SVM has been adopted to the tracking problem, and the designed discriminant function explicitly includes the motion of the object. Moreover, a budget mechanism is incorporated to their online SVM formulation for memory efficiency. Following this work, many variants have been proposed, such as [95], which mainly designs a generic framework for structured learning to combine weak classifiers. Furthermore, a part-based and latent structured SVM model is proposed in [116], and they achieve favorable results against [45], since the drift problem is alleviated by the proposed part-based model. Moreover, a dual linear SVM model is presented in [85] to improve the tracking and to achieve the near real-time computation while high dimensional features are efficiently handled in their SVM framework.

In the last decade, representation learning methodologies have changed their direction to deep learning as initiative studies [32, 18, 96] show the power of deeply learned representations during the classification task. Following the same line of research, other computer vision tasks, such as object segmentation, tracking and detection, has started to employ deep neural networks due to their hierarchical representation learning power in supervised and unsupervised ways.

In DLT [65], an unsupervised learning is carried out by using stacked denoising autoencoders (SDAE). For the tracking phase, a classification layer is added to the SDAE architecture and tracking is accomplished using a particle filter framework. A selective update strategy is applied and the whole network is updated when necessary. In [55], 2-layer CNN and an RBF classifier are used, and the object is tracked by utilizing a single example. In [7], simultaneous object tracking and recognition is achieved by motivating theories of perception. Their model includes two interacting pathways, identity and control, whereas object appearance is modeled by using Deep Boltzman Machines and the motion is estimated using particle filtering. The study in [33] improves the idea of [7] with an additional gaze-control strategy.

In [38], a CNN architecture is trained by setting the input as the current and the previous region of interest of the object. The desired output is a Gaussian-shaped mask centered on the accurate spatial translational shift of the object. In this way, tracking is achieved giving the current and the previous frame around the region of interest as input to the architecture and resulting shift is accepted as the motion of the object. In [64], a CNN architecture is used and learned purely online to accomplish the tracking task and achieve the-state-of-the-art performance. Moreover, the same idea with a single CNN layer is proposed in [66] by the help of more image cues. In [51], a pre-trained CNN model is utilized as the input feature to the SVM and a discriminative saliency map is obtained by the target-specific gradient of the saliency map with respect to the input image. Finally, target localization is performed by the extracted saliency map.

Although correlation filter based methods have recently dominated the benchmark challenges such as [59, 35, 36], the recently proposed deep CNN-based method Multi-Domain Network (MDNet) [84] has achieved the best results and become the winner on VOT2015 challenge [35]. The main idea is to train a special CNN network consisting of convolutional layers as well as three fully-connected layers. The last convolutional layer is separately learned for each training video. In the tracking phase, the positive and negative samples are gathered around the target object, and the network is fine-tuned by initializing the last layer randomly. The most probable target location is estimated as the one which has the highest classification score among the densely sampled candidates around the previously estimated object location. Although the method has impressive tracking results in many benchmarks, it suffers from the computational complexity, since the evaluation of each candidate sample on the network is required to obtain corresponding confidence scores.

2.2 Generative Tracking Methods

Unlike the discriminative tracking approaches, generative methods are intended to calculate the joint probability of the instance and the label. In other words, an appearance model is described for the object and optionally for the background. The object location is estimated as the one which contains the test instance with the most similarity to the appearance model. The test instances can be picked in a particle filter framework or densely sampled from the instantaneous region of interest. The similarity calculation may differ according to the obtained model. Once the object location is estimated, the model could be updated or stop updating if certain conditions are satisfied, such as significant appearance change or occlusion detection. Although the generative tracking algorithms differ considerably in many respects, Figure 2.4 attempts to provide a conceptual overview with a reference to the method in [5].

A representation for the object is required for the appearance model construction for all of the generative methods. The representation can be raw image intensities or any hand-crafted or learned features for the defined object at the beginning of the tracking, and hence all of the object instances with different appearance variations span a subspace in the high dimensional feature space. At this point, the problem can be reduced to the online subspace learning. For instance, the study in [91] proposes an online subspace learning method for visual tracking based on incremental update for the principal component analysis (PCA). This technique provides to update the learned subspace throughout the video frames as the target appearance changes. On the other hand, it is also possible to represent the object appearance in terms of the statistics of the feature dimensions, such as the probability density functions [21]. In

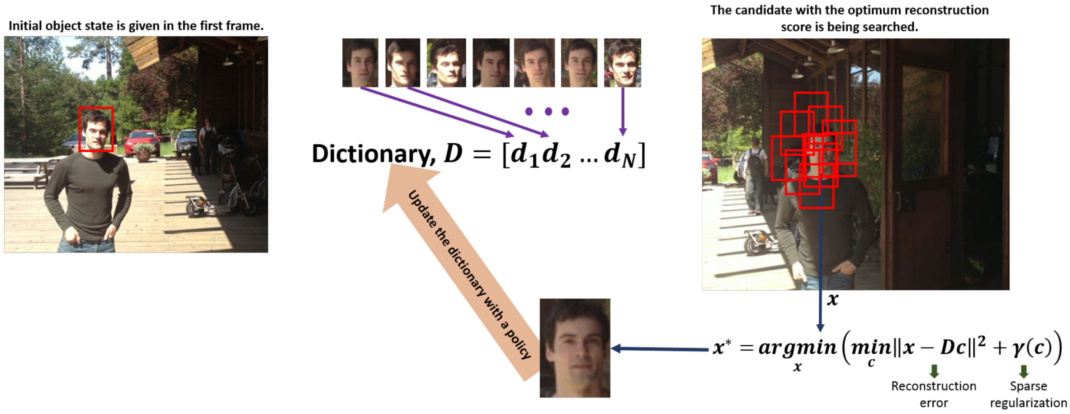

Initial object state is given in the first frame.
The candidate with the optimum reconstruction score is being searched.
Dictionary, $D = [d_1 d_2 \dots d_N]$
Update the dictionary with a policy
x
$x^* = \underset{x}{argmin}\left(\underset{c}{min}\|x - Dc\|^2 + \gamma(c)\right)$
Reconstruction error
Sparse regularization

Figure 2.4: A representative illustration of a generative visual tracking framework, which is drawn by getting an inspiration from the framework of L1APG [5] tracker. Many other designs might differentiate from the depicted figure. However, the general indication of a presence of the object is the closeness of the reconstruction error to zero.

that study, the localization is performed by the mean-shift algorithm, which is carefully customized for the tracking problem. Since the statistics of the appearance are not affected by the rotation and non-rigid motion (when compared to the object representations which are not invariant to the severe non-rigid motion and rotation, such as HOG and Haar-like features), the tracking accuracy is increased when compared to the normalized correlation based template matching algorithms. Visual Tracking Decomposition (VTD) tracker [61] makes use of more than one basic appearance model as well as multiple motion models. VTD employs sparse PCA technique [31] to activate the appearance model which is claimed to be effective in summarizing the object appearance.

Sparsity constraint has been shown to improve certain problems, such as face recognition [110]. Hence, a more detailed investigation of its usage in the visual tracking problem is performed in [81]. In that work, an $L1$ regularized least squares problem is solved and the object candidate with the least projection error is estimated as the object for the next frame. The sparsity constraint, which is satisfied by the $L1$ regularization, serves for finding a low dimensional subspace of the object appearance which is represented by high dimensional features. Moreover, the non-negativity constraint on the sparse reconstruction coefficients helps avoiding the clutter on the background. The template dictionary is updated by removing the template with the smallest sparse coefficient and adding new templates extracted from the current frame.

Since aforementioned method [81] is computationally infeasible for high dimensional data, a more sophisticated approach [71] improves this method [81] by proposing a two stage sparse optimization technique in order to jointly minimize the reconstruction error and maximize the discriminative power. On the other hand, a minimum error bound efficient sparse tracker is proposed in [82] by effectively selecting the samples in the particle filter framework. Furthermore, [82] has also an occlusion detection mechanism and do not update the tracker in a possible occlusion scenario. In order to further increase the efficiency of the optimization procedure, proximal gradient optimization method [5] is adopted in the sparse visual tracking problem and it achieves better results than the tracker in [82], while operating at higher frame rates. This study [82] is the first approach among other sparse trackers to obtain real-time speed on a standard desktop computer, since the approximate proximal gradient method generally has good convergence behavior for the objective functions, which are the summations of one smooth and one non-differential function and directly matches the sparse reconstruction problem. For the $L1$ norm-based trackers, the update is performed by removing the oldest or the unlikely templates and adding the new ones from the current frame. Nevertheless, one update pace might not be enough to represent the dynamic changes of the object appearance. In order to handle this problem, some authors [114] propose utilization of three dictionaries with different lifespans: short, medium and long. The short one represents the current object appearance, whereas the long one ensures the robustness of the model. Three dictionaries are utilized to learn the sparse coefficients for the current observation and all of the dictionaries are properly updated according to separate sampling policies. This strategy brings substantial improvements in the tracking accuracy.

Similar to the sparsity constrained discussion here, another useful idea might be the non-negativity of the coefficients to reconstruct the target sample as well as the non-negativity of the values of the templates in the dictionary, since this property is inherent to the video frames. If one can find a set of templates and coefficients to reconstruct the dictionary with the non-negativity constraints, it is very likely that the resulting dictionary elements will hopefully highlight the parts of the object. This approach is called Non-negative Matrix Factorization (NMF), and applied to the visual tracking problem in [113] by proposing an online NMF framework. Although there exist further efforts to improve this idea, such as [119] with additional sparsity constraints, the online NMF algorithm effectively performing for visual tracking is still unexplored.

Another sparsity-based solution to the visual tracking problem is multi-task learning framework [125]. In [125], a joint sparsity constraint is forced in such a way that the resulting sparse coefficients will not only be sparse themselves but also their usage for different samples will be sparse as well. Concretely, a few dictionary templates should be enough to represent all the templates of the object. Hence, the misaligned cropping of the object template, which is a potential reason for a tracking failure, is avoided due to its inconsistent sparse representation relative to the majority of the object templates. Following this work, a less restricted version of the method in [125] is presented in [126] to cover the outlier object appearances, while additional regularization terms are incorporated to the optimization problem. The detailed explanations and the reasoning behind the importance of low rank representations can be examined in the aforementioned studies.

Deep convolutional neural networks (CNN) have also been applied to the visual tracking problem in the context of sparse optimization in [108]. In that study, a feature selection process is performed on particular convolutional layers. The foreground object mask is reconstructed by the standard $L1$ regularized cost function similar to [5], where the foreground mask reconstruction is performed by looking for the coefficients corresponding to the selected convolutional feature maps. These feature maps can be considered as the dictionary for the foreground object mask.

2.3 Hybrid Methods

Hybrid methods that utilize both a set of discriminative and generative appearance models are also proposed in the literature. Moreover, custom neural network architectures are trained for visual tracking purposes. This section discusses the extensions of the generative and discriminative methods since they cannot be solely categorized as either discriminative or generative.

2.3.1 Correlation Filter Based Visual Tracking Methods

Correlation filter based (CFB) visual tracking methods are mainly based on the template matching idea which has a basis on the matched filter theory and they are frequently employed in radar applications [103]. Its arrival to the visual tracking community and practical usage in real world video sequences occurred quite lately after 2010, although many studies have been performed to identify the objects in simplified scenarios. The main goal is to find a “good" template, which is called as *correlation filter* in such a way that the correlation (or equivalently the convolution) of this filter and the candidate object instance should output a good correlation plane. The goodness of this correlation is measured with respect to its closeness to a peaky-shaped signal. The peaky-shaped signal is also known as the desired response. If the candidate object instance is shifted from the actual center of the object, then the filter should produce a response plane with the same shift amount on the peak location. An illustration of CFB tracking framework is presented in Figure 2.5 and 2.6. Since correlation filter is designed with respect to a desired correlation plane, it is also perceived as a discriminative regressor model. The difference between the naive correlation operation and the computed correlation filter can be visualized in Figure 2.6. As Figure 2.6 demonstrates, the correlation filter H, which is calculated by minimizing a cost function that considers a desired optimal correlation response, outputs a clear response when H is correlated with the object patch Y. On the other hand, the naive correlation of the object by itself produces multiple local maximums, and the correlation response has higher values in the background parts of the object patch when compared to the calculated correlation filter.

It should be noted that the correlation operation requires $\mathcal{O}(N^2)$ multiplications (N is the signal length) and might be problematic for long signals or large images. Thus, instead of (linear) correlation, circular correlation is mostly preferred for both finding the correlation filter and operating the correlation operations due to two reasons: (1) One can perform this operation in frequency domain with element-wise multiplications according to the Convolution Theorem, and Fast Fourier Transform has $\mathcal{O}(Nlog(N))$ complexity [87], (2) If the operations are reduced to the element-wise multiplications, the filter learning problem becomes trivial for most learning-based

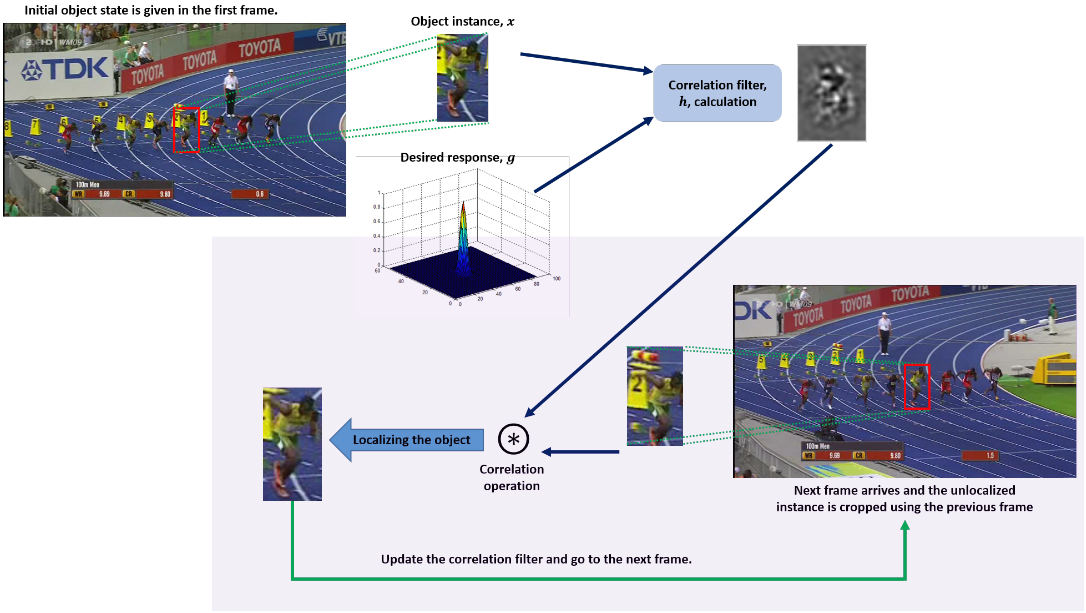

Initial object state is given in the first frame.
Object instance, x
Correlation filter, h, calculation
Desired response, g
Localizing the object
Correlation operation
Next frame arrives and the unlocalized instance is cropped using the previous frame
Update the correlation filter and go to the next frame.

Figure 2.5: A representative illustration of a correlation filter based visual tracking framework. The aim is to find a correlation filter such that the resulting correlation response of this filter and the unlocalized object observation will have a peak around the true center of the object.

methods. Due to these two reasons, although typical object motions are not circular in their translation, convolution operation is performed circularly while sacrificing from performance at the boundary locations.

2.3.1.1 Fundamental Correlation Filters

There exist preliminary efforts on designing and learning a correlation filter by using a set of training examples, such as Minimum Average Correlation Energy (MACE) filters [78], Optimal Tradeoff Filters (OTF) [89] and Unconstrained MACE (UMACE) [93]. Nevertheless, all of these methods have some major disadvantages. For instance, OTF and MACE require single desired correlation result, the centered filter and training examples. In UMACE, the lack of the constraints (or the supervision) is handled by removing the constraint on the desired correlation response to be a a particular value, instead by making it as high as possible. Obviously, those limited supervision makes them highly sensitive to noise, since there is no constraint or utilization of a prior information regarding the correlation response at other locations of the training examples. Hence, these family of correlation filter based approaches

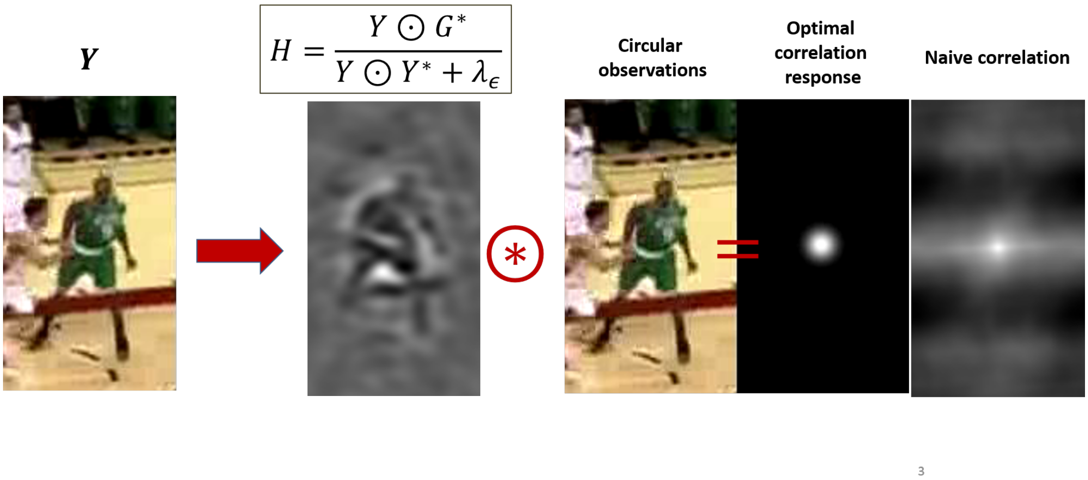

Y
$H = \frac{Y \odot G^*}{Y \odot Y^* + \lambda_\epsilon}$
Circular observations
Optimal correlation response
Naive correlation
3

Figure 2.6: Illustrating the effectiveness of using a desired correlation response.

are not discriminative. In the following subsections, discriminative correlation filter based methods are discussed.

2.3.1.2 Discriminative Correlation Filters

In order to overcome the previous correlation filter designs, a simple yet effective method, Average of Synthetic Exact Filters (ASEF) [15], exploits convolution theorem such that individual exact filters are obtained by dividing the desired response to the training images in the frequency domain. Then, the exact filters are averaged to extract a generalizing unique filter. Although the resulting correlation filter is not optimal for any cost function, the authors [15] present promising results for handling eye localization task, while they base their design on aggregation theory. For proper operation, there should be enough number of training examples. However, this might be a trouble for the tracking problem, since the correlation filter should have been designed in the very first frame.

A seminal work on discriminative correlation filters is proposed in [14] for visual tracking by Bolme *et.al.*, named as Minimum Output Sum of Squared Error (MOSSE). In MOSSE, a correlation filter is computed by the cost function to minimize the sum of squared errors between the desired correlation response and the correlation of the filter to be estimated and each training example. Since the circular correlation and the Convolution Theorem allow to operate in frequency domain, there exists a closed form expression for the correlation filter. If we look from the perspective of the previous studies on correlation filters, such as MACE and UMACE [78, 93], as a set of linear constraints, MOSSE [14] has intensive amount of constraints, *i.e.* the value of the correlation responses at all possible circular translations of each training example. Thus, the foreground and the background are learned discriminatively. Due to this property, it implicitly performs dense and circular template matching. Moreover, the closed form solution can be computed efficiently in the frequency domain and amenable to the simple moving average based updates. The aforementioned properties make MOSSE attractive. As a result, any researcher working on visual tracking problem should address this issue first in order to further develop or apply the machine learning and signal processing tools.

In [50], the correlation filter design in MOSSE [14] is perceived as a linear ridge regression problem. This perception is utilized to extend this idea to the ridge kernel regression in [50], where the kernel matrix is confined to be circulant by selection of appropriate kernels. Their proposed method has also a closed form solution in the frequency domain and consists of element-wise multiplications and divisions. Their tracking localization results are significantly better than the linear version in MOSSE.

2.3.1.3 Correlation Filters with Multiple Channels

Until now, a single correlation filter design is discussed, since the only information provided is the raw image intensities. However, there are alternative and robust versions of the raw image intensities including edges, oriented gradients and color channels. The kernelized correlation filter (KCF) based tracking is proposed in [48] by multi-channel extension of the tracker in [50]. Moreover, results on extensive amount of tracking sequences have shown impressive performance due to the use of multiple channels. In their study, HOG feature channels are employed by representing each pixels or a set of pixels with many orientation gradients. Apart from this study, the multiple channel correlation filters are applied to the object detection and alignment problems in three simultaneous studies [49, 40, 13], although these three works require expensive matrix inversions. It should be noted that the closed form solution dealing with both multiple channels and multiple training examples either in the dual or primal form of the kernel based formulation require intensive amount of computations and there is no efficient closed form solution in the frequency domain. Thus, the weighted averages are taken throughout the frames to compute a generalizing correlation filter model. Unlike the incorporation of maximum margin loss to the correlation filters in [90] with tedious and time-consuming computations, the study in [130] incorporate the SVM framework as support correlation filters with an increase in the efficiency even for multiple channels.

2.3.1.4 Scale Adaptability

The computation of multiple channel correlation filters with more than one template is formulated in [10] by an iterative optimization method. Moreover, [10] provides a scale adaptability by the posterior probability of a scale value according to a base scale. Discriminative Scale Space Tracker (DSST) [26], the winner of VOT2014 Challenge [59], has also a scale estimation stage in the context of correlation filters where the multiple scale search is efficiently fulfilled in the frequency domain by following the same idea in the correlation filter response calculation. In addition, DSST employs the multiple HOG channels as in [48] with an heuristic moving average based update stage. In [53], object detection proposals helps to improve the scale and aspect ratio of the target object unlike the search of single aspect ratio within multiple scales in most works. In [98], an adjustable Gaussian window operates spatial windowing to suppress the boundaries of the object patch instead of the cosine window utilized in most of the CFB studies, and the standard deviation of the window provides a scale estimation. Another scale adaptive kernelized method [69] also estimates the scale of the object by a multi-scale search and the optimum scale is decided according to the scale at which the correlation response is the best.

2.3.1.5 Other Extensions of Correlation Filters

There exist also methods which are either a combination of more than one concept including correlation filters or other tracking approaches. For instance, [41] attacks to one of the weak aspects of correlation filters, which is the boundary problem. In other words, imperfect training examples are inherently generated, since circular translation and the actual translation are assumed to be the same operations in the context of correlation filter based methods. To handle this issue, the study in [41] avoids the imperfect training examples by the help of a novel cost function containing a masking matrix. On the other hand, [28] proposes to penalize the boundaries of the correlation filters by a function in the spatial domain and the filter is obtained in the frequency domain by an iterative optimization process. [28] performs significantly better than [41] and achieves the best results with the use of pre-trained convolutional feature maps [27]. Following this work, an adaptive decontamination framework [25] is presented to discard the inappropriate training samples by learning the weights of these samples and the correlation filter model simultaneously. As a result, significant performance increase is achieved [25] .

Continuous convolutional operators tracker CCOT [30] is presented for the design of correlation filters in the continuous domain in order to cope with multiple feature sizes. Their formulation effectively employs the use of more than one convolutional layer with different size rather than naively resizing the feature maps to a reference size. This continuous formulation helps to obtain satisfactory results and makes this tracker the winner of VOT2016 Challenge [36]. [99] proposes different sparsity related loss functions unlike the conventional $L2$ loss utilized by the CFB tracking methods. Their proposed method relatively improves the results compared to the baselines while computationally more efficient than its state-of-the-art counterparts.

Although the sparse dictionary related trackers have become obsolete after the rise of CFB trackers, the study in [124] formulates the sparse dictionary learning by collecting circulant target templates and solves the problem efficiently in the frequency domain as in the case of CFB methods. Another drawback of the correlation filter based methods is due to the assumption that the desired correlation response is a peaky-shaped Gaussian. However, any other response with a peaky behaviour could be desirable. To address this issue, [11] proposes to learn both the correlation filter and the target response, concurrently. This simultaneous learning improves the tracking localization in benchmark sequences.

2.3.1.6 Learned Representations for Correlation Filters

As the impact of deep learning methods has increased in the recent years, such learned features trained for object classification tasks have been employed in many other tasks as it is stated earlier in Section 2.1, especially for the discriminative trackers. These pre-trained deep features have also been utilized by the correlation filters, except for their fully-connected layers, since fully-connected layers corrupt the shift invariance property, existence of which is an obligation for correlation filters. Hence, convolutional layers of the pre-trained networks are exploited in [27]. In that work, the convolutional feature maps are employed as the multiple channels of the correlation filters and they empirically try to determine the layers that are useful for correlation filter based visual tracking. The shallow layers are found out to be effective for improving the results. [77] merges the information in multiple convolutional layers, whereas in [76], multiple convolutional layers are hierarchically exploited to estimate the object location in a coarse-to-fine manner. Multiple channels are also wisely merged in [30] in the continuous domain. The main conclusion for the selection of layers is superiority of the combination of lower and higher layers, since shallow layers encode the low level features such as edges or small parts of the object, while the higher levels contain more holistic information about the object.

2.3.2 Methods based on neural networks

Recently, various deep architectures with customized layers or objective functions have been proposed. Among them, Siamese networks [20] have been popular in order to learn feature embeddings for a specific task with a contrastive loss. The main objective of a Siamese network is to learn a model which will output a feature embedding such that the instances belonging to the same class should have a proximity in the learned space, whereas the instances of different classes should be as far as possible from each other. An application of this concept to the visual object tracking is proposed in [101] where the network learns to output similar features for various appearances of the target object and dissimilar ones for the target and non-target samples. Once the model is learned, the feature vector of the first object appearance is compared to the candidates to find the best match without any update mechanism.

Nevertheless, evaluating many candidates are quite expensive. Hence, a CNN model is introduced in [47], which directly learns to output the relative location of the object with respect to a reference object instance and avoids the expensive candidate evaluations and the feature matching phase. Unlike the model in [47] employing fully connected layers, a fully-convolutional neural network is presented in [9]. In [9], the object template and the test frame are passed through the same convolutional layers, and their correlation is obtained by the sliding window approach. The sliding window stage is operated in the convolutional layer format, since the standard deep learning libraries are efficiently exploited in order not to sacrifice much from the computation time.

Another popular neural architecture that is worth to mention is Recurrent Neural Networks (RNNs) [34], which is a useful neural network model, especially in natural language processing. Recently, it has been started to be used for particular computer vision tasks, including visual tracking. For instance, RNNs are employed in [23] in order to estimate the confidence map of the target object by modeling the spatial relationships between the object and the background. Another spatial perspective is to spatially model the object structure. The study in [37] successfully applies this idea to the visual tracking problem in order to assist the CNN layers. Unlike the use of RNNs for the spatial relationships, two concurrent works, [56] and [42], propose to learn an RNN model to directly estimate the motion of the object by modeling their RNNs to learn the relationships between the frames sequentially. Nevertheless, the visual tracking experiments are conducted on the simulation data, and they lack the performance on the standard benchmarks, such as VOT challenges [59, 35, 36] or Online Tracking Benchmarks [111, 112].

2.4 Ensemble of Trackers

Combination of multiple online trackers is another line of research. For instance, multiple correlation trackers are executed at different parts of the object in [73]. A part-based version of MOSSE [14] has been proposed in [102] to accomplish object detection task. Reliable patches are tracked in [70] using KCF [48] as the base tracker. The work in MEEM [120] selects the best SVM-based discriminative tracker according to an entropy minimization criterion. Markov Chain Monte Carlo sampling is also used to sample trackers and combine them [54]. Various trackers with mixed feature types are combined in [100]. Hybrid methods combining generative and discriminative approaches are also proposed in [127, 118]. Improving the CNN-based method MDNet [84] described in Section 2.1, in [83], a tree-structure stores the different appearances in the nodes of the tree as CNN models. This provides a robustness to significant appearance changes of the target. The object state estimation is performed by combining the trackers in the nodes with the help of a specially designed policy.

In this chapter, the state-of-the-art visual object tracking methods are discussed in detail with a novel taxonomy taking the recent advances in the literature into account. The presented efforts typically handle the well-known challenges of visual tracking up to a certain level in specific scenarios. Throughout this chapter, the crucial issues related to the advantages and disadvantages of all of the mentioned methods are discussed while considering these challenges. In this thesis, the aim is to find solutions for the problems of the existing methods by utilizing correlation filter based tracking formulation. Firstly, the inefficient localization step of discriminative track-by-classification methods (they generally evaluate a classifier in all candidate locations) is avoided due to the choice of correlation filter based trackers. Unlike the generative methods, the discrimination of foreground and background regions exists in the adopted studies for this thesis. Moreover, the proposed ensemble tracking method that is described in Chapter 5 addresses the non-optimal learning rate problem of correlation filter based trackers, while handling to track objects with various appearance changes. The proposed spatial window learning method that is described in Chapter 4 improves the correlation quality in the case of occlusion and object deformations. Finally, the feature learning framework presented in Chapter 6 address the challenges of abrupt appearance changes, object deformations and the utilization of imperfect training examples by learning robust features to these challenges.

In brief, in this thesis, the proposed methods exploits the correlation filter based methods, obtains favorable tracking performance against the existing counterparts as well as other family of trackers, such as discriminative track-by-classification methods, while maintaining the computational complexity of the proposed trackers as low as possible for their use in practical applications.

CHAPTER 3

CORRELATION FILTER BASED TRACKING

The main focus of the thesis is to provide state-of-the-art improvements over single object tracking algorithms by the help of correlation filter theory. In order to achieve this goal, a popular correlation filter based tracking formulation is followed in this study, namely *linear multiple channel correlation filters*, since linear formulation [14, 26] facilitates its further development and performs favorably against the kernelized version [48, 50] in terms of accuracy and efficiency. In our proposals, we mainly follow two frameworks: (1) Discriminative Scale Space Tracker (DSST for short) [26] and (2) Continuous Convolution Operators Tracker (CCOT for short) [30]. Hence, the formulations and necessary derivations will be included here after the introductory basics about the single channel linear correlation filters.

3.1 Single Channel Correlation Filters for Visual Tracking

Following the correlation filter taxonomy presented in Section 2.3.1, the pioneering method is the single channel linear correlation filter formulation, which minimizes the sum of squared error (MOSSE) [14] of the correlation response, and its convenient formulation is summarized here.

Beforehand, the correlation and convolution theorem will be reminded, since they will be often utilized throughout the derivations.

$$c[n] = \sum_{m} a[m]b[n-m] = \mathcal{F}^{-1}\{A \odot B\} \tag{3.1}$$

$$c[n] = \sum_{m} a[m]b[n+m] = \mathcal{F}^{-1}\{A^* \odot B\} \tag{3.2}$$

Equations (3.1) and (3.2) are convolution and correlation theorem for two one-dimensional discrete signals, $a[.]$ and $b[.]$ with the equal length, respectively. The signal $a[n-m]$ is assumed to be the circularly translated version of $a[n]$ by an integer amount m to the right unless stated otherwise. The superscript * denotes the element-wise complex conjugate. (3.1) and (3.2) are obviously corollary of each other. The symbol $\odot$ denotes element-wise multiplication, whereas $\mathcal{F}$ and $\mathcal{F}^{-1}$ are Discrete Fourier Transform (DFT) and inverse DFT, respectively. All capital letters denote the signals in the frequency domain, while the lower case letters represent the signals in the spatial domain. The extension of the theorem to two-dimensional case is straightforward, since the DFT operation has a separable 2-dimensional definition. Henceforth, it is assumed that any derivation for 1-dimensional signals can be extended to 2-dimensional signals, if one can split the operations into two dimensions.

3.1.1 Correlation Filter Computation

At the beginning of the tracking, a set of training examples, $\{x_i\}_{i=1}^{N}$, are assumed to exist, where N denotes the number of examples. Each x_i represents an $M \times M$ two-dimensional object patch. Moreover, $\{g_i\}_{i=1}^{N}$ are the desired correlation responses for a filter h and the training examples. Concretely, if the object x is centered with respect to the bounding box, the desired response g is also centered, while g should be shifted from the center by the amount which x is shifted from the center. When the correlation filter and the training examples are circularly correlated, the resulting responses should resemble the desired responses, $\{g_i\}_{i=1}^{N}$, as much as possible. This resemblance is enforced by the following loss function [14]:

$$h_{opt} = \arg\min_{h} \sum_{i=1}^{N} ||h \circledast x_i - g_i||^2, \tag{3.3}$$

where $\circledast$ is the circular correlation operation, and h_{opt} is the resulting correlation filter. According to the correlation theorem, (3.3) can be written in the frequency domain as follows:

$$H_{opt} = \arg\min_{H} \sum_{i=1}^{N} \|H^* \odot X_i - G_i\|^2. \tag{3.4}$$

Since the operations are element-wise multiplications (denoted by $\odot$), the optimization of each element of the signal H can be performed separately. The term inside the $argmax$ in the above equation can be rewritten as:

$$\mathcal{L} = \sum_{i=1}^{N} H^* \odot H \odot X_i^* \odot X_i + G_i^* \odot G_i - H^* \odot X_i \odot G_i^* - H \odot X_i^* \odot G_i. \tag{3.5}$$

If the derivative of the function in (3.5) is taken with respect to H^* for all of its elements, we obtain[1]

$$\frac{\partial \mathcal{L}}{\partial H^*} = \sum_{i=1}^{N} H \odot X_i^* \odot X_i - X_i \odot G_i^*. \tag{3.6}$$

When (3.6) is equated to zero, H is obtained as follows:

$$H = \frac{\sum\limits_{i=1}^{N} X_i \odot G_i^*}{\sum\limits_{i=1}^{N} X_i^* \odot X_i}. \tag{3.7}$$

In (3.7), H is the correlation filter which minimizes the cost in (3.3) in the frequency domain. It is notable that the calculation of H has complexity of $\mathcal{O}(M^2 log(M))$ due to the Fast Fourier Transform algorithm with $\mathcal{O}(M log(M))$ complexity. If the operations were performed in the image domain, $M^2 \times M^2$ matrix operations would be required, *i.e.*, matrix inversion and matrix multiplications due to the need for linear least square solution. This efficiency provides the correlation filter usage in the tracking problem with a real-time operation opportunity.

3.1.2 Object Localization

Once the filter H_t is obtained at the t^{th} frame, the unlocalized object patch Z_{t+1} is element-wise multiplied by H_t, and the estimated response g_{t+1}^{est} is obtained simply by taking its inverse DFT:

$$g_{t+1}^{est} = \mathcal{F}^{-1}\{H_t^* \odot Z_{t+1}\} \tag{3.8}$$

---

[1] H and H^* are regarded as independent from each other [14] and $\frac{\partial H}{\partial H^*}$ is equal to zero.

In (3.8), it is obvious that the correlation response g_{t+1}^{est} would be exactly g_i, if $Z_{t+1} = X_i$ and there exist one training example. For localization, the pixel location which gives the highest value of g_{t+1}^{est} is determined as the shift from the center.

3.1.3 Model Update

As it can be observed from (3.7), the optimum filter is computed by only adding some terms to the numerator and denominator of (3.7). Hence, the following update is performed throughout the frames with a learning/update rate γ conveniently:

$$\begin{aligned} A_{t+1} &= (1-\gamma)A_t + \gamma(X_{t+1} \odot G_{t+1}^*), \\ B_{t+1} &= (1-\gamma)B_t + \gamma(X_{t+1} \odot X_{t+1}^*), \end{aligned} \tag{3.9}$$

where H_{t+1} at the $t+1^{th}$ frame is:

$$H_{t+1} = \frac{A_{t+1}}{B_{t+1}}, \tag{3.10}$$

and X_{t+1} is the object patch cropped after the localization and G_{t+1} is the desired response in the frequency domain with a centered Gaussian since the object patch is already localized.

The computations of the model update and the localization steps can be operated so efficiently that the visual tracking algorithm relying on MOSSE is able to run beyond real-time. Nevertheless, a single feature map (*e.g.* raw image intensities) is not enough to maintain accurate tracking of a target object when the appearance changes significantly degrade the representation power of the correlation filter. Furthermore, MOSSE framework is not appropriate for significant scale changes of the object. Due to these challenges, tracking methods that support multiple feature maps and include multi-scale search steps [69, 26] are proposed. In the following section, a state-of-the-art correlation filter based tracking method with the aforementioned capabilities is explained and employed in the proposed tracking algorithms.

