Regularized Ensemble Correlation Filter Tracking

A THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

BY

MONIKA JAIN

Visual Conception Group

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY

NEW DELHI- 110020

&

Signal Processing, Artificial Intelligence and Vision Technologies

Faculty of Engineering

QUEENSLAND UNIVERSITY OF TECHNOLOGY

AUSTRALIA

June, 2022

THESIS CERTIFICATE

The work contained in this thesis entitled, **Regularized Ensemble Correlation Filter Tracking**, has also been submitted to Queensland University of Technology, Australia, as a part of the Queensland University of Technology (QUT), Australia & Indraprastha Institute of Information Technology (IIIT), Delhi joint PhD program. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Advisors' Name

1) Dr. A V Subramanyam, Associate Professor, Depts. of ECE & CSE, IIIT Delhi

2) Prof. Clinton Fookes, Faculty of Engineering, School of Electrical Engineering & Robotics, QUT Australia

3) Prof. Sridha Sridharan, Emeritus Professor, Principal Research Fellow, Faculty of Engineering, School of Electrical Engineering & Robotics, QUT Australia

4) Dr. Simon Denman, Senior Lecturer, Faculty of Engineering, School of Electrical Engineering & Robotics, QUT Australia

Place: New Delhi

Date: June 2022

ACKNOWLEDGEMENTS

I am very grateful to my advisors, Dr. A V Subramanyam, Dr. Simon Denman, Prof. Clinton Fookes, and Prof. Sridha Sridharan, whose invaluable feedback on my analysis and framing helped to improve this thesis. Their kind words and guidance always kept me motivated.

I would like to thank the Queensland University of Technology and Indraprastha Institute of Information Technology for their financial support during the development of this thesis.

I also want to thank Dr. Tarun Bansal, Dr. Harshala Gammulle and Dr. Tharindu Fernando Warnakulasuriya for never letting me feel alone in Australia.

Most importantly, I would like to express my gratefulness and love to my husband, Akshay Jain, without his continuous support this work would not have been possible. A special thanks to Paa, Mumma and my siblings for their regular encouragement and unconditional support.

Monika Jain (PhD16114)

ABSTRACT

Correlation Filter based visual trackers have demonstrated tremendous progress in object tracking. These trackers primarily use hierarchical features learned from multiple layers of a deep network. However, issues related to background awareness, deterministic aggregation of these features from various layers, difficulties in estimating variations in scale or rotation of the object being tracked, as well as challenges in effectively modelling the object's appearance over long time periods leaves substantial scope to improve performance. Such issues lead to poor discriminative power and rapidly drift the tracker away from the target.

We propose ensemble and regularization techniques to achieve a strong discriminative ability for object trackers. We first obtain an ensemble of weak correlation filters using an adaptive weighing strategy and an appearance model pool to adapt to large appearance changes. Further, the target scale and rotation parameters are estimated using a dedicated affine correlation filter.

Our experiments reveal that each feature channel encodes a different appearance cue of the target and not all channels are equally important during different tracking steps. To this end, we propose a graph based adaptive channel weighing strategy that assigns weight to each channel based on the similarity of encoded appearance information. In order to model the channel importance and improve background awareness simultaneously, we introduce a sparse spatio-temporal regularizer with adaptive channel weights. We further kernelize the tracker to attain real-time performance. We extensively evaluate the trackers using publicly available datasets — Object Tracking Benchmark (OTB100), Visual Object Tracking (VOT) Benchmark 2016, VOT Benchmark 2017, Tracking Dataset, VOT Benchmark 2019, Temple Color 128, UAV123, LaSOT, and GOT10k.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

VOT	Visual Object Tracking
CNN	Convolutional Neural Network
CF	Correlation Filter
LSTM	Long Short Term Memory
FFT	Fast Fourier Transformation
DFT	Discrete Fourier Transform
HOG	Histogram of Oriented Gradients
FHOG	Felzenszwalb's Histogram of Oriented Gradients
ADMM	Alternating Direction Method of Multipliers
KCF	Kernelized Correlation Filter
BACF	Background Aware Correlation Filter Tracker
STRCF	Spatial-Temporal Regularized Correlation Filters
ASRCF	Adaptive Spatially-Regularized Correlation Filters
OTB	Object Tracking Benchmark
IV	Illumination Variation
OCC	Occlusion
SV	Size Variation
SV	Size Variation
MC	Motion Change
MB	Motion Blur
FM	Fast Motion
IPR	In-Plane Rotation
OPR	Out-of-Plane Rotation
DEF	Deformation
OV	Object Out-of-View
BC	Background Clutter
LR	Low Resolution
CLE	Center Location Error
POC	Partial Occlusion
FOC	Full Occlusion
IOU	Intersection-Over-Union
TC128	Temple Color-128
LaSOT	Large-scale Single Object Tracking Benchmark
UAV123	Unmanned Aerial Vehicle-123
AUC	Area Under Curve
EAO	Expected Average Overlap
OS	Overlap Success
DP	Distance precision
PCA	Principal Component Analysis
AOS	Average Overlap Success
ADP	Average Distance Precision

EAO	Expected Average Overlap
AR	Accuracy Robustness
XQDA	Cross-view Quadratic Discriminant Analysis
SE-Net	Squeeze-and-Excitation Network
CR	Channel Regularization
A	Accuracy
R	Robustness
AUC	Area Under Curve
IST	Iterative Soft Thresholding
STR	Sparse Temporal Regularization
SSR	Sparse Spatial Regularization
CI	Channel Importance
S	Success
p	Precision
SOT	Single Object Tracking
MOT	Multiple Object Tracking

LIST OF SYMBOLS

$\mathbf{y}$	Vectorized Gaussian shaped label
$\mathbf{y}_m$	Vectorized Gaussian shaped label obtained after m cyclic shifts
$\mathbf{x}$	Features
$\mathbf{x}_k$	Target Feature Channel
$\mathbf{x}_m$	Target Feature Channel obtained after m cyclic shifts
K	Total number of feature channels
k	Channel index
$\mathbf{h}$	Learned correlation filter
$\mathbf{h}_k$	Correlation Filter Channel
q_k	Scalar weight for response channel k
$*$	Spatial correlation operator
$\top$	Transpose
β	Channel regularization parameter
$\mathbf{P}$	Binary matrix that crops out the foreground region
W	Width of each feature channel
H	Height of each feature channel
$\mathbf{z}$	Incoming Test Frame
$\mathbf{r}$	Correlation Filter Response
p_1	x-axis coordinates
p_2	y-axis coordinates
$\mathbf{l}$	Target location
$\mathbf{l}^u$	Target location for layer u
fg	Foreground
bg	Background
u	Layer Index
q^u	Location predicted corresponding to layer u
$\mathbf{r}_{scale}$	Scale Filter Response
$scale$	Scaling Factor
$\mathbf{r}_{rot}$	Rotation Filter Response
$\mathbf{b}$	Bounding box
t	Frame index
$box^{(t)}$	Vectorized bounding box predicted at frame (t)
app	Vectorized appearance sample in the model pool
$dist$	Mahalanobis distance
$\hat{}$	Discrete Fourier transform
κ	Number of background patches
W'	Width of the input search window
λ	Filter regularization parameter
$\mathbf{G}$	Auxiliary variable matrix
$\mathbf{F}$	Orthonormal matrix
μ	Penality factor

$\mathbf{S}$	Lagrange multiplier
$\mathbf{I}$	Identity matrix
$\mathbf{w}$	Spatial weights
$\mathbf{W}$	Diagonal matriz of $\mathbf{w}$
Z	Weight matrix
σ	Decay speed controlling parameter
G	Weighted nearest neighbor graph
$\mathbf{D}$	Degree matrix
Tr(.)	Trace of a matrix
$\mathbf{L}$	Laplacian matrix
θ	Temporal regularization parameter
α	Regularization parameter
$\mathcal{O}$	Computational cost
$\mathbf{B}_w$	Sparse vector that learns the spatial variations
$\mathbf{B}_h$	Sparse vector that learns the temporal variations
ζ	Regularization parameter
$lambda_1$	Regularization parameter
$lambda_2$	Regularization parameter
ω	Online learning rate
$\mathcal{S}$	Soft Thresholding Operator
Ω	Kernel matrix
$\mathbf{r}_{Gaussian}$	Correlation filter response for Gaussian kernel
$\mathbf{r}_{Polynomial}$	Correlation filter response for polynomial kernel
$\mathbf{r}_{Linear}$	Correlation filter response for linear kernel

CHAPTER 1

Introduction

1.1 Motivation

Visual Object Tracking (VOT) is the task of tracking an object within a video. Typically, in a single object tracker scenario, the groundtruth bounding box of the target object is known in the first frame and a VOT algorithm is required to predict the bounding box in subsequent frames to track the target. An overview of the tracking task is provided in Figure 1.1.

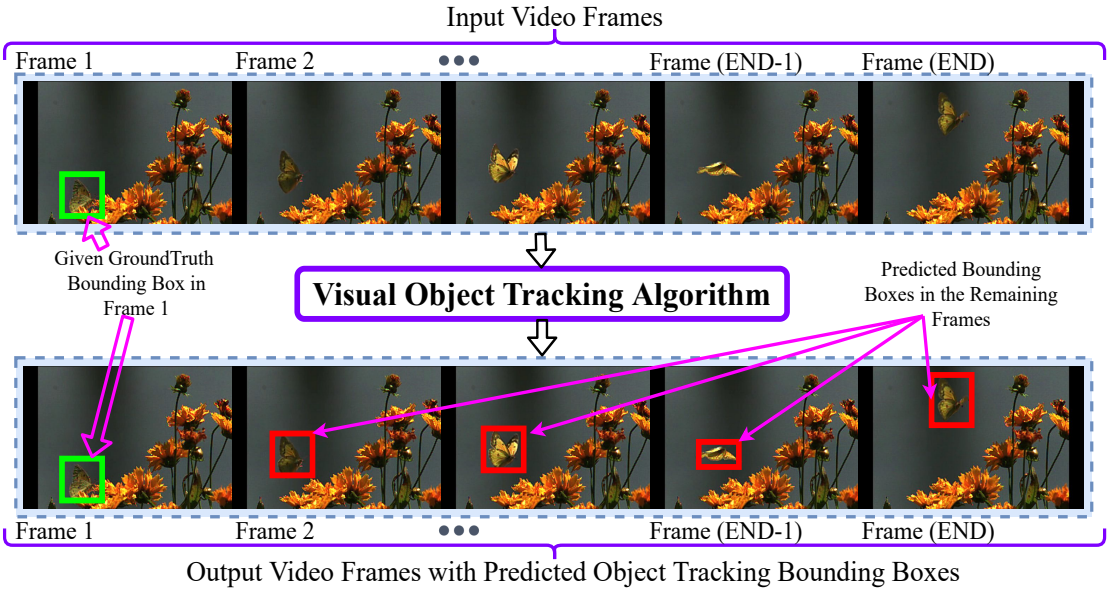


Figure 1.1: Visual Object Tracking: The bounding box is provided to the tracker in the first frame. The tracking algorithm then seeks to locate the object contained in the bounding box in subsequent frames.

Broadly, most existing tracking algorithms are based on Kalman filter, particle filters, end-to-end Convolutional Neural Network (CNN), Correlation Filters (CF) learned using hand-crafted features, and hybrid approaches. This thesis focuses on CF based and hybrid approaches. CNN based trackers offer robust tracking due to their rich hierarchical feature representation, but may suffer from heavy computational complexity due to their high number of parameters. On the other hand, CF based trackers typically use shallow features that model the target appearance with less discriminative power,

but offer real time tracking due to efficient operations in the frequency domain (Kumar *et al.*, 2005). To exploit the advantages from both of these tracker classes and suppress the drawbacks, hybrid trackers often use CFs in conjunction with 3-dimensional deep features extracted using a pre-trained CNN.

The use of Correlation Filters (CFs) for tracking in conjunction with the deep features facilitates target appearance modelling with more generalized and discriminative features, as compared to using hand-crafted features, along with real time speed due to fast computations in the Fourier domain (Kumar *et al.*, 2005). Although computationally inexpensive, learning CFs in the frequency domain incurs many disadvantages. Negative training samples are generated by circularly shifting the base patch in the frequency domain (Kumar *et al.*, 2005). All these samples are impaired by boundary effects and do not represent the actual background as they are merely shifted versions of the base patch. Training the CF with these imitated and limited negative samples can result in an over-fitted CF that adapts poorly to rapid visual deformations of the target object, and drifts easily during challenges like similar object presence, background clutter, occlusion and out-of-plane movements (Danelljan *et al.*, 2015*b*). To alleviate these issues, CF trackers with spatial and temporal constraints have been proposed in Danelljan *et al.* (2015*b*); Kiani Galoogahi *et al.* (2017); Dai *et al.* (2019); Li *et al.* (2018*c*). However, these works (Danelljan *et al.*, 2015*b*; Kiani Galoogahi *et al.*, 2017; Dai *et al.*, 2019; Li *et al.*, 2018*c*) do not investigate the contribution of different input feature channels while learning the CF. Additionally, the spatial and temporal constraints based CF trackers are not derived in combination with kernel tricks as the constraints may not allow the straight forward use of the circulant matrix structure (Huang *et al.*, 2020).

This research is a significant step forward towards the development of CF based trackers that successfully employ spatio-temporal regularizations and kernel tricks to overcome the shortcoming of the classical CF formulation.

1.2 Research Aims

Visual object tracking is a widely scrutinized research problem in the field of computer vision and video analytics. Recently, end-to-end deep learning based approaches have achieved tremendous success in visual object tracking. However, these approaches are computationally expensive and need huge volumes of training data. Correlation Filters

(CFs) are an alternate and highly efficient solution for object tracking with no strict data requirements. CF based object trackers use deep features extracted using a pre-trained Convolutional Neural Network (CNN). These trackers model the target appearance with generalized and discriminative deep features, whilst achieving real time speed due to fast computations in the Fourier domain (Kumar *et al.*, 2005). This thesis aims at developing new CF based tracking methods that successfully employ spatio-temporal regularizations and kernel tricks. To contribute to this main aim, the goals of this thesis are recognised by the following four (4) research aims.

Aim 1: To investigate LSTM guided ensemble correlation filter tracking with appearance model pool

Despite achieving state-of-the-art performance, existing CNN trackers still have some limitations. Most of these methods only use the output of a single Convolutional Neural Network (CNN) layer for target location estimation. Compared to such trackers, multi-layer trackers perform better Qi *et al.* (2016); Ma *et al.* (2015*a*). This is because features from deeper layers can capture rich category level semantic information, which is useful for object classification and are not the optimal representation for visual tracking since spatial details captured by earlier layers are also important for accurately localizing the targets. On the other hand, as the features in the earlier layers are more generic rather than discriminative as ones in the later layers, methods based on features from earlier layers are likely to fail in challenging scenarios. To achieve better tracking performance, it is thus imperative to combine features from different layers to best represent and separate foreground objects from the background clutters. Therefore, we aim to combine activations from deep layers with those of earlier layers (that capture more spatial information), to obtain a more reliable prediction. However, the prediction accuracy of each layer may vary from frame to frame and combining all the predictions with equal weight may result in inaccurate tracking. Thus, we aim to develop a reliable mechanism to compute the contribution of each layer when estimating the target's final location. To this end, we analyse Long Short Term Memory (LSTM) network based ensemble CF tracking. CF trackers are combined with an appearance model pool to avoid faulty filter updates. The tracker performance is further improved by using dedicated filters to compute the scale and rotation of the target.