3.2 Discriminative Scale Space Tracker (DSST)

DSST is an extension of MOSSE [14] in which only the first term of the cost function in (3.11) is minimized with one feature map (*i.e.* image intensity) as it is formulated in the previous section. Here, the feature maps $\{x^1, ..., x^d\}$ correspond to the training example x, which consists of particular feature maps, such as HOG orientation maps or deep feature maps with the same dimension as the object patch. The desired correlation mask of the training example x is denoted by g. The desired cost function to be minimized is given below to cover all of the training examples until time t:

$$\mathcal{L}(h_t) = \sum_{i=1}^{t} \left\| (\sum_{l=1}^{d} h_t^l \circledast x_i^l) - g_i \right\|^2 + \lambda_\epsilon \sum_{l=1}^{d} \|h_t^l\|^2 \tag{3.11}$$

Here, λ_ϵ is the control parameter for $L2$ regularization term of the filter. As (3.11) suggests, a set of filters $\{h_t^l\}_{l=1}^d$ should be estimated such that the correlation operation between h_t^l's and x_t^l's are summed and the error between the desired response g_i's and the summed correlation results $\sum_{l=1}^{d} h_t^l \circledast x_i^l$ should be minimized under the $L2$ regularization of the correlation filters. Although there exist a closed form solution for the correlation filters in (3.11), it is not as efficient as the one for the single channel case in the frequency domain. Fortunately, there exists a closed form solution in the frequency domain for one training example, *i.e.* $t = 1$. In order to obtain the efficient solution, (3.11) can be rewritten in the frequency domain using Parseval's relation by assuming one training example[2]:

$$\mathcal{L}(H) = \left\| \sum_{l=1}^{d} H^{l*} \odot X^l - G \right\|^2 + \lambda_\epsilon \sum_{l=1}^{d} \left\| H^l \right\|^2. \tag{3.12}$$

Remark: During the derivations, the complex terms have the following interpretations: X^l and H^l are $M \times M$ complex signals, $X^l[w]$ and $H^l[w]$ are complex scalar values at the w^{th} discrete frequency index of the signal X^l and H^l, and $X[w]$ and $H[w]$ are $d-$dimensional complex vectors at the discrete frequency index w. Yet, the desired correlation response, G, is a single feature map in the frequency domain and $G[w]$ is a complex scalar for each frequency index, w.

If (3.12) is explicitly written by summing over all the frequency components to obtain its norm,

$$\mathcal{L}(H) = \sum_{w} \left| \sum_{l=1}^{d} H^{l*}[w] X^l[w] - G[w] \right|^2 + \lambda_\epsilon \sum_{l=1}^{d} \left| H^l[w] \right|^2 \tag{3.13}$$

---

[2] The time subscript is dropped for clarity.

is obtained. Since each frequency component w has a d-dimensional complex vector for X and H, (3.13) can be converted to:

$$\mathcal{L}(H) = \sum_{w} \left| X[w]^H H[w] - G[w]^* \right|^2 + \lambda_\epsilon \|H[w]\|^2, \tag{3.14}$$

where Y^H is the Hermitian transpose of the complex vector Y. All the terms in (3.14) depends on one frequency index. Hence, the minimization can be evaluated for each frequency independently, and separate quadratic minimization terms are obtained for each frequency w as:

$$\mathcal{L}(H[w]) = H[w]^H X[w] X[w]^H H[w] + G[w] G[w]^* - G[w]^* H[w]^H X[w] - \tag{3.15}$$

$$G[w] X[w]^H H[w] + \lambda_\epsilon H[w]^H H[w]. \tag{3.16}$$

If the gradient with respect to H^H is taken and equated to zero,

$$(X[w] X[w]^H + \lambda_\epsilon I_d) H = X[w] G[w]^* \tag{3.17}$$

is obtained for each frequency component, where I_d is the $d \times d$ identity matrix. By exploiting the matrix inversion lemma and dropping the frequency index within the square brackets, the final correlation filters are obtained as:

$$H^l = \frac{X^l \odot G^*}{\sum\limits_{k=1}^{d} X^k \odot X^{k*} + \lambda_\epsilon} = \frac{A^l}{B + \lambda_\epsilon}, \forall l \in 1, ..., d \tag{3.18}$$

At each time instant, the filter H^l is updated by applying moving average to the numerator and denominator of (3.18) separately via:

$$\begin{aligned} A_t^l &= (1-\gamma) A_{t-1}^l + \gamma (G_t^* \odot X_t^l), \\ B_t &= (1-\gamma) B_{t-1} + \gamma \sum_{k=1}^{d} X_t^k \odot X_t^{k*}, \end{aligned} \tag{3.19}$$

where γ is the model update rate. The correlation of an object patch z_{t+1} and the model H^l is calculated using the updated numerator A_t^l and denominator B_t of H^l in the frequency domain using:

$$c = \mathcal{F}^{-1} \left\{ \frac{\sum\limits_{l=1}^{d} A_t^{l*} \odot Z_{t+1}^l}{B_t + \lambda_\epsilon} \right\}, \tag{3.20}$$

where the spatial domain correlation mask is obtained by taking the inverse Fourier transform. The new location of the object in the next frame is estimated as the location giving the maximum value at c in (3.20).

For scale estimation, DSST extracts $\tilde{d}$-dimensional HOG features for S scale factors. In order to achieve this aim, the base size of the target is multiplied by the scale factor. The corresponding region is cropped and described by the $\tilde{d}$-dimensional features similar to the location estimation procedure. Then, the scale correlation filter h_{scale} is calculated for the scale samples $x_s \in \mathcal{R}^{\tilde{d} \times S}$. The optimal scale is determined as the scale index giving the highest value on the correlation response of the test instance z_{scale} and h_{scale}. The moving average based update is again employed for the scale filter as well.

It should be pointed out that the location and scale estimation have the complexity of $\mathcal{O}(dM^2log(M))$ and $\mathcal{O}(\tilde{d}Slog(S))$, respectively, if the object is $M \times M$, the number of feature channels is d, the scale range is S and the scale feature dimension is $\tilde{d}$. The same complexity is also valid for the model update stage. Hence, this method can be operated in real-time in a single CPU, when those feature dimensions and the size of the object patch are kept within practical bounds. Nevertheless, its performance can be further increased at the expense of the computation time. One of the recently proposed trackers which aims to boost the performance of DSST-like [26] trackers is explained in the next section.

3.3 Continuous Convolution Operators for Visual Tracking

In order to achieve the state-of-the-art results, the most recent developments on the correlation filter based tracking literature should be examined. As it will be described in the following chapters, deep convolutional networks are learned specifically targeting at the correlation filter loss function. In the employment of the deep features or any other feature such as HOG, the feature maps are interpolated to a reference scale or an aspect ratio to operate the correlation filter formulations. For instance, the deep features extracted from different layers of a convolutional network have different dimensions. However, all of them should have the same spatial dimensions to calculate the filters and to operate the localization step efficiently in the frequency domain. This goal is fulfilled by explicitly resampling all feature maps to a common resolution. Yet, the resampling adds redundant data and introduces artifacts. To overcome this problem, a continuous domain formulation is proposed in [30] and obtains promising results on many benchmark datasets. The detailed explanation for Continuous Convolution Operator Tracker (CCOT) [30] is summarized in this section, since CCOT is utilized to evaluate the effectiveness of our learned features, *i.e.*, deep convolutional feature maps.

In order to operate in the continuous domain, $L^2(T)$ space of complex-valued periodic functions $g : \mathcal{R} \rightarrow \mathcal{C}$ with period $T > 0$ are the signals under consideration. Moreover, $L^2(T)$ is a space consisting of square Lebesgue integrable functions. The inner product and the circular convolution are defined as follows [30]:

$$\begin{aligned} < g, h >&= \frac{1}{T}\int_0^T g(t)h^*(t)dt, \\ g * h(t) &= \frac{1}{T}\int_0^T g(t-s)h(s)ds. \end{aligned} \tag{3.21}$$

In the above equation, $*$ is the circular convolution operator in the continuous domain.

In theory, a family of exponential functions with infinite quantity, *i.e.* $\{e^{j\frac{2\pi kt}{T}}\}_{-\infty}^{\infty}$, are required as the orthonormal basis for $L^2(T)$. Fourier coefficients of $g \in L^2(T)$ are $\hat{g}[k] =< g, e^{j\frac{2\pi kt}{T}} > \forall k$ such that $g = \sum\limits_{k=-\infty}^{\infty} \hat{g}[k]e^{j\frac{2\pi kt}{T}}$. Moreover, Parseval's formula and the convolution theorem are also satisfied as follows:

$$\begin{aligned} &\|g\|^2 = \|\hat{g}\|^2, \quad where \quad \|g\|^2 =< g, g > \quad and \quad \|\hat{g}\|^2 = \sum_{k=-\infty}^{\infty} |\hat{g}[k]|^2, \\ &\widehat{g * h} = \hat{g}\hat{h}, \quad and \quad \widehat{gh} = \hat{g} * \hat{h}, \quad where \quad \hat{g} * \hat{h}[k] = \sum_{l=-\infty}^{\infty} \hat{g}[k-l]\hat{h}[l] \end{aligned} \tag{3.22}$$

Unlike the constant dimension assumption for all of the feature maps in the previous section, each training sample x_j is here allowed to have the feature maps with different dimensions as $x_j^d \in \mathcal{R}^{N_d}, \forall d \in \{1, 2, \cdots, D\}$. To implicitly model the signals in the continuous domain, the interval $[0, T)$ is assumed to be the support interval. For each feature map d, the interpolation function is expressed as:

$$J_d\{x^d\}(t) = \sum_{n=0}^{N_d-1} x^d[n]b_d\left(t - \frac{T}{N_d}n\right), \tag{3.23}$$

where $b_d \in L^2(T)$ is the interpolation function. With the above equation, the discrete feature map $x^d[n]$ is mapped to the continuous domain as $J_d\{x^d\}(t)$ by the help of an interpolation function. The ultimate goal here is to find a linear convolution (or a correlation) operator S_h such that a sample x is mapped to a target confidence response $s(t) = S_h\{x\}(t)$. Since there exist D feature maps, the correlation filters $h = (h^1, h^2, ...h^D) \in L^2(T)$ are intended to be estimated. Here, h^d is the continuous filter for feature channel d. The convolution operator in the continuous domain is described as:

$$S_h\{x\} = \sum_{d=1}^{D} h^d * J_d\{x^d\}. \tag{3.24}$$

Although the initially given signals are discrete, they are first converted to the continuous domain using the operation $J_d\{x^d\}$. Moreover, there should be continuous desired values g_j for each training example x_j. The correlation filter cost function for m training examples is defined in the continuous domain by:

$$E(h) = \sum_{j=1}^{m} \alpha_j \|S_h\{x_j\} - g_j\|^2 + \sum_{d=1}^{D} \|w \odot h^d\|^2. \tag{3.25}$$

Here, α_j represents the importance of the sample x_j, and w is a spatial penalty function to regularize the correlation filter in the spatial domain. This penalty function is selected as the one which has high values around the boundary of the region of interest while it gets lower values near the center similar to the idea in SRDCF tracker (*c.f.* [28] for details).

In order to learn the filter set h minimizing the cost in (3.25), the frequency domain is mostly utilized. By the previously given properties of the Fourier coefficients, $\widehat{J_d\{x^d\}}[k] = X^d[k]\hat{b}_d[k]$ are Fourier coefficients of the interpolated feature maps where $X^d[k] = \sum_{n=0}^{N^d-1} x^d[n] e^{-\frac{2\pi nk}{N^d}}, \quad k \in \mathcal{Z}$ is DFT of x^d. The confidence function has the Fourier coefficients as:

$$\widehat{S_h\{x\}}[k] = \sum_{d=1}^{D} \hat{h}^d[k] X_d[k] \hat{b}_d[k]. \tag{3.26}$$

Thus, the cost function in (3.25) becomes:

$$E(h) = \sum_{j=1}^{m} \alpha_j \|\sum_{d=1}^{D} \hat{h}^d X_j^d \hat{b}_d - \hat{g}_j\|^2 + \sum_{d=1}^{D} \|\hat{w} * \hat{h}^d\|^2 \tag{3.27}$$

by applying Parsevals relation and using (3.26). Although the above formulations are in the continuous domain and require infinitely many number of Fourier coefficients, h is approximated by a finite amount of Fourier coefficients $\{\hat{h}^d[k]\}_{-K^d}^{K_d}$ for the filter of the feature map h^d by assuming that $\hat{h}^d[k] = 0$ for $|k| > K_d$. The K_d value is set to $\frac{N_d}{2}$. After this point, the terms in (3.27) are converted to the linear matrix, and vector multiplications and a set of normal equations are obtained. Since these equations are in so high dimensional space that the matrix inversion is infeasible, the Conjugate Gradient Descent is utilized to iteratively optimize the cost. For the other implementation details, the reader should refer to [30]. Once the object is localized, a multi-scale search is adopted to find the best matching scale by looking at the correlation responses at every scale. The best scale is selected as the one with the highest correlation response. If the best scale is different than the current one, the region of interest is cropped with the estimated scale and the same localization operations are performed for the next frame.

Thus far, the correlation filters that will be dealt are summarized: single channel linear correlation filter, multiple channel correlation filters and continuous convolution operators, all of which will be utilized in the proposed frameworks to further extend these methods for better tracking performance.

CHAPTER 4

LEARNING SPATIAL WINDOWS FOR CORRELATION FILTER BASED TRACKING

Correlation filters have recently attracted the visual tracking community due to mostly its efficiency and the simplicity of their update equations as explained in Section 3.2. Nevertheless, correlation filter based methods suffer from the following drawback: At each frame, a template filter is correlated with the region of interest in the current frame; although, there is an update strategy, the filter and the current region of interest might have significant dissimilarities due to the partial occlusion of the object, motion model mismatches or abrupt appearance changes of the target (*e.g.*, appearance of a rolling ball, a walking person, a partially occluded face). One alternative to cope with this situation is to increase the learning rate of the model. Yet, such a strategy has two major drawbacks: (1) Overfitting to the latest object appearance and (2) inappropriate nature of the features for the additive model update.

In the ideal case, the ultimate goal is obviously to have both an object description invariant to any kind of changes and zero update process. In the current line of research, there exists no such feature type which is invariant to all kinds of appearance changes. Thus, improvements to handle the appearance changes are often studied by proposing further extensions to a base framework. To this end, we opt to selectively permeate the extracted features. Numerous efforts have attempted to design feature selection algorithms. The pioneering examples for the tracking problem are MIL [3] and ODFS [122], where a subset of the features are to be selected according to their discrimination capabilities between the background and object.

In the context of the correlation filters, each component of the multiple feature maps are two-dimensional discrete signals. Moreover, the correlation response is mostly effected by the spatial arrangement of the pixel values. In order to avoid the described challenges, we propose a new spatial windowing method by reducing a cost function which penalizes the dissemblance of the correlation output to a peaky shaped signal. In other words, the differences between the template filter and object appearance at the current frame are suppressed by the estimated masking values in such a way that correlating the remaining (robust) appearances of the object and the correlation filter is able to yield a sharply peaked output. Although hard assignments for the window to be learned could be a nice choice (*e.g.*, multiplying the value at each location with 1 or 0), the problem statement is relaxed to the continuous valued outputs in order to efficiently formulate the window learning problem.

Concretely, we deal with the problem of estimating a spatial window for the object observation such that element-wise multiplication of the object observation (or filter) and the extracted spatial window improves the tracking performance by suppressing the irrelevant regions of the object (*e.g.*, occlusions or motion model mismatch) and highlighting the parts of the object which supports the similarity with the utilized correlation filter for the tracking task. Likewise, it is possible to design the window learning framework for the correlation filter as well. Hence, the spatial window learning on the correlation filter is also proposed and compared with respect to the case for the windowing on the object observation.

The efficacy of the proposed approach can be visualized by a typical example in Figure 4.1. In this figure, a rolling ball with hexagons is to be tracked. As the ball rolls, the locations of the rotating hexagons change rapidly. Hence, a significant mismatch in the location of the hexagons between two consecutive frames occurs. As it is illustrated in Figure 4.1, the highest location of the correlation output is shifted from the center when the conventional correlation filtering [26] is applied, whereas the correlation output gives an accurate highest location by using the proposed windowing strategy. The inferior localization of the conventional approach is due to the abrupt appearance changes of the ball, since rolling causes the location of the hexagons on the ball to move abruptly. The proposed window suppresses the hexagons (*cf.* bottom left in Figure 4.1), which are moving faster than the ball itself.

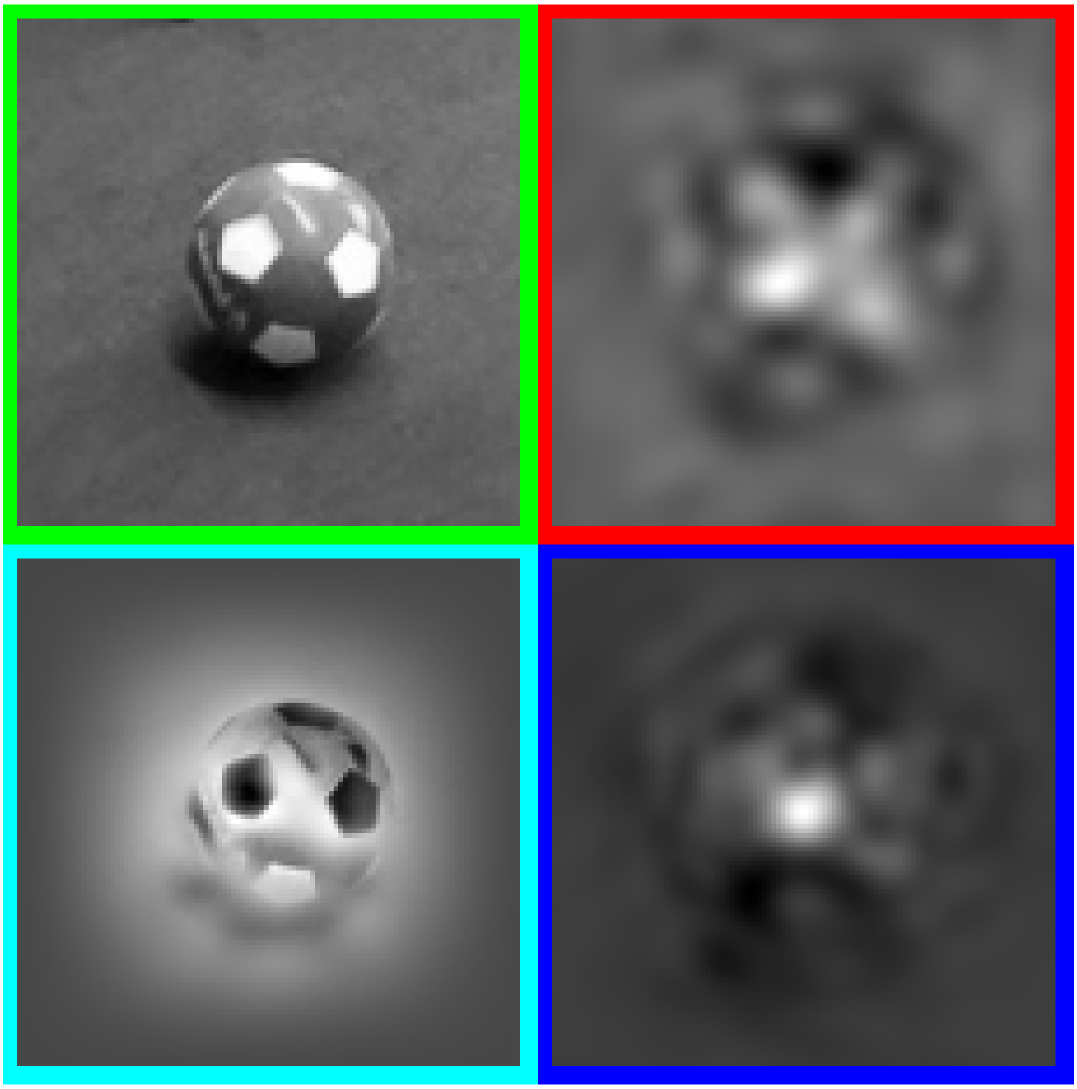


Figure 4.1: *Ball sequence,* frame # 87. **Top-left:** The object patch. **Top-right:** Correlation result using the conventional Hanning window. **Bottom-left** The estimated window by the proposed method. **Bottom-right:** Correlation result after the proposed method. (Dark values indicate low values for window function and the correlation results.) The correlation score is much sharper and located at a better position (*i.e.* non-translation)

In the following sections, windowing applied to the instantaneous object observation and to the correlation filter are explained, respectively.

4.1 Spatial Windowing Applied to the Object Observation

Windowing is an element-wise multiplication operation with the signal under concern before DFT calculation to reduce the frequency leakage [87]. Moreover, it is also used in correlation filter based trackers to suppress the boundaries due to the circular correlation operation [48, 26]. For a localized object patch x_t, the correlation output $\hat{g} = h_t \circledast x_t$ should give the desired response g as in (3.3) for the currently available correlation filer h_t. Nevertheless, $\hat{g}$ might have multi-peaks or an undesired maximum location due to the challenges, such as partial occlusion, appearance changes of the object and motion model mismatches, which will cause a drift of the tracking result from the actual location in the next frame.

To this end, we propose an online window estimation method to alleviate the adverse effects mentioned above by reducing the cost below in order to increase the similarity between $\hat{g}$ and g:

$$\epsilon(w) = \|h_t \circledast (w \odot x_t) - g\|^2, \tag{4.1}$$

where w is the unknown window that should reduce this cost function. In other words, we try to estimate a window, which will be multiplied by the signal x_t element-wise as $f_t = x_t \odot w$, instead of other (constant) windowing functions (*e.g.*, Gaussian or Hanning) such that the resulting correlation $\hat{g} = h_t \circledast (x_t \odot w)$ has more resemblance to the peaky shaped signal g than the other fixed windowing functions will cause. Here, $h_t = \mathcal{F}^{-1}\{A_t^1/(B_t + \lambda_\epsilon)\}$ is the $P \times P$ correlation filter for the image intensity features (the first feature channel in Eq. (3.18)). For reducing (4.1), gradient descent with a step size of γ_{SW} can be utilized as:

$$w[i] \leftarrow w[i] - \gamma_{SW} \frac{\partial \epsilon}{\partial w[i]}, \tag{4.2}$$

where each entry of the discrete signal w is perceived as an unknown parameter. The number of parameters to be optimized are as many as the number of the pixels in the window patch.[1] We propose a method for computing the gradient of the cost function with respect to the elements of the spatial window w. During the derivation of the gradient computation, the signals will be considered as 1-dimensional and the correlation output $z[n]$ of two signals $x[n]$ and $y[n]$ will be denoted as $z[n] = \sum_i x[i]y[i+n]$

---

[1] Throughout this chapter, w denotes the signal as vector whereas $w[i]$ denotes the i^{th} element of the signal w and we drop the time dependency subscript t for the clarity of the derivation.

as it has already been defined in Section 3.1 [2]. The cost function can be rewritten as:

$$\epsilon(w) = \sum_n \left(\sum_i h[i]w[i+n]x[i+n] - g[n] \right)^2 \tag{4.3}$$

by summing the squared values of the elements of the difference signal between the correlation output function and the desired response g.

Taking the partial derivative of $\epsilon(w)$ with respect to $w[m]$ gives:

$$\begin{aligned} \frac{\partial \epsilon(w)}{\partial w[m]} &= \sum_n (\sum_i h[i]w[i+n]x[i+n] - g[n])h[m-n]x[m], && (4.4) \\ &(\sum_n k[n]h[m-n])x[m] = s[m]x[m], && (4.5) \end{aligned}$$

if we define the following intermediate signals, $s[m]$ and $k[n]$:

$$\begin{aligned} s[m] &\triangleq \sum_n k[n]h[m-n], \\ k[n] &\triangleq \sum_i h[i]w[i+n]x[i+n] - g[n]. \end{aligned} \tag{4.6}$$

Since both of the intermediate signals s and k are calculated by correlating two signals, they can be evaluated efficiently in the frequency domain using Convolution Theorem as follows:

$$k = \mathcal{F}^{-1}\left\{ \mathcal{F}\left\{ w \odot x \right\} \odot H^* - G \right\}, \tag{4.7}$$

$$s = \mathcal{F}^{-1}\left\{ K \odot H \right\}. \tag{4.8}$$

Using $k[m]$ and $s[m]$, partial derivatives are quite efficiently calculated by element-wise multiplication in the image domain as:

$$\frac{\partial \epsilon(w)}{\partial w[m]} = s[m]x[m] \tag{4.9}$$

Thus far, the derivation of the gradient calculation is obtained for the window signal, which will be searched by the gradient descent procedure described in (4.2) due to the fact that other optimization methods, such as Conjugate Gradient, requires to

---

[2] The proofs can be extended to 2-dimensional signals by adding the second subscript as the second dimension, and valid for 2-dimensional case as well since the corresponding Fourier and inverse Fourier operations are separable for two dimensions.

reformulate our cost function as a linear system and to perform high-dimensional matrix and vector multiplications, where dimensions of the matrices and vectors are proportional to the number of pixels in the object patch. A discussion about our selection of optimization as gradient descent is included in Section 4.6.

4.2 Object Localization Algorithm for Windowing on the Object Observation

In the implementation of correlation filters to the visual tracking problem, at each frame, the translational object motion is determined as the difference between the center of the rectangular region of interest and the location giving the highest correlation score in that region. To calculate this correlation, the object observation is windowed by Hanning window to suppress the object boundaries. In our windowing algorithm, a pre-alignment procedure is applied to the window estimated in the previous frame, since Hanning window is more smooth than the estimated window and can handle small motions.

The overall flow of the proposed method with this pre-alignment procedure is described in Algorithm 1, where $\phi(I_t, \delta, w)$ extracts the object bounding box x_t from the frame I_t at the location δ by using the estimated object size of [26] and outputs $x_t \odot w$, and $T(I, \eta)$ circularly translates the image I by η vector. Since the object is not localized at frame t, the extracted object patch x_t and previously calculated window w_{t-1} might be substantially misaligned due to the motion of the object. To approximately align the object x_t and w_{t-1}, the object is localized beforehand as $\tilde{\delta}_t$ by extracting the object patch using the Hanning window w_h. The motion vector $\tilde{\delta}_t - \delta_{t-1}$ is utilized for adjusting the previously learned spatial window w_{t-1}, *i.e.*, it is circularly translated by this shift. At this point, w_{t-1} is ready to be utilized for the actual location estimation. After finding the final object location δ_t by using the aligned window, the raw object patch x_t is extracted; this patch is used for obtaining the new spatial window.

Algorithm 1 Windowing on the object observation

```
Input: Start a tracker with the position $\delta_1$ and initial size.
    Frames of a video sequence: $I_1, ..., I_N$
    Hanning and rectangular windows: $w^h$ and $w^r$
    Initialize the correlation filter $H_1$ at the first frame.
Output: Bounding boxes $\{\delta_2, ..., \delta_N\}$ $\forall t \in \{2, 3, \cdots, N\}$
    Perform localization and update the correlation filter:
1: for t from 2 to $N$ do
2:     Crop $f_t$ with $\delta_{t-1}$: $f_t = \phi(I_t, \delta_{t-1}, w^h)$
       Extract $d$ features maps $\{f_t^1, ..., f_t^d\}$
3:     Localize the object using:
       $c = \mathcal{F}^{-1}\left\{(\sum_{l=1}^{d}(A_{t-1}^l)^* \odot F_t^l)/(B_{t-1} + \lambda_\epsilon)\right\}$
       $\tilde{\delta}_t = [i^* j^*] = \underset{i,j}{\operatorname{argmax}}\, c(i, j)$
4:     Translate the window $w_{t-1} \leftarrow T(w_{t-1}, \tilde{\delta}_t - \delta_{t-1})$
5:     Crop $f^t$ with $\delta_{t-1}$: $f_t = \phi(I_t, \delta_{t-1}, w_{t-1})$
6:     Repeat step 3 and find the final location as $\delta_t$
7:     Crop $x_t$ with $\delta_t$: $x_t = \phi(I_t, \delta_t, w^r)$
       Extract $d$ features maps $\{x_t^1, ..., x_t^d\}$
8:     Update the model as in (3.19):
       $A_t^l = (1 - \gamma)A_{t-1}^l + \gamma(G_t)^* \odot X_t^l$,
       $B_t = (1 - \gamma)B_{t-1} + \gamma \sum_{k=1}^{d} X_t^k \odot (X_t^k)^*$, ($\gamma$ : 0.025 as in [26])
9:     Reduce $\epsilon(w)$ of Eq.(4.1) using the gradient of $w$ in Eq.(4.2) and assign $w$ to $w_t$: $w_t \leftarrow w$
10: end for
```

4.3 Spatial Windowing Applied to the Filter

Windowing operation can be applied to the correlation filter model h_t instead of its use for the object observation x_t as in (4.1). Similar to (4.1), we also propose learning a window for the correlation filter model as below:

$$\epsilon(w) = \|(w \odot h_t) \circledast x_t - g\|^2, \tag{4.10}$$

The gradient of this cost can be derived analogously with minimal changes in the final gradient formulation.

The cost function in (4.10) can be rewritten as $\epsilon(w) = \sum_n \left(\sum_i h[i]w[i]x[i+n] - g[n] \right)^2$ by summing the squared values of the elements of the difference signal between the correlation output function and the desired response g.

Taking the partial derivative of $\epsilon(w)$ with respect to $w[m]$ gives:

$$\frac{\partial \epsilon(w)}{\partial w[m]} = \sum_n (\sum_i h[i]w[i]x[i+n] - g[n])h[m]x[m+n], \tag{4.11}$$

$$(\sum_n k[n]x[m+n])h[m] = s[m]h[m], \tag{4.12}$$

if the intermediate signals k and s are defined as follows:

$$\begin{aligned} s[m] &\triangleq \sum_n k[n]x[m+n] \\ k[n] &\triangleq \sum_i h[i]w[i]x[i+n] - g[n]. \end{aligned} \tag{4.13}$$

Since both of the intermediate signals s and k are calculated by correlating two signals, they can be evaluated efficiently in the frequency domain using Convolution Theorem as follows:

$$k = \mathcal{F}^{-1} \left\{ X \odot \mathcal{F} \left\{ w \odot h \right\}^* - G \right\} \tag{4.14}$$

$$s = \mathcal{F}^{-1} \left\{ K^* \odot X \right\} \tag{4.15}$$

Using $k[m]$ and $s[m]$, partial derivatives are quite efficiently calculated operating element-wise multiplication in the image domain as:

$$\frac{\partial \epsilon(w)}{\partial w[m]} = s[m]h[m]. \tag{4.16}$$

Although the two costs, (4.1) and (4.10), seem to influence the tracker in the same manner, windowing on the filter as in (4.10) causes the tracker to adapt to the most recent observation x_t without increasing the learning rate of the model. To be more precise, $w \odot h_t$ (the model which is more appropriate for the most recent observation x_t) determines the location of the object, while h_t is slowly updated as in the standard correlation filter based trackers [26, 48] and not directly used for the localization.

This is due to the fact that $w \odot h_t$ outputs a more similar correlation plane to g when correlated with x_t and more appropriate for the most recent object appearance, while h_t represents all of the object observations until the t^{th} frame. Moreover, we can also expand the search range of the tracker (the size of the cropped object patch), since h_t can be perceived as the localized object template in the spatial domain. Thus, we increase the search space to three times as the object size[3].

4.4 Object Localization Algorithm for Windowing on the Filter

Although windowing operation on the object observation in Algorithm 1 requires an alignment stage for the previously learned window, there is no need for such a process in windowing operation on the filter, since the filter is already an aligned template of the object appearance. Hence, there are only two additional steps to be performed compared to the standard correlation filter based tracking algorithms. One of them is the windowing operation on the filter and the other one is the window learning step which reduces the cost in (4.10). All of the necessary steps are given in Algorithm 2.

4.5 Implementation Details

Since the linear correlation filter based trackers [14, 26] are compatible with our cost function, we modified DSST [26] and use it due to its superior performance on VOT2014 [59]. DSST exploits both the image intensities and HOG features and uses an efficient scale search (please refer to [26] or Section 3.2 for details) to estimate the object size. The windowing is performed only for the image intensities. w_h, which is the starting point of the gradient descent, is Hanning window. The number of maximum iterations and the learning rate γ_{SW} are set to 100 and 0.1, respectively. The iterations are stopped when the cost function does not decrease more than 0.001 of the cost value at the previous iteration.

---

[3] In the works [26, 48], the search space is selected as two times larger than the object size.

4.6 Computational Complexity of the Window Learning

Window learning has a computational complexity of $O(P^2 log(P) \times N)$ (N is the number of iterations, and P is the side length of a square object patch). The major overhead is FFT operation throughout the gradient calculation. In our MATLAB implementation, the execution speed is 5 fps in a 3.2 GHz desktop computer.

It should be noted that (4.1) can be rewritten as $||Aw - b||^2$, where the matrix $A = Cdiag(x_t)$. Here, C is the circulant matrix generated from h_t and $diag(x_t)$ is the diagonal matrix with diagonal entries consisting of x_t. There might be infinitely many solutions due to the possible rank deficiency of A (*e.g.*, if x_t or DFT of h_t has at least one zero element.). Yet, an exact solution to the minimization of (4.1) can cause overfitting to the most recent observation. On the other hand, rank deficiency can be avoided by using an $L2$ regularization on w; in this case, the calculation of the closed form solution with that regularization term is costly (leading to prohibitively slow running times) due to the operations on high dimensional matrices. An alternative cost function to (4.1) with a regularizing term can also be formulated as $||Aw - b||^2 + \lambda_{reg}||w - w^h||^2$, where w^h is the Hanning window.

Based on preliminary experiments, no significant performance difference is observed between the reduction of the proposed cost in (4.1) with fixed number of iterations and the minimization of the cost with the aforementioned regularizing term. Therefore, we prefer reducing the proposed cost in (4.1) by a pre-defined fixed number of gradient descent iterations, and mitigate overfitting while achieving high computational efficiency without losing much from performance.

4.7 Experimental Analysis on the Learned Spatial Windows

To demonstrate the effect of the proposed windowing, we compare the proposed window against Hanning window, which is used in the baseline tracker, in terms of PSR (Peak to Sidelobe Ratio) and localization error per frame for *Skating* sequence in Figure 4.2. PSR is computed as $PSR = \frac{C_{max} - \mu_C}{\sigma_C}$ where C_{max}, μ_C and σ_C are maximum, mean and standard deviation of correlation output C, respectively. In almost all of the

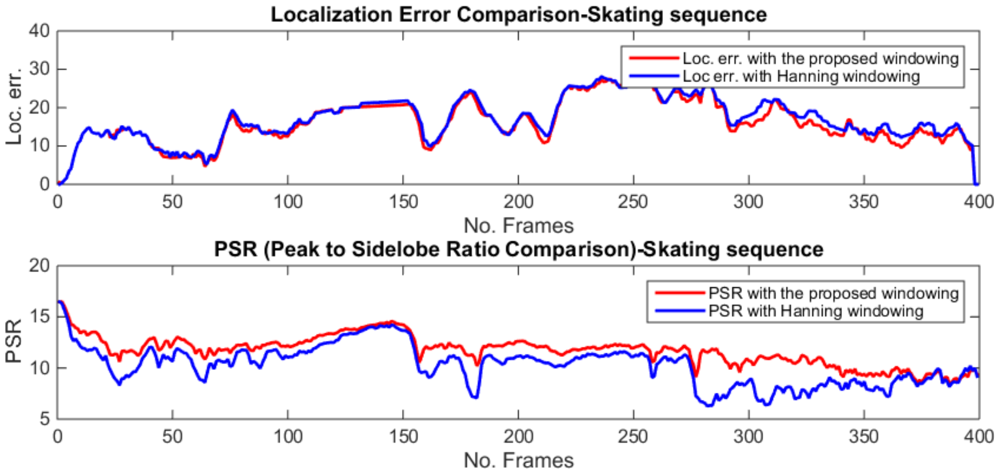


Figure 4.2: Comparison of Peak to Sidelobe Ratio and localization error (Euclidean dist. btw. ground truth and the predicted locations) between the proposed and Hanning windowing. (*Skating* sequence)

frames, the proposed window outperforms Hanning window in terms of PSR values and localization error. This is an experimental evidence that indicates the reduction of the cost function in (4.1) improves the tracking quality (PSR), which also enhances the tracking performance and will be extensively analyzed later in the next chapter.

In Figure 4.3, example observations and their corresponding windows are presented. With a closer look at these observations and their estimated windows, it can be seen that the most valuable parts of the toy tiger are found to be its cheeks and the black part on its nose. In almost all of the frames shown in Figure 4.3, these parts are consistently highlighted by having relatively higher values than the remaining parts. Moreover, the black region on its nose has lower values when it is occluded by the leaves.

Figure 4.4 shows further sample windows extracted by the proposed method. Although the main goal of the proposed windowing technique is to increase the correlation quality, the windows extracted by the proposed algorithm are intuitively meaningful in most cases. In the 1^{st} column, the face is partially occluded by the magazine and the window forms a fictitious eye in the occluded eye region in order to be able to improve a sharp peak. In the 2^{nd} column, the car undergoes an occlusion due to the branches of the tree, where we observe relatively dark values in some parts of these

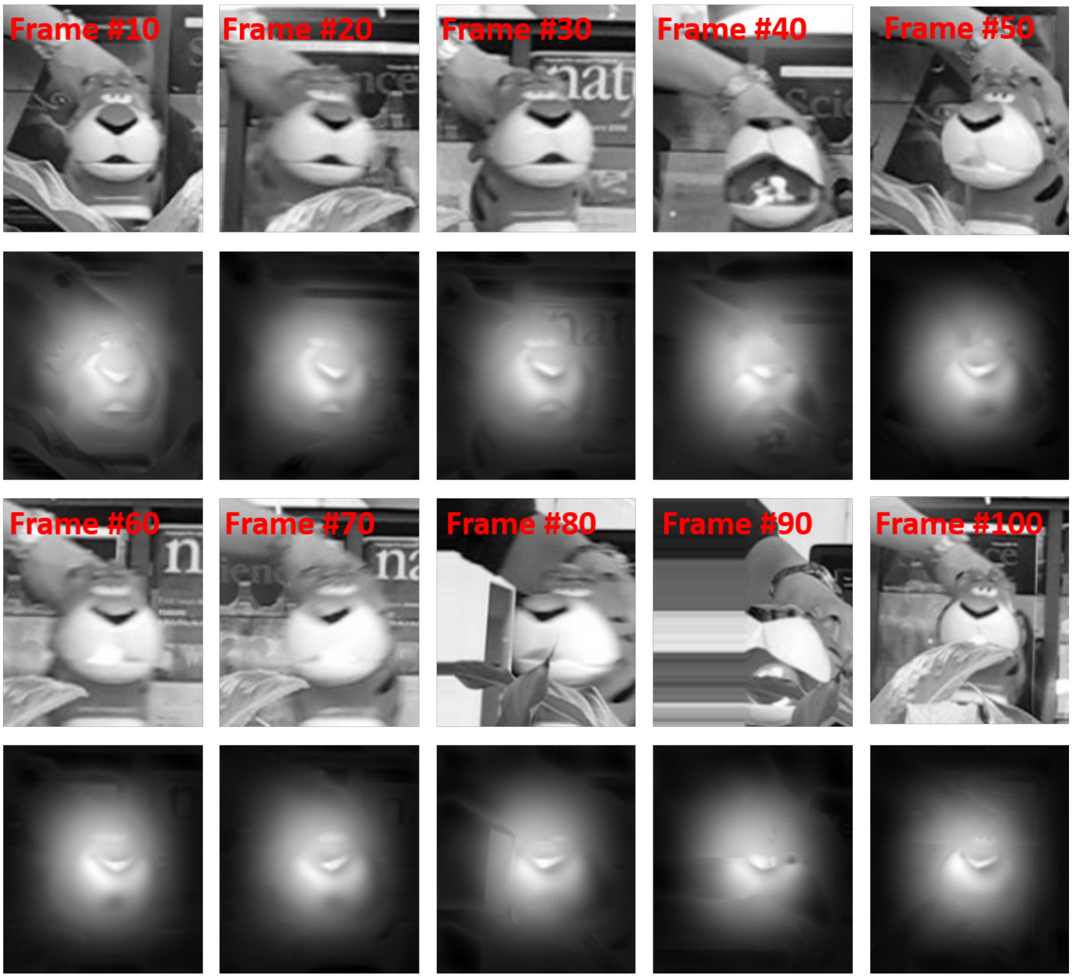


Figure 4.3: Visual examples from *Tiger* sequence. The first and the third rows present the observations, and the second and the fourth rows illustrate the calculated spatial windows.

occluded regions in the window, indicating a suppression of the occluded parts. The window in the 3^{rd} column has higher values in the pants of the woman which probably holds more similarity than the other regions to the corresponding filter and the appearance of this part does not change significantly. The window of the torus hold by the man (the 4^{th} column) has relatively higher values in the object regions of the bounding box, since it is probably the most resembling part to the correlation filter of the object.

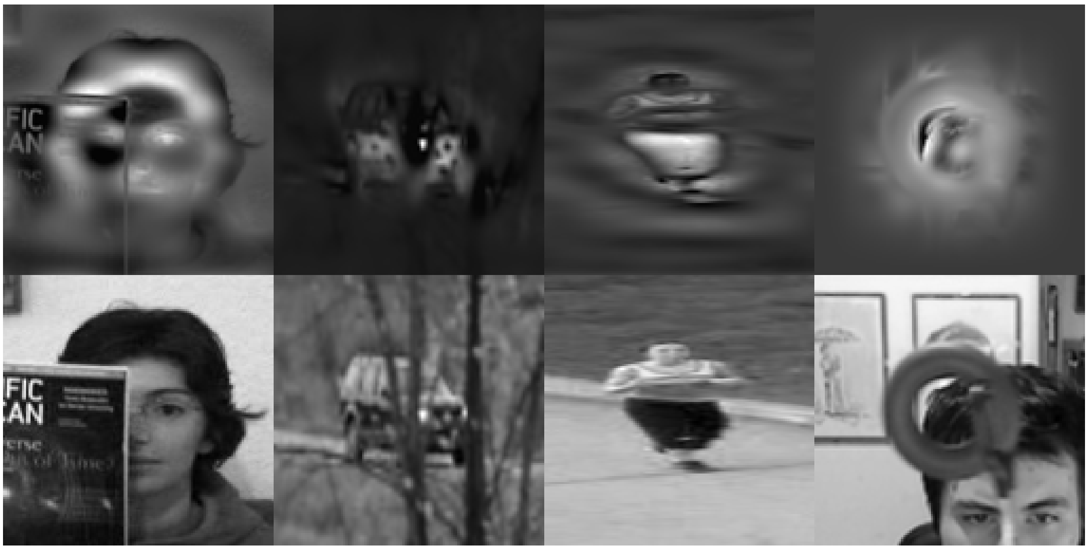


Figure 4.4: Visual examples from the sequences *Face Occluded 1, Car, Jogging, Torus*). 1^{st} **row**: spatial windows extracted by the proposed method. 2^{nd} **row**: corresponding objects.

4.8 Discussions

In this chapter, a novel window learning method is presented to increase the accuracy of a correlation filter based tracking approach. For this purpose, an efficient gradient formulation is provided and applied to visual tracking to suppress or highlight the irrelevant or useful regions of the target object patch. When the utilized features are not invariant enough to the encountered appearance changes of the object, the proposed method is able to increase the correlation quality and the tracking performance due to some important issues. For instance, the features that are not robust against the appearance changes deteriorate the object localization. The learned window is able to suppress those harmful features at the relevant locations. Moreover, the background regions similar to the object patch can also be eliminated due to the proposed cost function that penalizes the degeneration of the correlation response especially in the case of high corelation values in the regions other than the center of the object location. The same argument can also be valid for the case of object occlusion.

Based on the aforementioned discussions and the observed increase in the tracking performance, the proposed method significantly alleviates the adverse effects of some challenges (*e.g.* occlusion, object deformation and background clutter) of visual tracking. In our experiments on VOT2015 dataset, the number of failures are decreased by 6% when the spatial windowing is applied to the object observation. Furthermore, the overlap precision of OTB-2015 dataset (this performance metric will be explained in Section 5.3.4) is improved by 2.5% if the spatial windowing is applied to the correlation filter.

In this chapter, the proposed algorithms serve for improving the individual tracking quality. In the next chapter, an ensemble of trackers framework is presented. Moreover, the improved tracker of this chapter is also integrated into the proposed ensemble tracker. Hence, the tracking performance analysis and the experimental results in the benchmark sequences are explained in detail in Chapter 5.

Algorithm 2 Windowing on the filter

```
Input: Start a tracker with the position $\delta_1$ and initial size.
    Frames of a video sequence: $I_1, ..., I_N$
    Initialize Hanning and rectangular windows: $w^h$ and $w^r$
    Initialize the correlation filter $H_1$ at the first frame.
Output: Bounding boxes $\{\delta_2, ..., \delta_N\}$ $\forall t \in \{2, 3, \cdots, N\}$
    Perform localization and update the correlation filter:
for t from 2 to $N$ do
    Crop $f_t$ with $\delta_{t-1}$: $f_t = \phi(I_t, \delta_{t-1}, w^h)$
    Extract $d$ features maps $\{f_t^1, ..., f_t^d\}$
    Apply the previously learned window on the filter:
    $H^{\hat{l}}_{t-1} = \mathcal{F}\{\mathcal{F}^{-1}\{A^l_{t-1}/(B_{t-1} + \lambda_\epsilon)\} \odot w_{t-1}\}$
    (if $w_{t-1}$ does not exist, use $w_{t-1} = w^h$)
    Localize the object using:
    $c = \mathcal{F}^{-1}\left\{(\sum_{l=1}^{d}(H^{\hat{l}}_{t-1})^* \odot F^l_t)\right\}$
    $\delta_t = [i^* j^*] = \underset{i,j}{\operatorname{argmax}}\, c(i,j)$
    Crop $x_t$ with $\delta_t$: $x_t = \phi(I_t, \delta_t, w^r)$
    Extract $d$ features maps $\{x_t^1, ..., x_t^d\}$
    Update the model as in (3.19):
    $A^l_t = (1-\gamma)A^l_{t-1} + \gamma(G_t)^* \odot X^l_t$,
    $B_t = (1-\gamma)B_{t-1} + \gamma \sum_{k=1}^{d} X^k_t \odot (X^k_t)^*$, ($\gamma$ : 0.025 as in [26])
    Reduce $\epsilon(w)$ of Eq.(4.10) using the gradient of $w$ and assign $w$ to $w_t$: $w_t \leftarrow w$
end for
```

CHAPTER 5

TREE-STRUCTURED ENSEMBLE OF TRACKERS

This chapter presents a tree-structured ensemble of trackers framework, which is a novel combination tracker, and capable of frequently switching in an ensemble of correlation filters. This frequent switching allows the tracker to quickly adapt to various target object appearances, and hence, becoming less prone to drift and occlusion without sacrificing much from the computational efficiency. To be more precise, the proposed technique partitions the space of target object appearances (or equivalently the feature space) by using a binary tree as in [104, 88], where at each node, a separate expert tracker (*i.e.*, a correlation filter based tracker in this study) is employed. Each newly received target instance is propagated from the root to the leaves of this tree by applying a binary search and run only those "relevant" expert trackers independently, and combine their decisions for a final decision. Here, the collection of all of the expert trackers yields the aforementioned ensemble of cardinality $2^{D+1} - 1$ (D is the tree depth) and our algorithm combines only $D + 1$ many ensemble trackers to produce the final tracking decision, which only results in a linear complexity $O(D \times E)$ (E is the complexity for correlation), *i.e.*, it is not $O((2^{D+1} - 1) \times E)$.

Notably, the expert trackers of the ensemble are hierarchically represented over the partitioning tree (from the root to the leaves), each of which is specialized in a certain union of appearance subspaces growing from leaves to the root, *i.e.*, in a certain cluster of appearances. Hence, since each target instance is coupled with certain ($D + 1$ many) tree nodes as well as the corresponding certain expert trackers, we obtain a frequent switching mechanism in the ensemble of correlation filter based trackers during video stream. This switching follows a check over the tree determining which subset of the ensemble is the most specialized to the observed target instance. Then, the final decision is obtained through the combination of only those specialized expert trackers. The result is an indirect, but a quick (almost sudden) and non-linear adaptation to the target with controllable and low computational complexity. In this sense, the learning rate in the proposed tracking scheme is varying (not fixed) that is tuned to the object appearance. This approach addresses the trade-off, where a slow model update causes the model to become quickly outdated, whereas faster updates result in over-learning; unfortunately, one observes a drift in both cases. We emphasize that our strategy is rather radical, since it significantly differs from the literature, where the most popular techniques often attempt to learn appearance changes by iterating the most recently learned single model that is maintained during the sequence. Although combination of trackers idea has been previously exploited in certain studies [120, 54, 100, 118, 127], our tree-structured method is applicable to any kind of visual tracking method without requiring a discriminative or a generative model [118, 127], a training phase [100, 120] or a tracker sampling routine [54].

5.1 Proposed Tree-structured Ensemble Tracking Framework

The proposed ensemble tracking method partitions the target appearance space into subspaces by a binary tree, whose nodes store individual expert trackers specialized in certain target appearances. Each expert tracker is activated when its corresponding subspace contains the most recent observation. In our framework, for only the purpose of determination of the subset of expert trackers to combine at frame $t + 1$, the vectorized object patch x_t (cropped by the object location δ_t from frame t) is sequentially propagated from the root node to the leaves by checking their alignments with the corresponding *separator functions* (details of separator function will be explained next) at the nodes. Therefore, similar x_t's get clustered over time at the nodes in finer details as the depth increases. Since we have a separate expert tracker at each node that is trained by only those x_t's who visited that node, expert trackers then get specialized to target appearances (hence the name "expert"). As a result of this propagation of instances over the tree via separator functions, the complete high dimensional space of target appearances is partitioned in a data adaptive manner.

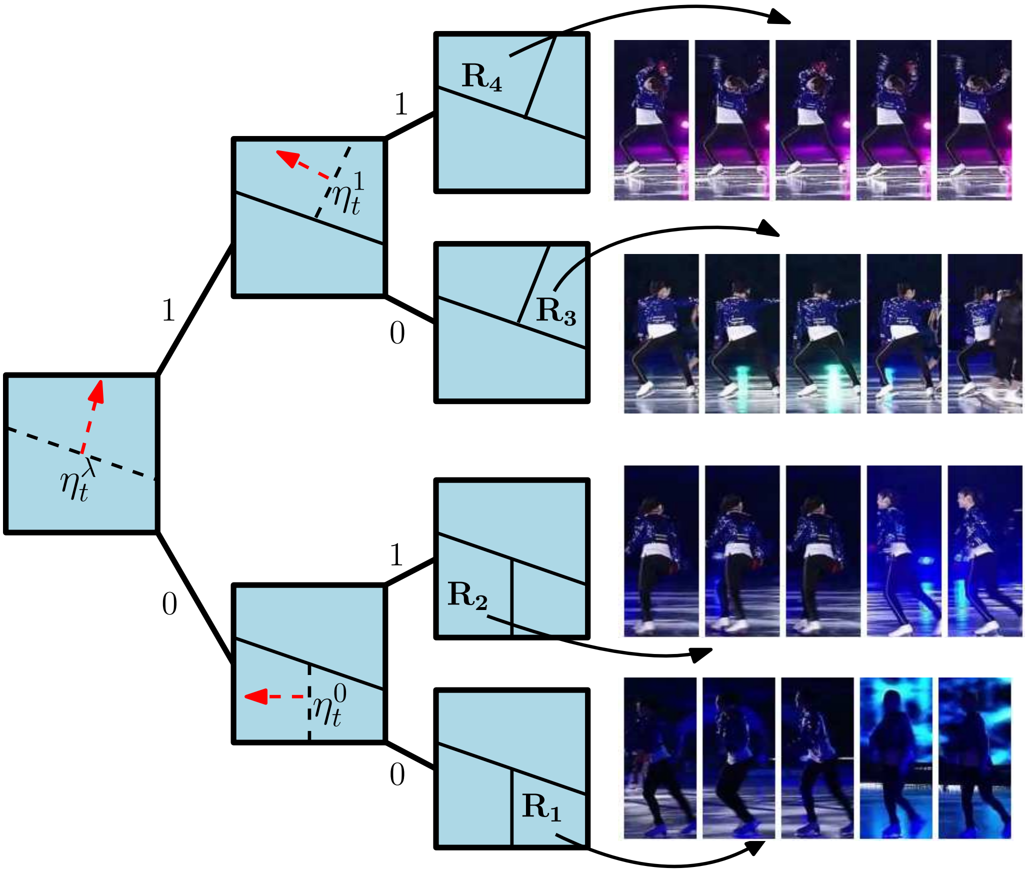


Figure 5.1: Visualization of the complete (no-pruning) tree with depth 2. At each frame; an object appearance x_t is processed through the tree by applying the separator function η_λ^S at each visited node S. Each row of the image matrix shows the object patches dropped to the leaves of the tree ($\{R_1, R_2, R_3, R_4\}$). Note that object poses with similar lightening conditions and appearances are clustered at the same nodes as desired. (*Skating* sequence from VOT2014 dataset [59]).

During initialization of tracking for each sequence, (i) a binary tree of depth D is constructed, (ii) a separate expert tracker T^{init} for each node is initialized. For this purpose, the first observation x_1 (the initial bounding box with raw image intensities) is utilized to extract $P \times P$ feature maps $\{x_1^1, ..., x_1^d\}$. Then, the correlation filters $\{h_1^l\}_{l=1}^d$ are calculated by using (3.18). This is the initial model for T^{init}, and it is set to all expert trackers $T_1^m = T^{init}$ where m ranges the nodes of the tree from 1 to $2^{D+1} - 1$ at the first frame.

After the initialization, we locate which nodes are visited by an object patch x_t as described above by checking its alignments with the corresponding node-specific sepa-

Algorithm 3 Proposed Ensemble Tracker

```
Input: A tracker $T^{init}$ in the first frame using the ground-truth bounding box, $\delta_1$, a tree with a depth of $D$.
    Initialize trackers: $T_1^i \leftarrow T^{init} \quad \forall i \in \{1, \cdots, 2^{D+1} - 1\}$
Output: Bounding boxes $\{\delta_2, ..., \delta_N\}$ $\forall t \in \{2, 3, \cdots, N\}$
    Perform localization and update the trackers:
for t from $1$ to $N - 1$ do
    Process $x_t$ (cropped by $\delta_t$ from frame $t$) through the tree by using the separator functions of the nodes, $\eta_t^{u_k}$'s $\forall k \in \{0, ..., D\}$ and obtain the visited node index set $W_t = \{u_0, u_1, ..., u_D\}$
    for $k$ from $u_0$ to $u_D$ do
        Update the separators $\eta_{t+1}^{u_k}$ by changing the value of $\tau_{t+1}^{u_k}$ to obtain equal sized subspaces using (5.4).
        Run the tracker $T_t^{u_k}$ of the node $u_k$ for the observation $x_{t+1}$ (cropped by $\delta_t$ from frame $t + 1$) with (3.18), and obtain the location estimation.
    end for
    Generate weight vector $w$ using (5.8)
    Combine the results using (5.3)
    Update the expert trackers $\{T_{t+1}^{u_k}\}_{k=0}^{D}$ in $W_t$ as in (3.19)
    Final tracking decision $\delta_{t+1}$ at each frame
end for
```

rator functions via

$$\eta_t^{u_k}(x_t) \triangleq \mathcal{I}\{v_{u_k}^T x_t < \tau_t^{u_k}\}, \tag{5.1}$$

where u_k denotes the selected node index at level k, v_{u_k} is the normal vector to the separating hyperplane (which defines a separator function and it is randomly initialized in the beginning from a normal distribution) at the node u_k of level k, $\tau_t^{u_k}$ is an adaptive threshold (this will be explained in Section 5.2.1) initialized to zero, and $\mathcal{I}\{.\}$ is the indicator function, which outputs 1 when its argument is true, 0 otherwise. If $\eta_t^{u_k}(x_t) = 1$, then the instance x_t travels through the right child of the node u_k; and through the left child, otherwise.[1] Serial application of these separator functions

---

[1] u_{k+1}, which is the next node index in the decision path, is assigned as the selected child according to the output of $\eta_t^{u_k}(x_t)$.

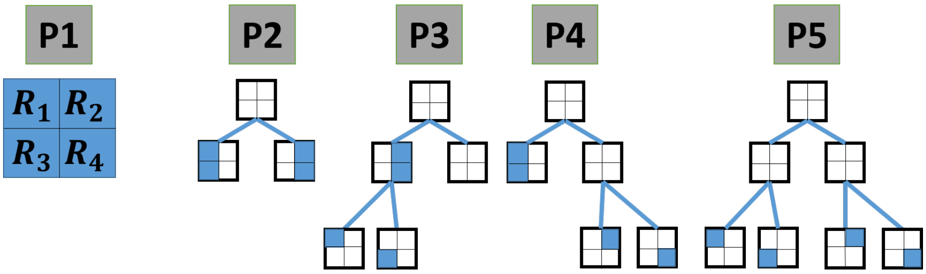


Figure 5.2: All possible partitions $\{P_1, ..., P_5\}$ are illustrated for a depth-2 tree in a 2-dimensional space. The union of the blue regions is the complete space.

to the observation x_t yields the index set of the visited nodes $W_t = \{u_0, u_1, ..., u_D\}$. Hence, the appearance space is partitioned into subspaces, and each of these subspaces corresponds to realizations of a different target appearance. A visualization of this tree with depth two and the resulting subspaces are illustrated in Figure 5.1 as a 2-dimensional space, and the flow of the proposed method is given in Algorithm 3.

For each newly observed object appearance x_{t+1}, an expert tracker is utilized during localization and updated only if it is at one of the nodes visited by x_t. As a result, each expert tracker is specialized for only those target appearances that are aligned with its v_{u_k} vector as well as the vectors in the corresponding branch. The general approach in a single model tracking methodology [26, 48] is to keep record of one tracker and update it at an appropriate pace to stay tuned to the most recent target appearance. On the contrary, our approach is different and novel in the sense that we exploit a dynamic switching mechanism among small subsets of (only $D + 1$ many) expert trackers with respect to the most recent target appearance and combine only those active expert trackers in the selected subset at each time. By this switching, we achieve superior adaptation to the target that is robust to sudden appearance changes and also able to recall the past appearances. Hence, the selection of learning rate is also implicitly performed by the introduced switching.

5.2 Combination of the Expert Tracker Outputs

We now introduce a specific combination over the expert trackers at the visited nodes at each time to produce a final boosted tracking decision. To this end, we first note that a combination over such expert trackers is mathematically and exactly equivalent to another combination over certain “partition trackers”. As it will be clear in this section, each partition tracker is essentially a union of expert trackers and can be accepted as a higher level of specialization for a specific set of target appearances. Our intuition is to learn these appearances during the course of the tracking, register them as partition trackers over our binary partitioning tree and combine them. Hence, the resulting ensemble tracker learns the target appearances in a data adaptive manner starting from simple models to more sophisticated ones. We emphasize that although the number of such partition trackers is doubly exponential (a very large class), since the same exact combination can be obtained over only $D + 1$ many visited expert trackers, our implementation will be computationally highly efficient, *i.e.*, an only linear complexity with the depth of the tree.

We start with observing that each pruning of our tree yields a specific partition of the observation space consisting of the regions at the corresponding leaf nodes (after the pruning). For instance, the complete tree (null-pruning or no-pruning) yields the finest partition with 2^D regions or the root node itself (the complete pruning) yields the complete space as another partition with a single region. Note that the total number of such prunings/partitions is doubly exponential, *i.e.*, $N_{\mathcal{P}} = O((1.5)^{2^D})$ [88].

Let $\mathcal{P} = \{P_i\}_{i=1}^{N_{\mathcal{P}}}$ be the set of all possible partitions. A partition $P_i \in \mathcal{P}$ consists of disjoint regions, *i.e.*, $P_i = \{R_{i1}, R_{i2}, \cdots, R_{iN_{P_i}}\}$, where the regions R_{ik}'s correspond to the leaf nodes of the subtree obtained after the pruning which generates the partition P_i. For example, $\{R1, R2, R3, R4\}$ and $\{R1 \cup R2, R3, R4\}$ are two partitions (out of 5 possibilities) from the depth-2 tree in Figure 5.1. In Figure 5.2, all of the partitions for depth-2 tree is pictured. For any partition at time t, a partition tracker is defined as the tracker that is the union of the expert trackers at the leaf nodes of the subtree that generates the partition we are considering. For instance, if the observation x_t is in R_1 in Figure 5.2, then the partition tracker P_2 matches the output of the expert tracker at the leaf child in the subtree generating the partition P_2 in Figure 5.2, since the left child is responsible for the region $R_1 \cup R_3$ that includes R_1. As a result, for every partition P_i, a partition tracker f_{P_i}[2] is obtained as $f_{P_i}(x_{t+1}) = T_t^{ik}(x_{t+1})$, if $x_t \in R_{ik}$,

---

[2] Tracker decisions are denoted by bounding boxes that are generated by the expert trackers. $T_t^{ik}(x_{t+1})$ is the bounding box decision of the expert tracker T_t^{ik} at the node which is responsible for the region R_{ik} for the unlocalized object observation x_{t+1}.

where $P_i = \{R_{i1}, ..., R_{iN_{P_i}}\}$. The observation x_t can belong to only one region from a partition by definition.

Based on this formulation, we introduce our ensemble tracker as a specific linear combination of partition trackers as [88]

$$f(x_{t+1}) = \sum_{i=1}^{N_{\mathcal{P}}=O((1.5)^{2^D})} \mu_i f_{P_i}(x_{t+1}). \tag{5.2}$$

In this definition, μ_i is the time varying weight[3] of the partition tracker P_i, and defined as $\mu_i = \alpha(i) \exp(\sum_{k=1}^{N_L^i} Q_t^{ik})$. Q_t^{ik}'s are the qualities of the leaf trackers within the partition P_i (defined in (5.5)), N_L^i is the number of leaves in P_i, and $\alpha(i)$, which is an implicit consequence of the weighting strategy (in Section 5.2.3), represents the prior probability of the partition P_i[4]. The proposed ensemble tracker is actually a *mixture of experts* (MOE) [80] algorithm, where the set $\mathcal{P}$ acts as an ensemble of *experts* and therefore, our proposed tracker satisfies the convergence, robustness and optimality MOE-results, if the ground-truth tracked locations were provided after observing every frame as in online learning scenario (cf. [17, 104, 57]). Moreover, the time varying weight values, μ_i, are designed in such a way that by definition, we first favor the simpler partition trackers due to the scarcity of the data in the beginning, but then learn the most sophisticated ones as more data is observed. This is a desired property because the partition trackers are of various complexities, *i.e.*, each of them has different number of parameters to be learned.

The computation of (5.2) requires to run $N_{\mathcal{P}} = O((1.5)^{2^D})$ partition trackers in parallel which is computationally infeasible even for moderate depth values. However we perform an efficient implementation with linear complexity in depth D, *i.e.* requiring to run only $D+1$ expert trackers [104], [88]. First, we note that although there are $N_{\mathcal{P}}$ expert trackers in the combination (5.2), there are only $D + 1 = |\{f_{P_i}(x_{t+1})\}_{i=1}^{N_{\mathcal{P}}}|$ [5] unique possible decisions, which are readily accessible from the tree nodes, since the number of visited nodes is $D+1$ at each time. Therefore, for each newly received observation, this efficient implementation is based on combining only the expert tracker decisions from the root to the leaf on the corresponding branch achieving complexity

---

[3] The dependency on time is dropped in the notation for simplicity

[4] $\alpha(i) = 2^{-(N_L^i + n_{P_i} - 1)}$, where n_{P_i} is the total number of the leaves in P_i that have depth less than the depth of the complete tree (c.f. [57]).

[5] $|\cdot|$ denotes the cardinality of a set.

$O(D \times E)$ instead of $O((1.5)^{2^D} \times E)$ (E is the expert tracker complexity). Therefore, the same exact combination in (5.2) can be effectively written as

$$f(x_{t+1}) = \sum_{k=0}^{D} w_k T_t^{u_k}(x_{t+1}), \tag{5.3}$$

where $W_t = \{u_0, u_1, ..., u_D\}$ is the set of the visited node indices by the instance x_t, $T_t^{u_k}(x_{t+1})$ is the decision of the expert tracker at the corresponding region which is represented by the node u_k. The weight w_k is collected out of the weights μ_i's as $w_k = \sum_{i=1}^{N_{\mathcal{P}}} \mu_i \mathcal{I}\{f_{P_i}(x_{t+1}) = T_t^{u_k}(x_{t+1})\}$. Since each region partitioned by the tree is represented by a particular node, w_k corresponds to the sum of the weights of the partition trackers (this summation is also of complexity $O(D)$), where u_k is a leaf out of $N_{\mathcal{P}} = O((1.5)^{2^D})$ different partitions. In Section 5.2.3, we explain the details of the $O(D)$ implementation of the efficient combination in (5.3), *i.e.*, the calculation of the weights w_k's.

Thus, we achieve the linear complexity $O(D \times E)$ by using the combination in (5.3), instead of naively using the initial definition in (5.2), where both of the definitions generate exactly the same tracking decision for any instance of the target object. From the perspective of (5.3), our technique frequently switches among the subsets of the ensemble of expert trackers depending on which subset is the most specialized in the target appearance; and combines the decisions of the expert trackers in the selected subset to produce the final tracking decision.

5.2.1 Tree Balancing

The goal of the proposed tree-structured ensemble tracking is to represent different object appearances at the nodes of the tree. Hence, we need a balanced partitioning of the object appearances. In order to achieve a balanced partitioning in the observation space, we update the threshold $\tau_t^{u_k}$ value of all of the visited separator functions $\eta_t^{u_k}(\cdot)$ at the nodes u_k by using

$$\tau_{t+1}^{u_k} \leftarrow (\tau_t^{u_k} N_t^{u_k} + v_{u_k}^T x_t)/(N_t^{u_k} + 1), \tag{5.4}$$

where $N_t^{u_k}$ corresponds to the number of visits to the node u_k until the t^{th} frame. Namely, the update in Eq. (5.4) is the cumulative moving average, which approximately causes half of the observations $\{x_t\}_{t=1}^{N_{length}}$ to produce $v_{u_k}^T x_t < \tau_t^{u_k}$, and the other half to produce $v_{u_k}^T x_t > \tau_t^{u_k}$ at the node u_k.

5.2.2 Quality Metric of Expert Trackers

In order to measure the tracker quality, Peak-to-Sidelobe-Ratio (PSR) is computed as $PSR = \frac{C_{max}-\mu_C}{\sigma_C}$ where C_{max}, μ_C and σ_C are maximum, mean and standard deviation of correlation output C, respectively. The PSR values of the trackers are modified for the qualities of the expert trackers. This metric represents the confidence of the estimated location, *i.e.*, sharper response is more confident. To control the tracking quality and prevent overflow, a quality decay parameter is added to PSR metric; and also an effective aging value is employed to speed up the convergence towards the leaves to obtain

$$Q_t^{u_k} = \frac{PSR_t^{u_k}}{\kappa} \exp\left(-\frac{N_t^{u_k}}{\lambda}\right). \tag{5.5}$$

In this equation, $Q_t^{u_k}$ is the proposed tracker quality metric of the node u_k at time t, and λ and κ are the effective aging threshold and quality decay parameters. This metric controls the tracking quality of the expert trackers, and emphasizes the closer nodes to the leaves as time passes.

5.2.3 Calculation of the Weights w_k's

The efficient way of calculating the weights w_k's is described below.[6] For the related proof, one should refer to the prediction study in [57] and modify the node probabilities with the tracking quality $\exp(Q^{u_k})$. For this purpose, the global qualities (g_k's) of the expert trackers are first calculated at each node, u_k, in $W = \{u_0, ..., u_D\}$ through the back-recursion (from the leaves to the root)

$$g_k = \frac{1}{2}\exp(Q^{u_k}) + \frac{1}{2} g_k^{left} g_k^{right}, \tag{5.6}$$

where Q^{u_k} is the local tracking quality of the node u_k in W, g_k^{left} and g_k^{right} are the global qualities of the left and right children of the node u_k. By the following

---

[6] For clarity, the time subscript is dropped in this subsection.

recursion, auxiliary qualities σ_k's are first calculated for each node u_k [57]

$$\sigma_k = \frac{1}{2} g_k^{sibling} \sigma_{k-1}, \tag{5.7}$$

where $g_k^{sibling}$ refers to the previously calculated global quality of the sibling of the node u_k. Finally, we obtain the combination weights as:

$$w_k = \exp(Q^{u_k}) \sigma_k. \tag{5.8}$$

We here presented a recursion to efficiently obtain the weight w_k and this recursion calculates $w_k = \sum_{i=1}^{N_{\mathcal{P}}} \mu_i \mathcal{I}\{f_{P_i}(x_{t+1}) = T_t^{u_k}(x_{t+1})\}$, proven in [57] to be the efficient way of calculating the final combination result in (5.2).

5.3 Performance Evaluation of the Proposed Spatial Windowing and Ensemble of Trackers

In this section, extensive experiments on benchmark datasets are reported for both the proposed spatial windowing method and the proposed ensemble tracker. First, implementation details of the proposed ensemble tracker is explained. Second, experimental results of the ensemble tracker is presented for VOT2014 [59] dataset. Then, the tracking performances of the proposed tracker based on spatial windowing and the ensemble tracker along with their combination are analyzed for VOT2015 [35] dataset. Finally, the experimental results of OTB-2015 [112] dataset are presented for all of the proposed tracker configurations, which are compared against the state-of-the-art visual trackers.

5.3.1 Implementation Details and Computational Complexity

For the proposed tree-structured ensemble tracker, DSST [26] is used as the expert tracker, and the parameters are set as in [26], except for the learning rate[7]. Since the number of visits to a node decreases exponentially with the depth level l with 2^{-l}, we set the learning rate of the tracker at the level-l as $\gamma \times 2^l$. In the quality calculation

---

[7] Note that the proposed framework can work with any tracker of the literature as the expert tracker.

of an expert tracker, κ value is set to the average qualities of all the expert trackers at each frame for stability. Aging parameter λ is set to 100. The initial patch size is twice as large as the initial ground truth object size as in [26]. For the proposed ensemble tracker, we use a constant tree depth of 3 throughout the evaluation on VOT2014 and VOT2015 datasets, since the performance saturates at higher depth levels and lower depth levels leave an improvement gap for the moving objects in the tested sequences. We study varying depth values on OTB-2015, cf. Section 5.3.4.

Implementation details of the proposed spatial windowing method are presented in the previous chapter. The performance evaluation of the spatial windowing method (both on the object observation and the correlation filter) are reported in this chapter, because the proposed ensemble tracker and spatial windowing method have been combined and the tracking performance of this combined tracker is analyzed here.

Notably, the windowing applied to the correlation filter is analogous to the one applied to the object observation. Yet, the latter one enables the search range to be larger, since the correlation filter can be perceived as the localized object template and it is restricted by the proposed windowing operation. Namely, the tracker jumps can be avoided in the case of a larger search range in the object observation and a correlation filter regularized by the learned window. Thus, windowing on the correlation filter has favorable performance against windowing on the object observation. In our preliminary experiments, the tracking accuracy of the integration of the tracker that operates windowing on the correlation filter into the proposed ensemble method was not observed to be much different from the one for the object observation. Hence, only the integration of the windowing on the object observation into the proposed ensemble tracker is tested for the combination of those two approaches.

5.3.2 Experimental Results of the Proposed Ensemble Tracker on VOT2014 Dataset

5.3.2.1 Quantitative Results

In our comparisons, we use the performance metrics of VOT 2014 [59], *i.e.*, the average accuracy and robustness scores. For a predicted object region and its ground

Table5.1: Average accuracy results per sequence. *, ** and *** indicate first, second and the third rankings of the methods. Red means the proposed method has an accuracy better than or equal to the baseline tracker DSST.