Aim 2: To investigate channel graph regularized correlation filters for tracking

The investigation in Aim 1 led to developing a reliable mechanism to compute the contribution of each Convolutional Neural Network (CNN) layer when estimating the target's final location. However, features extracted from each of these CNN layers contain multiple channels. Training Correlation Filters (CFs) with multi-channel CNN features is a challenging task. Each CNN feature channel encodes a different attribute of the target where some channels may offer more informative features for tracking, while others with less useful information may degrade the tracking and eventually lead to tracker drift Danelljan *et al.* (2015*b*). To address this issue of channel importance, feature selection Xu *et al.* (2019*a*), adaptive importance maps Li *et al.* (2018*a*) and reliability learning Sun *et al.* (2018*a*) methods have already been proposed which inspires us to develop a Channel Regularized CF tracker that learns an adaptive channel importance for each feature channel, such that the feature channels that encode useful appearance cues should be assigned higher weights and the less important feature channels should be suppressed through lower weights. However, a detailed analysis revealed that using the proposed Channel Regularization, the feature channels that produced similar filter responses were getting assigned dissimilar weights and few feature channels with weak filter responses were getting assigned higher weights, as shown in Figure 5.2. This can lead to higher contribution of the poor feature channels and lower contribution of the stronger feature channels during filter training, resulting in a filter with poor discriminative ability. This is corrected using the proposed Channel-Graph regularization that monitors the similarity between different feature channels and accordingly assigns similar weights to similar feature channels and higher weights to rich feature channels. Hence, preserving the similarity of importance between different feature channels Zhou *et al.* (2016) while increasing the contribution of stronger feature channels during the filter training.

Aim 3: To investigate importance guided sparse spatio-temporal regularized correlation filters for tracking

We further improve the proposed formulation from Aim 2 by introducing an efficient spatial and temporal regularizations, increasing awareness of previous and spatially adjacent observations. The challenges with spatial regularization based CF trackers are

that they either have fixed spatial weights (Danelljan *et al.*, 2015*b*), or learn spatial weights that are similar to some reference weights (Dai *et al.*, 2019). Likewise, temporal regularization based CF tracker imposes a constraint such that the current learned filter is similar to the previous filter (Li *et al.*, 2018*c*). However, due to continuous temporal and spatial variations in a tracking sequence, the filter and spatial weights in a tracking step will not be identical to their reference counterparts. This motivates us to develop a sparse spatially and temporally regularized CF tracker. Using spatial regularization, we aim to suppresses the effects of unfavorable background information and boundary effects in the learned filter. The temporal regularization aims to help the filter adapt to appearance changes, preventing drift. We will also model the sparse spatial and sparse temporal variations to enhance the discriminative ability of the learned filters.

Aim 4: To investigate temporally regularized multi-kernel correlation filters for tracking

In the recent research, Correlation Filter (CF) trackers with spatial and temporal constraints have been proposed (Danelljan *et al.*, 2015*b*; Kiani Galoogahi *et al.*, 2017; Dai *et al.*, 2019; Li *et al.*, 2018*c*), including the trackers investigated in Aims 2 and 3. The limitation of such methods is that they are not used in combination with the kernel model (Henriques *et al.*, 2014) as the constraints may break the circulant matrix structure, making it computationally expensive. To overcome this limitation, we aim to introduce the kernel trick to the temporally regularized CF tracker, learned using multiple kernels. The proposed tracker will capture the non-linearity in the appearance features while retaining the circulant matrix structure.

1.3 Thesis structure

The remainder of this thesis is organised as follows.

Chapter 2 provides an overview of topics related to correlation filter based visual object tracking. The strengths and weaknesses of existing methods are discussed in detail.

Chapter 3 explains all the evaluation metrics and datasets that are used to assess the object trackers developed in this thesis.

Chapter 4 introduces a novel LSTM guided ensemble correlation filter based tracker with an appearance model pool, along with an ablation study for different modules. We also investigate evaluation of the tracker across multiple standard tracking datasets.

Chapter 5 presents channel graph regularized correlation filters for visual object tracking. We investigate two different methods of computing the adaptive channel weights for the features used to train the filters.

Chapter 6 proposes an importance guided sparse spatio-temporally regularized correlation filter tracker. In this chapter, we model the sparse spatial and temporal variations along with the channel importance during each tracking step.

Chapter 7 investigates a temporally regularized multi-kernel correlation filter for visual object tracking.

Chapter 8 concludes the thesis by summarizing the key contributions, comparing the proposed trackers with each other, and discussing avenues for future work.

1.4 Contributions of this research

The overarching innovation of this thesis is the investigation of how classical Correlation Filter (CF) based tracking can be improve with the help of ensemble of CFs, spatio-temporal constraints and kernel tricks. The use of ensemble of CF trackers helps in modelling and combining the useful target appearance information encoded by different layers of a pre-trained deep network. On the other hand, the spatio-temporal constraint results in background aware CFs that can model the target appearance changes over a long period of time. The combination of temporal constraint with kernel tricks will help in modelling the large appearance changes while offering real-time tracking speed. The detailed investigation of each of the improvements suggested above lead to the following contributions of this thesis.

1. We analyse LSTM based ensemble Correlation Filter (CF) tracking. CF trackers are combined with an appearance model pool to avoid faulty filter updates. The tracker performance is further improved by using dedicated filters to compute the scale and rotation of the target.

2. The CFs learned in the above work are trained using multi-channel deep features. However, each feature channel encodes different appearance information and these may not be equally important for tracking at different times. This motivates us to introduce a channel regularized CF tracker that learns adaptive channel

importance for each feature channel. Thus, feature channels that encode useful appearance cues are assigned higher weights and the less important feature channels are suppressed through lower weights.

3. To further improve tracking accuracy, we investigate spatially and temporally regularized CFs. The spatial regularization suppresses the effects of unfavorable background information and boundary effects in the learned filter. The temporal regularization helps the filter adapt to appearance changes, preventing drift. We also model sparse spatial and sparse temporal variations to enhance the discriminative ability of the learned filters.

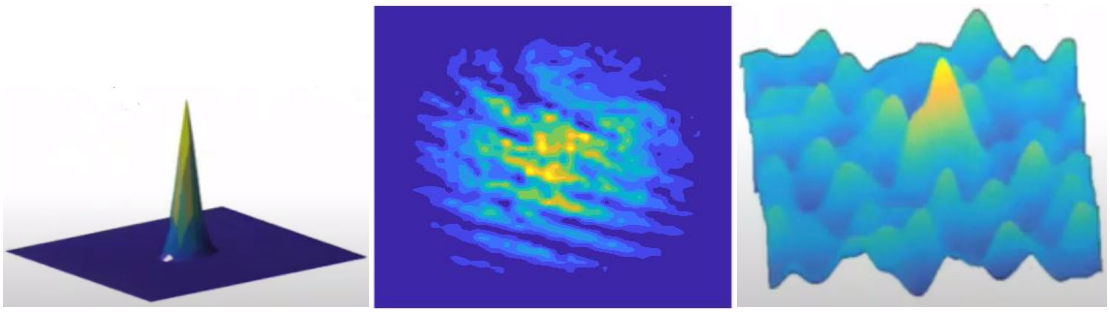
4. We further introduce the kernel trick to the temporally regularized correlation filter tracker, learned using multiple kernels. The proposed tracker captures the non-linearity in the appearance features and retains the circulant matrix structure.

CHAPTER 2

Related Work

2.1 Introduction

The key contribution of this thesis is the development of algorithms that are capable of efficiently tracking an object in an input video through various visual challenges. In order to fulfill the task, we utilise Correlation Filters (CFs), Long Short Term Memory (LSTM) networks, Spatio-Temporal Constraints, Channel Regularization, Kernel Tricks and pretrained deep networks that enable the extraction of rich and highly descriptive representational features of the target object. As such, this thesis connects to the following broad topics: CF based object tracking using deep features, spatio-temporal constraints, channel regularization and kernel tricks. In this chapter, we first give a summary of overall development of visual object trackers and then explain the classical CF formulation followed by an overview of the popular CF based tracking approaches. We also provide a review of current state-of-the-art developments in each of the broad topic areas, and summarise their strengths and weaknesses.



(a) Gaussian Shaped Label ($\mathbf{y}$) (b) Target Feature Channel ($\mathbf{x}_k$) (c) Correlation Filter Channel ($\mathbf{h}_k$)

Figure 2.1: Exemplar Images Showing Different Components of a Classical Correlation Filter

2.1.1 Evolution of Visual Object Tracking

The early representative methods of single object tracking are optical flow methods Horn and Schunck (1981); Lucas *et al.* (1981); Bouguet *et al.* (2001), filters meth-

ods (Kalman and Particle) Maybeck (1990); La Scala and Bitmead (1996); Julier and Uhlmann (2004); Nummiaro *et al.* (2003) and kernel-based methods Comaniciu and Meer (2002); Bradski (1998). However, complex calculations and low accuracy limited their further development.

To overcome the shortcoming of previous trackers, many innovative and efficient Correlation Filter (CF) based trackers were proposed Bolme *et al.* (2010); Danelljan *et al.* (2014*b*); Li and Zhu (2014); Danelljan *et al.* (2014*a*); Bertinetto *et al.* (2016*a*); Ma *et al.* (2015*b*); Kiani Galoogahi *et al.* (2015); Danelljan *et al.* (2015*b*,*a*, 2016*c*); Ma *et al.* (2015*a*); Danelljan *et al.* (2016*d*, 2017*a*); Bhat *et al.* (2018); Dai *et al.* (2019); Tang and Ling (2019); Lu *et al.* (2019*a*); Henriques *et al.* (2012); Dalal and Triggs (2005); Henriques *et al.* (2014).

The basic CF tracker, MOSSE Bolme *et al.* (2010), tracks an object at 615 FPS, but had room for improvement in both CFs and feature extraction. This provided the basis for a series of improvements like target scale estimation, use of hand crafted features Danelljan *et al.* (2014*b*); Li and Zhu (2014); Danelljan *et al.* (2014*a*); Bertinetto *et al.* (2016*a*); Ma *et al.* (2015*b*); Dong *et al.* (2016), use of kernel tricks Henriques *et al.* (2012); Dalal and Triggs (2005); Henriques *et al.* (2014); Huang *et al.* (2020), and elimination of boundary effects Kiani Galoogahi *et al.* (2015); Danelljan *et al.* (2015*b*, 2016*c*).

Correlation filters-based tracking methods performed better than traditional methods, but manual features are difficult to cope with the dynamic environment. Therefore, to achieve a balance between speed and accuracy, deep learning based trackers gradually gained popularity. The early deep learning trackers were based on offline training and online fine-tuning Wang and Yeung (2013); Wang *et al.* (2015*b*); Nam and Han (2016); Wang *et al.* (2015*a*); Lu *et al.* (2018). However, online fine-tuning caused trackers to run slowly which is difficult to meet real-time requirements. This gave rise to use of pre-trained Convolutional Neural Network (CNN) for tracking Hong *et al.* (2015); Held *et al.* (2016*a*); Nam *et al.* (2016). But, the pre-trained model focused more on extracting semantic features of each image, while object tracking tasks pay more attention to predicting the location of the labeled object in the video sequence. Therefore, to adjust the models by tracking datasets, recurrent neural network Cui *et al.* (2016); Ning *et al.* (2017); Redmon *et al.* (2016); Fan and Ling (2017*b*), Siamese network based trackers Tao *et al.* (2016); Li *et al.* (2018*b*); Wang *et al.* (2017*b*); Dong *et al.* (2020); Shen

et al. (2020); Dong *et al.* (2018, 2021); Zhu *et al.* (2018*a*); Li *et al.* (2019*a*); Valmadre *et al.* (2017*b*); Wang *et al.* (2017*a*); Zhu *et al.* (2018*b*); Xu *et al.* (2020); Gao *et al.* (2020) and regularized correlation filter based trackers trained using convolutional features were proposed Danelljan *et al.* (2015*a*); Ma *et al.* (2015*a*); Danelljan *et al.* (2016*d*, 2017*a*); Bhat *et al.* (2018); Dai *et al.* (2019); Lu *et al.* (2019*b*); Liu *et al.* (2020); Xu *et al.* (2019*a*); Rahman *et al.* (2020); Li *et al.* (2020*a*,*c*); Fu *et al.* (2020); Ma *et al.* (2020); Kristan *et al.* (2018, 2019); Javed *et al.* (2021).

Siamese networks have attracted great attention due to its outstanding performance. However, most of the Siamese-based methods only differentiate targets from non-semantic background, which is susceptible to interference from similar objects. Moreover, the templates of trackers based on the Siamese network cannot be updated which limits its tracking accuracy in complex environment (Valmadre *et al.*, 2017*a*; Kristan *et al.*, 2016*a*; Bertinetto *et al.*, 2016*b*). This is overcome by correlation filter trackers with dynamic weights and spatio-temporal regularizations that improves the discriminative ability of the filters and appearance model pool that facilitates easy template update, as shown in Chapter 4.

2.1.2 Classical Correlation Filter Tracker Formulation

The numerical formulation for a classical correlation filter tracker is given by,

$$E(\mathbf{h}) = \frac{1}{2}\left\| \mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k * \mathbf{h}_k \right\|_2^2 + \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{h}_k\|_2^2, \tag{2.1}$$

where K is the total number of feature channels, $\mathbf{y} \in \mathbb{R}^{T\times 1}$ is the desired Gaussian shaped correlation filter response. $\mathbf{x}_k \in \mathbb{R}^{T\times 1}$ is the vectorized feature and $\mathbf{h}_k \in \mathbb{R}^{T\times 1}$ is the learnable vectorized filter for the k^{th} channel. λ is a regularization parameter and $*$ is the spatial correlation operator. $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$ is the matrix of filters from all K channels (Bolme *et al.*, 2010). Exemplar images showing the Gaussian shaped correlation filter response ($\mathbf{y}$), feature channel ($\mathbf{x}_k$) and the unknown correlation filter channel ($\mathbf{h}_k$) are shown in Figure 2.1.

Before vectorizing $\mathbf{x}_k$ and $\mathbf{h}_k$, their original dimension is $W \times H$, where W is the width and H is the height of each channel. To generate the training samples for the correlation filter, we consider the circularly shifted versions of $\mathbf{x}$ along the dimensions

W and H. Let us denote each shifted sample as $\mathbf{x}_{a,b}$ (where $(a,b) \in \{0, 1, ..., W-1\} \times \{0, 1, ..., H-1\}$) to which we assign a Gaussian function label $y(a,b)$, given by

$$y(a,b) = e^{-\frac{(a-W/2)^2+(b-H/2)^2}{2\sigma^2}}, \tag{2.2}$$

where σ is the kernel width. A correlation filter $\mathbf{h}$, of the same size as $\mathbf{x}$, can be learned by minimizing Equation 2.1. Following Naresh Boddeti *et al.* (2013), Equation 2.1 can be solved using a Fast Fourier Transformation (FFT) for each feature channel. In the frequency domain, the learned filter for the k^{th} channel can be written as,

$$\hat{\mathbf{h}}_k = \frac{\hat{\mathbf{y}} \bigodot conj(\hat{\mathbf{x}}_k)}{\sum_{k=1}^{K} \hat{\mathbf{x}}_k \bigodot conj(\hat{\mathbf{x}}_k) + \lambda} \tag{2.3}$$

where $\hat{}$ represents the Discrete Fourier Transform (DFT), $k \in \{1, ...,K\}$, $\hat{\mathbf{y}}$ is the DFT of $\mathbf{y} = \big\{y(a,b)|(a,b) \in \{\mathbf{0}, \mathbf{1}, ..., W-\mathbf{1}\} \times \{\mathbf{0}, \mathbf{1}, ..., H-\mathbf{1}\}\big\}$, $conj$ indicates the complex conjugate, $\bigodot$ is the element-wise product, $\hat{\mathbf{h}}_k$ is the DFT of k^{th} channel of the learned correlation filter, and $\hat{\mathbf{x}}_k$ is the DFT of k^{th} channel of the vectorized features.