	Proposed	DSST [26]	CT [121]	DGT [16]	FRT [1]	IVT [91]	KCF [48]	MIL [3]	SAMF [69]	Struck [45]
ball	0,51	0,45	0,39	0,81*	0,69	0,33	0,76***	0,46	0,78**	0,57
basketball	0,64**	0,58	0,54	0,5	0,43	0,43	0,64**	0,61***	0,75*	0,61***
bicycle	0,63**	0,61	0,56	0,63**	0,5	0,72*	0,63**	0,54	0,62***	0,42
bolt	0,56*	0,54**	0,42	0,49	0,43	0,4	0,49	0,51	0,56*	0,53***
car	0,76*	0,74**	0,37	0,57	0,42	0,65	0,71***	0,42	0,51	0,43
david	0,8***	0,81**	0,4	0,53	0,46	0,69	0,82*	0,5	0,82*	0,6
diving	0,41**	0,44*	0,24	0,34***	0,3	0,23	0,25	0,24	0,25	0,28
drunk	0,59**	0,55	0,48	0,67*	0,44	0,51	0,54	0,45	0,57**	0,5
fernando	0,4	0,38	0,39	0,61*	0,34	0,39	0,41***	0,46**	0,39	0,38
fish1	0,45***	0,35	0,36	0,56*	0,41	0,26	0,42	0,4	0,5**	0,35
fish2	0,31**	0,31**	0,21	0,48*	0,28	0,2	0,27	0,22	0,3***	0,21
gymnastics	0,64*	0,56***	0,48	0,58**	0,52	0,56	0,54	0,26	0,54	0,49
hand1	0,27	0,5	0,32	0,63*	0,41	0,29	0,56**	0,43	0,55***	0,35
hand2	0,44	0,52*	0,2	0,52*	0,39	0,34	0,5**	0,4	0,46***	0,3
jogging	0,79***	0,79***	0,77	0,66	0,64	0,72	0,8**	0,2	0,82*	0,77
motocross	0,45**	0,36	0,22	0,49*	0,18	0,25	0,37	0,22	0,4***	0,26
polarbear	0,47	0,51	0,6	0,81*	0,63	0,45	0,78**	0,46	0,71***	0,62
skating	0,61**	0,59***	0,51	0,39	0,52	0,56	0,68*	0,25	0,45	0,52
sphere	0,92*	0,92*	0,61	0,85	0,66	0,38	0,9**	0,57	0,88***	0,7
sunshade	0,78*	0,77**	0,41	0,52	0,47	0,76	0,76***	0,43	0,76***	0,78*
surfing	0,82**	0,82**	0,66	0,64	0,78	0,69	0,8***	0,38	0,8***	0,91*
torus	0,85**	0,82	0,55	0,83	0,58	0,7	0,86*	0,43	0,84***	0,51
trellis	0,82**	0,81***	0,33	0,48	0,53	0,54	0,8	0,42	0,83*	0,53
tunnel	0,75**	0,79*	0,2	0,44	0,41	0,3	0,69***	0,33	0,55	0,32
woman	0,8*	0,77**	0,57	0,54	0,68	0,47	0,74	0,26	0,76**	0,75
# of top three ranks	20	15	0	12	0	0	16	2	20	4
# of failures	32***	29**	78	25*	83	69	33	57	32	54

truth at frame t, accuracy is defined as $s_t = \frac{area(R_P \cap R_G)}{area(R_P \cup R_G)}$, where R_P and R_G are the predicted and groundtruth object regions, respectively. Average accuracy per sequence is calculated by averaging these accuracy scores over time. If a tracker fails, *i.e.*, accuracy score decreases to zero, then the tracker is re-initialized (please refer to VOT2014 challenge paper [59] for further details). The other metric, *i.e.*, robustness, measures the number of failures per frame. Once the tracking results are obtained, they are ranked according to one of these two performance metrics; and the trackers with statistically insignificant results are merged. This case is denoted as *ranking measure*. We report the total number of failures at the last row of the Table 5.1 for each method.

Table 5.1 reports the average accuracies of the top performing trackers on VOT2014 challenge per sequence, which first demonstrate that our ensemble tracker is competitive with respect to other methods while often being (in 20 out of 25 sequences) superior. Moreover, the proposed method outperforms the single DSST in most of the sequences when DSST does not fail, which proves the efficacy of the introduced model switching approach. Hence, we indeed boost the performance of DSST [26], which is a winner of VOT 2014. On the other hand, in terms of the number of failures, the proposed ensemble tracker and the DSST perform similarly, which is intuitive, since DSST is employed as an expert tracker in this thesis (cf. the last row in Table 5.1).

In order to better illustrate the significance of the proposed weighting strategy described in Section 5.2.3, we also run the proposed ensemble tracker by using equal weights for active trackers at each frame, *i.e.*, $w_i = 1/(D+1)$. In this setting, this naive weighting performs significantly worse than the proposed weighting method. The average accuracies per sequence are shown in Table 5.2.

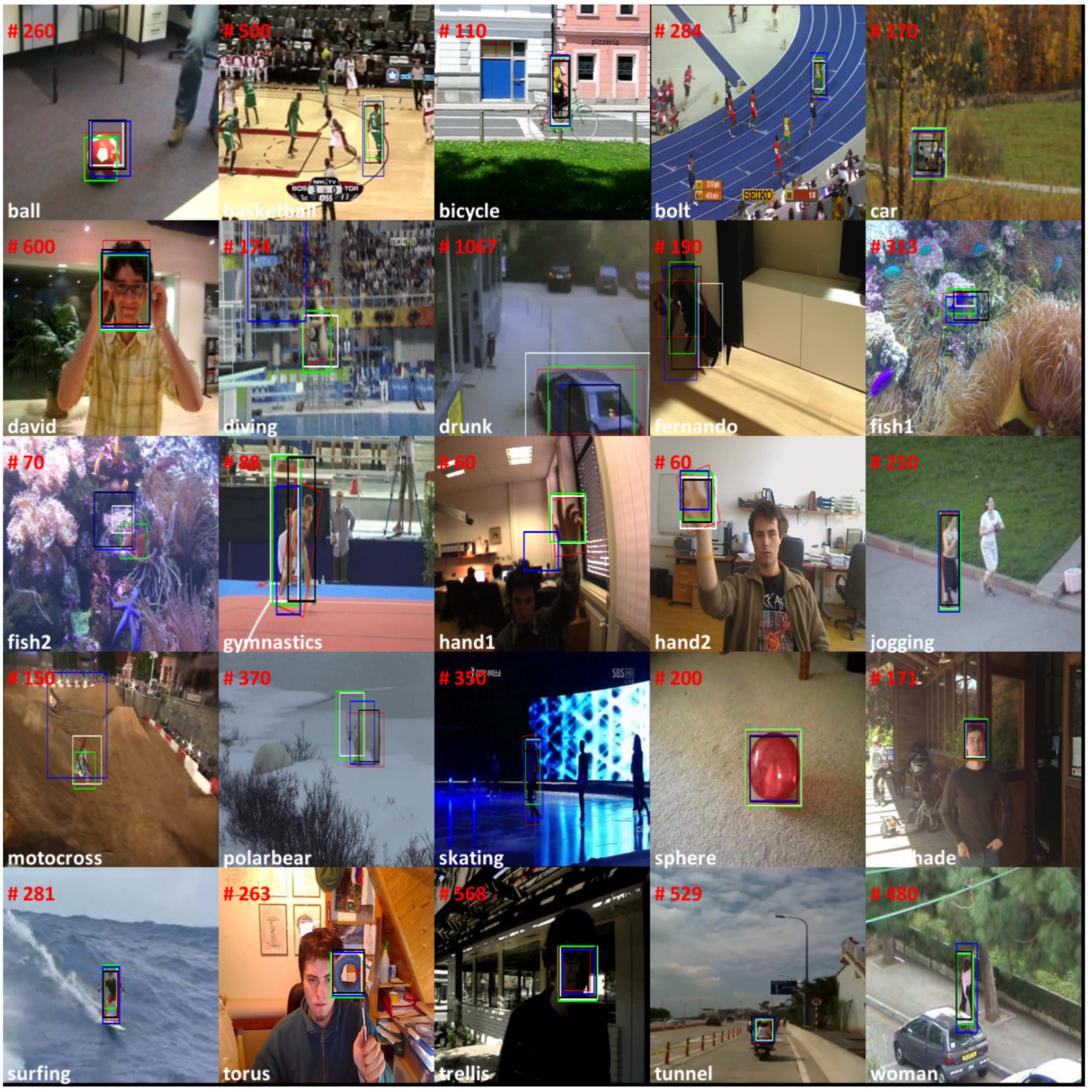


Figure 5.3: Example frames showing the tracking of the correlation filter based algorithms. Red: Ground truth, White: DSST [26], Black: SAMF [69], Blue: KCF [48], Green: Proposed Method. The name of each sequence is on the bottom left corner of the example patches.

Table5.2: Comparison of our proposed weighting strategy and naive weighting. Gray cells indicate better performance.

	ball	basketball	bicycle	bolt	car	david	diving	drunk	fernando	fish1	fish2	gymnastics	hand1	hand2	jogging	motocross	polarbear	skating	sphere	sunshade	surfing	torus	trellis	tunnel	woman	mean
Equal weighting	0,61	0,66	0,62	0,55	0,75	0,75	0,33	0,58	0,42	0,41	0,31	0,55	0,34	0,37	0,79	0,42	0,40	0,60	0,92	0,77	0,85	0,84	0,81	0,66	0,75	0,60
Our weighting strategy	0,51	0,64	0,63	0,56	0,76	0,8	0,41	0,59	0,4	0,45	0,31	0,64	0,27	0,44	0,79	0,45	0,47	0,61	0,92	0,78	0,82	0,85	0,82	0,75	0,8	0,62

Table 5.3 reports the average performances of the compared trackers per attribute. These attributes are camera motion, illumination change, occlusion, size change and motion change. Among the compared methods, we achieve the best accuracy ranking. The proposed ensemble tracker again outperforms the baseline tracker DSST in average per attribute. Table 5.4 shows the comparison of correlation filter based trackers in terms of average accuracy calculated by weighting all the frames equally. The proposed ensemble maintains its superior performance against DSST. Figure 5.4 visualize the success rate plot of the proposed method and other correlation filter based tracking methods. Success rate indicates the ratio of number of successfully tracked frames over the total number of frames. Successfully tracking is defined as having an accuracy score below a certain threshold, which is varied to plot these performance curves.

Table5.3: Average accuracy ranking results calculated using per attribute. Red, blue and green indicate the best, second and third ranking.

	camera_motion		**illum_change**		**occlusion**		**size_change**		**motion_change**		**Overall**	
	A-Rank	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**
Proposed	3.00	0.66	2.00	0.75	2.50	0.67	3.00	0.55	3.00	0.65	2.75	0.65
DSST	3.50	0.64	2.00	0.75	3.00	0.62	4.50	0.54	3.50	0.65	3.42	0.63
CT	9.00	0.43	9.50	0.38	8.50	0.43	8.67	0.36	8.67	0.42	8.64	0.42
DGT	4.00	0.56	6.50	0.47	6.50	0.48	2.50	0.58	3.50	0.58	4.00	0.57
FRT	6.50	0.50	6.50	0.48	6.00	0.52	7.50	0.42	7.00	0.48	7.08	0.48
IVT	9.00	0.47	6.50	0.57	8.50	0.40	8.00	0.41	8.00	0.49	7.67	0.48
KCF	2.00	0.67	2.50	0.75	3.00	0.65	2.50	0.58	2.00	0.68	2.83	0.66
MIL	9.00	0.41	9.50	0.38	10.00	0.28	8.00	0.37	8.67	0.41	9.11	0.40
SAMF	2.50	0.66	3.50	0.67	3.00	0.61	2.50	0.57	3.00	0.67	3.00	0.65
Struck	6.50	0.53	6.50	0.51	4.00	0.58	7.50	0.41	7.00	0.51	6.33	0.51

Table5.4: Average accuracy of the proposed and compared correlation filter based methods over successfully tracked frames.

Methods	Proposed	DSST [26]	SAMF [69]	KCF [48]
Accuracy %	64.3	62.8	64.7	65.5

The proposed ensemble method performs better than the baseline algorithm in terms of accuracy in 75% of the sequences (please see Table 5.1). In the rest of the sequences, the drawback is the relatively too short sequence lengths and the resulting slow rate of convergence. Note that we combine many expert trackers and learning the combination weights naturally need more frames, which we aim to avoid by using a more sophisticated clustering in the leaves of the tree in a future study. Such a clustering idea will also reduce the number of failures. Since especially the sequences *diving, hand1, hand2* are short sequences ($\sim$250 frames), the convergence is not well achieved. The relation between the performance of the proposed method and the sequence length is illustrated in Figure 5.5. As the sequence length increases, the proposed method gains more robustness than the expert tracker. When the accuracy of the baseline algorithm is greater than half percent, we achieve greater or equal performance compared to the baseline as expected. Hence, the superior performance of the ensemble of trackers is achieved when the baseline tracker performs moderately well.

5.3.2.2 Qualitative Results

A clear visual result is demonstrated in Figure 5.1 to indicate different appearances of the target. To create the object patches, the tree depth is set to 2 and the proposed ensemble tracker is run for *skating* sequence. Each target object is saved to the leaves of the tree throughout the frames and it is observed that similar patches corresponding to similar appearance of the object are sent to the same leaves of the tree as expected. This helps individual tracker specialize better for particular sets of appearances.

Some visual tracking results are illustrated in Figure 5.3 to compare the proposed

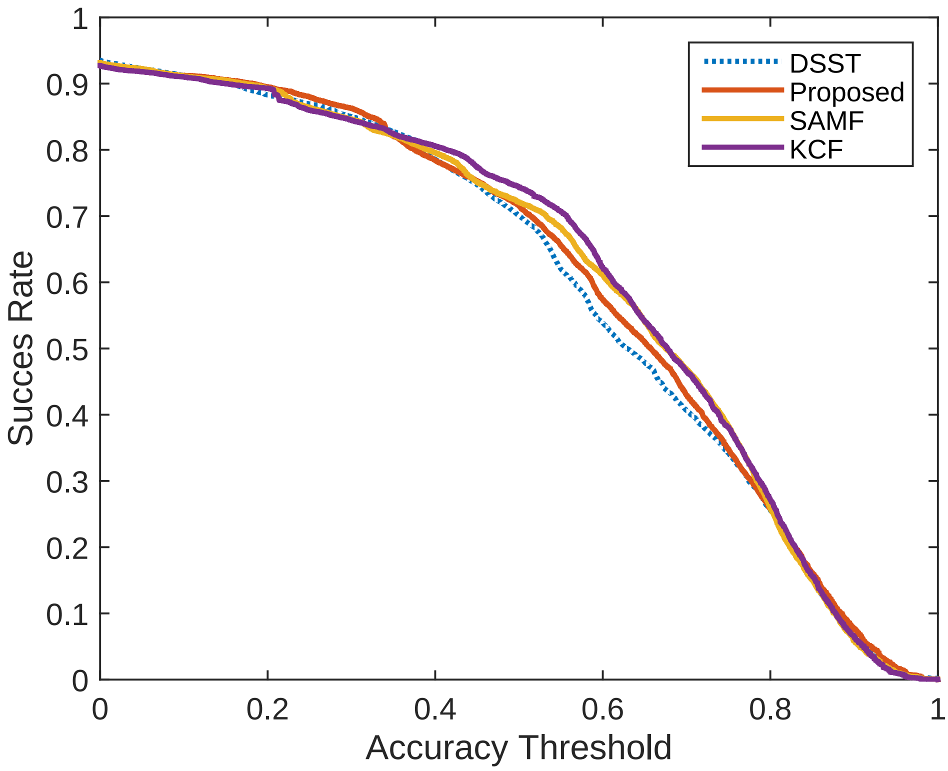


Figure 5.4: Average success rate plot of the proposed and the compared correlation filter based method. **X-axis** shows the accuracy threshold and **Y-axis** indicates the ratio of # of successfully tracked frames over the total # of frames.

method and the correlation filter based trackers in the VOT2014 sequences. As clearly demonstrated, objects are localized better compared to the expert tracker DSST as well as the others. For instance, the appearance of the car changes drastically throughout the video frames in the *drunk* sequence (the second row and the third column in Figure 5.3). Hence, a tracking algorithm which continuously update its model tends to fail in such a case. Nevertheless, since the proposed ensemble method registers various object poses and appearances in the tree nodes and accordingly partition the appearance space, better localization is obtained through the specialized expert trackers.

In the *basketball* sequence, there are many different poses of the basketball player and he makes movements like out-of-plane rotation, which contradicts the abilities of correlation based trackers. Nevertheless, our method localizes the player better than the other correlation filter based methods. Finally, the sequence *fernando* includes

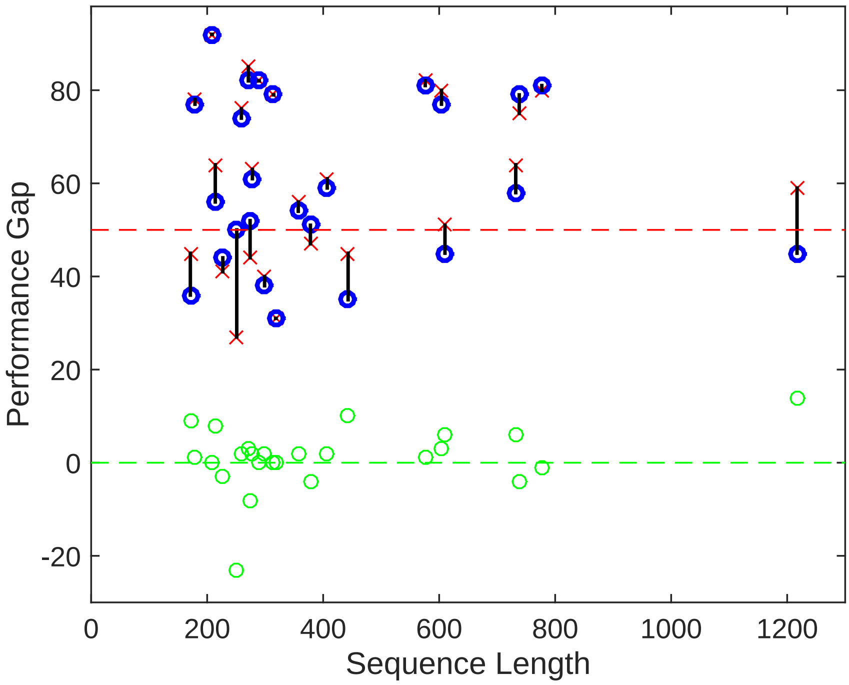


Figure 5.5: For green circles, **y-axis**: the average accuracy difference between the proposed method and the baseline tracker and **x-axis**: the length of the sequences on VOT2014 dataset. Blue and red markers indicate the performance of the baseline tracker and the proposed method respectively. Horizontal red line separates the sequences as below or above 50% of average accuracy.

severe occlusion, where the proposed technique achieve superior tracking.

5.3.3 Experimental Results of the Proposed Ensemble Tracker and Spatial Windowing Based Tracker on VOT2015 Dataset

We test the performance of the proposed tracker combinations and their state-of-the-art counterparts on VOT2015 [35] dataset including 60 video sequences with various attributes, such as illumination change, motion change, occlusion, size change. The utilized performance metrics are the same as the metrics of VOT2014 dataset.

5.3.3.1 Performance Evaluation

We evaluate our four different trackers; namely SW_Filter, SW_Obs, Ensemble and Ensemble_SW_Obs. Among these trackers, SW_Filter (described in Section 4.3 and in Algorithm 2) is the tracker with spatial windowing applied to the correlation filter while SW_Obs (described in Section 4.1 and in Algorithm 1) performs windowing on the object observation. Ensemble (described in Section 5.1 and in Algorithm 3) is our tree-structured ensemble tracker, which integrates DSST [26] as the expert tracker. Finally, Ensemble_SW_Obs is again our ensemble tracker, yet the expert tracker is SW_Obs.

We compare our tracker combinations with the following state-of-the-art methods: KCF2 [48], KCFDP [53], MTSA-KCF [10], sKCF [98], DSST [26], MEEM [120], MUSTER [52], SME [67], TRIC [109]. Among the compared correlation filter based trackers with a single expert; KCF-MTSA is a multi-template and scale adaptive extension of the original KCF work in [48], KCF2 is the Visual Object Tracking Committee implementation of KCF with improved scale support as well as sub-cell peak estimation. KCFDP helps to detect better aspect ratios for different scales. sKCF improves KCF by allowing an adjustable Gaussian window. DSST is the best performing algorithm in the VOT2014 challenge [59]. Among the compared trackers with multiple experts, TRIC is a part-based tracker; where each part location is determined by a regressor model, MUSTER exploits a short-term tracker (a correlation filter based tracker) and a long term tracker (based on key-point matching), SME and MEEM are the trackers with multiple experts, where each expert is a correlation filter based tracker and an SVM based tracker, respectively.

5.3.3.2 Evaluation with respect to the baseline

Table 5.5 shows the per-sequence average of the results. According to these results, Ensemble_SW_Obs has the first ranking in terms of the sum of average accuracy and robustness rankings, while achieving less number of failures without sacrificing the tracking accuracy. Moreover, our remaining tracker configurations outperform DSST. Table 5.6 reports the per-attribute analysis of our compared trackers where the aver-

Table5.5: VOT2015 [35] per sequence experimental results for our proposed tracker configurations and DSST [26]. Acc., Rob. and Rank. mean accuracy, robustness and ranking, respectively. The definitions of these metrics are included in Section 5.3.2.1.

Methods	**Acc.**	**Failures**	**Acc. Rank.**	**Rob. Rank.**	**Avg. Rank.**
Ensemble_SW_Obs	.52	1.97	1.58	1.55	1.57
SW_Filter	.50	2.53	1.70	2.0	1.85
SW_Obs	.53	2.41	1.23	2.02	1.63
Ensemble	.51	2.56	1.82	2.03	1.93
DSST [26]	.54	2.56	1.35	2.32	1.83

Table5.6: VOT2015 [35] per attribute experimental results for our proposed tracker configurations and DSST [26].

	Acc. Rank.	**Rob. Rank.**	**Avg. Rank.**
Ensemble_SW_Obs	1.00	1.00	1.00
SW_Filter	1.00	1.33	1.17
SW_Obs	1.00	1.33	1.17
Ensemble	1.00	1.67	1.33
DSST	1.17	2.17	1.67

aging is applied per-attribute. Our proposed trackers also perform favorably against DSST for this evaluation. Analyzing this table, VOT Toolkit finds no significant difference between our trackers in terms of accuracy while discriminating them clearly in terms of robustness ranking. This result helps to make comparisons between the configuration of trackers conveniently by only looking at the robustness values.

When total number of failures is concerned (the last column of Table 5.8), applying the spatial windowing on the correlation filter causes to have less number of failures than applying the window on the object observation (5.4% robustness increase) since applying the window on the filter allows us to increase the search space. This situation can also be observed from Figure 5.6, where SW_Filter is closer to the top right corner than SW_Obs.

Table5.7: VOT2015 [35] accuracy results for our proposed tracker configurations and the compared trackers. Red, blue and green indicate first, second and third rankings, respectively.

	label camera motion		label empty		label illum. change		label motion change		label occlusion		label size change		Mean		Weighted mean		Pooled	
	A-Rank	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**	**A-Rank**	**Overlap**
SW_Filter	1.00	0.53	1.00	0.56	2.00	0.61	1.00	0.43	1.00	0.41	1.00	0.46	1.17	0.50	1.17	0.50	1.00	0.51
DSST	1.00	0.56	1.00	0.61	1.00	0.67	1.00	0.49	8.00	0.38	1.00	0.52	2.17	0.54	2.17	0.54	1.00	0.55
Ensemble_SW_Obs	1.00	0.54	1.00	0.57	1.00	0.65	1.00	0.46	1.00	0.43	1.00	0.49	1.00	0.52	1.00	0.52	1.00	0.53
SW_Obs	1.00	0.54	1.00	0.60	1.00	0.68	1.00	0.49	1.00	0.41	1.00	0.49	1.00	0.54	1.00	0.53	1.00	0.54
Ensemble	1.00	0.53	1.00	0.56	1.00	0.67	1.00	0.47	7.00	0.37	1.00	0.47	2.00	0.51	2.00	0.51	1.00	0.51
MEEM [120]	1.00	0.49	1.00	0.58	9.00	0.48	1.00	0.47	1.00	0.47	9.00	0.36	3.67	0.47	3.67	0.48	1.00	0.50
KCF2	1.00	0.49	1.00	0.54	8.00	0.49	1.00	0.46	1.00	0.47	9.00	0.37	3.50	0.47	3.50	0.48	1.00	0.49
kcfdp [53]	1.00	0.48	1.00	0.56	1.00	0.70	1.00	0.46	1.00	0.40	1.00	0.47	1.00	0.51	1.00	0.50	1.00	0.50
kcf_mtsa [10]	7.00	0.48	1.00	0.57	8.00	0.54	5.00	0.43	1.00	0.44	5.00	0.40	4.50	0.48	4.50	0.48	1.00	0.49
sKCF [98]	1.00	0.49	1.00	0.57	8.00	0.48	1.00	0.42	1.00	0.46	8.00	0.37	3.33	0.47	3.33	0.47	1.00	0.49
muster [52]	1.00	0.54	1.00	0.58	8.00	0.50	1.00	0.48	1.00	0.42	1.00	0.45	2.17	0.50	2.17	0.51	1.00	0.52
tric [109]	7.00	0.45	1.00	0.54	9.00	0.45	1.00	0.43	1.00	0.45	9.00	0.33	4.67	0.44	4.67	0.45	1.00	0.46
SME [67]	1.00	0.56	1.00	0.60	1.00	0.69	1.00	0.51	1.00	0.45	1.00	0.52	1.00	0.56	1.00	0.55	1.00	0.56

Table5.8: VOT2015 [35] robustness results for our proposed tracker configurations and the compared trackers. Red, blue and green indicate first, second and third rankings, respectively.

	label camera motion		label empty		label illum. change		label motion change		label occlusion		label size change		Mean		Weighted mean		Pooled	
	R-Rank	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**	**R-Rank**	**Failures**
SW_Filter	2.00	64.00	1.00	31.00	3.00	7.00	3.00	50.00	2.00	25.00	2.00	31.00	2.17	34.67	2.17	42.46	2.00	141.00
DSST	2.00	69.00	4.00	37.00	3.00	7.00	6.00	61.00	12.00	28.00	2.00	33.00	4.83	39.17	4.83	47.88	9.00	163.00
Ensemble_SW_Obs	2.00	64.00	1.00	25.00	1.00	6.00	2.00	48.00	3.00	21.00	1.00	32.00	1.67	32.67	1.67	40.43	1.00	132.00
SW_Obs	2.00	65.00	4.00	30.00	3.00	9.00	2.00	50.00	2.00	23.00	2.00	36.00	2.50	35.50	2.50	43.26	6.00	149.00
Ensemble	2.00	68.00	4.00	33.00	2.00	9.00	10.00	66.00	3.00	22.00	1.00	36.00	3.67	39.00	3.67	47.54	6.00	157.00
MEEM [120]	1.00	42.00	1.00	24.00	1.00	5.00	1.00	38.00	3.00	26.00	1.00	30.00	1.33	27.50	1.33	32.15	1.00	107.00
KCF2	1.00	57.00	4.00	38.00	2.00	7.00	2.00	54.00	2.00	22.00	1.00	26.00	2.00	34.00	2.00	41.75	2.00	146.00
kcfdp [53]	1.00	58.00	5.00	41.00	2.00	10.00	2.00	55.00	1.00	23.00	1.00	37.00	2.00	37.33	2.00	45.04	2.00	152.00
kcf_mtsa [10]	1.00	65.00	1.00	32.00	2.00	7.00	10.00	60.00	1.00	23.00	2.00	30.00	2.83	36.17	2.83	44.37	2.00	147.00
sKCF [98]	2.00	69.00	10.00	40.00	2.00	6.00	10.00	66.00	3.00	30.00	2.00	32.00	4.83	40.50	4.83	49.44	7.00	165.00
muster [52]	1.00	56.00	1.00	33.27	2.00	8.00	2.00	45.00	2.00	21.00	1.00	28.00	1.50	31.88	1.50	39.03	2.00	132.27
tric [109]	2.00	50.80	1.00	25.67	2.00	6.93	2.00	45.00	1.00	19.47	1.00	27.93	1.50	29.30	1.50	35.47	2.00	119.27
SME [67]	1.00	54.00	2.00	27.00	2.00	9.00	1.00	48.00	1.00	16.00	1.00	31.00	1.33	30.83	1.33	37.55	1.00	126.00

5.3.3.3 Comparisons to the-state-of-the-art trackers

We also compare our proposed trackers against the state-of-the-art counterparts, described in Section 5.3.3.1. In Table 5.7 and 5.8, the trackers are compared in terms of the average accuracy and robustness rankings. In the last six columns, different averaging methodologies are presented. **Mean** and **Weighted mean** are averages of attributes either according to the weights of the attributes or equal weighting to each attribute, while **Pooled** corresponds to averaging per-frame results of the super-sequence obtained by concatenating all of the sequences.

Among the compared trackers, our best configuration (Ensemble_SW_Obs) shares the first ranking in terms of the average accuracy with SME and KCFDP. Nevertheless, average number of failures for KCFDP is significantly larger than our algorithm and SME. In robustness ranking, our Ensemble_SW_Obs tracker has the second best ranking after SME and MEEM in the weighted mean, while the first ranking is shared among MEEM, SME and Ensemble_SW_Obs in terms of the pooled average robustness. MEEM has an accuracy ranking of 3.67, significantly below the best ranking trackers, in spite of having the least number of failures. Furthermore, SME and MEEM trackers run every tracker in the ensemble and remove the oldest expert in a predefined period. On the contrary, the proposed ensemble tracker runs and updates only relevant trackers to the instantaneous object appearance, hence it does not require an expert removal procedure.

Although accuracy and robustness are complementary measures, our best performing tracker achieves a good trade-off between these two measures in this diverse dataset with 60 sequences. This trade-off can be visually observed from the Accuracy-Robustness plot in Figure 5.6. In this figure, the two axes serve for visualizing the accuracy and robustness in the descending order. Hence, closeness to the top right corner indicates better performance among the compared trackers. In this plot, our ensemble tracker with spatial windowing, SME and KCFDP trackers are closer than the remaining ones. Notably, SME requires trajectory consistency for each tracker in the expert quality calculation in the ensemble, while such a consideration is not necessary in the proposed ensemble.

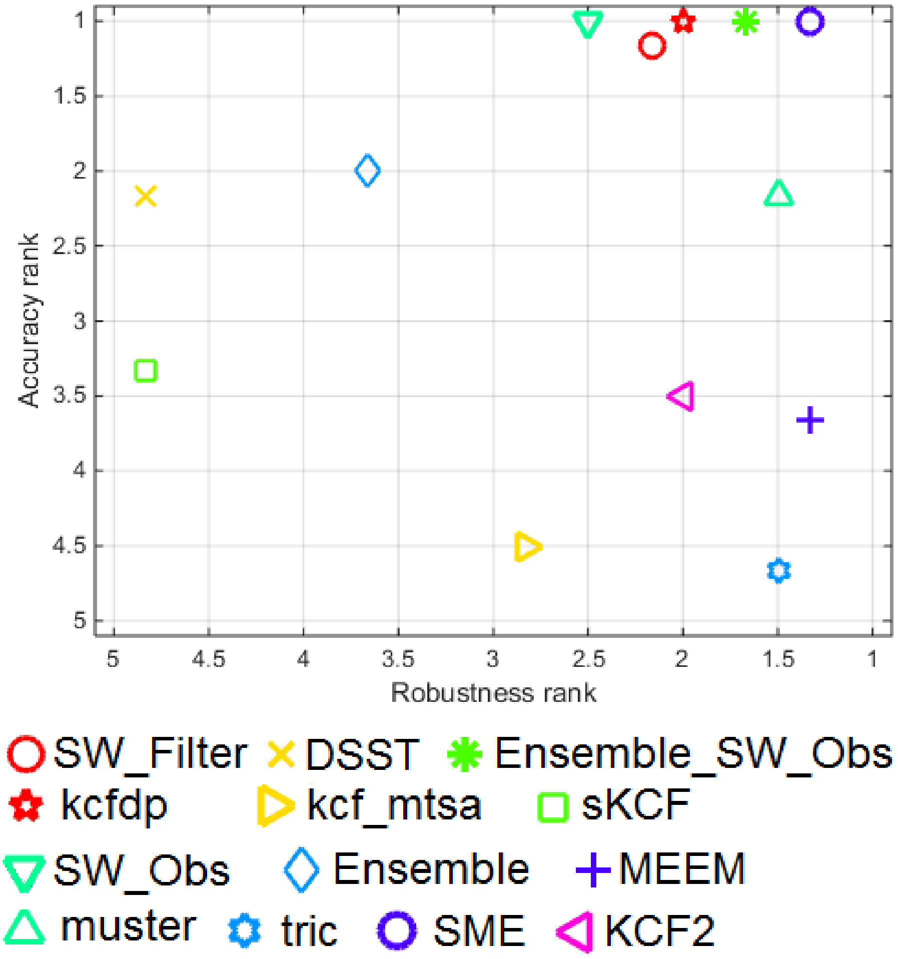


Figure 5.6: Accuracy-robustness ranking plot for the-state-of-the-art comparison.

5.3.4 Performance Evaluation on OTB-2015 Dataset

We evaluate the performance of the proposed tracker configurations on OTB-2015 [112] dataset consisting of 100 video sequences. For OTB, success curve is computed by the ratio of successfully tracked frames according to a threshold on the overlap ratio, and defined as the intersection over union of the predicted and ground-truth bounding boxes. The trackers are ranked according to the Area-Under-Curve (AUC) score of the success curve. Overlap precision (OP) orders the trackers according to the value of the average success for the threshold 0.5. Distance precision (DP) is calculated by the percentage of frames with a center localization error smaller than 20 pixels.

Table 5.9 shows the performance comparison for different configurations of our pro-

Table5.9: OTB-2015 results for different configurations of the proposed method. Red, blue and green indicate 1^{st}, 2^{nd} and 3^{rd} rankings, respectively.

	OP	DP	AUC
Ensemble_SW_Obs (D=4)	63.9	71.0	54.2
Ensemble (D=4)	63.9	69.0	53.6
Ensemble_SW_Obs (D=3)	62.1	69.5	53.1
Ensemble (D=3)	62.4	69.7	52.9
Ensemble_SW_Obs (D=2)	62.8	70.5	52.8
Ensemble (D=2)	61.0	68.6	51.9
SW_Filter	62.2	69.0	51.7
SW_Obs	60.8	69.1	51.5
DSST [26]	60.6	67.7	51.1

posed trackers in terms of OP, DP and AUC. In terms of AUC, the tracking performance improves as the number of depth (D) increases from 2 to 4. In our preliminary experiments on VOT2015 dataset, the tracking performance saturated after $D = 3$, while the performance saturates after $D = 4$ on OTB-2015. This is probably due to two factors: (1) VOT2015 has a measure technique that re-initiates the trackers after a failure while OTB-2015 does not, and (2) the average sequence length of OTB-2015 is significantly greater than VOT2015 has. Remarkably, our spatial windowing method enhances the tracking performance of the proposed ensemble tracker with different tree depths. Moreover, application of window learning on the correlation filter (SW_Filter) improves the OP values of the baseline tracker DSST, and has superiority over the application of window learning on the object observation (SW_Obs).

Our proposed tracker Ensemble_SW_Obs with $D = 4$ is compared against 3 trackers SRDCF [28], deepSRDCF [27] and the tracker proposed in [12]. SRDCF is a correlation filter based method which penalizes the boundaries of the correlation filter with a spatial mask in the training stage and employs HOG feature maps. deepSRDCF extends SRDCF and utilizes deep convolutional features. The work in [12] jointly learns correlation filters and the target correlation response. It has tree different configurations as DCF_AT, KCF_AT and SAMF_AT. Figure 5.7 reports the success curves for the compared trackers. We achieve competitive or superior performance against

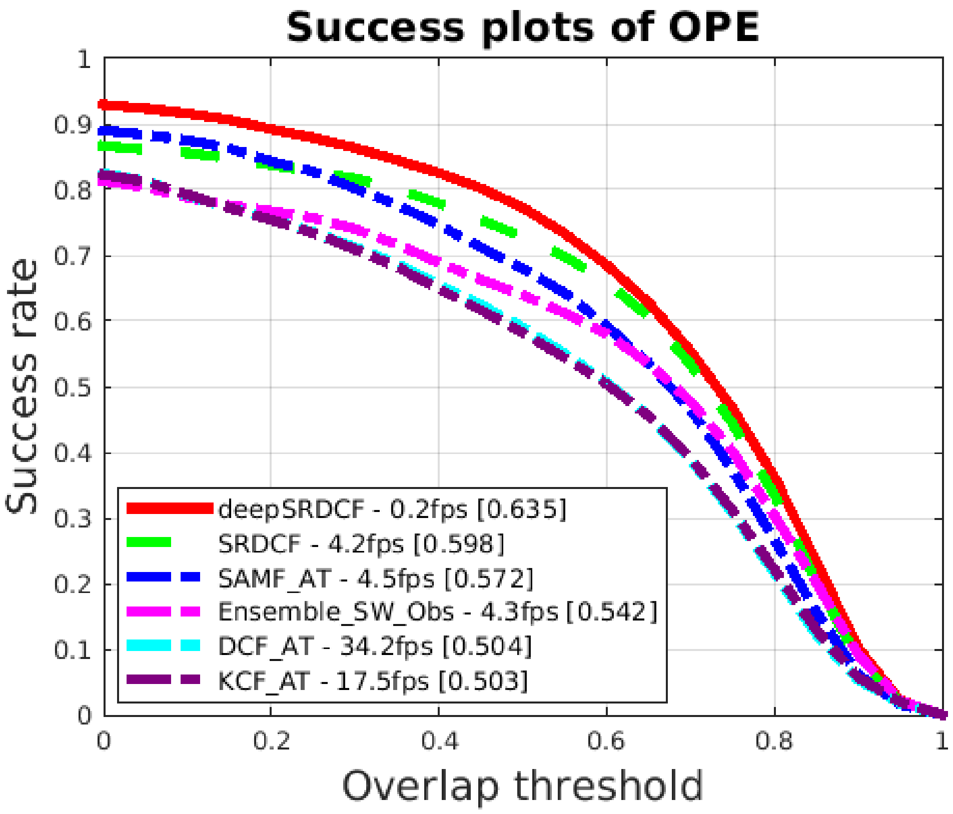


Figure 5.7: Success curves of the proposed method with the tree depth 4 and the compared trackers for OTB-2015 dataset.

DCF_AT, KCF_AT and SAMF_AT. Figure 5.7 also lists running speeds of the trackers extracted from the result files that are released by the corresponding authors. The proposed tracker is run on Intel Xeon E5-2623 @3.00GHz, and has a good trade-off between accuracy and speed among the compared methods. Moreover, the proposed method has a comparable average running speed against its counterparts.

CHAPTER 6

LEARNING DEEP CONVOLUTION FEATURES FOR CORRELATION FILTERS

Recent works on correlation filters mainly concentrate on either improving the correlation technique or exploiting the most appropriate hand-crafted or deep features which are provided by the pre-trained networks for object recognition tasks. However, a general deep feature learning strategy dedicated to correlation filter based tracking frameworks is still unexplored.

To this end, we carry out training a fully-convolutional neural network (CNN) which is based on a loss function by exclusively taking care of the correlation filter based tracking formulation. By integrating the learned feature maps into the-state-of-the-art correlation filter based trackers, a considerable amount of performance increase has been observed in contrast to the hand-crafted features. Moreover, the proposed network, which produces few feature maps, has a comparable performance against the large number of deep feature maps of object recognition networks. It should also be emphasized that when the proposed learning framework is adopted in fine-tuning the convolutional layers of VGG-M network [18], the performance of the state-of-the-art tracker CCOT [30] is substantially boosted by a large margin in terms of the expected average overlap metric of VOT2016 [36].

The goals of this work are threefold: (1) to derive the necessary formulations of the *convolutional features for correlation filters*, which will be abbreviated as CFCF following the initials of this phrase, (2) to implement our algorithm by presenting a CNN model and train this model with an appropriately generated dataset, and (3) finally to integrate the learned features to correlation filter based trackers which support multiple feature maps. Before going into the details of our formulations and the tracking implementation, convolutional layers will be summarized for the readers who are unfamiliar with this concept, because fully convolutional layers are exploited and learned by considering the correlation response quality.

6.1 Convolutional Layers

It is notable that a feature generation function $f_\theta(.)$ of the image patch I, which is typically integrated into the CFB trackers, should carry the shift invariance property, *i.e.*, if $I_\theta[x, y] = f(I[x, y])$ and $Y_\theta[x, y] = f_\theta(I[x - k\delta_x, y - k\delta_y])$, then $Y_\theta[x, y] \approx I_\theta[x - \delta_x, y - \delta_y]$ should be satisfied, where $I[., .]$ is a 2-dimensional discrete signal and k is the scale factor of the transformation function $f_\theta(.)$. Hence, in the proposed feature learning framework, fully-convolutional networks are trained. This kind of network is composed of various layer types including convolutional layers, non-linearities, batch normalization, local response normalization and pooling layers. All these layers do not violate this property. The reader should refer to the documentation of [105] for the layer types and their implementations.

A convolution layer is composed of the following tensors [105]:

$$x \in \mathcal{R}^{H \times W \times D}, \quad f \in \mathcal{R}^{H' \times W' \times D \times D''}, \quad y \in \mathcal{R}^{H'' \times W'' \times D''}, and \quad b \in \mathcal{R}^{D''} \tag{6.1}$$

where x is the input to the layer, y is the output of the layer, and f is the filter tensor of the convolutional layer and b is the bias vector.

The relation between x, y, f and b is defined as:

$$y[i''j''d''] = b_{d''} + \sum_{i'=1}^{H'} \sum_{j'=1}^{W'} \sum_{d'=1}^{D'} f_{i'j'd} x_{i''+i'-1, j''+j'-1, d', d''}. \tag{6.2}$$

In (6.2), the convolutional layer has a stride 1 and no padding. The formulations with the stride more than 1 and non-zero padding can be found in [105]. In the proposed custom model, there is no pooling between the convolutional layers and an appropriate padding is provided not to change the dimension of the signal. Nevertheless, the VGG-M model [18] with pooling and no padding is also trained with the proposed cost function as it will be explained in Section 6.7.3, although the sizes of the input and output are significantly different. This situation is handled by resizing the templates to a common size in order to implement the necessary multiplications and additions in the same image domain.

6.2 Convolutional Features for Correlation Filters

Utilization of both hand-crafted features or convolutional features extracted from pre-trained convolutional neural network architectures are specifically tuned for obtaining a sufficient summary of the object appearance. Such an efficient representation is expected to handle the trade-off between intra-class and inter-class invariance for classification purposes. Unfortunately, such representations of the object appearance are not designed for obtaining object features which will improve the correlation quality of a correlation filter based task for object tracking and detection. This observation has motivated learning convolutional features dedicated to serve for obtaining better correlation results. The representation that is aimed to be learned can be designed in such a way that they could become robust to visual appearance changes of the object of interest. Moreover, the learned model could partially handle the imperfect training example phenomenon of the correlation filter based trackers [26, 14], where actual translation is assumed to be identical to the circular translation in order to exploit the FFT algorithm along with the Convolution Theorem. These two improvements are possible, if the model is trained on a carefully generated dataset which contains the challenges of visual tracking.

By considering aforementioned issues, our aim is to minimize the following cost in (6.3) for all of the training examples. Yet, the cost in (6.3) is defined for a single training example for the sake of simplicity.

$$\mathcal{L}(\theta) = ||h_\theta \circledast x_\theta - g||^2 \tag{6.3}$$

In the above equation, x_θ and h_θ are the extracted features of the object appearance and the corresponding correlation filter, which is expected to represent the model of the object appearance. In addition, g is the desired correlation plane, and $\circledast$ is the correlation operator to be defined in (6.6). The overall flow of the proposed network is given in Figure 6.1. It is also obvious that x_θ and h_θ are functions of θ. Here,

θ stands for the parameters of the feature generation function $f_\theta(.)$. In other words, $x_\theta \triangleq f_\theta(x_{raw})$ and $y_\theta \triangleq f_\theta(y_{raw})$, where x_{raw} and y_{raw} are the raw object appearance, *i.e.*, raw image intensities. Moreover, the correlation filter h_θ is a function of y_θ. The relation between y_θ and h_θ for both the single and multiple feature channels are discussed in Chapter 3 and reminded here if necessary.

6.3 Backpropagation of the Loss for a Single Feature Map

It should be noted that although the two images of the running man in Figure 6.1 have almost the same appearance, their circular correlation output, denoted as g^c, is so imperfect that it may cause a drift, if the estimated motion at each frame is gradually shifted from the actual target location. Since correlation filters are learned by the circularly translated training examples, the correlation filter generated from y_{raw} might not be able to handle an abrupt motion change and an appearance change of the target object x_{raw}. Yet, a representation transformation on the object observation could be beneficial to mitigate the considered problems. Thus, we propose learning a model which extracts deep fully convolutional representations and deal with the cost in (6.3). In order to train the proposed architecture, we require the partial derivatives of the loss function with respect to every element of the parameter vector θ, *i.e.*, the gradient of the loss as $\nabla_\theta \mathcal{L}$. It is explicitly given below[1]:

$$\nabla_\theta \mathcal{L} = \frac{\partial \mathcal{L}}{\partial x_\theta}\frac{\partial x_\theta}{\partial \theta} + \frac{\partial \mathcal{L}}{\partial h_\theta}\frac{\partial h_\theta}{\partial y_\theta}\frac{\partial y_\theta}{\partial \theta} \tag{6.4}$$

In the above equation, $\frac{\partial x_\theta}{\partial \theta}$ and $\frac{\partial y_\theta}{\partial \theta}$ can be calculated by publicly available deep learning libraries which exploit backpropagation algorithm to dynamically calculate θ for every layer of the network. The partial derivatives, $\frac{\partial \mathcal{L}}{\partial x_\theta}$ and $\frac{\partial \mathcal{L}}{\partial h_\theta}$, Jacobian matrix $\frac{\partial h_\theta}{\partial y_\theta}$ are required to be derived to find the gradient of the loss, $\nabla_\theta \mathcal{L}$.

Before calculating the required terms, the correlation and convolution theorem will be reminded, since they will be utilized throughout the derivations.

$$c[n] = \sum_i a[i]b[n-i] = \mathcal{F}^{-1}\{A \odot B\} \tag{6.5}$$

---

[1] Although $\frac{\partial(.)}{\partial(.)}$ denotes partial derivative, we also use $\frac{\partial(.)}{\partial(.)}$ to magnify the independent variables and unify element-wise differentiation, partial derivative, gradient, Jacobian operations into one notation.

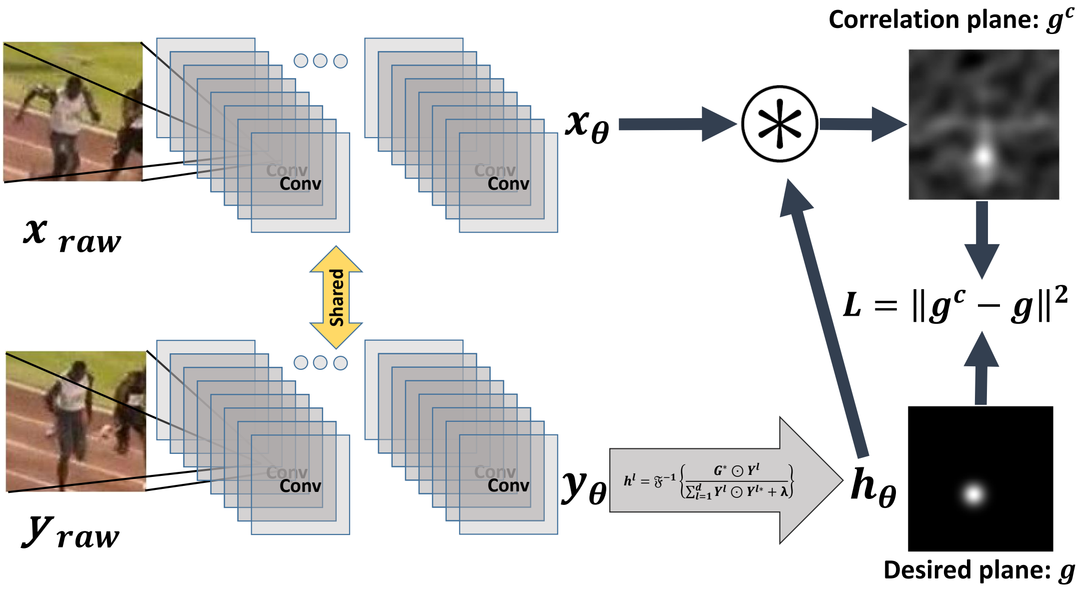


Figure 6.1: Block diagram of the proposed architecture. $\circledast$ indicates the circular correlation.

$$c[n] = \sum_i a[i]b[n+i] = \mathcal{F}^{-1}\{A^* \odot B\} \tag{6.6}$$

Equations (6.5) and (6.6) are convolution and correlation theorem for two one-dimensional signals, respectively. Equations (6.5) and (6.6) are obviously corollary of each other. $\odot$ is element-wise multiplication, $\mathcal{F}$ and $\mathcal{F}^{-1}$ are Discrete Fourier Transform (DFT) and inverse DFT, respectively. All capital letters denote the signals in Fourier domain. It should also be noted that any variable is assumed to be one-dimensional unless specified explicitly.

The cost function in (6.3) can be rewritten as below:

$$\mathcal{L} = \sum_n \left(\sum_i h[i]x[i+n] - g[n] \right)^2 \tag{6.7}$$

In the above equation, we drop θ term from both of the signals x and h for convenience. With some effort, derivative of (6.3) with respect to an arbitrary m^{th} element of the signal x, which we denote as $h[m]$, can be written as in the below equations (6.8) and (6.9).

$$\frac{\partial \mathcal{L}}{\partial h[m]} = \sum_n (\sum_i h[i]x[i+n] - g[n]) \frac{\partial \sum_i h[i]x[i+n]}{\partial h[m]} \tag{6.8}$$

$$\frac{\partial \mathcal{L}}{\partial h[m]} = \sum_n (\sum_i h[i]x[i+n] - g[n])x[m+n] \tag{6.9}$$

If we define the error signal as:

$$e[n] = \sum_i h[i]x[i+n] - g[n], \tag{6.10}$$

the derivatives will have better interpretation for the sake of both the time and frequency domain. By substituting this error signal into (6.9), the resulting derivative signal will have an efficient calculation in the frequency domain as follows by using (6.6):

$$\frac{\partial \mathcal{L}}{\partial h[m]} = \sum_n e[n]x[m+n] = [\mathcal{F}^{-1}\{E^* \odot X\}][m] \tag{6.11}$$

By similar efforts and utilizing the equation (6.5), $\frac{\partial \mathcal{L}}{\partial x[m]}$ relation can be obtained as follows:

$$\frac{\partial \mathcal{L}}{\partial x[m]} = \sum_n e[n]h[m-n] = [\mathcal{F}^{-1}\{E \odot H\}][m] \tag{6.12}$$

Until this point, we derived the partial derivatives of the cost function with respect to any element of the intermediate signals (which can be treated as vectors in general); hence, some important gradients are required for backpropagation. The derived gradients have computational complexity of $O(Plog(P))$ (P is the signal length), since we utilize some DFT properties. However, the most cumbersome term of $\nabla_\theta \mathcal{L}$ is the Jacobian matrix $\frac{\partial h}{\partial y}$ which has the following definition:

$$\frac{\partial h}{\partial y} = \begin{bmatrix} \frac{\partial h[1]}{\partial y[1]} & \frac{\partial h[1]}{\partial y[2]} & \cdots & \frac{\partial h[1]}{\partial y[N]} \\ \frac{\partial h[2]}{\partial y[1]} & \frac{\partial h[2]}{\partial y[2]} & \cdots & \frac{\partial h[2]}{\partial y[N]} \\ \cdots & & & \\ \frac{\partial h[N]}{\partial y[1]} & \frac{\partial h[N]}{\partial y[2]} & \cdots & \frac{\partial h[N]}{\partial y[N]} \end{bmatrix} \tag{6.13}$$

Hence, we require the individual partial derivatives of each element of h with respect to each element of y. Although this well-known definition of the Jacobian matrix is simple, the tedious point is that the relation between h and y are in the frequency domain, bringing a complicated burden since they have a non-linear and non-analytic relation as follows:

$$h = \mathcal{F}^{-1}\left\{\frac{\hat{G}^* \odot Y}{Y \odot Y^* + \lambda}\right\}, \tag{6.14}$$

which is the correlation filter for the single scale version of (3.18). In the above equation, $\hat{G}$ is the DFT of $\hat{g}$. The above filter is nothing but the correlation filter which minimizes the following cost:

$$\epsilon = ||h \circledast y - \hat{g}||^2 + \lambda||h||^2 \tag{6.15}$$

where $\hat{g}$ is the desired Gaussian-shaped correlation response with a sharp peak at its center, since the object template y is the localized object patch, and is used in the literature for most state-of-the-art correlation filter based tracking algorithms such as [14, 48, 26].

As the relation between h and y are in the frequency domain that includes complex numbers, the derivatives of complex variables are required. For some functions which are non-analytic (this is what we have), the derivative definitions do not apply. Moreover, chain rule of real functions does not apply properly. However, fortunately, we have real signals in hand and could approach the complex derivative issue from a different perspective. Hence, the conjugation operation in the frequency domain will reflect to the time domain as the time reversed version of the corresponding signal. The time reversal is circular due to DFT definition derived from Discrete Fourier Series (DFS). In the following equations, if a time reverse of the signal is given, it means that it is the circular time reverse, although it is not specified in the notation. In the following, we present an efficient property of the real signals that will be extremelly useful in our backpropagation derivation: The conjugated signal in the frequency domain has a relation to the signal itself as:

$$\begin{bmatrix} X[1]^* \\ X[2]^* \\ X[3]^* \\ \vdots \\ X[N]^* \end{bmatrix} = \begin{bmatrix} X[1] \\ X[N] \\ X[N-1] \\ \vdots \\ X[2] \end{bmatrix} = \begin{bmatrix} 1 & 0 & ... & ... & ... & 0 \\ 0 & ... & ... & ... & 0 & 1 \\ ... & & & & & \\ 0 & ... & ... & 1 & ... & 0 \\ 0 & 0 & 1 & ... & ... & 0 \\ 0 & 1 & ... & ... & ... & 0 \end{bmatrix} \begin{bmatrix} X[1] \\ X[2] \\ X[3] \\ \vdots \\ X[N] \end{bmatrix} = M \begin{bmatrix} X[1] \\ X[2] \\ X[3] \\ \vdots \\ X[N] \end{bmatrix} \tag{6.16}$$

if the signal x is real and X is its DFT. Here, the matrix M is an operator which simply converts the signal X to its circular time reversal. By using (6.16), we will replace the conjugate of the independent variable (derivative of which will be taken). This replacement allows us to use the standard chain rule and obtain an analytic form of the function under concern.

Equation (6.14) and its Jacobian can be written as follows:

$$h = F^H K(Y), \quad where \quad K(Y) = \frac{\hat{G}^* \odot Y(w)}{Y(w) \odot Y(-w) + \lambda} \tag{6.17}$$

$$\frac{\partial h}{\partial y} = \frac{\partial h}{\partial K(Y)} \frac{\partial K(Y)}{\partial Y} \frac{\partial Y}{\partial y} = F^H \frac{\partial K(Y)}{\partial Y} F \tag{6.18}$$

In the above equation, F and F^H are the DFT and inverse DFT matrices. The signal $Y(-w)$ represents the circular time reversal of the signal $Y(w)$ and it is equivalent to the conjugate of Y. We can write down the Jacobian matrix of $K(Y)$ with respect to $Y(w)$ by using the derivative of the division rule in an element-wise product manner since the multiplications are element-wise in $K(Y)$ function.

$$\frac{\partial K(Y)}{\partial Y} = diag\left(\frac{\lambda\hat{G}^* - \hat{G}^* \odot Y(w) \odot Y(w) \odot \dot{Y}(-w)}{(Y(w) \odot Y(-w) + \lambda)^2}\right) \tag{6.19}$$

Here, $\dot{Y}(-w)$ denotes the derivative of $Y(-w)$ with respect to $Y(w)$. We know from the property in (6.16) that $Y(-w) = MY(w)$. Thus, we can explicitly write (6.19) as follows:

$$\frac{\partial K(Y)}{\partial Y} = diag\left(\frac{\lambda\hat{G}^*}{(Y(w) \odot Y(-w) + \lambda)^2}\right) - diag\left(\frac{\hat{G}^* \odot Y(w)^2}{(Y(w) \odot Y(-w) + \lambda)^2}\right) M \tag{6.20}$$

Finally, we arrive at the derivation of the overall Jacobian as:

$$\frac{\partial h}{\partial y} = F^H \left(diag\left(\frac{\lambda\hat{G}^*}{(Y(w) \odot Y(-w) + \lambda)^2}\right) - diag\left(\frac{\hat{G}^* \odot Y(w)^2}{(Y(w) \odot Y(-w) + \lambda)^2}\right) M\right) F \tag{6.21}$$

The main problem with the equation (6.21) is that the dimensions of the matrices are proportional to the signal length P, hence making the computational complexity propotional to the P^2 even for one-dimensional case due to the matrix multiplication. Fortunately, this matrix is not directly used. In other words, the Jacobian $\frac{\partial h}{\partial y}$ is multiplied from left by $\frac{\partial \mathcal{L}}{\partial h}$ as follows:

$$\frac{\partial \mathcal{L}}{\partial h}\frac{\partial h}{\partial y} = \frac{\partial \mathcal{L}}{\partial h} F^H \left(diag\left(\frac{\lambda\hat{G}^*}{(Y(w) \odot Y(-w) + \lambda)^2}\right) - diag\left(\frac{\hat{G}^* \odot Y(w)^2}{(Y(w) \odot Y(-w) + \lambda)^2}\right) M\right) F \tag{6.22}$$

The term $\frac{\partial \mathcal{L}}{\partial h} F^H$ can be written as $\left(F\left(\frac{\partial \mathcal{L}}{\partial h}\right)^H\right)^H$. Moreover, if $c \triangleq \frac{\partial \mathcal{L}}{\partial h}$, $A \triangleq \frac{\lambda\hat{G}^*}{(Y(w)\odot Y(-w)+\lambda)^2}$, and $B \triangleq (\frac{\hat{G}^* \odot Y(w)^2}{(Y(w)\odot Y(-w)+\lambda)^2})$ are defined, then (6.22) reduces to:

$$\frac{\partial \mathcal{L}}{\partial h}\frac{\partial h}{\partial y} = c^H F^H (diag(A) - diag(B)M) F \tag{6.23}$$

$$\frac{\partial \mathcal{L}}{\partial h}\frac{\partial h}{\partial y} = (F^H(diag(A)^H - Mdiag(B)^H)Fc)^H \tag{6.24}$$

by knowing the fact that c is a real signal. Notably, all of the operations conducted here should become real in time domain due to the fact that the conjugation and/or element-wise multiplication and/or circular time reversal operations over the real signal in the frequency domain will not produce an imaginary part in the time domain. Due to this fact the Hermitian notation can be dropped from (6.24), and the gradient of loss with respect to y can be written as:

$$\frac{\partial \mathcal{L}}{\partial h}\frac{\partial h}{\partial y} = \mathcal{F}^{-1}\{A^* \odot \mathcal{F}\{c\} - M(B^* \odot \mathcal{F}\{c\})\} \tag{6.25}$$

Due to the fact that the M matrix makes the signal circularly time-reversed and that DFT of real signals are circularly conjugate symmetric, multiplying the DFT of a real signal with M from the left corresponds to the conjugation operation, the equation (6.25) can be written as follows:

$$\nabla_y \mathcal{L} = \frac{\partial \mathcal{L}}{\partial h}\frac{\partial h}{\partial y} = \mathcal{F}^{-1}\{A^* \odot \mathcal{F}\{c\} - (B \odot \mathcal{F}\{c\}^*)\} \tag{6.26}$$

The equation (6.26) is the final analytic expression for the gradient of the loss with respect to y for single feature map case. In the following section, the backpropagation derivation for the multiple feature maps will be presented.

6.4 Backpropagation of the Loss for Multiple Feature Maps

In the previous section, the backpropagation of the proposed loss function is presented for single feature map case. In this section, the required gradients in the case of multiple feature maps is derived. The correlation filter cost function for multiple feature maps is given as:

$$\epsilon = \left\| \sum_{l=1}^{d} h^l \circledast y^l - g \right\|^2 + \lambda \sum_{l=1}^{d} ||h^l||^2 \tag{6.27}$$

The minimizing set of filters are [26]:

$$h^l = \mathcal{F}^{-1}\left\{\frac{\hat{G}^* \odot Y^l}{\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda}\right\} \tag{6.28}$$

In the multiple feature channels case, we can define our backprobable cost function as follows:

$$\mathcal{L}(\theta) = \left\|\sum_{l=1}^{d} h_\theta^l \circledast x_\theta^l - g\right\|^2 \tag{6.29}$$

Since most parts of the gradient calculations are analogous to the one channel case, only the Jacobian part will be focused here. Similar to the previous part, we can define:

$$H^l = \frac{\hat{G}^* \odot Y^l}{\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda} \tag{6.30}$$

The Jacobian term $\frac{dh^l}{dy^k}$ is given as follows:

$$\frac{dh^l}{dy^k} = \frac{dh^l}{dH^l}\frac{dH^l}{dY^k}\frac{dY^k}{dy^k} = F^H \frac{dH^l}{dY^k} F, \tag{6.31}$$

where the Jacobian term $\frac{\partial H^l}{\partial Y^k}$ can be calculated in the frequency domain as:

$$\frac{\partial H^l}{\partial Y^k} = \mathcal{I}(l == k) diag\left(\frac{\hat{G}^*}{\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda}\right) - \tag{6.32}$$

$$diag\left(\frac{\hat{G}^* \odot Y^l \odot Y^{k*}}{(\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda)^2}\right) - diag\left(\frac{\hat{G}^* \odot Y^l \odot Y^{k}}{(\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda)^2}\right) M \tag{6.33}$$

where $\mathcal{I}(.)$ is the indicator function and outputs 1 if the argument is true and 0 otherwise. For simplicity, if $K_1^{lk} = \mathcal{I}(l == k) diag\left(\frac{\hat{G}^*}{\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda}\right)$, $K_2^{lk} = diag\left(\frac{\hat{G}^* \odot Y^l \odot Y^{k*}}{(\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda)^2}\right)$ and $K_3^{lk} = diag\left(\frac{\hat{G}^* \odot Y^l \odot Y^{k}}{(\sum\limits_{m=1}^{d} Y^m \odot Y^{m*} + \lambda)^2}\right)$ are defined, then (6.32) can be rewritten as follows:

$$\frac{\partial H^l}{\partial Y^k} = K_1^{lk} - K_2^{lk} - K_3^{lk} M \tag{6.34}$$

Following the same methodology in the previous section, $a^l = \frac{\partial \mathcal{L}}{\partial h^l}$ and $A^l = \mathcal{F}\{a\}$ are the gradient of the cost with respect to the correlation filter for the l^{th} channel and its DFT, respectively. The gradient of the cost $\mathcal{L}$ is also given as:

$$\frac{\partial \mathcal{L}}{\partial y^k} = \sum_{l=1}^{d} \frac{\partial \mathcal{L}}{\partial h^l} \frac{\partial h^l}{\partial y^k} \tag{6.35}$$

Replacing (6.34) and (6.31) to (6.35) yields:

$$\nabla_{y^k} \mathcal{L} = \frac{\partial \mathcal{L}}{\partial y^k} = \sum_{l=1}^{d} \mathcal{F}^{-1}\{(K_1^{lk} - K_2^{lk})^* \odot A^l - K_3^{lk} \odot A^{l*}\} \tag{6.36}$$

by the similar derivation in the previous section. The gradient of the loss with respect to x^l and h^l are

$$\frac{\partial \mathcal{L}}{\partial h^l} = \mathcal{F}^{-1}\{E^* \odot X^l\}, \tag{6.37}$$

$$\frac{\partial \mathcal{L}}{\partial x^l} = \mathcal{F}^{-1}\{E \odot H^l\} \tag{6.38}$$

where $E = \mathcal{F}\left\{\sum_{l=1}^{d} h^l \circledast x^l - g\right\}$. The final gradient of the loss with respect to the parameters θ of the network is[2]:

$$\nabla_\theta \mathcal{L} = \sum_{l} \frac{\partial \mathcal{L}}{\partial x_\theta^l} \frac{\partial x_\theta^l}{\partial \theta} + \sum_{l} \frac{\partial \mathcal{L}}{\partial y_\theta^l} \frac{\partial y_\theta^l}{\partial \theta}. \tag{6.39}$$

Thus far, we derived the necessary gradient terms required to backpropagate our signal under concern. By the derived formulations, we make the feature design of the well-known correlation filter based trackers become amenable to train a deep network model. In the next section, the proposed custom architecture design is detailed with the employed training and testing setups.

6.5 Dataset Generation from VOT2015 Dataset

In order to train a fully CNN model, 200K training examples are generated by utilizing the VOT2015 dataset [35], consisting of 60 sequences with different attributes. The bounding boxes of each object are provided for each frame. We crop approximately two times larger area of the object size and resize the images to the appropriate size of the network (101×101 in our experiments). To keep the aspect ratio of the objects, we crop the squares from the region of interests of the object, where the side

---

[2] Please note that x^l is replaced with x_θ^l to underscore the dependency of x^l on the network $f_\theta(.)$.

length of the square is $2 * \sqrt{W \times H}$ (W and H are the width and height of the object, respectively.). Generated y_{raw}'s center the object since these patches are indeed templates for us. However, x_{raw}'s are obtained by shifting the center of the object since our aim is to break the influence of the circular translation over the actual translation. The shift amount is determined by a random variable which is uniformly distributed in $[-0.3 \times W, 0.3 \times W]$ and $[-0.3 \times H, 0.3 \times H]$ for horizontal and vertical translations. The frame difference between y_{raw} and x_{raw} is a Gaussian random variable with standard deviation of 5 frames.

6.6 Proposed Custom architecture and Implementation Details

Since a medium scale dataset is generated, we prefer designing relatively small architecture with respect to the state-of-the-art networks of classification such as [18]. For this purpose, the input to our network is 3 channel input image in 101×101 dimensions. The architecture consists of 4 convolutional layers. All of these layers have a batch normalization layer after the convolutional layer part. The first three of them have a rectified linear unit (ReLU) [60] layer with a leak of 0.1 [115, 46]. To keep the spatial size of the feature maps constant, convolutional layers have the appropriate padding (*e.g.*, padding value is 1 for the 3×3 kernel sizes). The number of feature maps are shown in Figure 6.2. The final layer outputs the map which will be utilized for the correlation task ($x(\theta)$ and $y(\theta)$ of Figure 6.1).

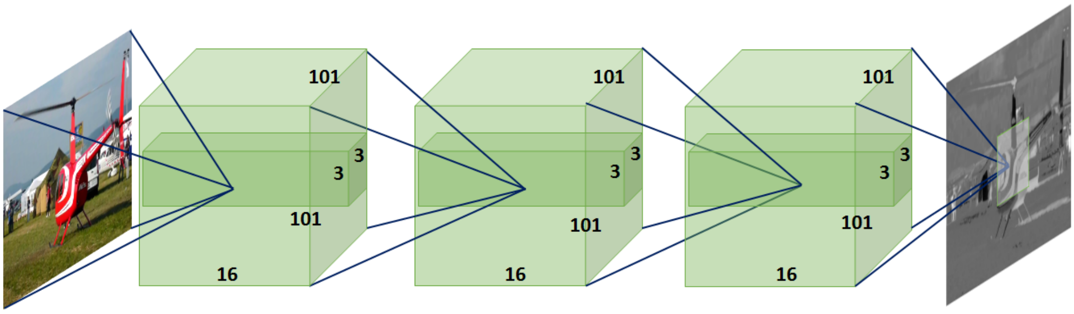
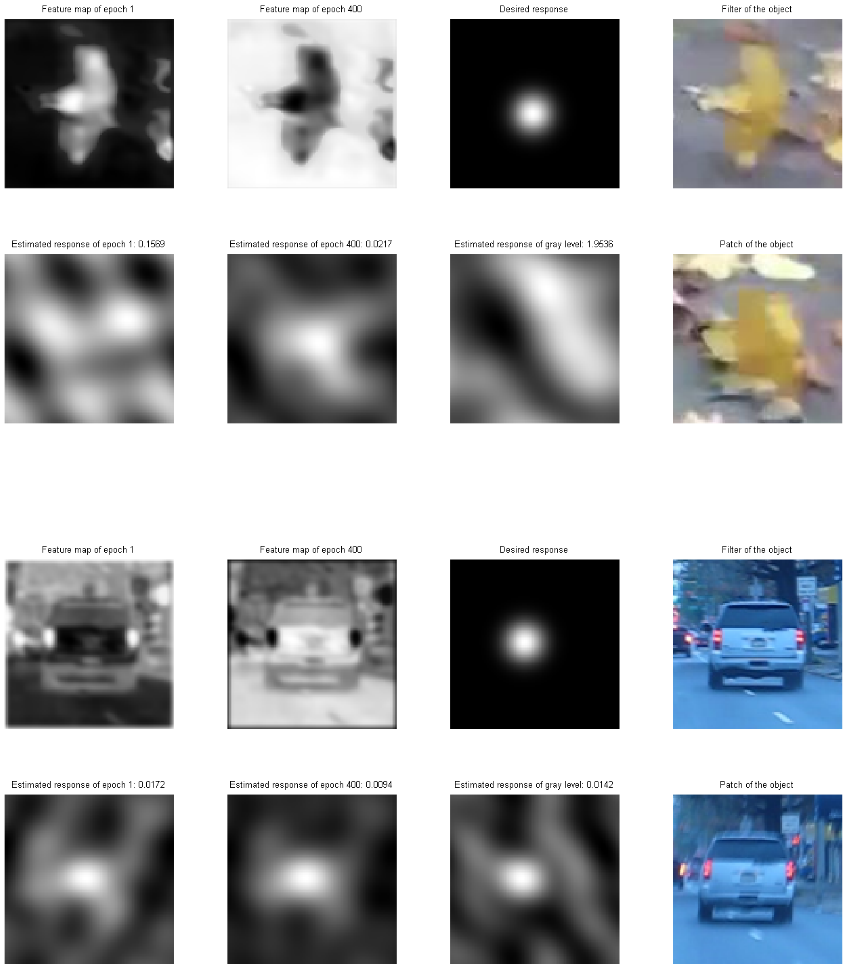


Figure 6.2: Architecture dimensions.

The aforementioned network is trained in the generated dataset with a batch size of 128, learning rate is decreased from 10^{-9} to 10^{-10} for 400 epochs. Momentum is set to 0.9, and a weight decay of 0.0005.



Feature map of epoch 1 | Feature map of epoch 400 | Desired response | Filter of the object

Estimated response of epoch 1: 0.1569 | Estimated response of epoch 400: 0.0217 | Estimated response of gray level: 1.9536 | Patch of the object

Feature map of epoch 1 | Feature map of epoch 400 | Desired response | Filter of the object

Estimated response of epoch 1: 0.0172 | Estimated response of epoch 400: 0.0094 | Estimated response of gray level: 0.0142 | Patch of the object

Figure 6.3: Visual illustrations from our trained network for single feature map. MSE scores of the correlation filter of the feature map extracted by our network after the first epoch, 400^{th} epoch and the correlation filter of the gray-level image intensities.

As it can be observed from Figure 6.3, we have a significant amount of decrease in the MSE cost with respect to both time (comparison between the first and 400^{th} epoch) and the gray level image intensity feature. This indicates that useful features are learned and they are targeting at reducing the vagueness of the correlation planes.