At test time, the features extracted from the incoming frame, $\mathbf{z}$, are of size $W \times H \times K$. The Fourier transformed CF response map, $\hat{\mathbf{r}}$, corresponding to $\mathbf{z}$ is calculated using

$$\hat{\mathbf{r}} = \mathrm{F}^{-1}\big(\sum_{k=1}^{K} \hat{\mathbf{h}}_k \bigodot conj(\hat{\mathbf{z}}_k)\big), \tag{2.4}$$

where F^{-1} is the inverse Fourier transform and $\hat{\mathbf{z}}$ is the Fourier transform of $\mathbf{z}$. The target location, $\boldsymbol{l} = (p_1, p_2)$, predicted using the CF is the position where the value of the corresponding filter response map (of size $W \times H$) is maximized and is given by

$$\boldsymbol{l} = (p_1, p_2) = \underset{x',y'}{argmax}\ \mathbf{r}(x', y'), \tag{2.5}$$

where p_1 and p_2 are the *x* and *y-axis* coordinates respectively. Following this calculation, at each new frame these locations are used to predict the target bounding box. This is followed by target scale estimation.

After predicting the target location and scale in the current frame, the correlation filters are updated to adapt to target appearance changes and facilitate accurate target localization in the next frame.

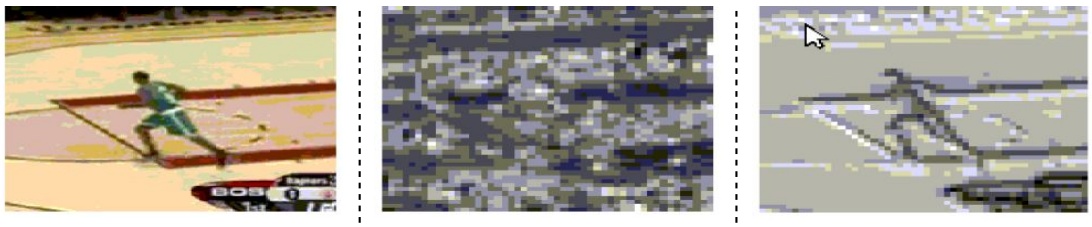
2.2 Correlation Filter based Object Tracking

Visual Object Tracking (VOT) is an active area of research in the field of computer vision. Tracking by detection is a popular way to achieve this task, where a binary classifier is learned to discriminate between the target (foreground) and background. These classifiers are usually learned online and updated incrementally. Unfortunately, such trackers are prone to model drift due to sampling ambiguity problems during updates. Although many improvements are suggested to overcome this issue (Babenko *et al.*, 2011; Gao *et al.*, 2014; Grabner *et al.*, 2008; Kalal *et al.*, 2012; Zhang *et al.*, 2014*a*; Hare *et al.*, 2016; Li *et al.*, 2014), the most popular approach is the Correlation Filter (CF) (Naresh Boddeti *et al.*, 2013). It alleviates sampling ambiguity by regressing the training samples to Gaussian labels enabling soft assignment. When learned in the Fourier domain, correlation filters are also computationally efficient.

Correlation filters have been widely explored to achieve fast visual tracking (Bolme *et al.*, 2010; Henriques *et al.*, 2012, 2015; Danelljan *et al.*, 2016*d*,*a*; Zhang and Suganthan, 2017; Wang *et al.*, 2018*a*) with several advances. In Danelljan *et al.* (2016*d*), a generic framework is proposed for learning discriminative convolution operators in the continuous spatial domain. This approach enables the integration of multi-resolution feature maps and is also capable of accurate sub-pixel localization. An improvement over Danelljan *et al.* (2016*d*) is proposed in Danelljan *et al.* (2016*a*), which revisits the core discriminative correlation filter formulation to overcome issues of over-fitting and computational complexity. They introduce a factorized convolution operator to reduce the number of parameters in the model to drastically reduce the memory and time complexity of learning. These trackers (Danelljan *et al.*, 2016*d*,*a*) are also among the top 5 ranked trackers on the VOT-2017 challenge Kristan2017a. In Bertinetto *et al.* (2016*a*) (staple), a simple combination of a correlation filter (using Histogram of Oriented Gradients (HOG) features) and a global color histogram is proposed, that offers tracking speed of up to 80 Frames Per Second (FPS). Other recent advances include clustering based ensemble tracking with a multi-scale kernelized correlation filter as proposed in Zhu *et al.* (2016), scale estimation as proposed in Danelljan *et al.* (2014*a*), spatio-temporal context learning proposed in Zhang *et al.* (2014*b*) and kernelized correlation filters proposed by Henriques *et al.* (2012), followed by its extension from single channel input features to multiple channels in Henriques *et al.* (2015) and Danelljan

et al. (2014*b*). However, these trackers use only one correlation filter learned over hand crafted features such as color attributes (Danelljan *et al.*, 2014*b*) and HOG (Dalal and Triggs, 2005). Such trackers do not maintain accurate object tracking during challenges like occlusions, object deformation and motion blur. This is because during challenging sequences, hand crafted features are insufficient for modelling the object's appearance, as illustrated in Figure 2.2. To address this issue, researchers have been motivated to explore deep learning for tracking, which has shown tremendous improvement compared to other classes of tracker (Li *et al.*, 2016; Qian *et al.*, 2018; Fan *et al.*, 2010; Wang and Yeung, 2013; Hong *et al.*, 2015; Song *et al.*, 2018).

Besides the above methods, many CF based trackers focus on modeling channel importance as each feature channel can make a dynamic contribution during each tracking step. However, these channel importance based CF trackers do not employ spatial (Danelljan *et al.*, 2015*a*,*b*, 2016*c*; Kiani Galoogahi *et al.*, 2015) or temporal (Dai *et al.*, 2019; Li *et al.*, 2018*c*) regularization. To combat the shortcomings of the existing regularization based (Dai *et al.*, 2019; Li *et al.*, 2018*c*) and channel importance based (Li *et al.*, 2018*a*; Sun *et al.*, 2018*a*; Zhou *et al.*, 2016) CF trackers, we investigate various classes of CF based trackers. A discussion of the evolution of correlation filter based tracking is provided in the subsequent sections.



(a) (b) (c)

Figure 2.2: Comparison of HOG features with features extracted using a pretrained Convolutional Neural Network: (a) Input image; (b) A HOG feature channel corresponding to (a); (c) A deep feature channel corresponding to (a)

2.2.1 Deep Feature based Correlation Filter Trackers

To improve their discriminative ability, CFs have evolved from using HOG (Dalal and Triggs, 2005) features and color names (Danelljan *et al.*, 2014*b*) to using deep Convolutional Neural Network (CNN) features (Ma *et al.*, 2015*a*; Qi *et al.*, 2016). Ma *et al.* (2015*a*) learns three CFs and Qi *et al.* (2016) learns six CFs over CNN features extracted from different convolutional layers of a VGG-Net (Simonyan and Zisserman,

2014). The target locations, predicted using each CF, are combined to compute one final location. Many part based (Liu *et al.*, 2016, 2015) and scale adaptive (Danelljan *et al.*, 2016*b*; Li *et al.*, 2017*a*; Li and Zhu, 2014; Danelljan *et al.*, 2014*a*) trackers have been proposed to handle challenges such as occlusion and size change. To further improve the tracking accuracy and robustness, particle filter based approaches (Zuo *et al.*, 2018) and continuous convolution filters that combine the feature maps with different spatial resolutions have also been proposed (Danelljan *et al.*, 2016*d*). Sui *et al.* (2018) introduces a CF based tracker that adopts joint learning to bridge the gap between circulant filtering and classical filtering approaches to enhance the discrimination performance of the filter.

Inspired by Qi *et al.* (2016) and Ma *et al.* (2015*a*), who both propose CNN based trackers that incorporate CFs, we investigate a deep learning based tracker that learns multiple CFs over multi-dimensional features extracted from the convolutional layers of VGG-Net (Simonyan and Zisserman, 2014). In addition to this, we introduce an appearance model pool to avoid faulty updates to the CFs and Felzenszwalb's HOG (FHOG) (Felzenszwalb *et al.*, 2010) based CFs that estimate target scale and rotation in each frame. We also use an Long Short Term Memory (LSTM) network as it models the temporal variations and retains information of the past appearance. Since the appearance of the object being tracked varies with time, an LSTM can inherently take these variations into account.

2.2.2 Correlation Filters with Spatio-Temporal Constraints

Although computationally efficient, CFs learned in the frequency domain are impacted by boundary effects that plague the circularly shifted training patches. This leads to sub-optimal training. Also, the learning is done solely using the shifted patches of the target, and background information is completely overlooked in the learning process. This results in an over-fitted CF that is prone to poor discrimination when encountering background clutter and occlusion (Danelljan *et al.*, 2015*b*). Several methods have been proposed to alleviate the aforementioned issues. Danelljan *et al.* (2015*b*) introduces a weighted regularization constraint in the CF formulation to penalize filter coefficients near the boundary. BACF (Kiani Galoogahi *et al.*, 2017) proposes generating real world positive and negative training samples by directly multiplying the filter with a binary

matrix. This results in improving the discriminative ability of the filter.

The above approaches have been used as a basis for many subsequent CF based trackers (Danelljan *et al.*, 2017*a*, 2016*d*; Li *et al.*, 2018*c*; Sun *et al.*, 2018*a*). Many other works propose different spatial constraints and use spatially larger training samples compared to the trained filter (Danelljan *et al.*, 2015*a*,*b*, 2016*c*; Kiani Galoogahi *et al.*, 2015). These approaches suppress the background information during training (Danelljan *et al.*, 2015*b*) and have demonstrated a significant reduction in boundary effects (Kiani Galoogahi *et al.*, 2015). Further advancements over SRDCF (Danelljan *et al.*, 2015*b*) are made in STRCF (Li *et al.*, 2018*c*) and ASRCF (Dai *et al.*, 2019). Li *et al.* (2018*c*) employs spatio-temporal constraints that utilize CFs learned in the previous frame to learn the CFs in the current frame. Dai *et al.* (2019) introduces an object aware spatial regularization that attempts to learn spatial weights that are similar to the reference spatial weights. The regularization terms in Li *et al.* (2018*c*) and Dai *et al.* (2019) make use of a reference to learn the CFs and spatial weights. However, the target appearance varies with every frame. Therefore, the spatial weights or CFs learned in consecutive frames should be constrained to be similar while still adapting to variations.

To combat the shortcomings of the above spatio-temporal regularization based (Dai *et al.*, 2019; Li *et al.*, 2018*c*) CF trackers, we investigate an object aware sparse spatial regularization and a sparse temporal regularization that also incorporates an adaptive channel importance estimation mechanism that assigns importance weights to each feature channel.

2.2.3 Correlation Filters with Channel and Graph Regularization

Training the CFs with multi-channel features has obvious advantages, however each feature channel may make a dynamic contribution during each tracking step. This is because the target in each sequence is different and its appearance varies throughout the sequence. Each feature channel encodes a different attribute of the target, and feature channels that encode useful visual cues may contribute to efficient tracking while channels that encode less useful information may degrade CF learning, eventually resulting in tracker drift.

Recently, Xu *et al.* (2019*a*) proposed an adaptive spatio-temporal-channel filter learning method that operates on a low dimensional manifold, with enhanced inter-

pretability of the learned model. This is achieved by reformulating the appearance learning model to incorporate group-sparse regularization and a temporal smoothness constraint. Li *et al.* (2018*a*) introduced a feature integration method for correlation filters, where the filters and importance maps are jointly learned in each frame. In this approach, an importance map for each feature is shared by all channels to avoid overfitting. Sun *et al.* (2018*a*) suggests that a CF trained for tracking should consider discrimination as well as reliability information to avoid model degradation, while the prior literature focuses only on the former. They propose a CF-based optimization method that jointly models discrimination and reliability information. Zhou *et al.* (2016) propose to learn a discriminative and robust dictionary that preserves the locality and similarity of the input to obtain more accurate visual tracking. This is obtained by explicitly considering the local geometric structure of the data, and results in the representation varying smoothly along the geodesics of the data manifold.

Besides these, graph theory (Liu *et al.*, 2019, 2020) has shown excellent performance in problems such as sparse feature extraction and dimensionality reduction (Lai *et al.*, 2015). Du *et al.* (2019) proposes a generic formulation that jointly learns the CFs and the channel reliability. To ensure high efficiency, the authors prove the upper bound of the objective function and estimate the channel weights efficiently by reformulating the objective function with it's upper bound. Li *et al.* (2018*c*) introduces a formulation that learns a robust multi-spectral feature representation for visual tracking. This formulation is driven by Alternating Direction Method of Multipliers (ADMM) and collaboratively uses multi-spectral information to learn an adaptive graph according to the intrinsic relationship between image patches. This allows the method to inherit advantages of both local and global graph models. However, these works do not learn the channel weights while preserving the similarity of importance between different feature channels. Consequently, similar feature channels are assigned dissimilar weights and feature channels with poor target representation are assigned higher weights, leading to tracking failure.

To learn adaptive weights for each feature channel and preserve the similarity of importance between feature channels, we use channel regularization and graph regularization, respectively. The source of inspiration for graph regularization is Zhou *et al.* (2016). The channel regularization is motivated by ridge regression wherein feature selection can be performed. In a similar way, we formulate our framework in a manner

in which channels important for tracking can be selected. Different works (Xu *et al.*, 2019*a*; Li *et al.*, 2018*a*; Sun *et al.*, 2018*a*; Zhou *et al.*, 2016) have explored this idea in different manners. Recently, in an independent work (Lu *et al.*, 2019*b*), a sparse regularizer was adopted to assign sparse weights to the channels. Though our channel regularization is similar in spirit, the overall formulation as channel-graph regularization is completely different from Lu *et al.* (2019*b*). The authors in Lu *et al.* (2019*b*) propose an L_1 channel regularization and use Accelerated Proximal Gradient (Tibshirani, 1996) to learn a sparse weight vector that assigns weights to each feature channel. The useful feature channels are assigned non-zero weights and the redundant information is suppressed by assigning zero weights to remaining channels. We employ L_2 channel regularization and channel-graph regularization to learn non-zero weight for each feature channel, while preserving the similarity of importance between different feature channels.

2.2.4 Correlation Filter Based Trackers with Kernel Trick

MOSSE (Bolme *et al.*, 2010) is one of the earliest trackers that learns a CF in the frequency domain using a few samples. It uses a single-channel gray-scale image to train the CF and offers impressive tracking speed. The limitations of MOSSE (Bolme *et al.*, 2010) have been addressed by many tracking algorithms proposed afterwards. To list a few, a multi-channel version of MOSSE (Bolme *et al.*, 2010) is proposed in Kiani Galoogahi *et al.* (2013). Henriques *et al.* (2012) proposed the use of high dimensional features with kernels. This is further improved by using HOG features in the Kernelized Correlation Filter (KCF) (Henriques *et al.*, 2014). To improve the discriminative ability of the KCFs, Danelljan *et al.* (2014*b*) use color attributes to learn adaptive CFs for tracking and maps the multi-channel features into a Gaussian kernel space. Scale adaptive KCF trackers are introduced in Li and Zhu (2014) and Bibi and Ghanem (2015), and an ensemble of KCF tracker is proposed in Uzkent and Seo (2018). A kernel based structured output correlation tracker is proposed in Hare *et al.* (2015). Zhang *et al.* (2016) introduces an output constraint transfer for the KCF. Tang and Feng (2015) extends KCF (Henriques *et al.*, 2014) to multiple kernels that enhance the model's distinguishability with the help of complementary features. A further improvement to this work is proposed in Tang *et al.* (2018) by taking advantage of the discriminative power spectrums of different features.

Although computationally efficient, KCF based trackers are not derived with spatial and temporal constraints as the constraints may not allow the straight forward use of the circulant matrix structure (Huang *et al.*, 2020). To this end, Huang *et al.* (2020) introduces a multi-kernel CF with spatial constraints that handles occlusion efficiently. Inspired by the above trackers, we derive a Temporally Regularized Multi-Kernel CF for tracking. The temporal constraint helps in obtaining more reliable filter coefficients for improved tracking and ensures that the tracker adapts to large appearance variations, preventing drift. Despite the constraints, we can kernelize and exploit the circulant matrix properties to speed up computations. We also investigate advantages of using multiple kernels in the proposed formulation as each kernel encodes a different attribute of the target.

2.3 Baselines Trackers

Section 2.2 and it's subsections provided an elaborate discussion about various classes of the Correlation Filter (CF) trackers along with their shortcomings and the improvements proposed in this thesis. To implement and evaluate the proposed improvements, we used various existing CF based trackers as our baseline. In this section, we provide an exhaustive list of all the baseline trackers used in this thesis, along with their numerical formulations.

BACF

The Background Aware Correlation Filter Tracker (BACF) (Kiani Galoogahi *et al.*, 2017) formulation can be given by,

$$E(\mathbf{h}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k * (\mathbf{P}^T \mathbf{h}_k)\right\|_2^2 + \frac{\lambda}{2}\sum_{k=1}^{K} \|\mathbf{h}_k\|_2^2, \tag{2.6}$$

where K is the total number of feature channels, $\mathbf{y} \in \mathbb{R}^{T\times 1}$ is the desired Gaussian shaped CF response. $\mathbf{x}_k \in \mathbb{R}^{T\times 1}$ is the vectorized feature and $\mathbf{h}_k \in \mathbb{R}^{T\times 1}$ is the vectorized filter for the k^{th} channel. λ is the regularization parameter and $*$ is the spatial correlation operator. $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$ is the matrix of filters from all K channels, $\mathbf{P} \in \mathbb{R}^{T\times T}$ represents a binary matrix that crops out the foreground region.

STRCF

The Spatial-Temporal Regularized Correlation Filters (STRCF) (Li *et al.*, 2018*c*) formulation can be given by,

$$E(\mathbf{h}, \mathbf{w}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k * \mathbf{h}_k\right\|_2^2 + \frac{1}{2}\sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2 + \frac{\theta}{2}\left\|\mathbf{h}^{(t)} - \mathbf{h}^{(t-1)}\right\|_2^2, \quad (2.7)$$

where θ is the regularization parameter, $\mathbf{w}$ are the spatial weights, $\frac{\theta}{2}\left\|\mathbf{h}^{(t)} - \mathbf{h}^{(t-1)}\right\|_2^2$ is the temporal regularization term, $\frac{1}{2}\sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2$ is the spatial regularizer. $\mathbf{h}^{(t)}$ and $\mathbf{h}^{(t-1)}$ are the CFs used in the t^{th} and $(t-1)^{th}$ frames respectively.

ASRCF

The Adaptive Spatially-Regularized Correlation Filters (ASRCF) (Dai *et al.*, 2019) formulation can be given by,

$$E(\mathbf{h}, \mathbf{w}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k * (\mathbf{P}^T\mathbf{h}_k)\right\|_2^2 + \frac{\lambda_1}{2}\sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2 + \frac{\lambda_2}{2}\|\mathbf{w} - \mathbf{w}_r\|_2^2, \quad (2.8)$$

where λ_1 and λ_2 are the regularization parameters, $\frac{\lambda_1}{2}\sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2$ is the spatial regularizer. $\mathbf{w}_r$ is the reference spatial weight and $\mathbf{w}$ is spatial weight to be learned.

KCF

The Kernelized Correlation Filters (KCF) (Henriques *et al.*, 2014) formulation is given by,

$$E(\mathbf{h}) = \frac{1}{2}\sum_{m=1}^{T} \left\|y_m - \mathbf{h}^\top \mathbf{x}_m\right\|_2^2 + \frac{\lambda}{2}\sum_{k=1}^{K} \|\mathbf{h}\|_2^2, \quad (2.9)$$

where $\mathbf{x}_m \in \mathbb{R}^{T\times 1}$ is the sample obtained after m cyclic shifts of $\mathbf{x}$, y_m is the m^{th} element of $\mathbf{y}$, and λ is the regularization parameter. The filter $\mathbf{h}$ is formulated as the linear combination of the training samples, $\mathbf{h} = \sum_{m=1}^{T} \boldsymbol{\psi}_i \phi(\mathbf{x}_m)$. Here $\phi(.)$ is a non-linear transformation. Using the Kernel Trick [16], we obtain,

$$\boldsymbol{\psi} = (\boldsymbol{\Omega} + \lambda I)^{-1}\mathbf{y} \quad (2.10)$$

where the vector $\boldsymbol{\psi} \in \mathbb{R}^{T\times 1}$ represents the solution of filter $\mathbf{h}$, $\boldsymbol{\Omega}$ is the kernel matrix with its elements defined as $\boldsymbol{\Omega}(m,n) = \phi(\mathbf{x}_m)^\top \phi(\mathbf{x}_n)$. Since $\boldsymbol{\Omega}$ is a circulant matrix, it can be calculated efficiently in the Fourier Domain (Henriques *et al.*, 2014). The objective function in Equation 2.10 can be equivalently expressed as,

$$\boldsymbol{\psi} = \mathrm{F}^{-1}\left(\frac{\hat{\mathbf{y}}}{\hat{\boldsymbol{\Omega}}_1 + \lambda I}\right), \tag{2.11}$$

where $\hat{}$ represents the Discrete Fourier Transform (DFT), the vector $\boldsymbol{\psi}$ contains all $\boldsymbol{\psi}_m$ coefficients and $\boldsymbol{\Omega}_1$ represents the first row of $\boldsymbol{\Omega}$. F^{-1} denotes the inverse discrete Fourier transfrom. During tracking, the target location for input $\mathbf{z}$ in the current frame t is determined using,

$$\hat{\mathbf{r}} = (conj(\hat{\boldsymbol{\Omega}}_{zx_{model}}) \odot \hat{\boldsymbol{\psi}}), \tag{2.12}$$

where $\hat{\mathbf{r}}$ is the DFT of CF response, $\mathbf{x}_{model}$ is the learned target appearance model, $\odot$ represents element-wise multiplication, $conj$ indicates the complex-conjugate, and $\boldsymbol{\Omega}_{zx_{model}}$ is the first row of the kernel matrix formed using $\mathbf{z}$ and $\mathbf{x}_{model}$. The location at which the response map, $\hat{\mathbf{r}}$, shows the maximum value is used to estimate the target location. The model at frame t is updated using,

$$\mathbf{x}_{model}^{(t)} = (1-\eta)\mathbf{x}_{model}^{(t-1)} + \eta\mathbf{x}^{(t)}, \tag{2.13}$$

$$\hat{\boldsymbol{\psi}}^{(t)} = (1-\eta)\hat{\boldsymbol{\psi}}^{(t-1)} - \eta\hat{\boldsymbol{\psi}}, \tag{2.14}$$

where η denotes the online adaptation rate.

2.4 Conclusion

In this chapter, we summarize various classes of Correlation Filter (CF) based trackers that successfully employ spatio-temporal regularizations and kernel tricks to overcome the shortcoming of the classical CF formulation. But these formulations can be further improved by investigating the techniques to avoid the faulty updates of the CFs and modelling the spatial and temporal variations along with the channel importance. This motivated us to explore an appearance model based CF formulation that prevents faulty filter updates and models the temporal variations using an Long Short Term Memory

(LSTM). We further investigate a channel graph regularization based CF tracker formulation that computes the contribution of different feature channels in tracking besides modelling the spatio-temporal variations. We also investigate temporally regularized CF tracker with kernel tricks, which was missing in the existing literature. Though the scope of this thesis is limited to CF based single object tracking, interested readers may wish to refer to Jr. and Belangour (2021) and Marvasti-Zadeh *et al.* (2021) for overview of end-to-end deep learning based tracking (Jr. and Belangour, 2021) and multi object tracking (Marvasti-Zadeh *et al.*, 2021).

CHAPTER 3

Tracker Evaluation Protocols

3.1 Introduction

Assessing the performance of a tracking algorithm with quantitative metrics is a challenging task. Many factors such as tracking speed, position estimation accuracy, robustness to a certain type of appearance change, and ease of use can be considered. Even in a single frame for which the tracking output and the groundtruth object state is available, there are several metrics to measure accuracy. When an algorithm loses track of the target object, it may resume tracking after failure using a re-detection module. If the re-detection module is absent, the tracker may fortuitously locate the target object again when the target reappears in the vicinity of the tracker's bounding box. In such case, simply observing the target bounding box location towards the end of an image sequence may not be fair since a tracker may have lost the target in the beginning but could have tracked the target successfully if it were initialized in a object state or frame.

Practically, a fair evaluation metric should provide the average performance of a tracker by considering its performance over the complete image sequence. Besides quantitative evaluation metrics, challenging tracking sequences are required to evaluate whether tracking algorithms perform robustly under different conditions. To accomplish this, we evaluate the trackers using several quantitative metrics on various publicly available tracking datasets that includes multiple visual challenges. In this chapter, we discuss the performance measures and tracking datasets that are used for evaluating the tracking algorithms in this thesis.

3.2 Evaluation using OTB Toolkit

The Object Tracking Benchmark (OTB) (Wu *et al.*, 2015) contains 100 fully annotated tracking sequences. To better analyse the strength and the weakness of the tracking algorithms, all the sequences are categorized according to 11 attributes: Illumination

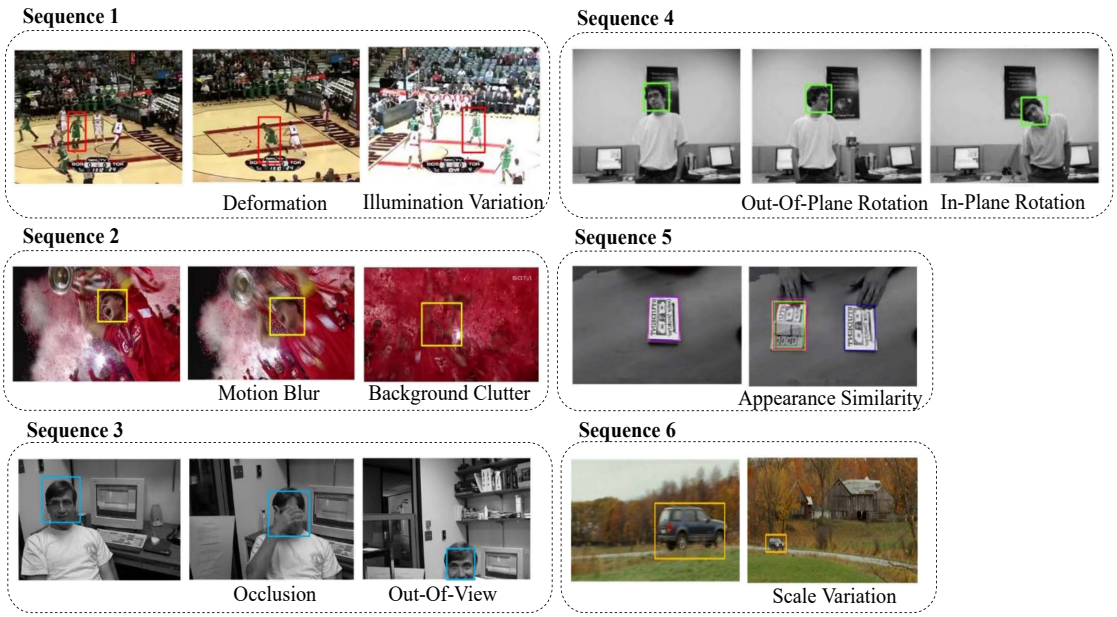


Figure 3.1: Visual Object Tracking Sequences Illustrating Different Challenges

Variation (IV), Occlusion (OCC), Size Variation (SV), Motion Change (MC), Motion Blur (MB), Fast Motion (FM), In-Plane Rotation (IPR), Out-of-Plane Rotation (OPR), Deformation (DEF), Object Out-of-View (OV), Background Clutter (BC), and Low Resolution (LR). The definition of each attribute is provided in Table 3.1 and the pictorial representation is provided in Figure 3.1. To evaluate the trackers on OTB (Wu *et al.*, 2015), we use the following metrics.

3.2.1 Precision plot

The Center Location Error (CLE) computes the Euclidean distance between the center location of the tracked target and the manually labeled groundtruth position in a frame. A precision plot is used to records the average CLE for all the frames in a sequence/dataset. When a tracking algorithm loses track of a target object, the output location can be random, and thus, the average error value does not measure the tracking performance correctly [5]. Therefore, the percentage of frames in which the estimated locations are within a given threshold distance of the groundtruth positions is a better metric to measure tracking performance [5], [34]. In this thesis, we measure the precision rate as the value of precision plot at a threshold distance of 20 pixels [5]. However, the CLE only measures the pixel distance and does not reflect the size and scale of the target object.

Table 3.1: List of Visual Challenges in Publicly Available Tracking Datasets

Attributes	Abbreviation	Attribute Description
Illumination Variation	IV	Significant illumination change in the target region
Scale Variation	SV	The ratio of the bounding boxes of the first frame and the current frame changes between the range [0.5, 2]
Occlusion	OCC	The target is partially (POC) or fully occluded (FOC)
Deformation	DEF	Shape change of the target object
Motion Blur	MB	The target region is blurred due to the target or camera motion
Fast Motion	FM	The motion of the target in consecutive frames is larger than 20 pixels
In-Plane Rotation	IPR	Target rotation in the image plane
Out-of-Plane Rotation	OPR	Target rotation out of the image plane
Out-of-View	OV	Target is fully or partially out of the frame
Background Clutter	BC	The background near the target has texture or color similar to to the target
Low Resolution	LR	The number of pixels inside the groundtruth bounding box are less than 400
Viewpoint Change	VC	Change in the point from where the target is being viewed
Aspect Ratio Change	ARC	The ratio of bounding box aspect ratio is outside the range [0.5, 2]

3.2.2 Success plot

Another commonly used evaluation metric is the Intersection-Over-Union (IOU) ratio between the tracked and the groundtruth bounding box [21]. At any frame t, given a tracked bounding box, $\mathbf{b}_t$, and the groundtruth bounding box, $\mathbf{b}'_t$, of a target object, the IOU is defined as $IOU = \frac{|\mathbf{b}_t \cap \mathbf{b}'_t|}{|\mathbf{b}_t \cup \mathbf{b}'_t|}$, where $|.|$ denotes the number of pixels in a region, $\cap$ represents the intersection operator, and $\cup$ represents the union operator. The average IOU can be used as a performance measure. In addition, the success rate is computed as the ratio of the number of successfully tracked frames (i.e., IOU between groundtruth and the tracked bounding box larger than a pre-defined threshold, typically $t_o = 0.5$) to the total number of frames in a sequence. As the threshold varies between 0 and 1, the success rate changes and the resultant curve is presented as success plot in this work.

3.3 Evaluation using the TC128 Toolkit

The Temple Color-128 (TC128) dataset (Liang *et al.*, 2015) contains 128 color sequences, 50 of which are taken from previous studies , and 78 are collected for TC128. The color sequences collected from the previous studies are insufficient for a thorough evaluation of color trackers. Therefore, 78 new color sequences are collected from the internet to increase the diversity and difficulty. Each sequence from TC128 (Liang *et al.*, 2015) contains the annotation for the groundtruth bounding box. In addition to the groundtruth bounding boxes, each sequence is also annotated by its challenging factors. Similar to OTB (Wu *et al.*, 2015), TC128 (Liang *et al.*, 2015) offers 11 visual challenges including Illumination Variation (IV), Scale Variation (SV), Occlusion (OCC),

Deformation (DEF), Motion Blur (MB), Fast Motion (FM), In-Plane Rotation (IPR), Out-of-Plane Rotation (OPR), Out-of-View (OV), Background Clutters (BC), and Low Resolution (LR). The definition for each attribute is provided in Table 3.1.

Following the protocol in OTB (Wu *et al.*, 2015), the performance evaluation over TC128 (Liang *et al.*, 2015) is done using the average success rate and precision rate, which are derived from the success plot and precision plot, respectively.