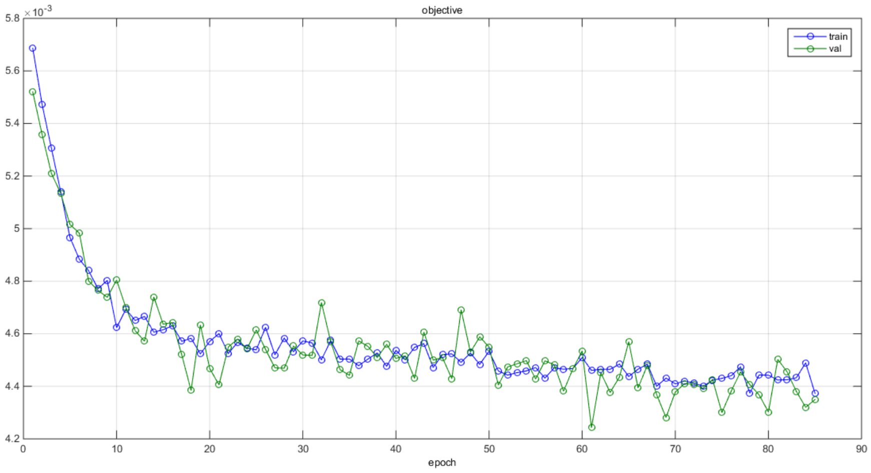


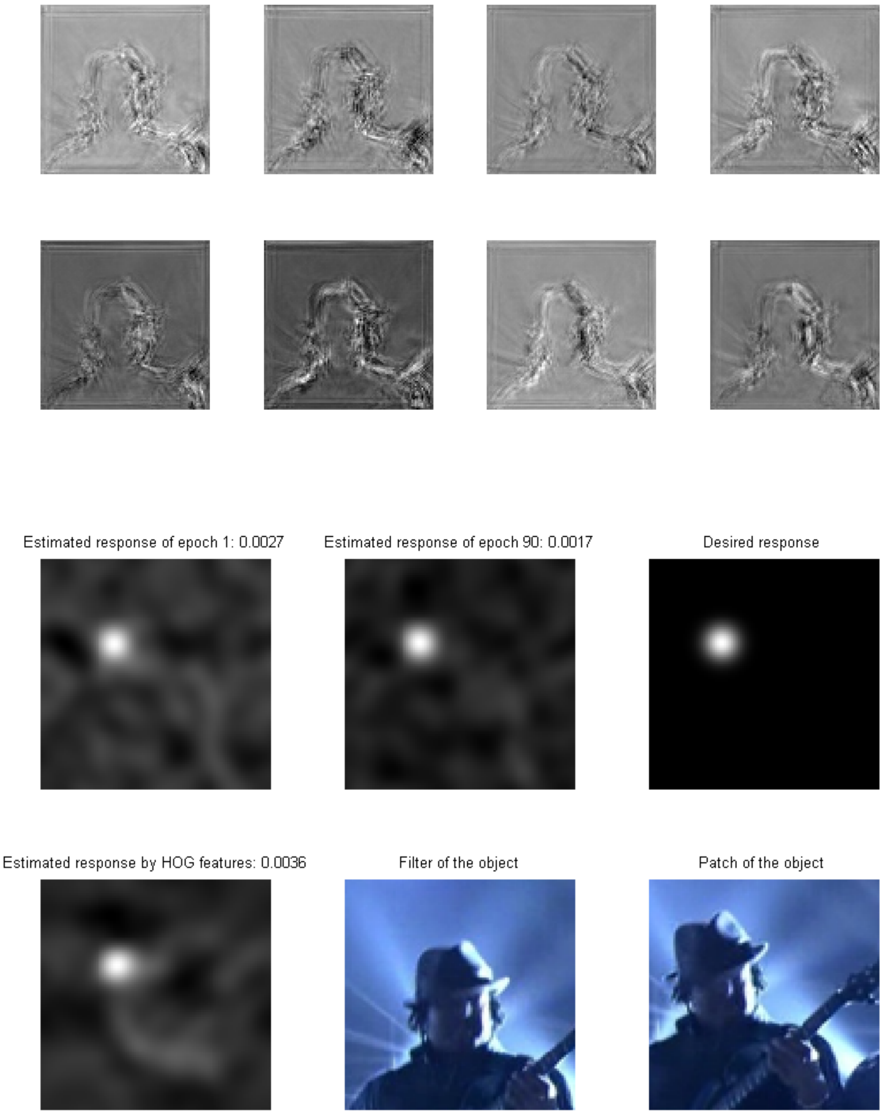
Figure 6.4: Cost function decrease during the training.

For the multiple feature maps design, we set 4 hidden layers where each hidden layer has 16 feature maps, ReLU units and batch normalization layers. Kernel sizes are selected as 9×9, 7×7, 5×5 and 3×3 in the increasing order of the layers. The final layer has 8 feature maps. Moreover, Figure 6.4 displays the loss function through the epochs during training.

Figures 6.5, 6.6, 6.7 and 6.8 illustrate the extracted features by the proposed model. It is notable that MSEs (*i.e.*, the loss value) between the desired and estimated response have less value for the proposed features than the HOG features even though there exist 8 features for the proposed model, while the number of features is 28 for HOG representation. As one can notice from these figures, the obtained features highlight edges in different orientations.

6.7 Tracking Application and Experimental Results on Benchmark Datasets

In this section, the experimental results on benchmark datasets are presented to show the representation power of the learned features. Moreover, the learned features are



Estimated response of epoch 1: 0.0027

Estimated response of epoch 90: 0.0017

Desired response

Estimated response by HOG features: 0.0036

Filter of the object

Patch of the object

Figure 6.5: Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features.

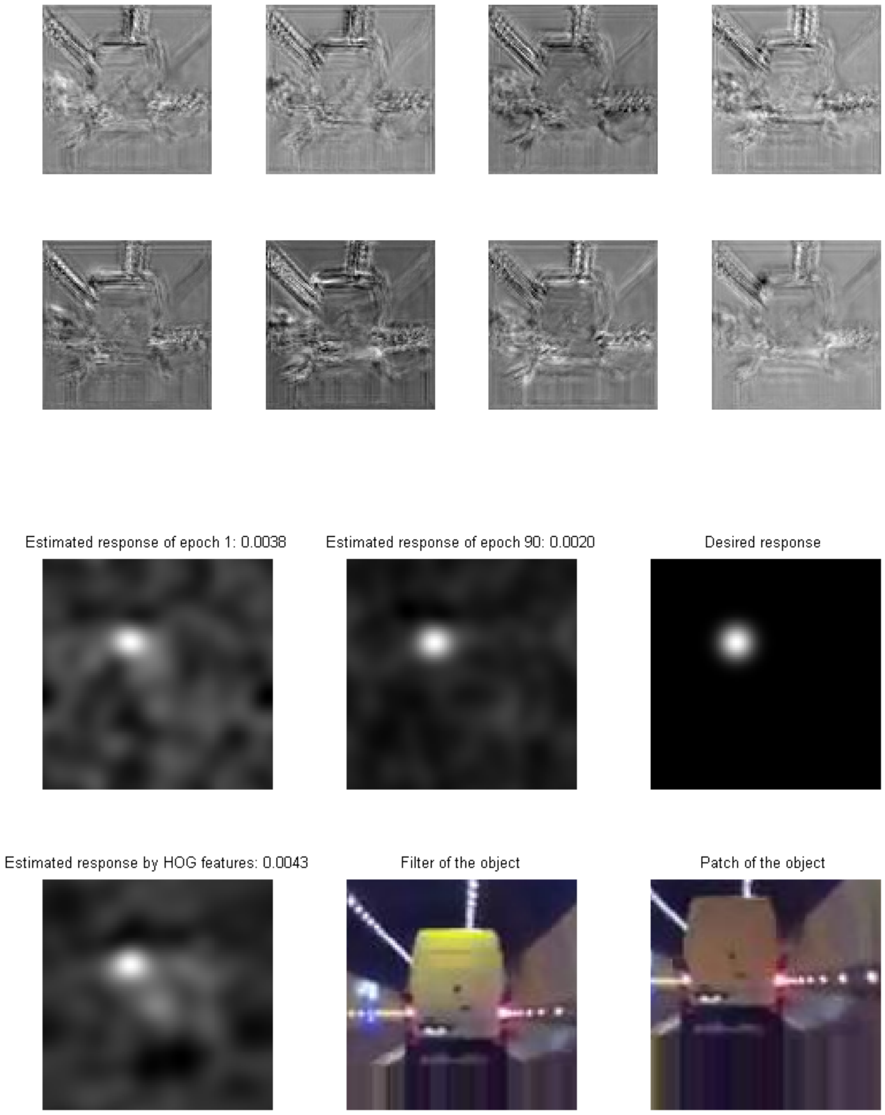

Estimated response of epoch 1: 0.0038
Estimated response of epoch 90: 0.0020
Desired response
Estimated response by HOG features: 0.0043
Filter of the object
Patch of the object

Figure 6.6: Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features.

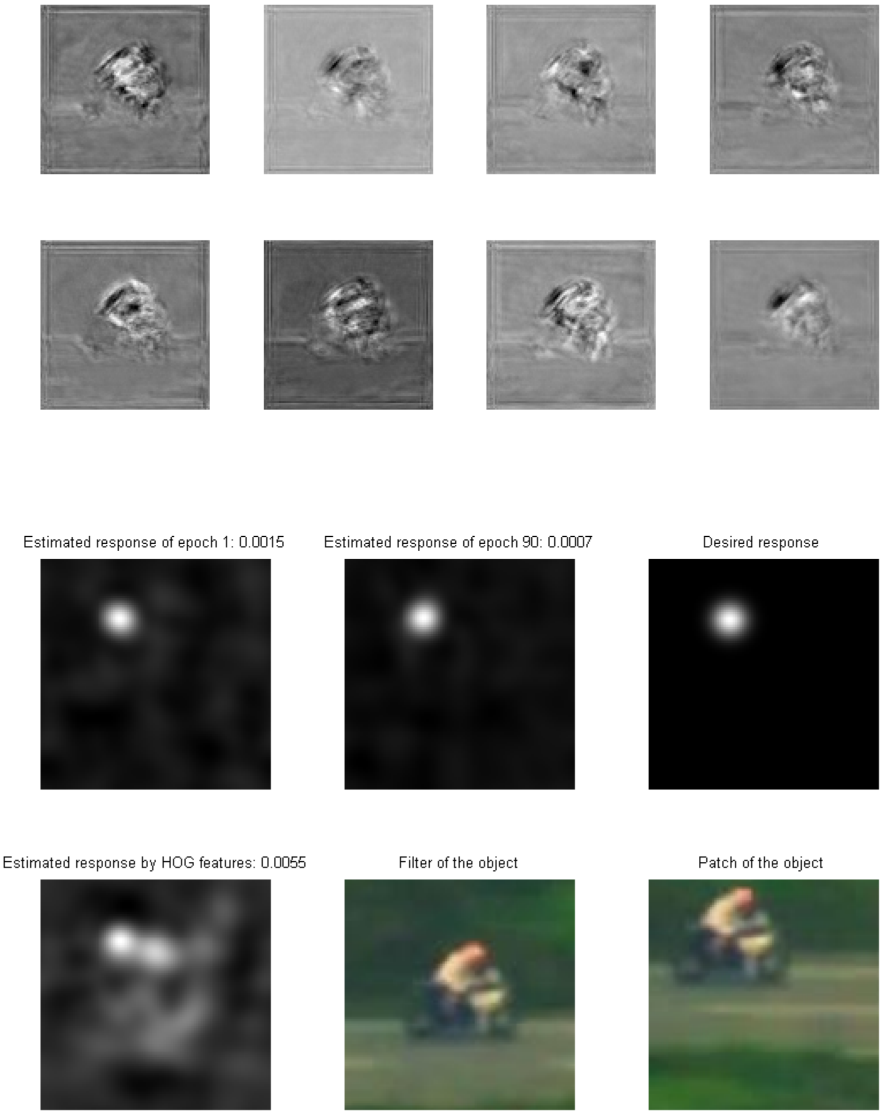


Figure 6.7: Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features.

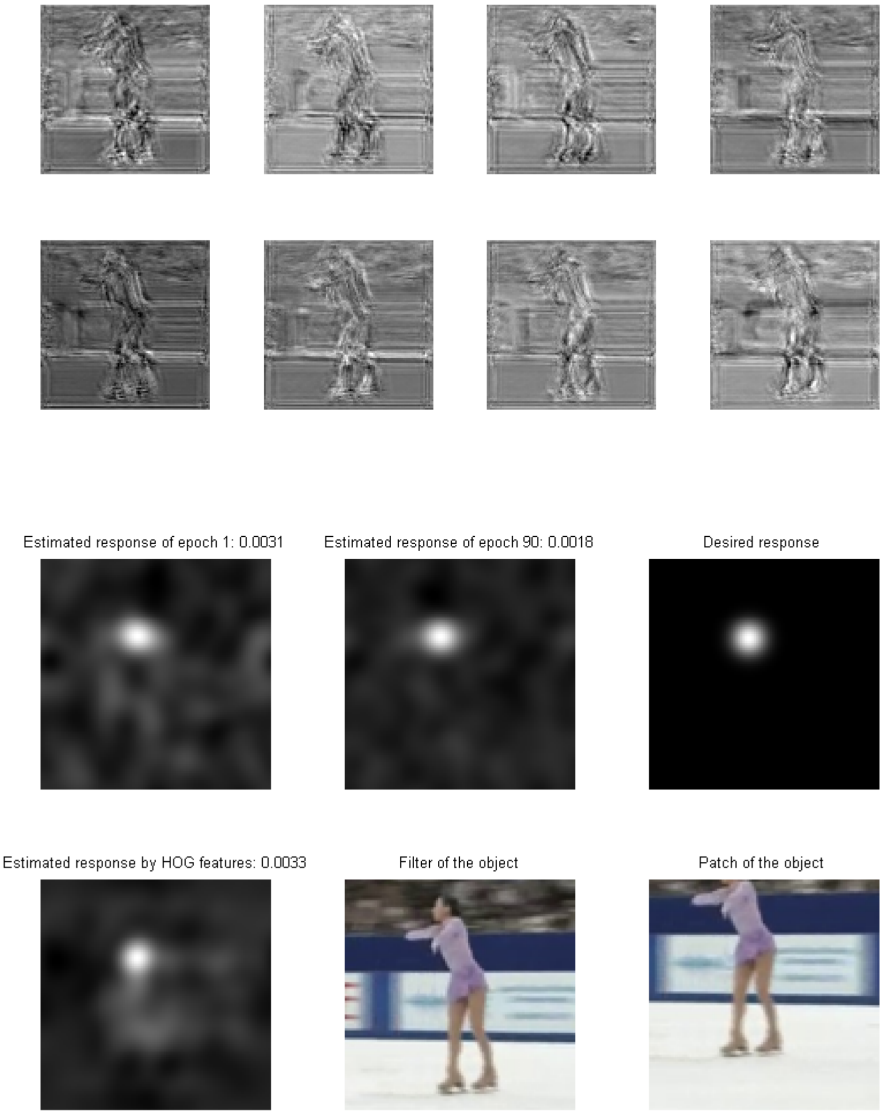

Estimated response of epoch 1: 0.0031
Estimated response of epoch 90: 0.0018
Desired response
Estimated response by HOG features: 0.0033
Filter of the object
Patch of the object

Figure 6.8: Visual illustrations for multiple feature maps. 1^{st} **and** 2^{nd} **rows**: extracted features. 3^{rd} **and** 4^{th} **rows**: MSE scores of the features extracted by the network after the 1^{st} epoch, 90^{th} epoch, desired response and the response by HOG features.

integrated into the state-of-the-art trackers, and compared against top performing trackers on the utilized datasets.

6.7.1 Experimental Results on OTB-2013 Benchmark in Comparison to the Hand-Crafted Features

We conducted tracking experiments on 40 sequences from [111] with 11 different attributes including illumination variation, out-of-plane rotation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out of view, background clutter and low resolution. It should be noted that the selected sequences do not exist on VOT2015 [35]. The trackers are evaluated in terms of the overlap and distance precision scores. Overlap score is the intersection-over-union of the ground truth bounding box and the predicted one. Distance precision is the Euclidean distance between the center of the ground truth and predicted bounding boxes. The plots are drawn for different thresholds and determined as the percentage of frames in a sequence where the overlap or distance precision is larger than a threshold.

In order to analyze the effects of feature maps on the tracking performance, DSST [26] is adopted as the baseline tracker. Different feature type configurations are tested as follows: Proposed CFCF (learned single feature map + gray level intensity + horizontal gradient + vertical gradient), proposed MCFCF (learned multiple feature maps + gray level intensity + horizontal gradient + vertical gradient), DSST (28 HOG feature maps), DSST_GRAY (only gray-level image intensities) and DSST_GRAY_GRADS (gray-level image intensities + the horizontal + vertical gradients). All the parameters of DSST are fixed including the scale estimation. The number of feature maps for these configurations are summarized in Table 6.1. Figure 6.9 demonstrates the overall results in terms of overlap success and distance precision. Figure 6.10 and 6.11 illustrate overlap success and distance threshold performances for different thresholds on 6 attributes from the utilized sequences, respectively.

When we analyze the overall results from Figure 6.9, it can be stated that a favorable performance against DSST algorithm with 28 feature maps is achieved by reducing the number of feature maps 7 times fewer. Moreover, we also have a significant performance increase with respect to the gray-level image intensity usage as well as the

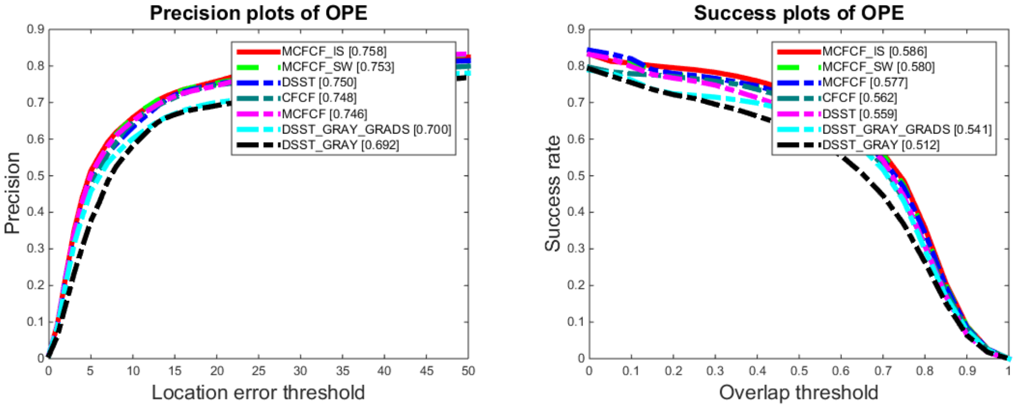


Figure 6.9: Overall results on 40 sequences from [111]. MCFCF_IS is MCFCF with an intelligent search algorithm that performs an object search in a region with a size 4 times larger than the object size, when the tracking quality (Peak-to-Sidelobe-Ratio) is below 10. MCFCF_SW is MCFCF with windowing algorithm of Chapter 4.

gray-level and gradient features. This is the experimental evidence that our network embed most of the necessary information to the intermediate layers and utilize this abstract information when necessary, thus avoiding the redundant use of gradients in all directions as in the case of DSST [26].

Figure 6.9 and the subsequent figures illustrating the per-attribute analysis in Figures 6.10 and 6.11 also demonstrate two extensions of our MCFCF framework. The first extension is our spatial windowing algorithm from Chapter 4, where a spatial window is learned from gray-level image intensities and applied to all feature maps. The details about this tracking methodology and the necessary derivation are already discussed in Chapter 4. The MCFCF tracker with spatial windowing is here called as MCFCF_SW. On the other hand, the second attempt to improve the tracking performance on MCFCF is called as MCFCF_IS which basically stops updating the correlation filter, if the correlation plane quality is below a certain threshold and search the object in a relatively larger area during this particular frame. Hence, it is possible to maintain tracking under different challenging conditions. These improvements are observable from Figures 6.9, 6.10 and 6.11.

Another frequently utilized interpretation of the OTB [111] dataset is mean distance precision (DP) and mean overlap precision (OP). Mean distance precision is the fraction of the frames with a center location error 20 pixels, and mean overlap precision

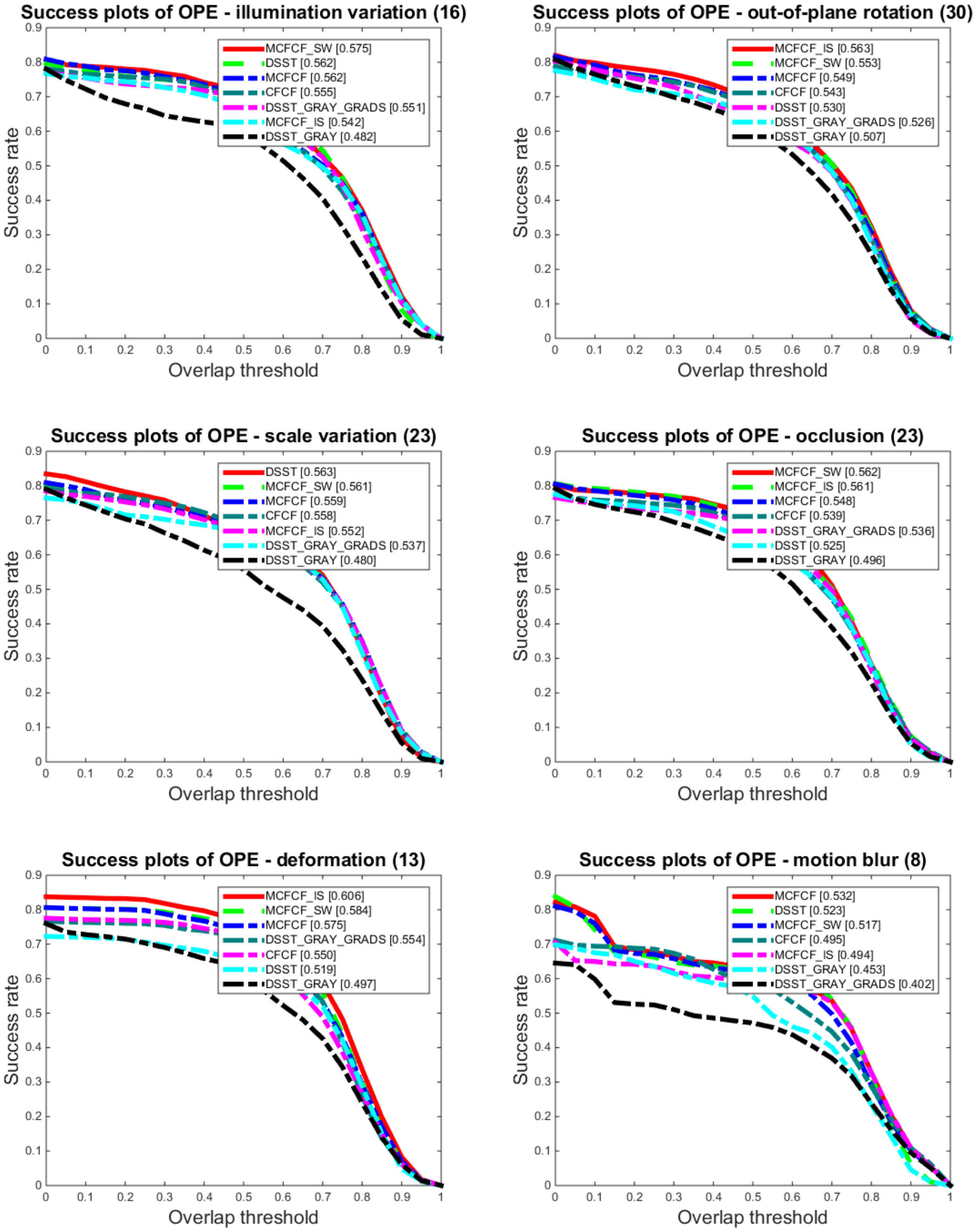


Figure 6.10: Success plots of 40 sequences from [111] for 6 attributes.

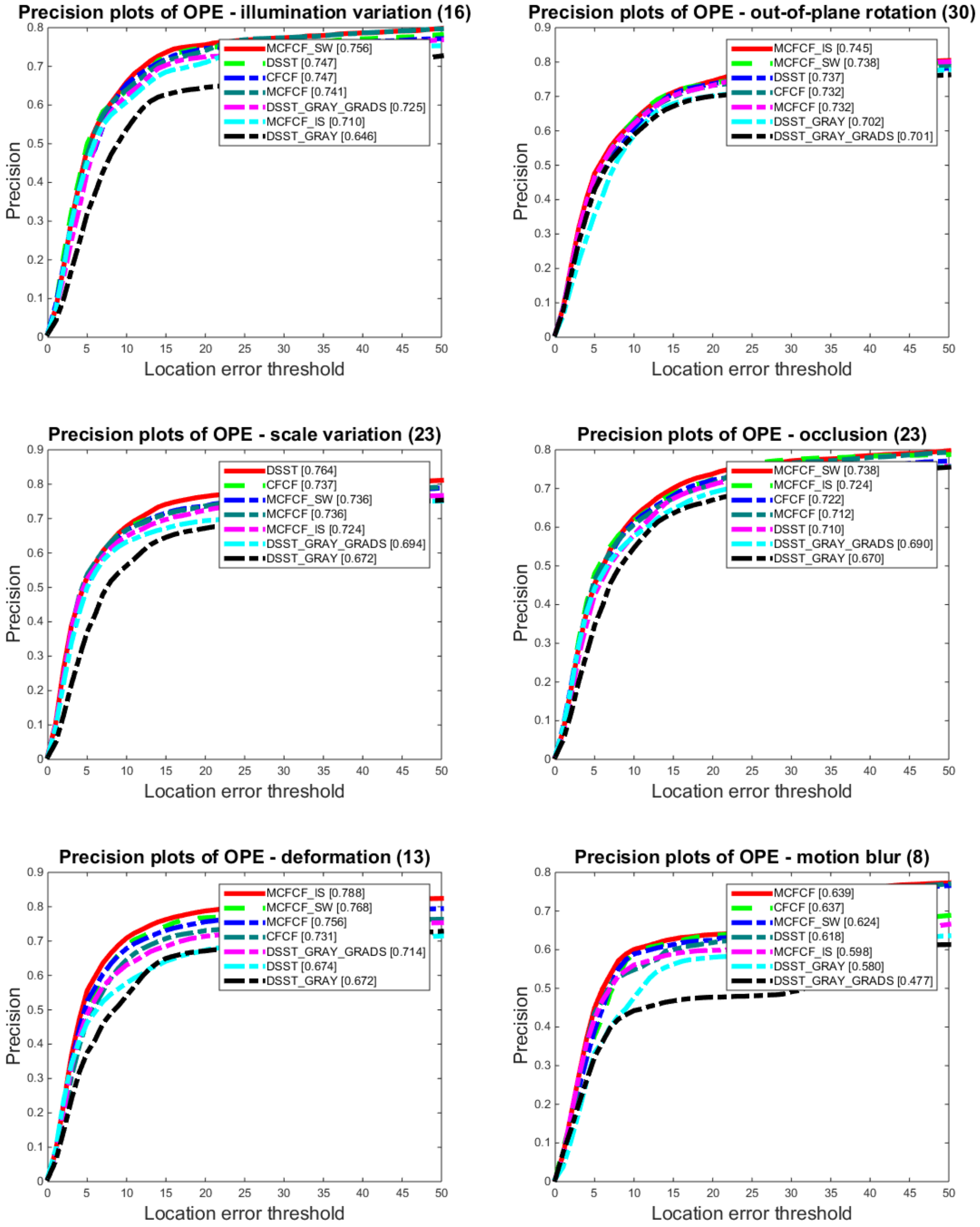


Figure 6.11: Precision plots of 40 sequences from [111] for 6 attributes.

Table6.1: Mean distance precision and mean overlap precision comparison for 40 sequences from OTB [111] for different features.

	DSST_GRAY	DSST_GRAY_GRADS	DSST	CFCF	MCFCF
Mean OP%	62.2	**67.4**	67.3	**70.4**	**70.9**
Mead DP%	69.2	70.0	**75.0**	**74.8**	**74.6**
# of feature maps	1	3	28	4	11

is the percentage of the frames exceeding the PASCAL criterion threshold of 0.5. In terms of these metrics (Table 6.1), proposed method achieves the same performance as DSST with 28 feature maps, while significantly outperforming DSST with only gray-level features and gray-level and gradient features.

6.7.2 Comparison with Respect to the State-of-the-Art Deep Features

For comparing the proposed learned features against recently proposed trackers, CCOT [30] (winner of VOT2016) method, which allows the use of multi-resolution feature maps, is adopted. For this purpose, the last layer features are integrated to the zeroth and first layers after the ReLU part, resulting in 27 feature maps compared to 611 feature maps of CCOT [30]. This configuration is also compared against deepSRDCF [27] (the 2^{nd} best of VOT2015 challenge) utilizing 96 feature maps of [18]. A recent work SiamFC [9], where a fully-convolutional model is trained for sliding window matching, is also compared with the proposed method.

Figure 6.12 presents OPE results on 40 sequences of [111]. Regarding average OP values (the left of Figure 6.12), the proposed 27 feature maps yield a close performance to CCOT with 611 features. Meanwhile, it outperforms deepSRDCF, which utilizes 96 feature maps. On the other hand, the proposed method performs favorably against deepSRDCF and SiamFC in terms of CLE values (the right of Figure 6.12). In Table 6.2, AUC values of OP are presented for 11 attributes. Although CCOT performs favorably against CCOT_CFCF, the feature maps utilized by CCOT_CFCF are significantly less than CCOT (The second column of Table 6.2). It is also notable that the proposed custom architecture is significantly smaller than VGG, employed by CCOT. Moreover, the proposed features perform close to CCOT, such as in the se-

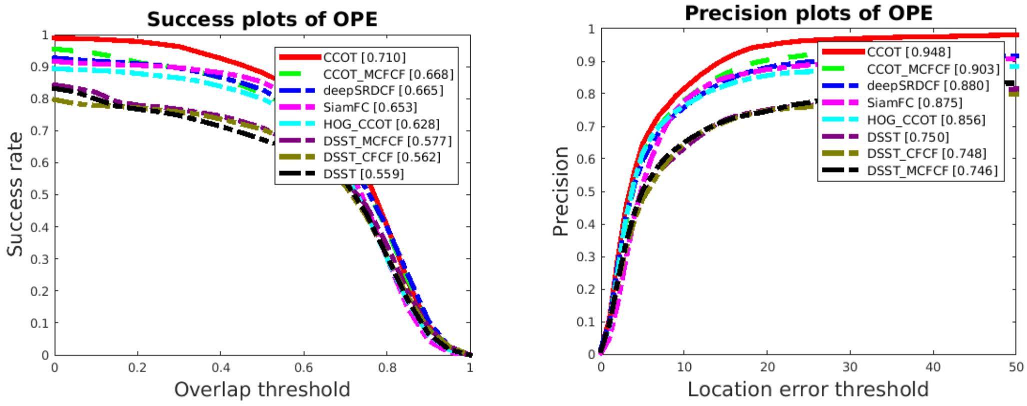


Figure 6.12: One-Pass-Evaluation (OPE) curves. **Left:** Overlap precision (OP) and **right**: center localization error (CLE). Avg. OPs are ordered according to Area-Under-Curve (AUC) while avg. CLEs are in the order of 20 pixel threshold.

Table6.2: AUC values for 11 attributes of the 40 test sequences [111]. FM means feature maps.

	# of FM	**IV**	**SV**	**OCC**	**DEF**	**MB**	**FM**	**IPR**	**OPR**	**OV**	**BC**	**LR**	**Avg.**
CCOT_MCFCF (**ours**)	27	0.63	0.66	0.63	0.70	0.69	0.63	0.64	0.64	0.58	0.65	0.62	0.67
CCOT [30]	611	0.70	0.70	0.70	0.70	0.76	0.71	0.69	0.69	0.76	0.68	0.75	**0.71**
deepSRDCF [27]	96	0.62	0.64	0.63	0.68	0.69	0.64	0.65	0.65	0.64	0.65	0.44	0.67
SiamFC [9]	128	0.57	0.61	0.63	0.60	0.60	0.60	0.65	0.63	0.65	0.60	0.63	0.65
HOG_CCOT	31	0.55	0.58	0.61	0.65	0.64	0.57	0.59	0.59	0.53	0.61	0.53	0.63
DSST_MCFCF (**ours**)	11	0.56	0.56	0.55	0.58	0.53	0.50	0.57	0.55	0.47	0.55	0.56	0.58
DSST_CFCF (**ours**)	4	0.56	0.56	0.54	0.55	0.50	0.47	0.56	0.54	0.50	0.52	0.50	0.56
DSST [26]	28	0.56	0.56	0.53	0.52	0.52	0.49	0.58	0.53	0.47	0.52	0.51	0.56

quences with scale variation (SV), deformation (DEF) and background clutter (BC) attributes.

6.7.3 Fine-tuning the Pre-trained Networks

In the previous two sections, a comprehensive performance analysis has been carried out to show the efficacy of the features learned by the custom network. Nevertheless, the learned model has been trained on a dataset generated by using VOT2015 dataset including 60 video sequences. 11 of the sequences on VOT2015 also exist on OTB-2013 Benchmark dataset [111]. Thus, it prevents the evaluation to be fulfilled on the full OTB-2013 sequences. Moreover, VOT2015 is not a large-scale dataset even though the generated samples are over $200K$. This situation discourages to train or fine-tune the state-of-the-art convolutional networks such as [18, 96]. In order to handle this situation, a new dataset is generated from the large-scale video sequences of ILSVRC challenge dataset [92]. In the 2015 challenge organized by ILSVRC, a new dataset is presented for the challenge called "Object Detection from Video", which has around 4000 videos. In each video, an object from 30 different classes acts and the bounding boxes for each frame is provided. This rich amount of annotated data is utilized to generate our $200K$ triplet samples (the desired response, the localized and unlocalized patches) as it is achieved for VOT2015.

Similar to the training protocol in Section 6.5, the proposed cost specifically targeting the correlation filter based tracking error is minimized by using the backpropagation. Unlike the network described in Section 6.6, VGG-M network proposed in [18] is exploited in such a way that this network is cut from the first fully-connected layer (*fc6*), since the built framework only accepts the convolutional layers due to their shift invariance property. Although any other network model could be selected, VGG-M is fine-tuned to fairly compare against the CCOT tracker [30], which utilizes the zeroth, the first and the fifth convolutional layers of VGG-M. In order to train convolutional layers of VGG-M, an auxiliary layer with 32 feature maps is added as the layer to be optimized by the CFCF cost function. This augmentation is necessary because the final convolutional layer of VGG-M has 512 feature maps and the training with respect to the proposed loss becomes infeasible. By the auxiliary layer with 32 layers, the

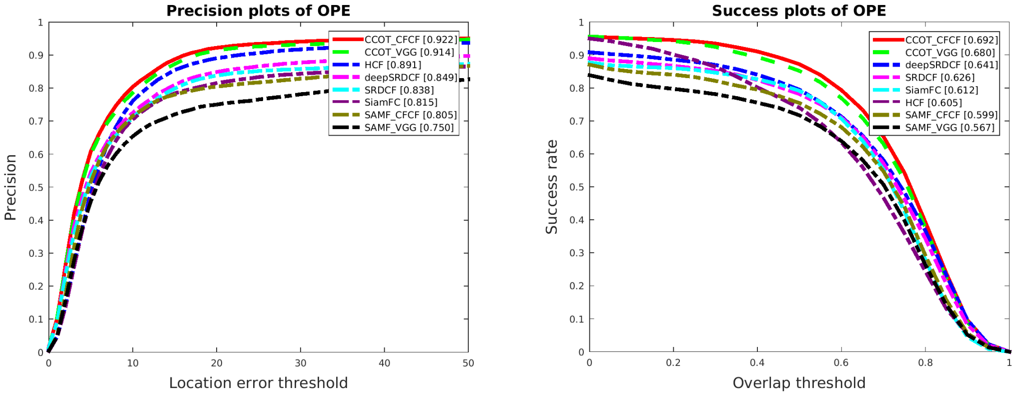


Figure 6.13: Average precision and success plots of 51 sequences in [111].

complexity is significantly reduced. The major assumption that is worth discussing here is the claim that the reduction of the cost in the top layer also helps to decrease the cost in the lower layers at least by the same amount. In Appendix A, this issue is addressed by showing the approximate equivalence of the correlation response error for the two consecutive layers.