3.4 Evaluation using the LaSOT Toolkit

The Large-scale Single Object Tracking Benchmark (LaSOT) (Fan *et al.*, 2019) consists of 85 object classes, which are divided into two parts. The first part contains 1,400 sequences from 70 object categories, most of which are chosen from the 1,000 classes from ImageNet [22]. The other portion comprises 150 sequences for the remaining 15 object classes. These 15 classes are carefully chosen to lie outside the object categories in ImageNet [22] and are intended to be non-overlapping with the 70 categories from the first pprtion. In order to analyze tracker performance, each sequence in LaSOT (Fan *et al.*, 2019) is annotated with the attributes listed in Table 3.1.

The evaluation on the LaSOT dataset is performed using the 150 test videos. We measure the tracker performance in terms of precision rate and success rate (as per OTB (Wu *et al.*, 2015) and TC128 (Liang *et al.*, 2015)). Since precision does not take object scale into consideration, an additional strategy to normalize the precision according to scale is adopted [67]. The resultant normalized precision ensures consistency of the evaluation across different target scales.

3.5 Evaluation on UAV123 Dataset

The Unmanned Aerial Vehicle-123 (UAV123) dataset (Mueller *et al.*, 2016) contains a total of 123 video sequences and more than 110K frames. In UAV123 (Mueller *et al.*, 2016), 103 sequences are collected using an off-the-shelf professional-grade UAV (DJI S1000) that captures different objects from altitudes varying between 5-25 meters. These sequences were recorded at frame rates between 30 and 96 Frames Per Second (FPS) and resolutions between 720p and 4K. A set of 12 sequences is captured from

a boardcam (with no image stabilization) mounted on a small low-cost UAV following other UAVs. These sequences are of lower quality and resolution and contain a reasonable amount of noise due to limited video transmission bandwidth. All these sequences are annotated with upright bounding boxes at 30 FPS. Another set of 8 synthetic sequences is captured by UAV simulator that automatically provides annotations at 30 FPS and a full object mask/segmentation.

The evaluation over the UAV123 dataset is done using the success plot, precision plot, success rate and precision rate, as per OTB (Wu *et al.*, 2015) and TC128 (Liang *et al.*, 2015).

3.6 Evaluation using GOT-10k Toolkit

The GOT-10k dataset (Huang *et al.*, 2019*a*) contains 420 test videos with 84 classes of moving objects and 31 forms of motion. For this dataset, we employ simple metrics with clear meaning for the evaluation of trackers. We choose the widely used success rate and Area Under Curve (AUC) of the success plot, which is the average of the success rates corresponding to the sampled overlap thresholds.

3.7 Evaluation on the Tracking Dataset

The Tracking Dataset, introduced by Tomas Vojir (Vojir *et al.*, 2014), consists of 77 sequences collected from the published literature (Kalal *et al.*, 2012; Babenko *et al.*, 2011). It includes sequences from the OTB Wu *et al.* (2015) and various VOT (Kristan *et al.*, 2017*b*, 2019) datasets. Hence, the Tracking Dataset is full of challenging sequences at a scale similar to that of OTB and VOT. Sequences in this dataset vary in length from dozens of frames to thousands and contain diverse object types including articulated and rigid objects. It has different scene settings including static/moving cameras, indoor/outdoor and varied lightning conditions.

To perform evaluations over this dataset, we use success rate, precision rate, average IOU and average CLE for challenges including Camera Motion (CM), Occlusion (OCC), Illumination Variation (IV), Low Resolution (LR), Size Change (SC) and Motion Changes (MC).

3.8 Evaluation using the VOT-Toolkit

To evaluate a tracker, the VOT toolkit (Kristan *et al.*, 2017*b*, 2019) applies a reset-based methodology, where the performance is measured in terms of Accuracy and Robustness (Čehovin *et al.*, 2016*b*). In the reset-based methodology, a failure is detected whenever a tracker predicts a bounding box with zero overlap with the ground truth. As a result, the tracker is re-initialized five frames after the failure. The average overlap between the groundtruth and the tracked bounding boxes during successful tracking periods is captured by the Accuracy measure, and the number of times the tracker loses the target (fails) during tracking determines the Robustness. To reduce the bias due to resets, the ten frames directly after re-initialization are ignored in the accuracy measure (Kristan *et al.*, 2016*b*). Another evaluation measure is Expected Average Overlap (EAO), which estimates the average overlap a tracker is expected to attain on a large collection of short term sequences with the same visual properties in the given dataset. This measure is used to address the increased variance and bias of the average overlap measure (Wu *et al.*, 2015), when evaluating over variable sequence lengths. The results are obtained for the baseline (reset based) and unsupervised (no-reset) experiments (Kristan *et al.*, 2017*b*).

CHAPTER 4

LSTM-AMP: LSTM Guided Ensemble Correlation Filter Tracking with Appearance Model Pool

4.1 Introduction

In this chapter, we propose a deep learning based tracker that predicts the target location based on multi-layer convolutional feature aggregation, scale and rotation estimation, and an appearance model pool. Compared to existing trackers that use the output of a single Convolutional Neural Network (CNN) layer for target location estimation, multi-layer trackers perform better (Qi *et al.*, 2016; Ma *et al.*, 2015*a*). This is because deep layers in a CNN capture only category-level semantic information, which is best suited to classification tasks. Tracking requires a feature representation that also contains spatial information for accurate target localization.

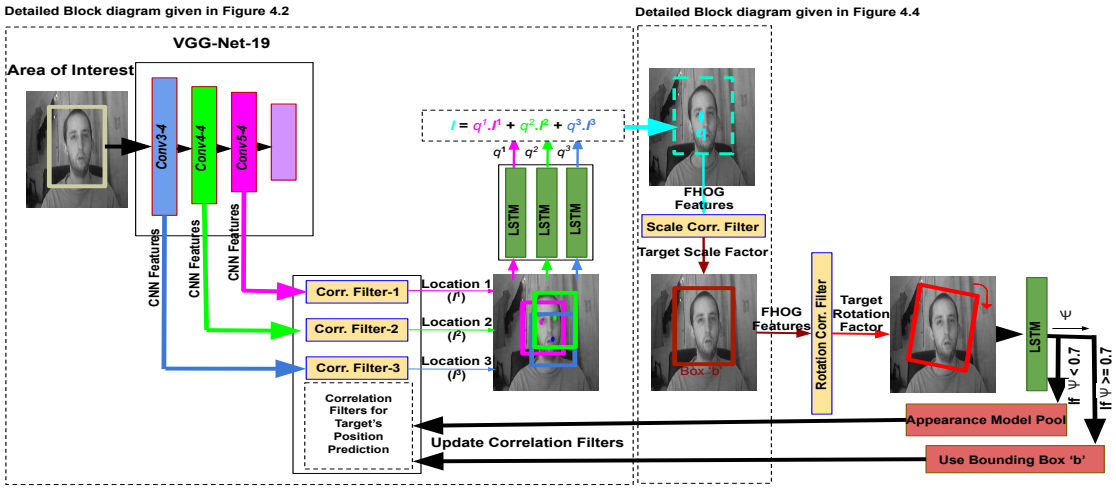


Figure 4.1: Block Diagram of the LSTM-AMP: To calculate target locations $\boldsymbol{l}^1$, $\boldsymbol{l}^2$ and $\boldsymbol{l}^3$, the features of *conv*3-4, *conv*4-4 and *conv*5-4 layers act as an input to Correlation Filters (CFs) 1, 2 and 3 respectively. To calculate the weight of $\boldsymbol{l}^1$, $\boldsymbol{l}^2$ and $\boldsymbol{l}^3$, an LSTM network is used, followed by target scale and rotation estimation. Finally, location CFs are updated using appearance model pool

Therefore, we combine activations from deep layers with those of earlier layers (that capture more spatial information), obtaining a more reliable prediction. However, the

prediction accuracy of each layer may vary from frame to frame and combining all the predictions with equal weight may result in inaccurate tracking. Thus, a reliable mechanism that can compute the contribution of each layer when estimating the target's final location is required. To this end, we use a Long Short Term Memory (LSTM) network that computes the prediction accuracy of each layer, and which is used to assign a weight to each layer. The target's final location is predicted as the weighted sum of predictions, using the above mentioned weights from different layers. However, during situations like illumination variation, occlusion or appearance change, ambiguous training samples may lead to a poor Correlation Filter (CF) update.

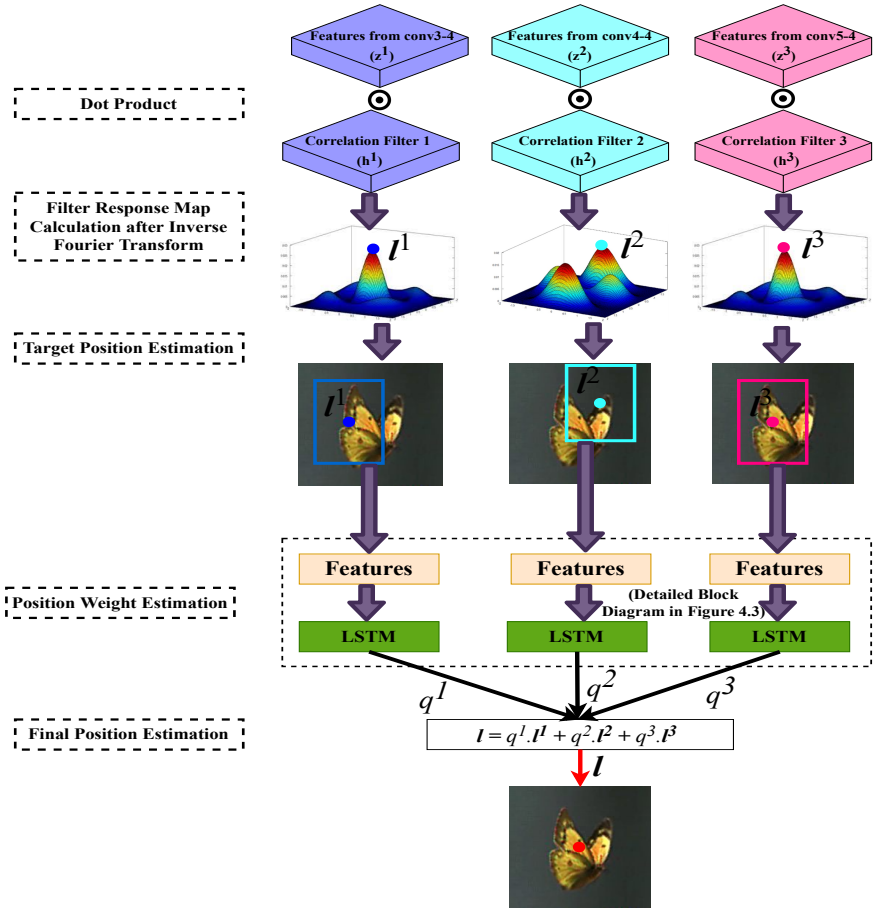


Figure 4.2: Target Location Prediction using the Correlation Filter and LSTM

This may result in tracking failure during longer sequences. To overcome this, it is imperative to introduce a corrective measure that can prevent faulty updates of the CF. Thus, we introduce an appearance model pool that stores the appearances of the object that matches most with the reference appearance and provides the CFs with the best update template during challenging scenarios. This helps prevent tracker drifting during occlusion and other challenges, resulting in successful long term tracking. Further,

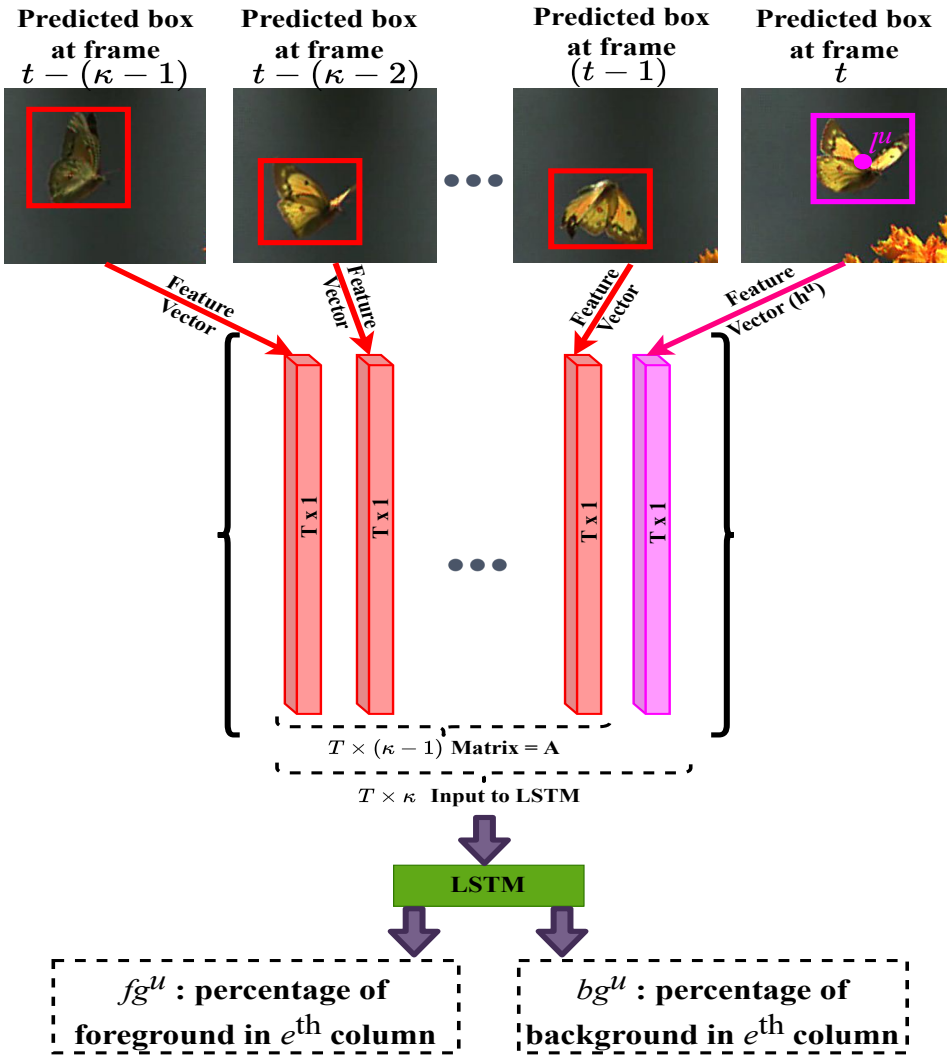


Figure 4.3: Using LSTM for Estimating Foreground ($fg\%$) and Background ($bg\%$) Percentages in the Target Bounding Box. Here, we compute fg and bg present in the magenta bounding box centered at position $l^{\mathbf{u}}$, predicted using convolutional layer u

during the course of tracking, the object can undergo scale and rotation variations. In order to address these variations, we use a strategy based on Felzenszwalb's Histogram of Oriented Gradients (FHOG) features (Felzenszwalb *et al.*, 2010), that estimates target scale and rotation in each frame.

To summarize, our major technical contributions in this chapter are as follows:

- We introduce an LSTM network that adaptively learns the weights to combine the target location predicted by each CF. This helps in improving the aggregation of predictions from different layers.
- We introduce an appearance model based correction module that avoids faulty updates to the CFs. This promotes efficient long term tracking.
- To further improve the performance, an FHOG feature (Felzenszwalb *et al.*, 2010) based CF is learned to estimate the target rotation in each frame.

Table 4.1: Overlap Success (OS) rate and Distance Precision (DP) rate calculated over the VOT2016 and VOT2017 datasets, for entire dataset and for selected videos. "Selected Videos" refers to videos with object size of between 200×200 and 250×250 pixels. The best and second best performance is shown in red and green color, respectively

	For VOT2016 Dataset				For VOT2017 Dataset			
	On Entire Dataset		On Selected Videos		On Entire Dataset		On Selected Videos	
	OS (%)	DP (%)	OS (%)	DP (%)	OS (%)	DP (%)	OS (%)	DP (%)
LSTM1-FC1	**35.20**	50.40	39.40	47.20	**29.60**	**43.60**	24.80	31.90
LSTM1-FC2	34.90	**50.70**	**40.40**	**51.00**	**28.60**	42.50	24.80	**32.40**
LSTM4-FC1	32.80	46.90	39.80	46.50	27.30	40.40	25.00	31.70
LSTM4-FC2	32.20	45.90	40.00	47.10	26.50	39.30	25.10	32.20
LSTM8-FC1	32.40	46.90	40.00	47.10	26.80	40.40	25.20	32.20
LSTM8-FC2	34.00	48.70	39.80	47.20	28.10	41.90	25.20	32.30
LSTM1-PCA-FC1	32.30	45.50	**41.00**	47.10	26.50	39.00	**25.90**	**32.90**
LSTM1-PCA-FC2	33.10	48.40	39.50	45.90	26.70	40.20	24.80	31.20
LSTM4-PCA-FC1	32.30	46.80	39.90	47.10	26.80	40.10	25.10	32.30
LSTM4-PCA-FC2	32.00	45.30	40.30	47.30	26.40	39.00	**25.40**	**32.40**
LSTM8-PCA-FC1	33.50	49.10	40.10	47.10	27.20	41.10	25.20	32.20
LSTM8-PCA-FC2	31.30	44.80	39.00	45.90	25.60	38.10	24.30	31.20
LSTM1-HOG	32.90	48.10	39.40	45.90	27.6	41.7	24.80	31.40
LSTM4-HOG	32.50	46.90	40.00	47.10	26.8	40.4	25.20	32.20
LSTM8-HOG (Proposed)	**36.50**	**53.70**	39.90	**50.30**	**28.60**	**42.80**	24.80	31.40
LSTM12-HOG	32.60	46.70	40.00	47.20	27.00	40.20	25.10	32.20

- We perform an extensive quantitative and qualitative analysis of our tracker and show competitive results on Object Tracking Benchmark (OTB100) (Wu *et al.*, 2015), Visual Object Tracking (VOT) - 2016 Dataset (Kristan *et al.*, 2016*c*), VOT-2017 Dataset (Kristan *et al.*, 2017*a*), Tracking Dataset (Vojir *et al.*, 2014) and UAV123 Dataset (Mueller *et al.*, 2016). We also conduct an ablation study to investigate the contribution of various components of our tracker in the tracking performance, demonstrating the benefits of the proposed approach.

The rest of this chapter is structured as follows. Section 4.2 describes the proposed tracker with each module in Figure 4.1 explained in detail. Section 4.3 shows evaluation of the LSTM component. Section 4.4 shows the experimental results obtained by testing the tracker on standard video datasets. It also shows the comparative analysis of the tracker with other recent trackers. Section 4.5 presents an ablation study that shows the contribution of each module to the overall tracker performance and Section 4.6 concludes the chapter.

4.2 LSTM guided ensemble tracking with appearance model pool

We propose a Convolutional Neural Network (CNN) feature based tracker that, in parallel, learns multiple Correlation Filters (CFs) over CNN features extracted from the hier-

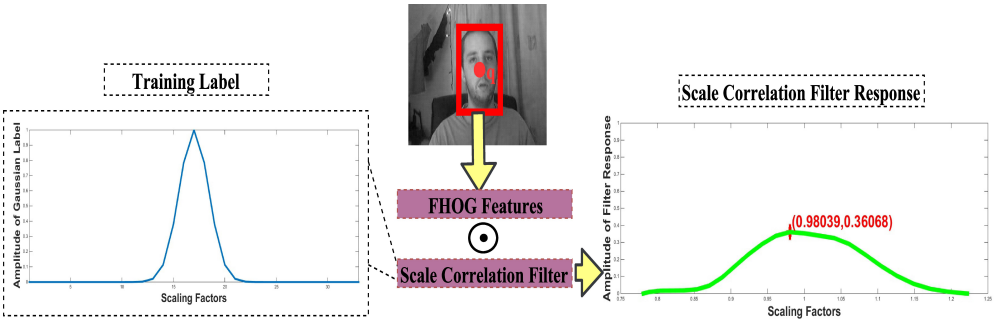


Figure 4.4: Target Scale Prediction using Scale Correlation Filter: Output scaling factor (*scale*) is equal to value where filter response is maximum (here *scale* = 0.98039)

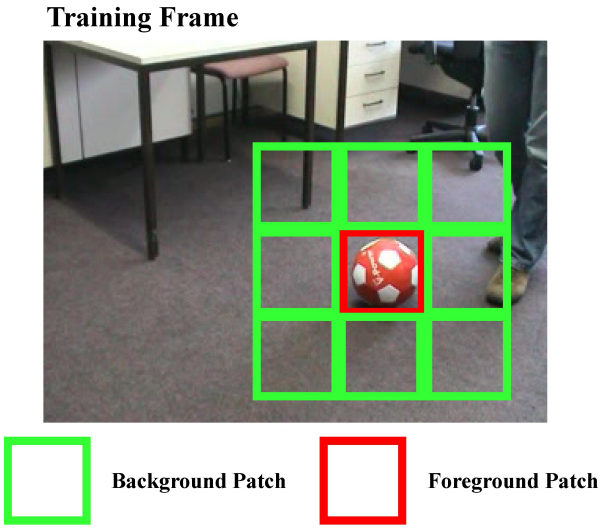


Figure 4.5: Foreground and background patch extraction to train the LSTM

archical convolutional layers of a VGG-Net (Simonyan and Zisserman, 2014). Each CF response contributes to estimating the position of the target in a frame. In addition, to estimate the target's scale and rotation, CFs are learned over Felzenszwalb's Histogram of Oriented Gradients (FHOG) features (Felzenszwalb *et al.*, 2010). For target localisation, robust and rich features are required to efficiently handle multiple challenges in a sequence. Therefore, VGG features are used (Simonyan and Zisserman, 2014). However, once the target is localised FHOG features are sufficient for scale and rotation estimation (Ma *et al.*, 2015*a*; Danelljan *et al.*, 2014*a*, 2017*b*), and their use helps reduce computational costs. Finally, an appearance model pool based correction module is added to prevent faulty updates to the CFs. The block diagram in Figure 4.1 shows how each component of the proposed tracker is combined to form an accurate ensemble tracker. The function of each component is subsequently explained in the sub-sections below.

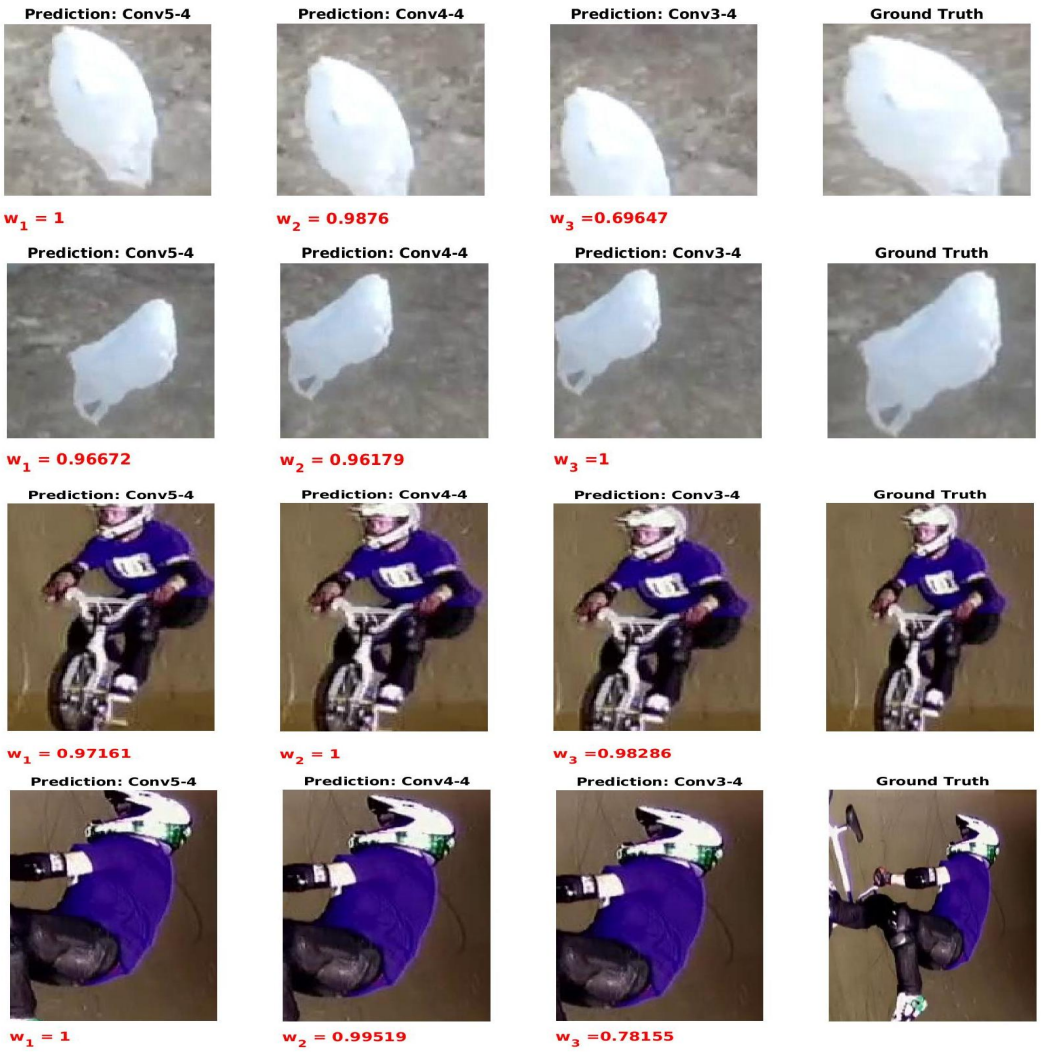


Figure 4.6: Estimating the weights of predictions corresponding to layer *Conv5-4* (column 1), *Conv4-4* (column 2) and *Conv3-4* (column 3). Row 1 is frame #5 and Row 2 is frame #26 from Sequence *'bag'*. Row 3 is frame #3 and Row 2 is frame #27 from Sequence *'bmx'*.

4.2.1 Feature Representation for Target Appearance

As suggested by prior literature, trackers based on hand crafted features such as HOG (Dalal and Triggs, 2005), SIFT (Lowe, 1999), and color histogram (Tian *et al.*, 2009; Adam *et al.*, 2006) often fail in sequences with multiple challenges. This is because such features are inadequate to model the rich appearance of the target. Therefore, to generate a feature representation that can sustain tracking during multiple simultaneous challenging conditions, a robust and rich feature representation of the target appearance is desirable. To resolve this, the majority of recent trackers are based on features extracted using CNNs. Amongst a number of CNNs like AlexNet (Krizhevsky *et al.*, 2012), R-CNN (Girshick *et al.*, 2014), CaffeNet (Jia *et al.*, 2014), and VGG-Net (Simonyan and Zisserman, 2014) (that are trained for image classification and object

Figure 4.7: Failure case in computing the weights of predictions corresponding to layer *Conv5-4* (column 1), *Conv4-4* (column 2) and *Conv3-4* (column 3). Row 1 is frame #14 of Sequence *'bag'* and Row 2 is frame #33 of Sequence *'bmx'*

recognition tasks), VGG-Net is the most popular network used by object tracking approaches to generate features representing the target appearance. This chapter also uses VGG-Net, whose input at frame t is the image patch cropped by a bounding box that is twice the size of the bounding box predicted at frame $t-1$, the center of both the boxes being the same (Qi *et al.*, 2016). Before feeding the image to the network, it is resized to 224×224.

4.2.2 Correlation Filters

A CF is trained corresponding to each convolutional layer, u, where $u \in \{1, 2, .., n\}$ and n is the number of layers used for predicting the target's location. For an input image patch, each convolutional layer gives a 3-dimensional feature matrix. For a layer u, let $\mathbf{x}^u$ be the feature matrix of dimension $W \times H \times K$, where W is the width, H is height and K is the number of channels. To generate the training samples for the CF, we consider all the circularly shifted versions of $\mathbf{x}^u$ along the dimensions W and H. Ignoring the subscript u here, let us denote each shifted sample as $\mathbf{x}_{a,b}$ (where $(a,b) \in \{0, 1, ..., W-1\} \times \{0, 1, ..., H-1\}$) that has a Gaussian function label $y(a,b)$ given by

$$y(a,b) = e^{-\frac{(a-W/2)^2+(b-H/2)^2}{2\sigma^2}}, \tag{4.1}$$

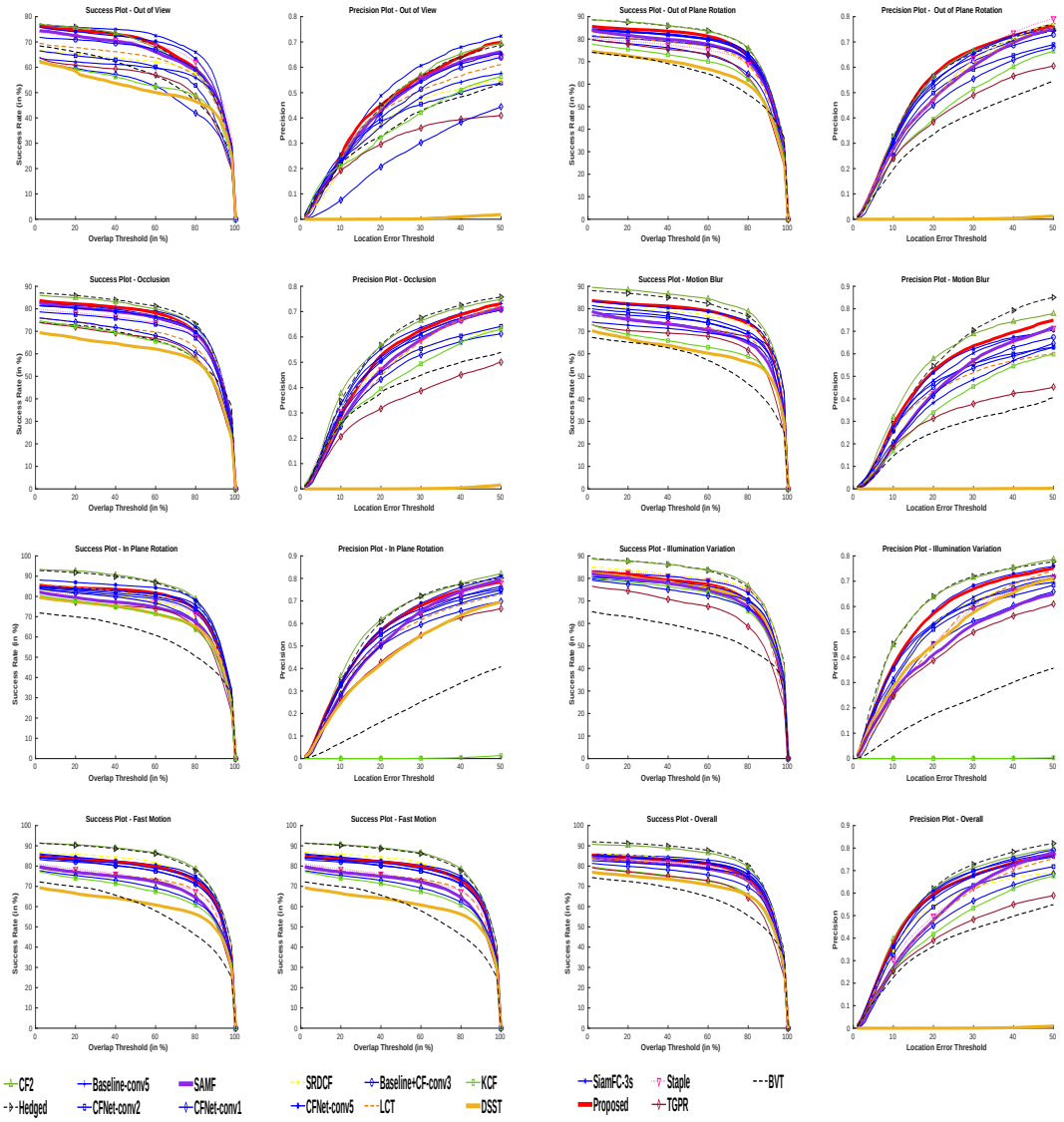


Figure 4.8: Success and Precision plots obtained over the OTB100 dataset (Wu *et al.*, 2015) for Object Out-of-View, Out-of-Plane Rotation, Occlusion, Motion Blur, In-Plane-Rotation, Illumination Variation, Fast Motion prone videos and overall performance.