The fine-tuned VGG-M network is integrated into the CCOT tracker by using its default hyper-parameters except for the number of iterations to find the filter. The default iteration number of CCOT is 5, whereas only 1 conjugate gradient iteration is performed except for the first frame (which has 100 iterations in both our case and the baseline CCOT configuration). Our preliminary experiments have shown that more than 1 iteration deteriorates the performance of the learned features due to the fact that the trained network becomes more robust to appearance and pose changes of the object and invariant to the changing conditions, such as illumination. Hence, the learned features help to double the computation speed, as the fps values are reported in Figure 6.21 for 100 videos of OTB-2015. The fps is measured in Intel Xeon E5 2623 3.0 GHz except for the CNN feature extraction which is performed by Nvidia Tesla K40 in MatConvNet [105].

The fine-tuned VGG-M network is tested on the full 51 sequences of OTB-2013 [111] and on the full 100 sequences of OTB-2015 [112], which is a superset of OTB-2013. The Figures 6.13 and 6.17 present the success and precision plots.

Moreover, Figures 6.14, 6.15, 6.16 and 6.18, 6.19, 6.20 compare the features by fine

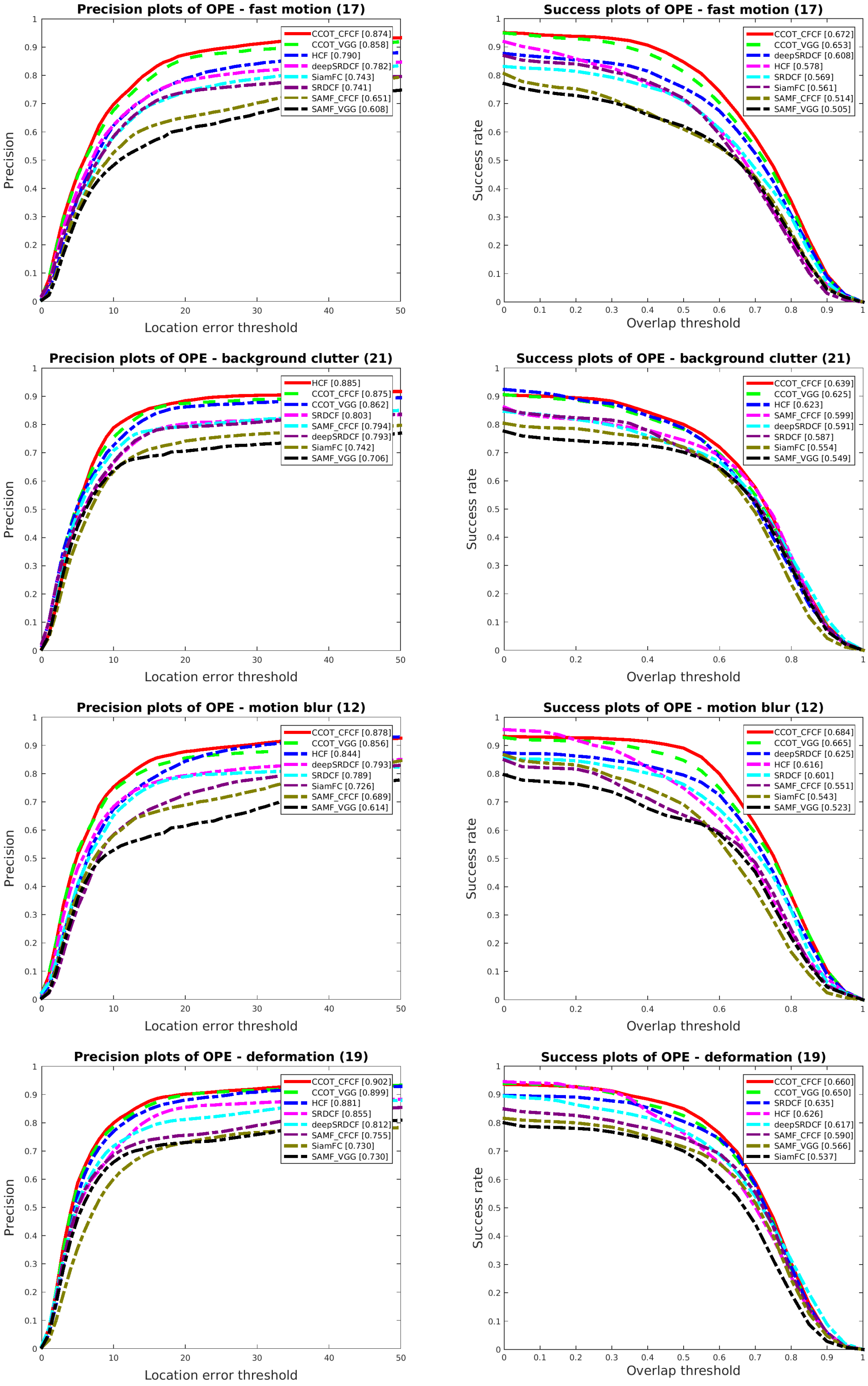


Figure 6.14: Precision and success plots of 51 sequences in [111] for 4 attributes: fast motion, background clutter, motion blur and deformation.

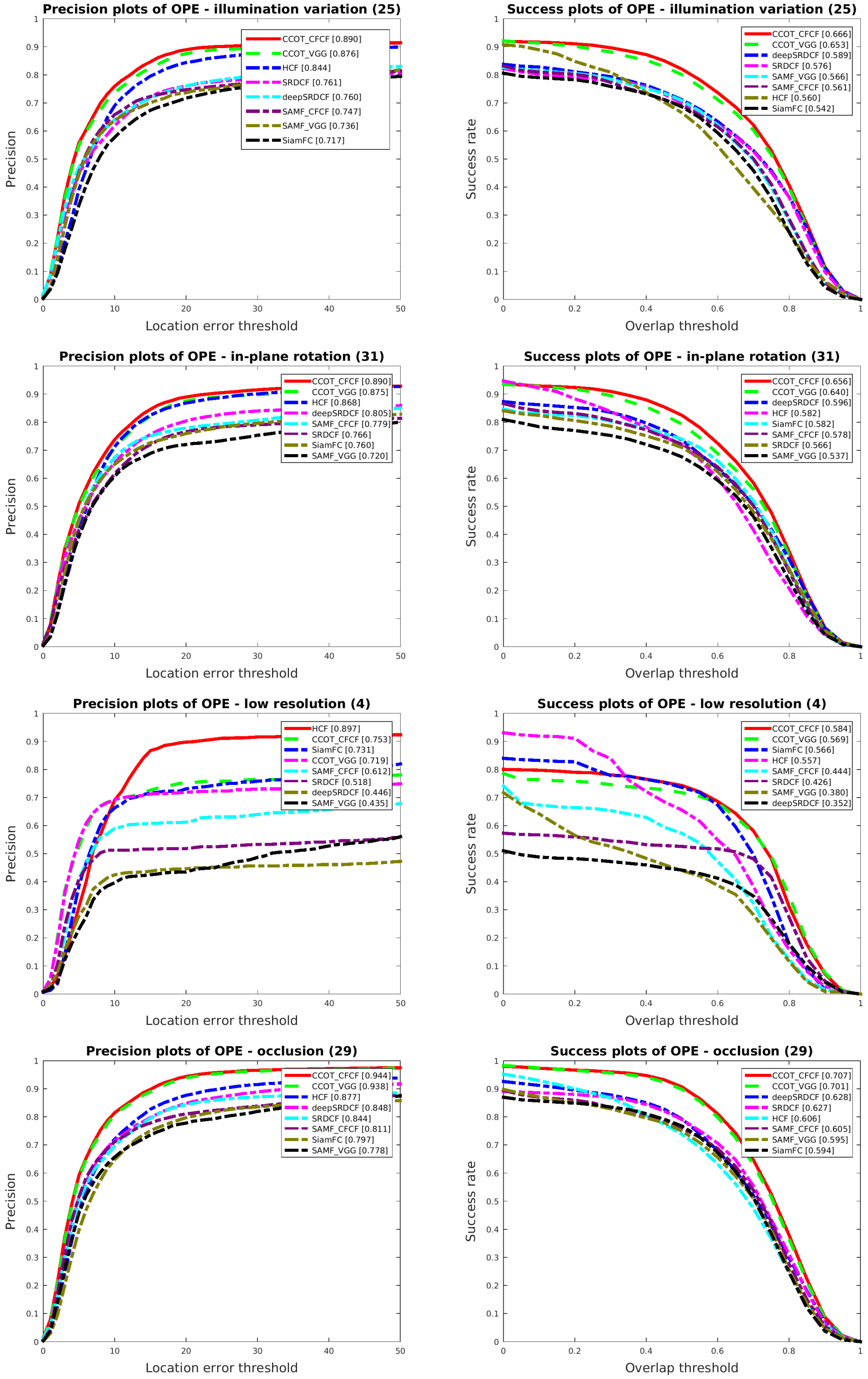


Figure 6.15: Precision and success plots of 51 sequences in [111] for 4 attributes: illumination variation, in-plane rotation, low resolution and occlusion.

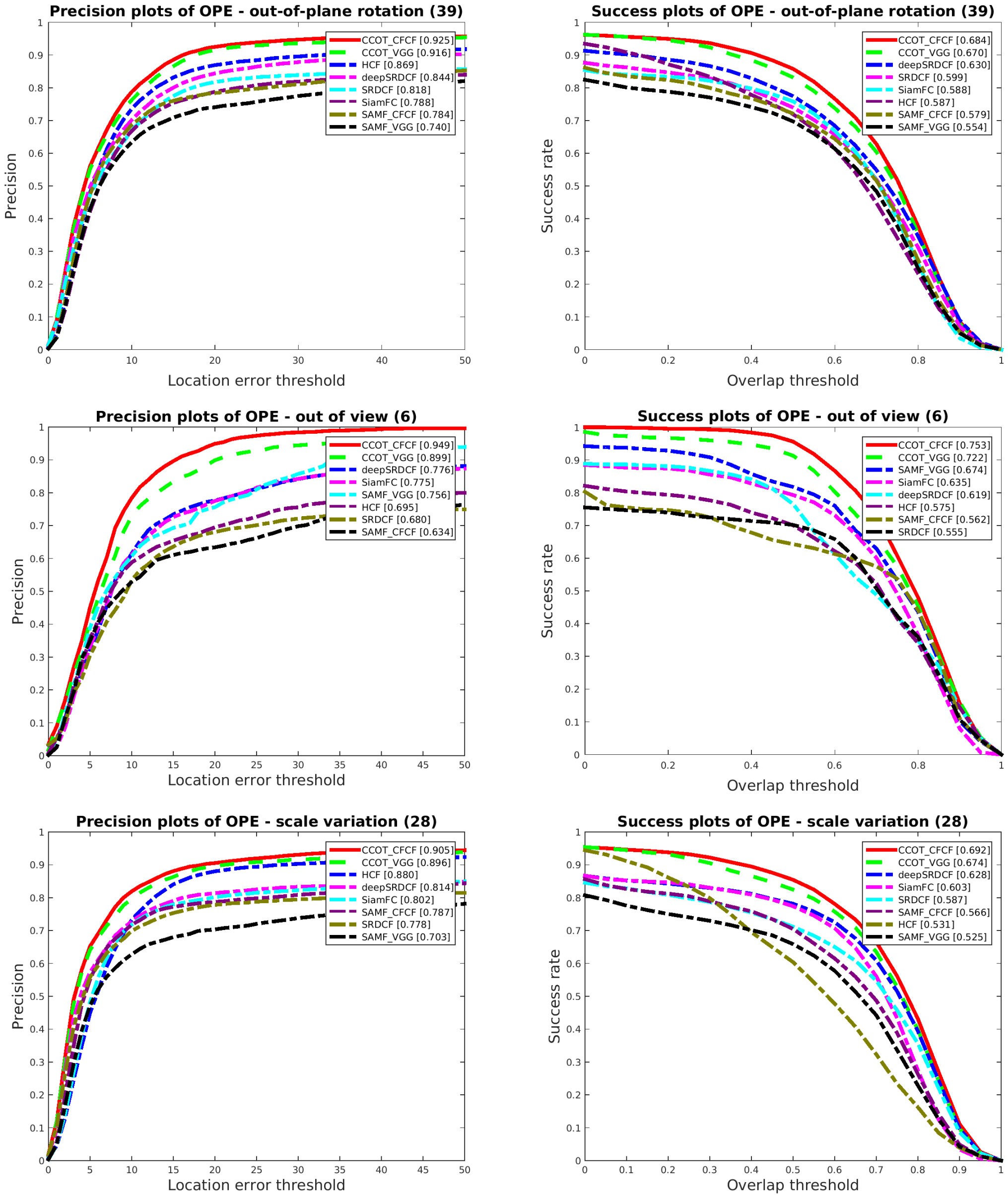


Figure 6.16: Precision and success plots of 51 sequences in [111] for 3 attributes: out-of-plane rotation, out-of-view and scale variation.

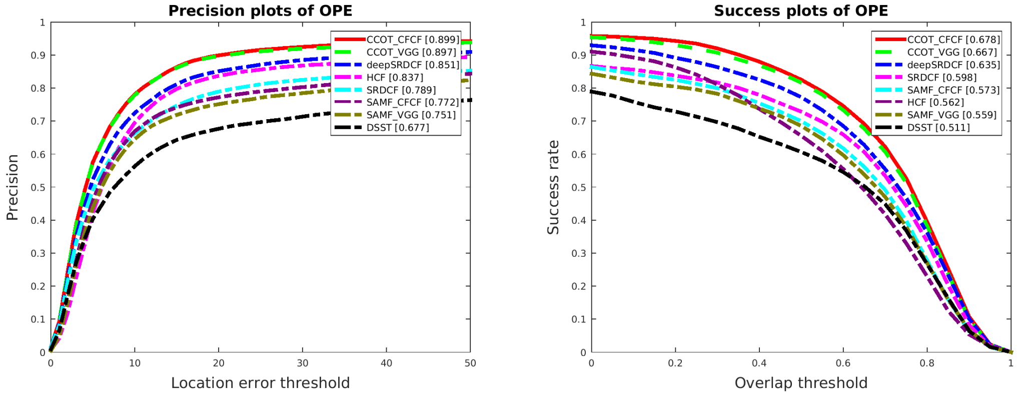


Figure 6.17: Average precision and success plots of 100 sequences in [112].

tuning VGG-M network with the proposed feature learning framework. For this purpose, the pre-trained VGG and the fine-tuned VGG networks (short for CFCF) are integrated into two state-of-the-art trackers. The first one is CCOT [30]. The baseline tracker [30] and the tracker with the fine-tuned features are named as CCOT_VGG and CCOT_CFCF, respectively. Secondly, SAMF [69] is utilized to integrate our learned features and named as SAMF_CFCF. SAMF [69] is the multi-scale extension of KCF [48] tracker that originally utilizes HOG orientations. The zeroth, first and fifth convolutional layer outputs are utilized after the ReLU layer as in CCOT [30]. Moreover, the pre-trained VGG network [18] trained on ImageNet [32] object recognition dataset for classification task is also integrated into SAMF, and this configuration is named SAMF_VGG.

Among the remaining state-of-the-art trackers, SRDCF [28] utilizes HOG orientations maps, and deepSRDCF [27] integrates deep VGG features into SRDCF tracker. DSST [26] is the winner tracker of VOT2014 challenge with an efficient scale search technique and it utilizes HOG orientation maps. Moreover, SiamFC [9] is a fully convolutional Siamese architecture that learns to estimate the object location when the two inputs to the network are an image template and a test frame. HCF [76] exploits convolutional layers of a CNN architecture by hierarchically utilizing the correlation filters designed for each layer.

The results implies that a considerable performance increase is obtained with the proposed features compared to CCOT and SAMF, which takes the features extracted

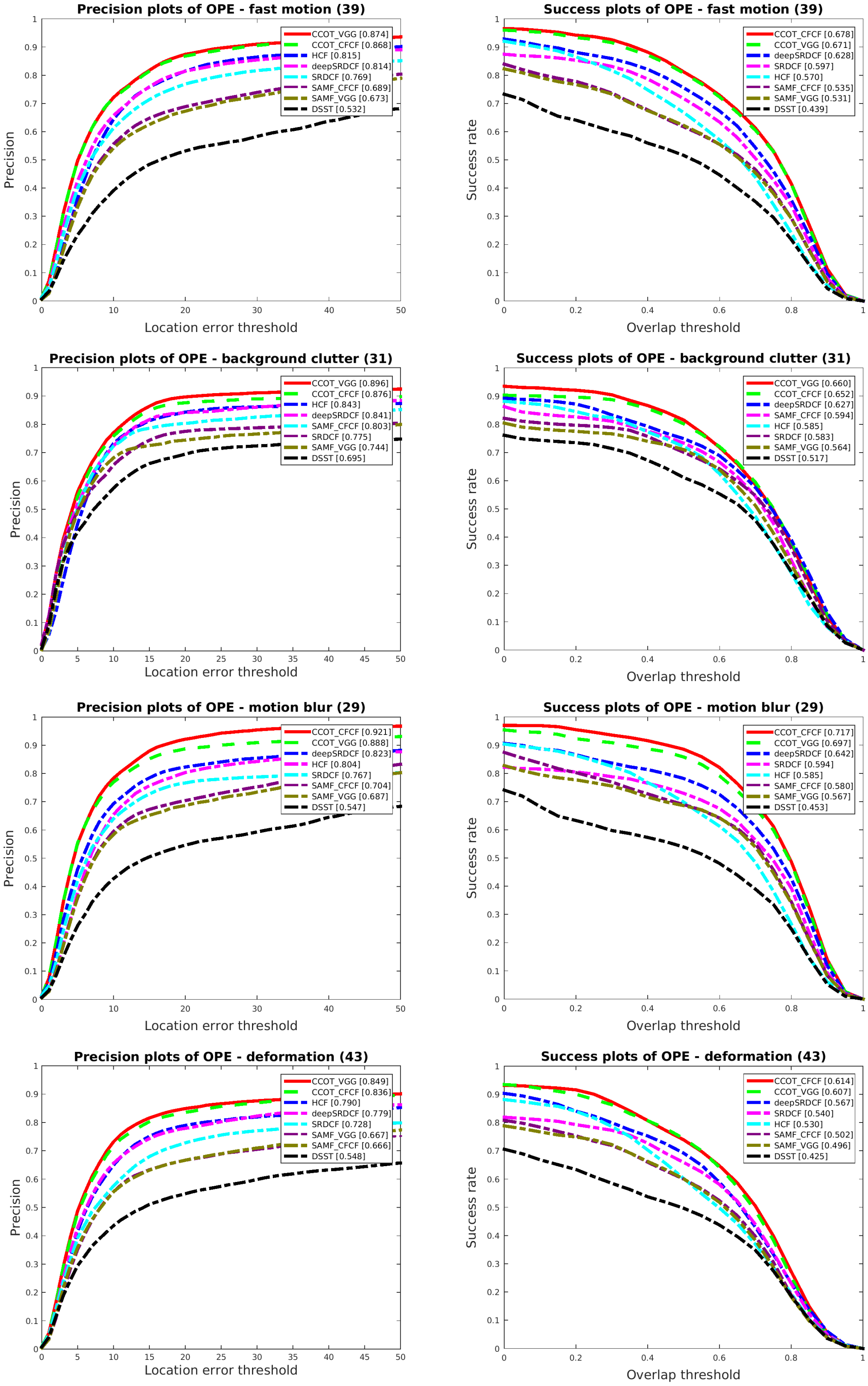


Figure 6.18: Precision and success plots of 100 sequences in [112] for 4 attributes: fast motion, background clutter, motion blur and deformation.

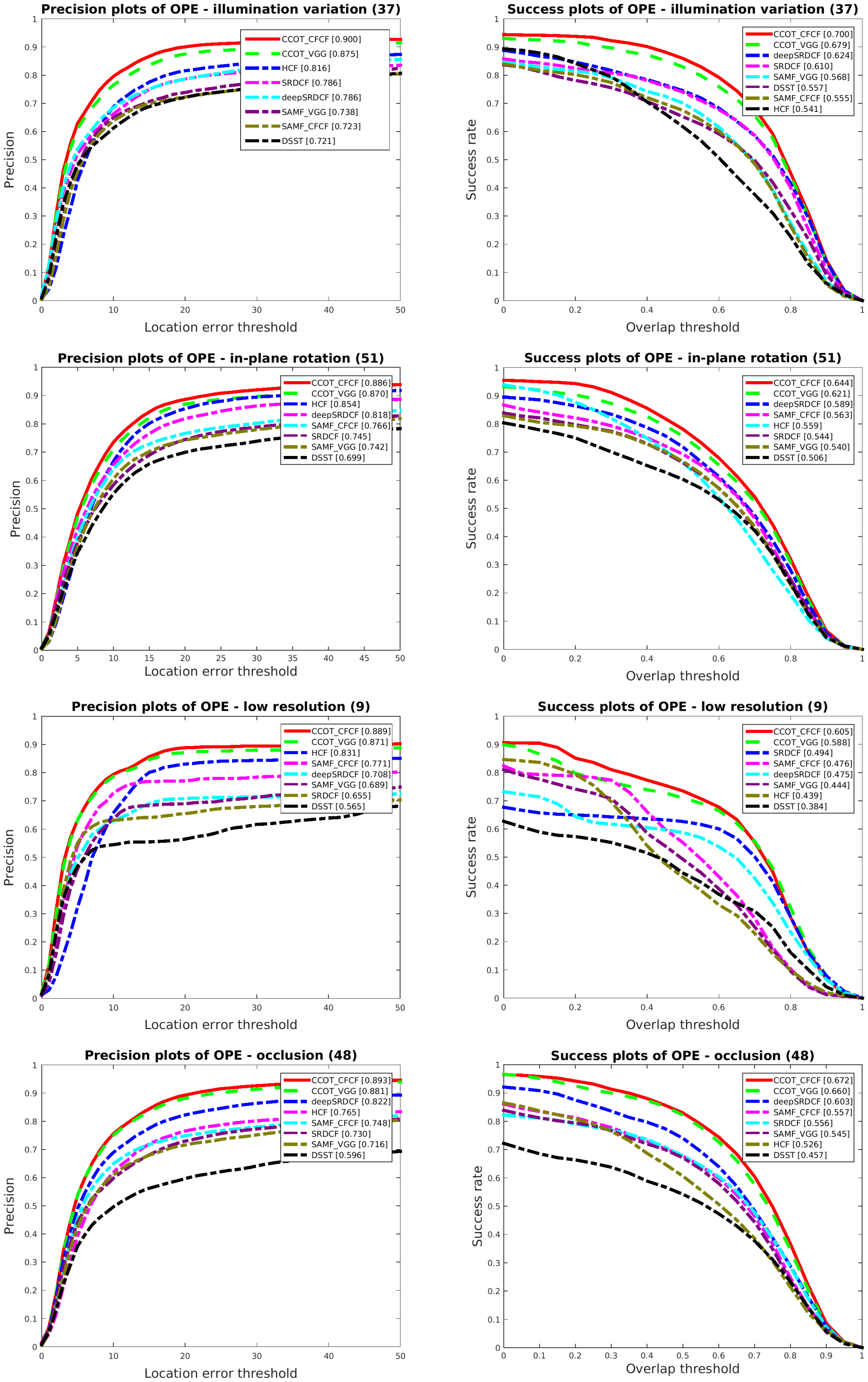


Figure 6.19: Precision and success plots of 100 sequences in [112] for 4 attributes: illumination variation, in-plane rotation, low resolution and occlusion.

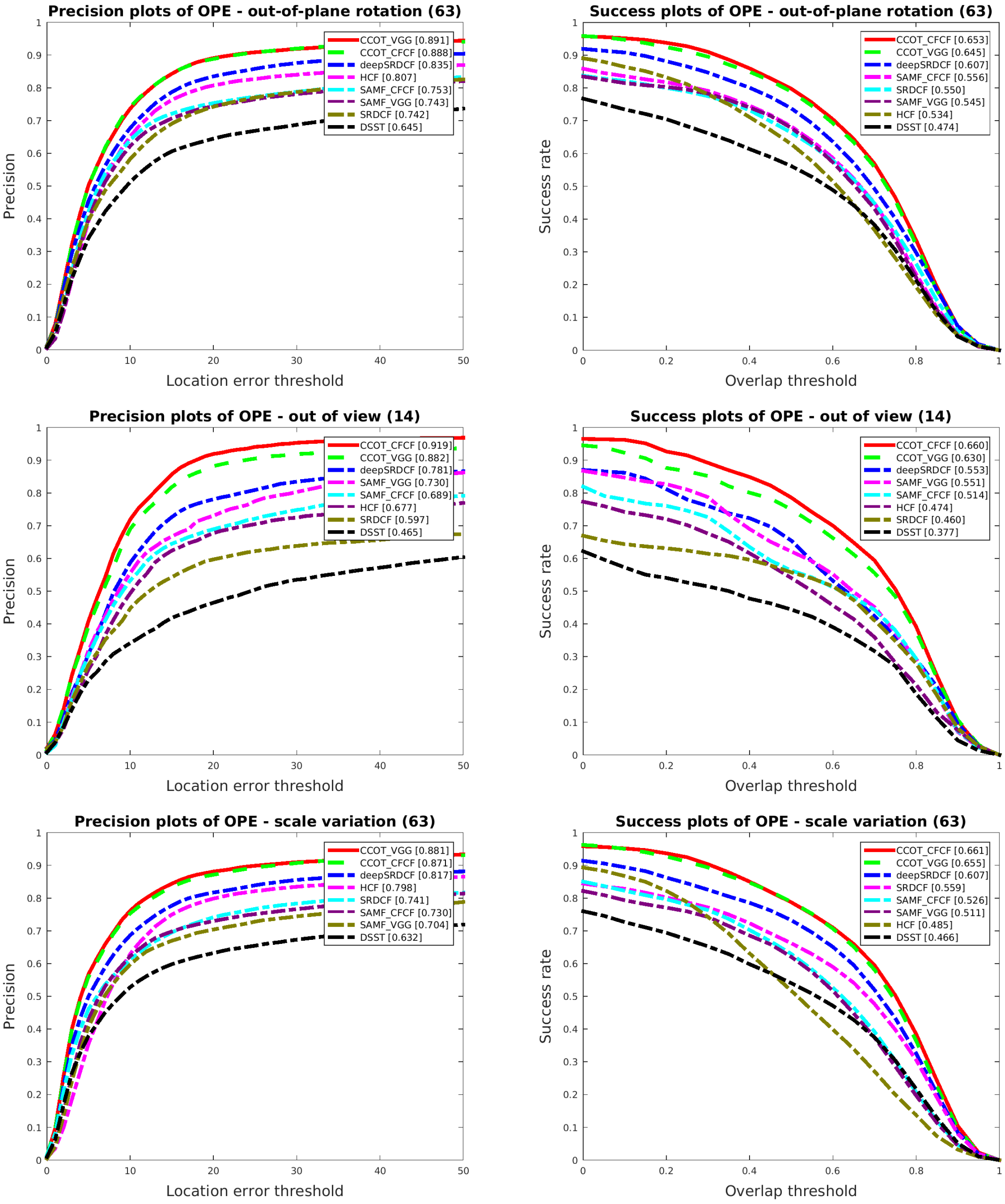


Figure 6.20: Precision and success plots of 100 sequences in [112] for 3 attributes: out-of-plane rotation, out-of-view and scale variation.

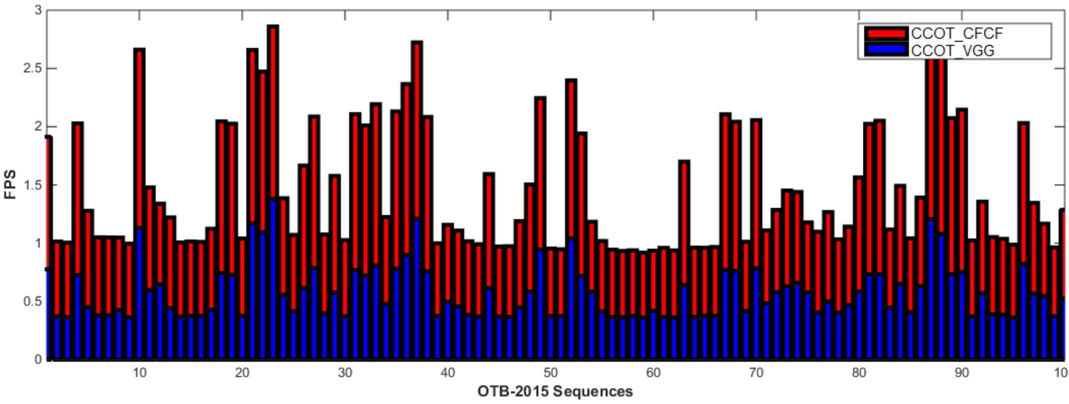


Figure 6.21: Speed comparison between CCOT_VGG (baseline) and the proposed tracker CCOT_CFCF on OTB-2015 sequences.

from the pre-trained network of VGG-M. CCOT is the winner tracking method of VOT2016 challenge, SAMF is the top second tracker of VOT2014 challenge and SRDCF with deep features is the top second tracker of VOT2015 challenge. Integrating only our features into the winners, SAMF and CCOT, and putting no further effort, the tracking performance is leveraged by a significant amount.

6.7.4 VOT2016 Performance Analysis of the Fine-tuned VGG-M

The proposed features integrated to CCOT has also been tested on VOT2016 challenge dataset including 60 videos.

The VOT2016 has quite a different tracking assessment technique including three major metrics. 1) Accuracy is the mean intersection over union of the frames in a sequence. 2) Failure is the mean number of failures per sequence. These two metrics are the raw metrics. The ranking of a particular metric (failure or accuracy) is measured by ordering the compared trackers with respect to that metric, and the statistically significant tracker rankings are merged. 3) Expected average overlap (EAO) is estimated for a selected range of sequence lengths. Concretely, a specific expected average overlap ϕ_{N_s} is estimated by averaging the accuracy values in the segments that are longer than N_s while discarding the segments shorter than N_s with no failure termination. The segments shorter than N_s with a failure are zero-padded, hence penalizing the failure case for that particular N_s length. These ϕ_{N_s} values are determined for the set $\{\phi_{N_s}\}_{N_{lo}}^{N_{hi}}$ and the final score is the mean of these expected

Table6.3: VOT2016 performance results

Trackers	**EAO**	**Acc. Rank**	**Rob. Rank**	**Acc. Raw**	**Fail. Raw**
CFCF	**0.3903**	1.98	2.27	**0.54**	**0.63**
CCOT [30]	0.3310	2.55	2.95	0.52	0.85
TCNN [83]	0.3249	1.97	3.92	0.54	0.96
SSAT [84, 36]	0.3207	1.62	3.80	0.57	1.04
MLDF [36]	0.3106	3.70	2.82	0.48	0.83
Staple [8]	0.2952	2.57	4.83	0.54	1.35
DDC [36]	0.2929	2.27	4.62	0.53	1.23
EBT [129]	0.2913	5.07	2.88	0.4	0.90
SRBT [36]	0.2904	3.73	4.47	0.50	0.125
STAPLEp [36]	0.2862	2.03	4.42	0.55	1.32
DNT [36]	0.2783	3.03	4.47	0.50	1.18

values in the set. Selection of N_{lo} and N_{hi} are performed according to the sequence length histogram of the dataset.

In order to make a fair comparison between the fine-tuned VGG features for our loss function and the VGG features utilized by CCOT, VOT2016 challenge configuration of CCOT is utilized. In that configuration, the zeroth, first and fifth convolutional layers of VGG are employed as well as the color names of [29] with 11 features and 31 HOG gradient maps in [28]. To keep the legends simple in the figures and tables, the proposed tracker configuration CCOT_CFCF is abbreviated as CFCF.

Table 6.3 reports the performance results of VOT2016 challenge. Among 71 participants of this challenge, we only show the top ten trackers and the proposed tracker CFCF ordered by the EAO metric, which unifies the robustness and accuracy of the trackers.

In Figure 6.22, the ranking results in Table 6.3 is pictured within a 2-dimensional plot. As the figure shows, the proposed tracker significantly outperforms all of the existing participants. Moreover, the proposed features improves the top tracker CCOT by

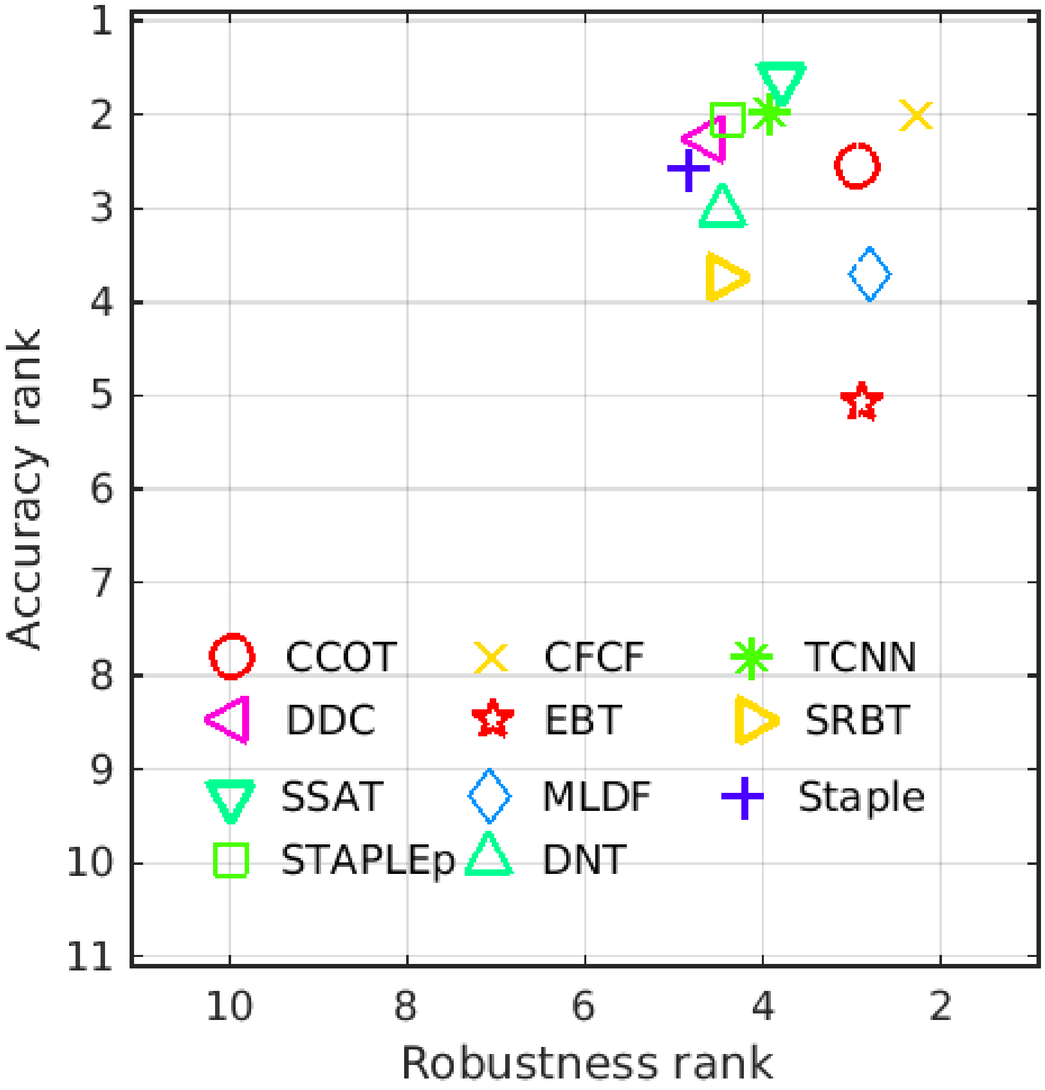


Figure 6.22: Accuracy-Robustness Ranking plot. Closeness to the top right indicates good tracking performance.

18.7% in terms of EAO. On the other hand, the number of optimization iterations is reduced to 1 from 5, bringing a significant decrease in the computation time. It is also attractive that the number of failures is decreased by 25% with respect to the CCOT. The raw accuracy performance is also improved by at least 3.5%.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Summary of the Thesis

This thesis addresses the problem of single object visual tracking, which mainly aims to estimate the state of a target object at each frame of a video when the first state is provided by the oracle. In particular, correlation filter based tracking methods are focused due to their notable performances in benchmark datasets, their efficient computational complexity for localization and simple model update. In order to leverage the tracking performance, three major frameworks have been proposed.

In the first proposal, the object patches are perceived as signals. The goal is to increase the correlation quality of the model filter and the object patch. This requirement results from the assumption that particular regions of the object are not appropriate to correlate, while some other regions are invariant to appearance changes of the object, hence they have the potential to increase the correlation quality. That said, a spatial window which suppresses or highlights the parts of the object patch or the correlation filter in the image domain is learned at each frame. For this purpose, an efficient gradient formulation for the spatial window is proposed with respect to the correlation quality of the signal. If the estimated spatial window by the gradient descent reduction is element-wise multiplied by the object patch or the correlation filter model in the image domain, a considerable increase is observed in the tracking accuracy and robustness. The conducted experiments are important evidences of the aforementioned assumptions. Moreover, the obtained windows have qualitative interpretations; concretely, the occluded part of the object patches have low values in these masks, whereas the salient regions such as edges or blobs typically have high values.

In the second proposal, a machine learning perspective is adopted by treating the object patches as points in the high dimensional feature space. Most correlation filter based tracking approaches, especially the pioneering studies MOSSE [14] and DSST [26], occupy a single model, *i.e.* a single correlation filter, which is updated by weighted averaging the individual correlation filters throughout the video frames. Nevertheless, the averaging of the correlation filters at every frame might have disadvantages. For instance, the correlation filter for the object patch of a woman dancing with her hands on her side and the correlation filter of the same woman leaned over the knees are averaged. Such an averaged correlation filter most likely deteriorates the performance of the localization in both poses of the woman, if they are encountered again during the video stream. A promising alternative is to keep record of separate models in an ensemble. Yet, the number of models, their inclusion and removal to the ensemble are cumbersome and prone to the hyper-parameters of such a framework as in [120]. To handle this situation, we organize the ensemble of trackers in a tree-structure such that the individual correlation filter models are stored in the tree nodes. Moreover, an effective combination algorithm that considers all of the possible combinations in a tree is proposed.

The presented ensemble of trackers method is able to handle the accurate localization of the target objects especially in challenging conditions, such as object deformations and occlusions, since only the relevant and most similar models to the instantaneous object appearances are activated by the proposed combination algorithm. The extensive evaluations in benchmark sequences have validated the effectiveness of the proposed framework. In addition, the spatial windowing methodology of the first proposal is also integrated into the ensemble tracking framework. The combined method performs significantly better than both of the proposals. Thus, one can experimentally state that the two frameworks have uncommon advantages and complementary to each other.

Both of the proposed methods that are summarized above are designed to increase the capabilities of the correlation filter based tracking techniques while further improvements are concurrently proposed in various studies, such as in [73] [10], [52], [28], [27], [30] and so on. Nevertheless, none of them explicitly takes care of beneficial features for the particular task, *i.e.*, the correlation filter based tracking, although few attempts [76, 27] have been proposed in the past to make full use of the features extracted from the pre-trained deep network model of the object recognition task. Hence, in this thesis, a third novel technique is proposed in order to train a deep fully convolutional CNN network for the correlation filter based tracking loss function. An efficient backpropagation algorithm is presented to train the deep model by using the stochastic gradient descent exploiting the Convolution Theorem for the circular convolution operation of the discrete signals. The reported results demonstrate that the learned features are much more amenable for the correlation operation than the pre-trained networks utilized by the state-of-the-art methods, such as [27] and [30].

7.2 Conclusions

The major success of the correlation filters for the localization of the object is due to the fact that dense samples are inherently learned. In other words, the correlation filter based tracking formulation regresses a desired correlation response for every possible shift of the training examples. At this point, a disadvantage arises: Existence of imperfect training samples. In order to generate training samples for every shift of the signal, the circular translation of the object patches are assumed to be the actual translations of the object. Training examples especially in the boundaries become far from being the actual ones. Hence, the boundaries of the object patches should be carefully considered. To handle the above issue, the studies in [41] and [28] cope with the boundaries by proposing the regularization of the boundaries. On the other hand, this issue is addressed by the proposed window learning method which is also a good candidate to suppress the boundaries of the object patch as well as to increase the correlation quality. It can be concluded that such a window-based approach improves the tracking quality of the conventional techniques. The experiments conducted on VOT2015 dataset have demonstrated that the proposed windowing method decreases the number of failures by 6%. Moreover, it improves the tracking accuracy over 2.5% on the OTB-2015 dataset.

Another important point that should be taken into account is the search range trade-off. If an object patch is cropped S times larger than the actual size of the object, and S is sufficiently large, then the tracker failures might be decreased in abrupt motion scenarios. Yet, the possibility of the tracker to jump to a similar object increases. This problem is also handled by the boundary regularization or window learning up to a certain degree. Nonetheless, as the appearance of the target significantly changes in the cases of severe object deformation, a single model is no longer adequate. Thus, one can wisely store multiple models similar to the proposed tree-structured ensemble tracking method. The reported results reveal that the presented ensemble tracker and its combination with the proposed window learning method significantly decreases the average number of failures by 23% without sacrificing from the accuracy of the tracker on VOT2015 dataset.

Although the utilization of multiple models is a powerful selection to handle the appearance change of the object, it is not a desirable alternative due to the practical issues, such as memory requirements and computational complexity, even though the proposed ensemble tracking method efficiently registers the models in the nodes of the tree. The proposed ensemble framework requires to run multiple trackers independently. Hence, it is possible to design an implementation to operate individual active expert trackers of the ensemble in multiple threads. By this way, it could be possible to avoid the slow computation time that depends on the depth of the tree.

The existing challenges includes the followings: abrupt motion, illumination change, scale variation, motion blur, partial occlusion, out-of-plane rotation, in-plane-rotation and deformation. Abrupt motion and deformation are partially handled with the proposed methods in this thesis as well as the state-of-the-art studies. However, the failure is inevitable in the cases where multiple difficulties exist in a sequence. Thus, a tracker should be able to solve a subset of these challenges to a great extend. For instance, if the designed tracker is convincing in terms of its robustness to severe deformations, then it could be much easier to handle the abrupt motion or scale variation. Thus, the proposed feature learning method targets at making the tracker as invariant as possible to the appearance changes of the object such as deformation and out of plane rotation. In such a case, it would be possible to consistently increase the tracking accuracy in most of the video sequences of the benchmark datasets.

It is notable that the integration of the learned convolutional feature maps to the state-of-the-art trackers DSST [26], SAMF [69] and CCOT [30] significantly increases the tracking accuracy by 3.4% (in OTB-2013 dataset), 1.4% (in OTB-2015 dataset) and 5.6% (in OTB-2015 dataset), respectively, in terms of the area-under-curve of the success plots. Remarkably, the learned features increase the tracking performance in Visual Object Tacking 2016 challenge dataset by over 18% in comparison to the pre-trained object classification network utilized by the winner of this challenge. This is the best result reported so far in tracking literature in terms of the VOT2016 Expected Average Overlap metric, which ranks the participating trackers in that challenge.

7.3 Future Work

In order to compete with the existing state-of-the-art trackers, the computational efficiency is somewhat sacrificed. If the tracking speed were concentrated, the quantitative performance values would be almost halved, such as the pioneering work of [14]. As the cutting edge performance gains are being obtained, the experience gathered from these gains should be transferred to the real-time or near real-time trackers. Our convolutional feature formulation can be considered as an example of transferring the deep convolutional networks to the real-time trackers, since our light weight and custom design leverages the tracking accuracy and it is convenient to utilize these features in a multiple channel correlation filter based tracker with few feature maps learned in a specific domain for the correlation task.

The proposed feature learning methodology is scalable to simpler or complex networks. Hence, any custom design is possible in other domains. For instance, deep fully convolutional networks can be trained in infrared images for the correlation task without confining the tracker to use pre-trained features of the object recognition task for the RGB cameras. It is also notable that a surveillance system requires more than one task including visual tracking, mid- and high-level action representations, object detection and recognition. For such a requirement, a deep network can be trained in a multi-task learning framework, where the fully convolutional parts are trained for both the correlation task and the object detection.

When the computational resources of a system are so limited that no time-consuming feature calculation is possible, the proposed tree-structured ensemble tracking method can be employed with limited number of tree nodes to model the existing poses of the object. Moreover, the tree pruning could be performed if the system is sure about the incorrect model in a tree node. As it has been demonstrated in the experiments, the depth of the tree is able to increase the tracking performance to a certain degree. The depth value at which the performance saturates is affected by the difficulty of the sequences and their lengths. Hence, the depth of the constructed tree is a controllable hyper-parameter, and it can be adjusted according to the performance and speed requirements.

The proposed window learning method can be combined with other existing correlation filter based tracking formulations to further increase the performance. One potential use of the window learning approach is that a surveillance system can learn windows offline or online for specific target object types. For instance, it is possible to suppress the propellers of a helicopter by the proposed windowing technique, since the propeller locations rapidly change and inappropriate for the correlation task. Another alternative is to design application specific hardware implementations of the correlation filter based tracking with the proposed windowing approach. Through such implementation, it is possible to select significantly fewer pixels than the total number of pixels in the object patch during the correlation operation to obtain more computational efficiency.

Finally, this thesis includes three major contributions to improve the tracking performance by concentrating on the correlation filter based trackers. Although the introduced frameworks are designed for correlation filters, they can be extended to be employed in other types of visual trackers or in other computer vision tasks. Namely, deep fully convolutional networks can be trained for object detection problem which is based on correlation filters. Moreover, the introduced ensemble tracking method can adopt any state-of-the-art visual tracker at the expense of computational complexity that increases linearly with the depth of the tree.

REFERENCES

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *IEEE Conference on CVPR*, 2006.

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

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

[4] Q. Bai, Z. Wu, S. Sclaroff, M. Betke, and C. Monnier. Randomized ensemble tracking. In *2013 IEEE International Conference on Computer Vision*, pages 2040–2047, Dec 2013.

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

[6] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Number 179 in Mathematics in Science and Engineering. Academic Press, 1988.

[7] L. Bazzani, N. de Freitas, H. Larochelle, V. Murino, and J.-A. Ting. Learning attentional policies for object tracking and recognition in video with deep networks. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 937–944, New York, NY, USA, June 2011. ACM.

[8] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr. Staple: Complementary learners for real-time tracking. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1401–1409, June 2016.

[9] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. Torr. Fully-convolutional siamese networks for object tracking. *arXiv preprint arXiv:1606.09549*, 2016.

[10] A. Bibi and B. Ghanem. Multi-template scale-adaptive kernelized correlation filters. In *IEEE ICCV Workshops*, December 2015.

[11] A. Bibi, M. Mueller, and B. Ghanem. Target response adaptation for correlation filter tracking. In *Computer Vision - ECCV 2016 - 14th European Confer-*

ence, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI, pages 419–433, 2016.

[12] A. Bibi, M. Mueller, and B. Ghanem. *Target Response Adaptation for Correlation Filter Tracking*, pages 419–433. Springer International Publishing, Cham, 2016.

[13] V. N. Boddeti, T. Kanade, and B. V. K. V. Kumar. Correlation filters for object alignment. In *CVPR*, pages 2291–2298. IEEE Computer Society, 2013.

[14] D. Bolme, J. Beveridge, B. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on CVPR*, pages 2544–2550, June 2010.

[15] D. Bolme, B. Draper, and J. Beveridge. Average of synthetic exact filters. In *IEEE Conference on CVPR*, pages 2105–2112, June 2009.

[16] Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, and S. Li. Robust deformable and occluded object tracking with dynamic graph. *IEEE Transactions on Image Processing*, 23(12):5497–5509, Dec 2014.

[17] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning and Games*. Cambridge University Press, New York, NY, USA, 2006.

[18] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

[19] Z. Chen, Z. Hong, and D. Tao. An experimental survey on correlation filter-based tracking. *CoRR*, abs/1509.05520, 2015.

[20] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 539–546, Washington, DC, USA, 2005. IEEE Computer Society.

[21] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE TPAMI*, 25(5):564–575, May 2003.

[22] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995.

[23] Z. Cui, S. Xiao, J. Feng, and S. Yan. Recurrently target-attending tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[24] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005.

[25] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. *CoRR*, abs/1609.06118, 2016.

[26] M. Danelljan, G. Hager, F. Shahbaz K., and M. Felsberg. Accurate scale estimation for robust visual tracking. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[27] M. Danelljan, G. Hager, F. Shahbaz K., and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCV Workshops 2015*, pages 621–629, 2015.

[28] M. Danelljan, G. Hager, F. Shahbaz K., and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV 2015*, pages 4310–4318, 2015.

[29] M. Danelljan, F. S. Khan, M. Felsberg, and J. v. d. Weijer. Adaptive color attributes for real-time visual tracking. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1090–1097, June 2014.

[30] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016.

[31] A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. *SIAM Rev.*, 49(3):434–448, July 2007.

[32] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[33] M. Denil, L. Bazzani, H. Larochelle, and N. de Freitas. Learning where to attend with deep architectures for image tracking. *Neural Comput.*, 24(8):2151–2184, Aug. 2012.

[34] J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[35] M. K. et.al. The visual object tracking vot2015 challenge results. In *ICCV Workshops*, 2015.

[36] M. K. et.al. The visual object tracking vot2016 challenge results. In *ECCV Workshops*, 2016.

[37] H. Fan and H. Ling. Sanet: Structure-aware network for visual tracking. *CoRR*, abs/1611.06878, 2016.

[38] J. Fan, W. Xu, Y. Wu, and Y. Gong. Human tracking using convolutional neural networks. *IEEE Transactions on Neural Networks*, 21(10):1610–1623, 2010.

[39] M. Felsberg. Enhanced distribution field tracking using channel representations. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

[40] H. K. Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. In *2013 IEEE International Conference on Computer Vision*, pages 3072–3079, Dec 2013.

[41] H. K. Galoogahi, T. Sim, and S. Lucey. Correlation filters with limited boundaries. *CVPR*, abs/1403.7876, 2014.

[42] Q. Gan, Q. Guo, Z. Zhang, and K. Cho. First step toward model-free, anonymous object tracking with recurrent neural networks. *CoRR*, abs/1511.06425, 2015.

[43] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *Proc. BMVC*, pages 6.1–6.10, 2006. doi:10.5244/C.20.6.

[44] K. Granstrom, C. Lundquist, and O. Orguner. Extended target tracking using a gaussian-mixture phd filter. *IEEE Transactions on Aerospace and Electronic Systems*, 48(4):3268–3286, October 2012.

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

[46] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE ICCV*, pages 1026–1034, Dec 2015.

[47] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 FPS with deep regression networks. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pages 749–765, 2016.

[48] J. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE TPAMI*, 2015.

[49] J. F. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[50] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *Proceedings of the ECCV*, 2012.

[51] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 597–606. JMLR.org, 2015.

[52] 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 *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 749–758, June 2015.

[53] D. Huang, L. Luo, M. Wen, Z. Chen, and C. Zhang. Enable scale and aspect ratio adaptability in visual tracking with detection proposals. In *Proceedings of the BMVC*, pages 185.1–185.12, September 2015.

[54] S. M. Jianming Zhang and S. Sclaroff. Tracking by sampling trackers. In *ICCV*, pages 1195–1202, Nov 2011.

[55] J. Jin, A. Dundar, J. Bates, C. Farabet, and E. Culurciello. Tracking with deep neural networks. In *Information Sciences and Systems (CISS), 2013 47th Annual Conference on*, pages 1–5, March 2013.

[56] S. E. Kahou, V. Michalski, and R. Memisevic. RATM: recurrent attentive tracking model. *CoRR*, abs/1510.08660, 2015.

[57] S. S. Kozat, A. C. Singer, and G. C. Zeitler. Universal piecewise linear prediction via context trees. *IEEE Transactions on Signal Processing*, 55(7):3730–3745, July 2007.

[58] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebehay, G. Fernandez, T. Vojir, A. Gatt, A. Khajenezhad, A. Salahledin, A. Soltani-Farani, A. Zarezade, A. Petrosino, A. Milton, B. Bozorgtabar, B. Li, C. S. Chan, C. Heng, D. Ward, D. Kearney, D. Monekosso, H. C. Karaimer, H. R. Rabiee, J. Zhu, J. Gao, J. Xiao, J. Zhang, J. Xing, K. Huang, K. Lebeda, L. Cao, M. E. Maresca, M. K. Lim, M. E. Helw, M. Felsberg, P. Remagnino, R. Bowden, R. Goecke, R. Stolkin, S. Y. Lim, S. Maher, S. Poullot, S. Wong, S. Satoh, W. Chen, W. Hu, X. Zhang, Y. Li, and Z. Niu. The visual object tracking vot2013 challenge results. In *2013 IEEE International Conference on Computer Vision Workshops*, pages 98–111, Dec 2013.

[59] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojir, and G. et al. Fernandez. The visual object tracking vot2014 challenge results, In IEEE ECCV Workshops, 2014.

[60] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in NIPS*, pages 1097–1105. Curran Associates, Inc., 2012.

[61] J. Kwon and K. M. Lee. Visual tracking decomposition. In *IEEE Conference on CVPR*, pages 1269–1276, June 2010.

[62] C. Leistner, A. Saffari, and H. Bischof. Miforests: Multiple-instance learning with randomized trees. In *Proceedings of the 11th European Conference on*

Computer Vision: Part VI, ECCV'10, pages 29–42, Berlin, Heidelberg, 2010. Springer-Verlag.

[63] G. Li and H. Wu. Scale adaptive ensemble tracking. In *2011 4th International Congress on Image and Signal Processing*, volume 1, pages 431–435, Oct 2011.

[64] H. Li, Y. Li, and F. Porikli. Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In *Proceedings of the BMVC*. BMVA Press, 2014.

[65] H. Li, Y. Li, and F. Porikli. Deeptrack: Learning discriminative feature representations online for robust visual tracking. *CoRR*, abs/1503.00072, 2015.

[66] H. Li, Y. Li, and F. Porikli. Robust online visual tracking with a single convolutional neural network. In D. Cremers, I. Reid, H. Saito, and M.-H. Yang, editors, *Computer Vision – ACCV 2014*, volume 9007 of *Lecture Notes in Computer Science*, pages 194–209. Springer International Publishing, 2015.

[67] J. Li, Z. Hong, and B. Zhao. Robust visual tracking by exploiting the historical tracker snapshots. In *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 604–612, Dec 2015.

[68] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel. A survey of appearance models in visual object tracking. *ACM Trans. Intell. Syst. Technol.*, 4(4):58:1–58:48, Oct. 2013.

[69] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *IEEE ECCV*, pages 254–265, 2014.

[70] Y. Li, J. Zhu, and S. C. Hoi. Reliable patch trackers: Robust visual tracking by exploiting reliable patches. In *IEEE Conference on CVPR*, June 2015.

[71] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski. Robust and fast collaborative tracking with two stage sparse optimization. In *IEEE ECCV*, volume 6314 of *Lecture Notes in Computer Science*, pages 624–637. Springer Berlin Heidelberg, 2010.

[72] H. Liu and F. Sun. Semi-supervised ensemble tracking. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1645–1648, April 2009.

[73] T. Liu, G. Wang, and Q. Yang. Real-time part-based visual tracking via adaptive correlation filters. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[74] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[75] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004.

[76] C. Ma, J. B. Huang, X. Yang, and M. H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, pages 3074–3082, Dec 2015.

[77] C. Ma, Y. Xu, B. Ni, and X. Yang. When correlation filters meet convolutional neural networks for visual tracking. *IEEE Signal Processing Letters*, 23(10):1454–1458, Oct 2016.

[78] A. Mahalanobis, B. V. K. V. Kumar, and D. Casasent. Minimum average correlation energy filters. *Appl. Opt.*, 26(17):3633–3640, Sep 1987.

[79] R. P. S. Mahler. Multitarget bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, Oct 2003.

[80] S. Masoudnia and R. Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

[81] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *IEEE ICCV*, pages 1436–1443, Sept 2009.

[82] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai. Minimum error bounded efficient l1 tracker with occlusion detection. In *CVPR 2011*, pages 1257–1264, June 2011.

[83] H. Nam, M. Baek, and B. Han. Modeling and propagating cnns in a tree structure for visual tracking. *CoRR*, abs/1608.07242, 2016.

[84] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *The IEEE Conference on CVPR*, June 2016.

[85] J. Ning, J. Yang, S. Jiang, L. Zhang, and M.-H. Yang. Object tracking via dual linear structured svm and explicit feature map. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[86] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, Jul 2002.

[87] R. W. Oppenheim, Alan V. nd Schafer and J. R. Buck. *Discrete-time Signal Processing (2nd Ed.).* Prentice-Hall, Inc, Upper Saddle River, NJ, USA, 1999.

[88] H. Ozkan, N. D. Vanli, and S. S. Kozat. Online classification via self-organizing space partitioning. *IEEE Transactions on Signal Processing*, 64(15):3895–3908, Aug 2016.

[89] P. Refregier. Optimal trade-off filters for noise robustness, sharpness of the correlation peak, and horner efficiency. *Opt. Lett.*, 16(11):829–831, Jun 1991.

[90] A. Rodriguez, V. N. Boddeti, B. V. K. V. Kumar, and A. Mahalanobis. Maximum margin correlation filter: A new approach for localization and classification. *IEEE Trans. Image Processing*, 22(2):631–643, 2013.

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

[92] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115(3):211–252, Dec. 2015.

[93] M. Savvides and B. V. K. V. Kumar. Efficient design of advanced correlation filters for robust distortion-tolerant face recognition. In *AVSS*, pages 45–52. IEEE Computer Society, 2003.

[94] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, Apr. 2017.

[95] C. Shen, G. Lin, and A. van den Hengel. Structboost: Boosting methods for predicting structured output variables. *CoRR*, abs/1302.3283, 2013.

[96] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[97] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, July 2014.

[98] A. Solis Montero, J. Lang, and R. Laganiere. Scalable kernel correlation filter with sparse feature integration. In *The IEEE ICCV Workshops*, December 2015.

[99] Y. Sui, Z. Zhang, G. Wang, Y. Tang, and L. Zhang. Real-time visual tracking: Promoting the robustness of correlation filter learning. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII*, pages 662–678, 2016.

[100] F. Tang, S. Brennan, Q. Zhao, and H. Tao. Co-tracking using semi-supervised support vector machines. In *ICCV*, pages 1–8, 2007.

[101] R. Tao, E. Gavves, and A. W. M. Smeulders. Siamese instance search for tracking. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1420–1429, 2016.

[102] R. Tokola and D. Bolme. Ensembles of correlation filters for object detection. In *IEEE Conference on WACV*, pages 935–942, Jan 2015.

[103] G. Turin. An introduction to matched filters. *IRE Transactions on Information Theory*, 6(3):311–329, June 1960.

[104] N. D. Vanli and S. S. Kozat. A comprehensive approach to universal piecewise nonlinear regression based on trees. *IEEE Transactions on Signal Processing*, 62(20):5471–5486, oct 2014.

[105] A. Vedaldi and K. Lenc. Matconvnet – convolutinoal neural networks for matlab. In *Int. Conf. on Multimedia, ACM*, 2015.

[106] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518 vol.1, 2001.

[107] T. Vojíř and J. Matas. *The Enhanced Flock of Trackers*, pages 113–136. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[108] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3119–3127, Dec 2015.

[109] X. Wang, M. Valstar, B. Martinez, M. H. Khan, and T. Pridmore. Tric-track: Tracking by regression with incrementally learned cascades. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4337–4345, Dec 2015.

[110] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE TPAMI*, 31(2):210–227, Feb 2009.

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

[112] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1834–1848, 2015.

[113] Y. Wu, B. Shen, and H. Ling. Visual tracking via online nonnegative matrix factorization. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(3):374–383, March 2014.

[114] J. Xing, J. Gao, B. Li, W. Hu, and S. Yan. Robust object tracking with online multi-lifespan dictionary learning. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[115] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *CoRR*, abs/1505.00853, 2015.

[116] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel. Part-based visual tracking with online latent structural learning. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2363–2370, June 2013.

[117] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4), Dec. 2006.

[118] Q. Yu, T. B. Dinh, and G. Medioni. Online tracking and reacquisition using co-trained generative and discriminative trackers. In *IEEE ECCV*, Berlin, Heidelberg, 2008. Springer-Verlag.

[119] H. Zhang, S. Hu, X. Zhang, and L. Luo. Visual tracking via constrained incremental non-negative matrix factorization. *IEEE Signal Processing Letters*, 22(9):1350–1353, Sept 2015.

[120] 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.

[121] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *IEEE ECCV*, ECCV'12, pages 864–877, Berlin, Heidelberg, 2012. Springer-Verlag.

[122] K. Zhang, L. Zhang, and M.-H. Yang. Real-time object tracking via online discriminative feature selection. *Image Processing, IEEE Transactions on*, 22(12):4664–4677, Dec 2013.

[123] K. Zhang, L. Zhang, and M.-H. Yang. Fast compressive tracking. *IEEE TPAMI*, 36(10):2002–2015, Oct 2014.

[124] T. Zhang, A. Bibi, and B. Ghanem. In defense of sparse tracking: Circulant sparse tracker. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[125] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via structured multi-task sparse learning. *International journal of computer vision*, 101(2):367–383, 2013.

[126] T. Zhang, S. Liu, N. Ahuja, M.-H. Yang, and B. Ghanem. Robust visual tracking via consistent low-rank sparse learning. *International Journal of Computer Vision*, 111(2):171–190, 2014.

[127] L. H. Zhong Wei and Y. Ming-Hsuan. Robust object tracking via sparsity-based collaborative model. In *IEEE Conference on CVPR*, CVPR '12, pages 1838–1845, Washington, DC, USA, 2012. IEEE Computer Society.

[128] Q. H. Zhou, H. Lu, and M. H. Yang. Online multiple support instance tracking. In *Face and Gesture 2011*, pages 545–552, March 2011.

[129] G. Zhu, F. Porikli, and H. Li. Beyond local search: Tracking objects everywhere with instance-specific proposals. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 943–951, June 2016.

[130] W. Zuo, X. Wu, L. Lin, L. Zhang, and M. Yang. Learning support correlation filters for visual tracking. *CoRR*, abs/1601.06032, 2016.

APPENDIX A

THE EFFECT OF A LAYER ON THE CORRELATION QUALITY OF THE PREVIOUS LAYER

In this part, it is analyzed that the correlation quality of a layer behaves analogous to the layer above it if some assumptions on the additive appearance noise hold. This noise can be perceived as the appearance difference between the template x and the test patch z. Convolutional layers have a set of 2-dimensional feature maps. In order to obtain another convolutional layer on top of the previous one, they are summed with a set of weight parameters.

For this purpose, X is two-dimensional DFT of a single feature map obtained from a network in a certain layer, *e.g.*, l^{th} layer, for the training example x. Similarly, z is the test patch with non-centered object. The single channel correlation filter for this training example is given as:

$$H = \frac{X \odot \hat{G}^*}{X^* \odot X + \gamma}, \tag{A.1}$$

where γ is the regularization parameter, and $\hat{G}$ is the DFT of the desired response $\hat{g}$ for the template x with a peak in its center location.

If the localized test sample z has the feature map in DFT domain as $Z = X + \mu$ with μ being the additive noise due to the appearance change of the object, then the resulting correlation error turns out to be:

$$\begin{gathered}\mathcal{E}_{single} = H^* \odot Z - \hat{G} \\ = \left(\frac{X^* \odot \hat{G}}{X^* \odot X + \gamma} \odot (X + \mu) - \hat{G} \right) \approx \frac{\mu X^* \odot \hat{G}}{X^* \odot X + \gamma}\end{gathered} \tag{A.2}$$

If the convolutional kernel at level $l - 1$ is assumed to be 1×1 with their values fixed to 1 and there exists only two feature maps, then we can split X as $X = X_1 + X_2$ by ignoring the bias terms. In this case, the feature map of the test example Z will be split as $Z = Z_1 + Z_2$, where $Z_1 = X_1 + \mu_1$ and $Z_2 = X_2 + \mu_2$. The μ_1 and μ_2 are the individual additive noises of the feature maps. Repeating the formulation in Section 3.2, the two correlation filters are:

$$\begin{aligned} H_1 &= \frac{X_1 \odot \hat{G}^*}{X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma} \\ H_2 &= \frac{X_2 \odot \hat{G}^*}{X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma} \end{aligned} \tag{A.3}$$

Hence, the correlation of the test sample z and the correlation filters yield:

$$\begin{aligned} \mathcal{E}_{multi} &= H_1^* Z_1 + H_2^* Z_2 - \hat{G} \\ &= \frac{X_1^* \odot \hat{G}}{X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma} \odot (X_1 + \mu_1) + \\ &\frac{X_2^* \odot \hat{G}}{X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma} \odot (X_2 + \mu_2) - \hat{G} \end{aligned} \tag{A.4}$$

By rearranging the terms and neglecting the effect of γ value, the error is reduced to:

$$\mathcal{E}_{multi} = \frac{\mu_1 X_1^* \odot \hat{G} + \mu_2 X_2^* \odot \hat{G}}{X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma} \tag{A.5}$$

In order to make a similarity between the (A.2) and (A.5), X is replaced with $X_1 + X_2$ and $\mu = \mu_1 + \mu_2$ in (A.2). Moreover, all of the terms are copied in (A.5). Finally, we obtain the following error for single and multiple channels:

$$\begin{aligned} \mathcal{E}_{multi} &= \frac{\mu_1 X_1^* \odot \hat{G} + \mu_2 X_2^* \odot \hat{G} + \mu_1 X_1^* \odot \hat{G} + \mu_2 X_2^* \odot \hat{G}}{X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma + X_1^* \odot X_1 + X_2^* \odot X_2 + \gamma} \\ \mathcal{E}_{single} &= \frac{\mu_1 X_1^* \odot \hat{G} + \mu_2 X_2^* \odot \hat{G} + \mu_1 X_2^* \odot \hat{G} + \mu_2 X_1^* \odot \hat{G}}{X_1^* \odot X_1 + X_2^* \odot X_2 + X_1^* \odot X_2 + X_2^* \odot X_1 + \gamma} \end{aligned} \tag{A.6}$$

As it can be observed from the errors of single and multiple channels, both of them are proportional to μ_1 and μ_2. Hence, if the sum of these two variables (*i.e.*, μ) decreases, μ_1 and μ_2 have also tendency to decrease.

The above derivation can be extended to more than two feature maps, where the same assumption would hold. If the two correlation response errors in two consecutive layers are almost the same, one can argue that the mitigation of the appearance noise in one of the layers is quite likely to reduce the correlation response error in the other one. Hence, training a fully convolutional model to reduce its correlation error with respect to the top layer will eventually increase the correlation quality in the lower layers. The experimental evaluation of the learned feature maps on the correlation filter based trackers has clearly shown that this analysis is practically valid in most of the tracking scenarios.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Gündoğdu, Erhan

Nationality: Turkish (TC)

Date and Place of Birth: 13.02.1987, Aydın, Turkey

Marital Status: Married

Phone: +905054488566

EDUCATION

Degree	Institution	Year of Graduation
M.S.	EEE Dept., METU	2012
B.S.	EEE Dept., METU	2010
High School	Adnan Menderes Anatolian High School, Aydın	2005

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
Oct. 2010 - Oct. 2012	EEE Dept., METU, Ankara	Teaching Assistant
June 2013 - Aug. 2013	Ortana Elektronik, Ankara	Engineer
Sept. 2013 - Present	Aselsan Inc., Ankara	Research Engineer

PUBLICATIONS

International Journals

- E. Gundogdu, H. Ozkan, A. A. Alatan, Extending Correlation Filter Based Visual Tracking by Tree-Structured Ensemble and Spatial Windowing, IEEE Transactions on Image Processing, Accepted, 2017.
- E. Gundogdu, Berkan Solmaz, Veysel Yucesoy, Aykut Koc, A. Aydın Alatan, A Joint Framework of Deep Metric Learning and Fine-Grained Object Recognition for Visual Surveillance, submitted to IEEE Transactions on Circuits and Systems for Video Technology on Jan. 31, 2017 (under revision).
- E. Gundogdu and A. A. Alatan. Good features to correlate for visual tracking. CoRR, abs/1704.06326, 2017 (under revision to be published in IEEE Transactions on Image Processing).
- Aykut Koc, Burak Bartan, Erhan Gundogdu, Tolga Cukur, Haldun M. Ozaktas, Sparse representation of two- and three-dimensional images with fractional Fourier, Hartley, linear canonical, and Haar wavelet transforms, Expert Systems with Applications, Volume 77, 2017, Pages 247-255, ISSN 0957-4174, http://dx.doi.org/10.1016/j.eswa.2017.01.046.

International Conferences

- E. Gundogdu, B. Solmaz, V. Yucesoy, and A. Koc, “Marvel: A large scale image dataset for maritime vessels,” in ACCV, 2016.
- E. Gundogdu and A. A. Alatan, "Spatial windowing for correlation filter based visual tracking," 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, 2016, pp. 1684-1688. doi: 10.1109/ICIP.2016.7532645
- E. Gundogdu, A. Koc and A. A. Alatan, "Object classification in infrared images using deep representations," 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, 2016, pp. 1066-1070.

- E. Gundogdu, H. Ozkan and A. A. Alatan, "Ensemble of adaptive correlation filters for robust visual tracking," 2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Colorado Springs, CO, 2016, pp. 15-22. doi: 10.1109/AVSS.2016.7738031

- E. Gundogdu, A. Koc, B. Solmaz, R. I. Hammoud and A. A. Alatan, Evaluation of Feature Channels for Correlation-Filter-Based Visual Object Tracking in Infrared Spectrum, 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, 2016, pp. 290-298. doi: 10.1109/CVPRW.2016.43

- E. Gundogdu, H. Ozkan, H. S. Demir, H. Ergezer, E. Akagunduz, S. K. Pakin, Comparison of Infrared and Visible Imagery for Object Tracking: Toward Trackers with Superior IR Performance, Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops, 2015.

- G. Tanisik and E. Gundogdu, Multiple Model Adaptive Visual Tracking with Correlation Filters, IEEE International Conf. on Image Processing (ICIP), 2015

- E. Gundogdu and A. Aydın Alatan, Feature Detection and Matching Towards Augmented Reality Applications On Mobile Devices, 3DTVCON, October, 2012

- M. K. et.al., “The visual object tracking vot2017 challenge results,” in ICCV Workshops, 2017.

- M. K. et.al., “The visual object tracking vot2016 challenge results,” in ECCV Workshops, 2016.

National Conferences

- E. Gundogdu and A. A. Alatan, Method for learning deep features for correlation based visual tracking, 2017 25th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 2017, pp. 1-4. doi: 10.1109/SIU.2017.7960171

- E. Gundogdu, B. Solmaz, A. Koç, V. Yücesoy and A. A. Alatan, "Deep distance metric learning for maritime vessel identification," 2017 25th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 2017, pp. 1-4. doi: 10.1109/SIU.2017.7960170
- E. Gundogdu, A. Koç and A. A. Alatan, "Infrared object classification using decision tree based deep neural networks," 2016 24th Signal Processing and Communication Application Conference (SIU), Zonguldak, 2016, pp. 1913-1916. doi: 10.1109/SIU.2016.7496139
- E. Gundogdu, Kızıl Ötesi Görüntülerde Hızlı Jeodezik İlgi Haritası, SIU 2015
- G. Tanisik and Erhan Gundogdu, Korelasyon Süzgeçleri ile Uyarlanır İki Modelli Görsel Hedef Takibi, SIU 2015
- E. Gundogdu and A. Aydın Alatan, Feature Detection and Matching Towards Augmented Reality Applications On Mobile Devices, Master Forum (MASFOR (Bogazici University Electrical and Electronics Engineering 100. anniversary), June, 2012
- E. Gundogdu and A. Aydın Alatan, “NESTLE: İç İçe Çemberler Kullanarak İlgi Noktası Algılama” SIU 2012