where σ is the kernel width. A CF $\mathbf{h}^u$ corresponding to the layer u, of the same size as $\mathbf{x}^u$, can now be learned by minimizing:

$$E(\mathbf{h}^u) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k^u * \mathbf{h}_k^u\right\|_2^2 + \frac{\lambda}{2}\sum_{k=1}^{K} \|\mathbf{h}_k^u\|_2^2, \tag{4.2}$$

where $k \in \{1, ..., K\}$ are the number of feature channels, '$\cdot$' operator is the inner product, λ is a regularization parameter and $\lambda \geq 0$. Since the above equation is similar to training the CF in Naresh Boddeti *et al.* (2013), it can be solved using a Fast Fourier Transformation (FFT) for each feature channel. In the frequency domain, the learned

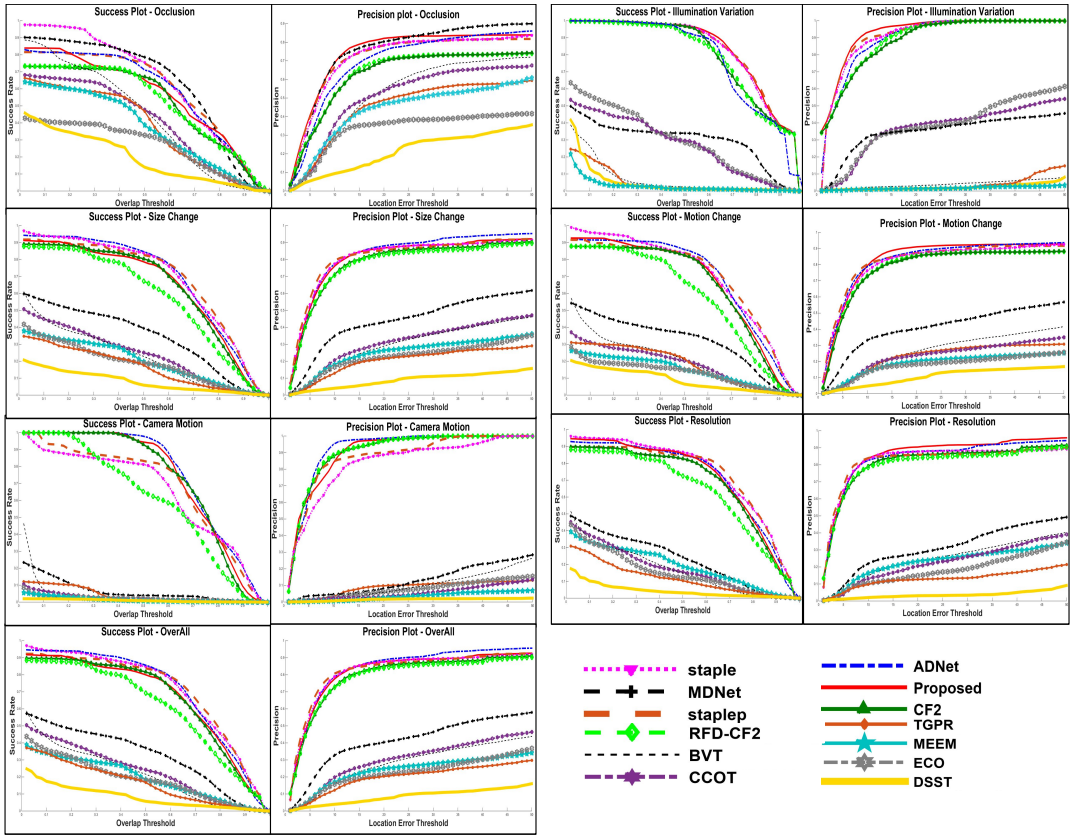


Figure 4.9: Success and Precision plots obtained over Tracking Dataset (Vojir *et al.*, 2014) for Occlusion, Illumination Variation, Size Change, Motion Change, Camera Motion, Low Resolution prone videos and overall performance.

filter for the k^{th} channel can be written as,

$$\hat{\mathbf{h}}_k^u = \frac{\hat{\mathbf{y}} \bigodot conj(\hat{\mathbf{x}}_k^u)}{\sum_{k=1}^{K} \hat{\mathbf{x}}_k^u \bigodot conj(\hat{\mathbf{x}}_k^u) + \lambda} \tag{4.3}$$

where $\hat{\mathbf{y}}$ is the Discrete Fourier Transform (DFT) of $\mathbf{y} = \{y(a,b)|(a,b) \in \{0,1,...,W-1\} \times \{0,1,...,H-1\}\}$, $conj$ indicates the complex conjugate, $\bigodot$ is the element-wise product, $\hat{\mathbf{h}}_k^u$ is the DFT of k^{th} channel of the CF learned for layer u, and $\hat{\mathbf{x}}_k^u$ is the DFT of k^{th} channel of the CNN features learned from layer u. The calculation shown above is valid for every convolutional layer. Now, assume for any layer u, features extracted from the new incoming frame are $\mathbf{z}^u$ of size $W \times H \times K$. The Fourier transformed CF response map, $\hat{\mathbf{r}}^u$, for the corresponding layer is calculated using

$$\hat{\mathbf{r}}^u = \mathrm{F}^{-1}\Big(\sum_{k=1}^{K} \hat{\mathbf{h}}_k^u \bigodot conj(\hat{\mathbf{z}}_k^u)\Big), \tag{4.4}$$

where F^{-1} is the inverse Fourier transform and $\hat{\mathbf{z}}^u$ is the DFT of $\mathbf{z}^u$. The target location, $\boldsymbol{l}^u = (p_1^u, p_2^u)$, predicted using this layer is the position where the value of the corresponding

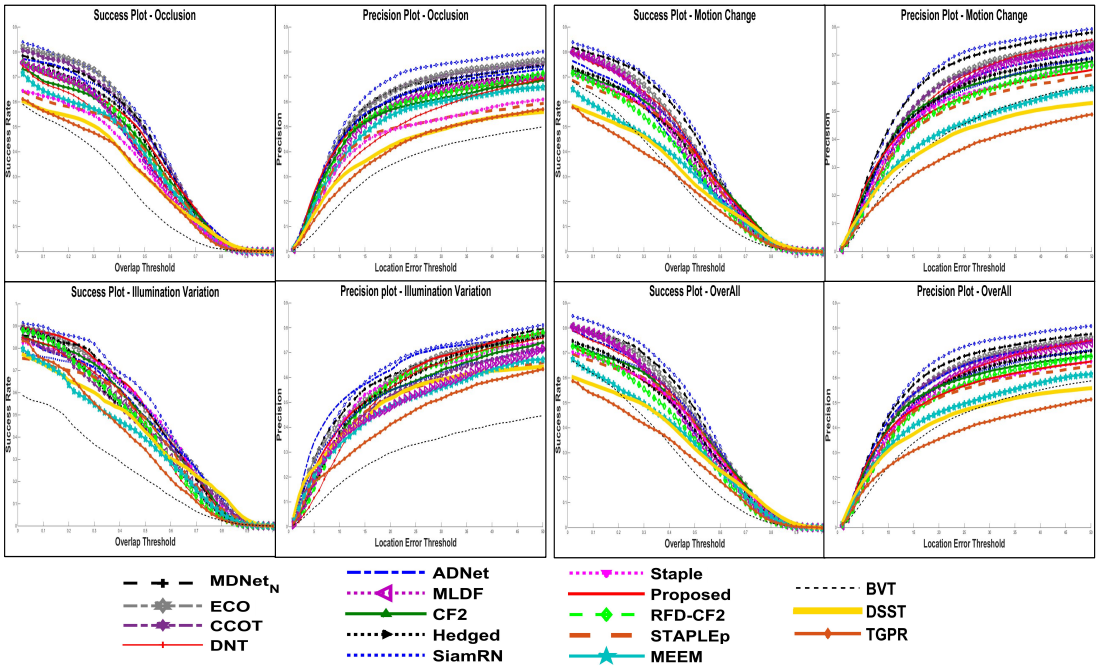


Figure 4.10: Success and Precision plots obtained over VOT-2016 dataset (Kristan *et al.*, 2016*c*) for Occlusion, Motion Change, Illumination Variation prone videos and overall performance. (Note: Zoom in for better view)

filter response map (of size $W \times H$) is maximized and is given by

$$\boldsymbol{l}^u = (p_1^u, p_2^u) = \underset{x',y'}{argmax}\ \mathbf{r}^u(x', y'), \tag{4.5}$$

where p_1^u and p_2^u are the x and y-*axis* coordinates respectively. Following this calculation, at each new frame, a target location is obtained corresponding to each convolutional layer. This can be seen in Figure 4.2. These locations are used to predict the final location of the target, the procedure for which is explained in the following sub-section.

4.2.3 Target Position Estimation using LSTM Network

As already discussed, for every frame, features from each convolutional layer u are used to predict a target location $\boldsymbol{l}^u = (p_1^u, p_2^u)$. In this work, $u \in \{1, 2, 3\}$, where $u = 1$ denotes layer conv3-4, $u = 2$ denotes layer conv4-4 and $u = 3$ denotes layer conv5-4. However, the predictions corresponding to the CFs operating over the different convolution layers may not be equally reliable. This is because, at each new frame, information gathered by the features from only one layer may not sufficiently represent the target. Therefore, to predict the final location of the target, locations predicted from

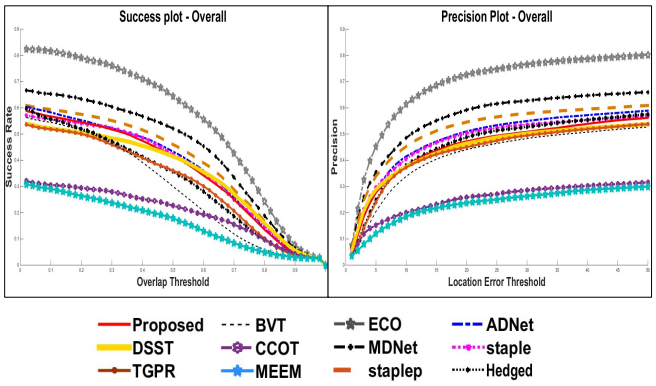


Figure 4.11: Overall success plots (left) and precision plot (right) calculated for UAV123 Dataset (Mueller *et al.*, 2016). (Note: Zoom in for better view)

all three layers are used. Also, since the information captured by the features from each layer may vary from frame to frame, the contribution (or weight) of each layer in the final prediction can not be generalized.

To address this issue, we use an LSTM network. An LSTM is trained that takes an input of dimension $T \times \kappa$, where columns denote the feature vector extracted from an image patch over the past κ consecutive frames. It gives two values as the output: the percentage of foreground present in the κ^{th} column; and the percentage of background in the same.

At any frame t, three image patches centered at location $\boldsymbol{l}^1$, $\boldsymbol{l}^2$ and $\boldsymbol{l}^3$ are cropped with the scale same as the scale in frame $t-1$. Corresponding to each of these patches, a feature vector of size $T \times 1$ is extracted. The features here can be hand crafted features or CNN features and T will depend on the type of feature used. In this work, T is a 1764 dimensional HOG feature vector. Let us denote these feature vectors as $\mathbf{z}^1$, $\mathbf{z}^2$ and $\mathbf{z}^3$ corresponding to position $\boldsymbol{l}^1$, $\boldsymbol{l}^2$ and $\boldsymbol{l}^3$ respectively. Now, let there be a matrix $\mathbf{A}$ of size $T \times (\kappa - 1)$. The columns of $\mathbf{A}$ contain the features of the past $\kappa - 1$ bounding boxes predicted by the tracker at frames $\{t-1, t-2, ..., t-(\kappa-1)\}$. To compute the weight of location $\boldsymbol{l}^u$, $\mathbf{z}^u$ is column-wise concatenated to the right end of $\mathbf{A}$ to form a $T \times \kappa$ matrix. This matrix is fed to the trained LSTM, which in turn gives us the percentage of foreground and background present in $\mathbf{z}^u$. This can be visualized in Figure 4.3. A higher percentage of foreground present in $\mathbf{z}^u$ implies that the majority of the corresponding bounding box is covering the target and hence it is likely to be an accurate box. Thus, its contribution in calculating the final position should be high. On the other hand, a lower percentage of foreground in a box implies that it is not a good prediction and hence its

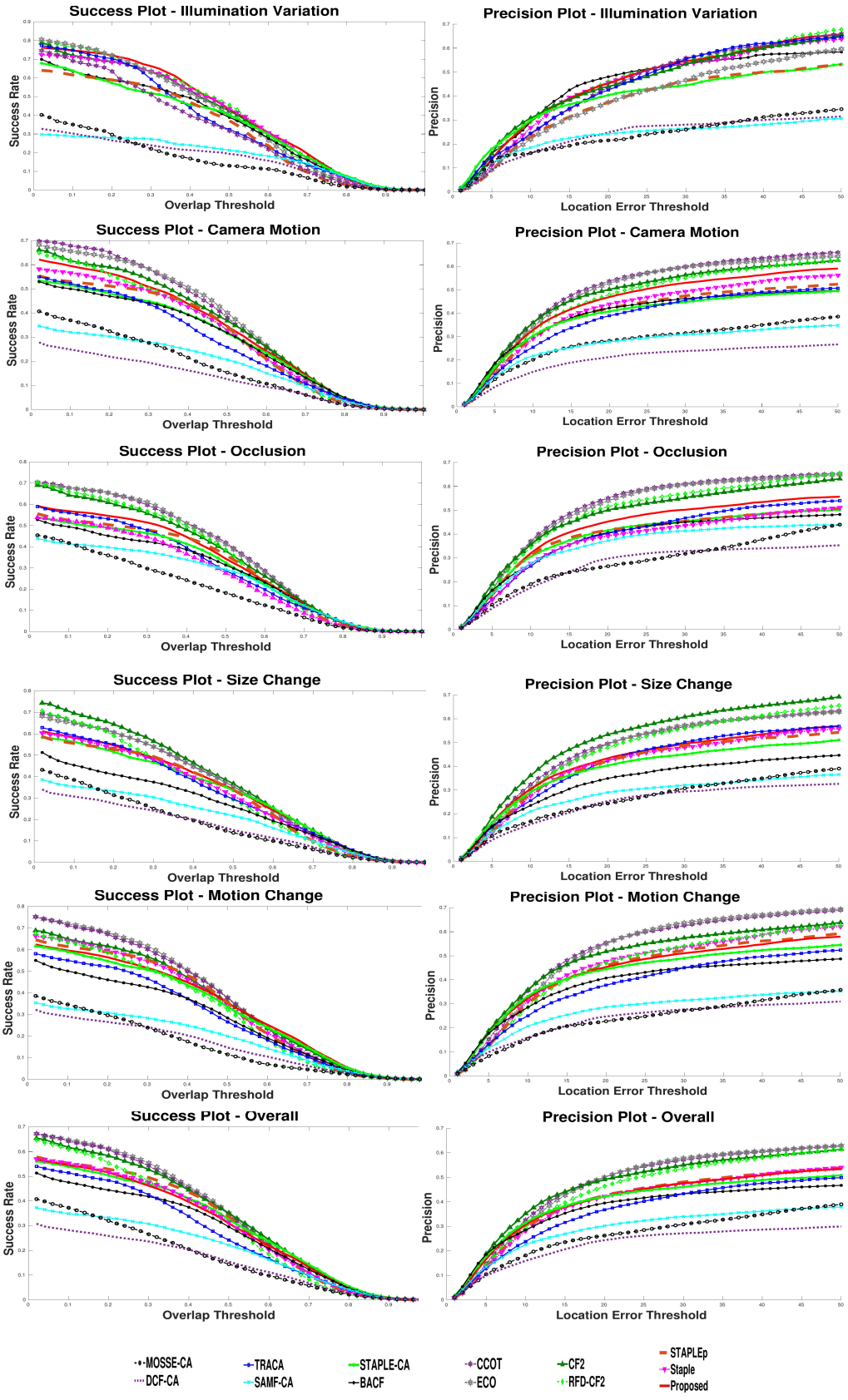


Figure 4.12: Success and Precision plots obtained over the VOT-2017 dataset Kristan *et al.* (2017*b*) for Illumination Variation, Size Change, Camera Motion, Motion Change, Occlusion Change prone videos and overall performance.

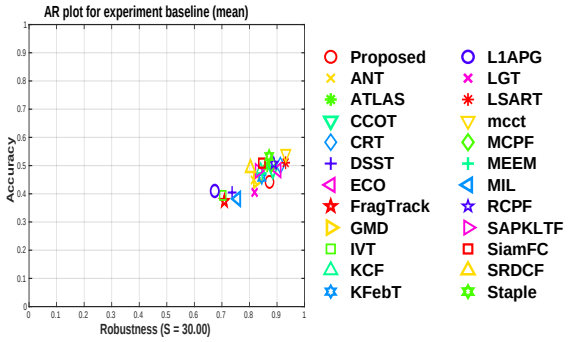
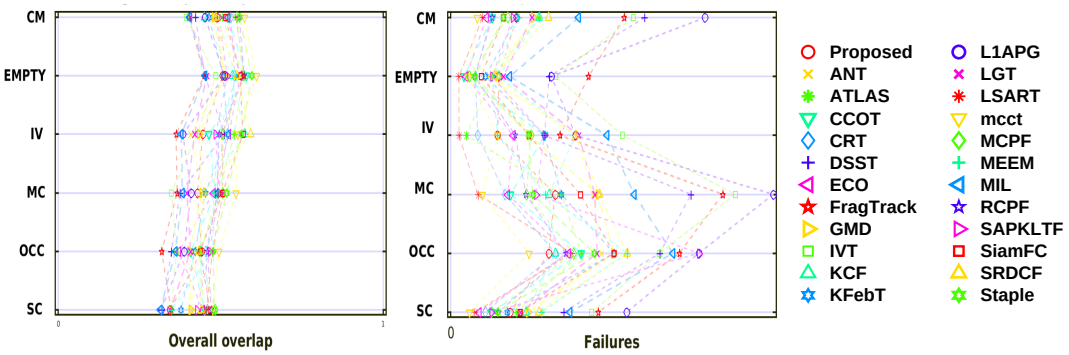


Figure 4.13: Mean Accuracy-Robustness (AR) Plot for experiments on baseline using the VOT toolkit



((a)) ((b))

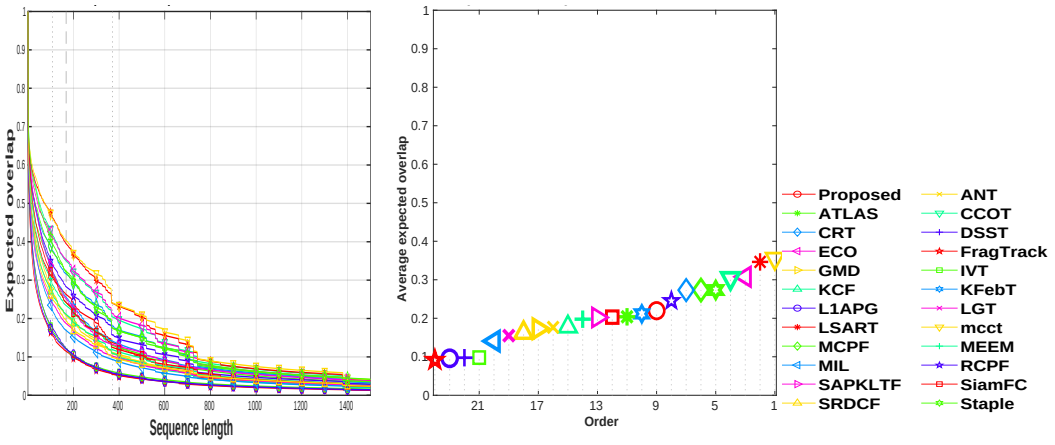
Figure 4.14: Baseline Experiments: (a) Tracker ordering for overlap and (b) Tracker ordering for failures

contribution should be low.

The benefit of including the features from previous bounding boxes in the input is that it enables the LSTM to compare the current bounding box with past observations. This is because the appearance of a target should not change much in κ frames (if κ is small). If the features of the current box are similar to features of the past $\kappa - 1$ boxes, the LSTM interprets the current bounding box as foreground and assigns a higher percentage to the foreground. On the other hand, if features of the current box do not match the features of the past $\kappa - 1$ bounding boxes, it is likely that the box location is erroneous and it is capturing the background. In such a case, the LSTM interprets the current bounding box as background and assigns a higher percentage to the background.

Let the foreground and background percentage present in the bounding boxes be denoted as fg^u and bg^u respectively, where u is the layer index and $u \in \{1, 2, 3\}$. The weight q^u of the location predicted corresponding to layer u can then be calculated as:

$$q^u = \frac{fg^u}{max(fg^1, fg^2, fg^3)} \tag{4.6}$$



((a)) ((b))

Figure 4.15: Baseline Experiments: (a) Expected overlap curves and (b) Expected overlap score

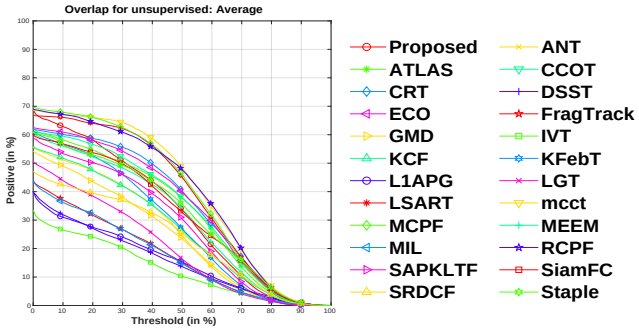


Figure 4.16: Average Overlap Plot for experiments on unsupervised using VOT toolkit

Once the weight of each location is calculated, the final location $\boldsymbol{l} = (p_x, p_y)$ can be given as follows.

$$\boldsymbol{l} = q^1.\boldsymbol{l}^1 + q^2.\boldsymbol{l}^2 + q^3.\boldsymbol{l}^3 \tag{4.7}$$

The procedure in Section 4.2.2 and 4.2.3 can be easily visualized using Figure 4.2 and 4.3. Information on training the LSTM is given in Section 4.2.4.

4.2.4 Target Scale Estimation

A CF learned over FHOG features of the target bounding box is used to estimate the target scale. FHOG features are a variant of HOG features (Dalal and Triggs, 2005) used by Felzenszwalb *et al.* (2010). As per the literature, it has shown superior performance compared to the original HOG features. If there are j orientations, FHOG features will be $3j + 5$ dimensional. There will be $2j$ contrast sensitive orientation channels, j contrast insensitive orientation channels, 4 texture channels and 1 all zero channel (used

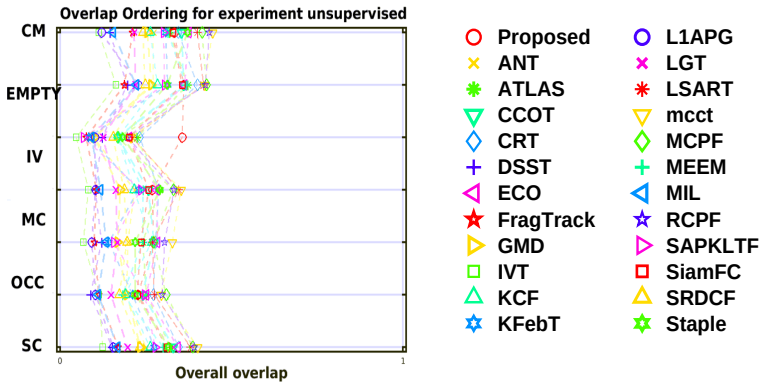


Figure 4.17: Overlap ordering for experiments on unsupervised for Camera Motion (CM), Empty, Illumination Variation (IV), Motion Change (MC), Occlusion (OCC), Size Change (SC) and Average

as a truncation feature). We use a standard value of $j = 9$ which results in a $W \times H \times 32$ dimensional feature vector, where $W \times H$ is the dimension of target image patch.

In this work, we learn a 2-dimensional CF filter of size $33 \times (W \times H \times 32)$, where the columns represent the dimension of the vectorized FHOG features and the row represents the number of scaling factors. For each training sample, it has Gaussian label of size 33×1, where the peak of the Gaussian corresponds to the target's actual or best scaling factor. The learning procedure remains the same as Equation 4.4.

At frame t, once we calculate the target's final position $\boldsymbol{l}$, the target is cropped by a bounding box centered at (p_x, p_y) having scale similar to the scale in frame $t-1$. The cropped patch is scaled by 33 different scaling factors and corresponding to each scaled patch, FHOG features are extracted and vectorized. These features are concatenated to form a $33 \times (W \times H \times 32)$ matrix. For the feature matrix, the response of the scale correlation filter is computed using Equation 4.4, where $\mathbf{z}$ and $\mathbf{h}$ are Fourier transforms of the FHOG feature matrix and scale CF respectively, and $k \in \{1, ...,(W \times H \times 32)\}$. This results in a 33×1 filter responses, $\mathbf{r}_{scale}$. The scale that corresponds to $max(\mathbf{r}_{scale})$ is considered the target's current scaling factor, $scale$.

After the scaling factor estimation, the target bounding box can be calculated as $\mathbf{b} = \{p_x - (p_3 * scale)/2, p_y - (p_4 * scale)/2, p_3 * scale, p_4 * scale\}$, where p_3 and p_4 are the width and height in frame $t-1$ respectively. For the remainder of this chapter, let us denote $p_x - (p_3 * scale)/2$, $p_y - (p_4 * scale)/2$, $p_3 * scale$, $p_4 * scale$ as b_1, b_2, b_3 and b_4, respectively. The procedure in this Section is illustrated by Figure 4.4.

Figure 4.18: Intermediate Frames Showing Tracker's Performance During Various Challenges: Row 1 - Size Change, Row 2 - Object Inclination, Row 3 - Occlusion, Row 4 - Shaking Camera, Row 5 - Illumination Variation. Proposed tracker is able to the track target during challenges where other trackers fail.

4.2.5 Target rotation estimation

Similar to the scale CF, we train a rotation CF. It is a 2-dimensional filter of size $9 \times (W \times H \times 32)$, where the columns represent the dimension of vectorized FHOG features and the rows represent the number of rotation factors. For each training sample, it has a Gaussian label of size 9×1, where the peak of the Gaussian corresponds to the factor by which the target rotates.

The rotation is estimated by cropping a patch using bounding box $\mathbf{b}$ and rotating it by 9 different rotation factors. Corresponding to each rotated patch, FHOG features are extracted and vectorized. Again, a matrix of $9 \times (W \times H \times 32)$ is formed corresponding to which, the response of the rotation CF is computed using Equation 4.4. As a result, a filter response, $\mathbf{r}_{rot}$, of size 9×1 is obtained, which is used to infer the target rotation. The rotation value that corresponds to $max(\mathbf{r}_{rot})$ is considered to be the target's current rotation factor, rot. The final target bounding box can then be calculated by rotating the box $\mathbf{b} = \{b_1, b_2, b_3, b_4\}$ by *rot* degrees.

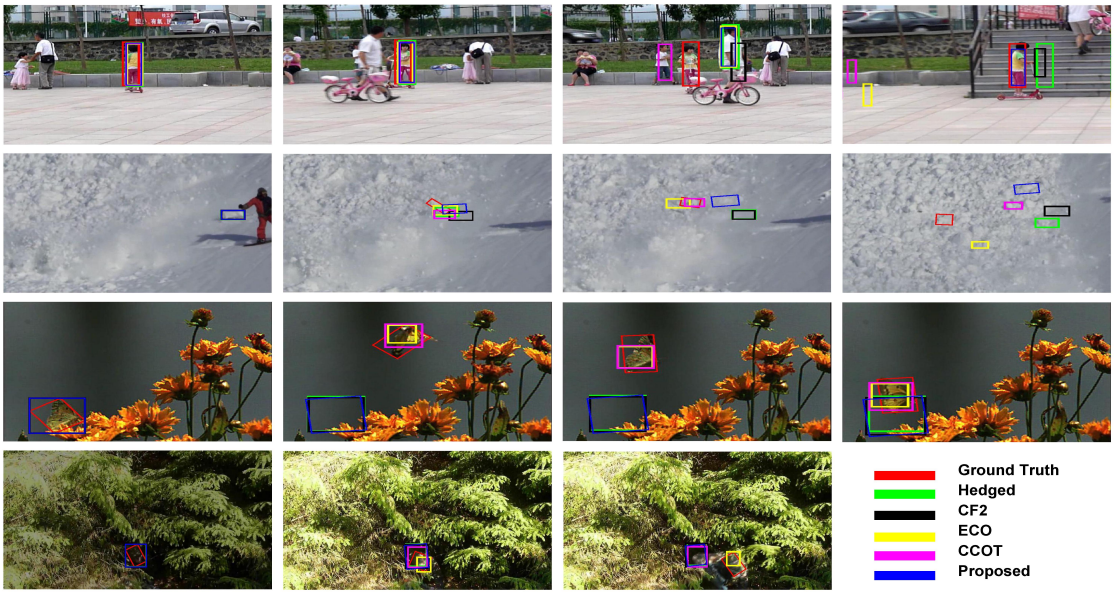


Figure 4.19: Intermediate Frames Showing Tracker's Failure During Various Challenges: Row 1 - Occlusion by object with similar appearance, Row 2 - Background Clutter, Row 3 - Tracker stuck on bright yellow flower due to poor HOG features of the dull colored object, Row 4 - Background Clutter

4.2.6 Appearance Model Pool

In most of the CF based trackers, once the final bounding box is predicted in a frame (say t), it is used to update the CF so that filter can learn the updated target appearance and make a suitable prediction in the next frame, $(t+1)$. This method is likely to fail if the predicted bounding box is faulty/inaccurate. This is because an inaccurate bounding box will lead to an erroneous update to the CF. In longer sequences, the error will keep accumulating, leading to tracking failure.

To avoid this, an appearance model pool is introduced. In this, at frame t, HOG features for the predicted bounding box $\mathbf{b} = \{b_1, b_2, b_3, b_4\}$, concatenated to the right end of $\mathbf{A}$ are fed to the LSTM trained in Section 4.2.3. Doing this, we obtain the percentage of foreground and background present in the predicted box $\mathbf{b}$. If the percentage of foreground is more than a threshold ζ, it is likely to be a good prediction and hence a good sample of the target's latest appearance. Such appearance samples are stored in the appearance model pool for later use. Since, the target appearance changes with time, storing older appearance samples in the pool may not be favourable. Therefore, to avoid computations on the old appearances, only the 7 most recent samples are stored.

In addition, at frame t, if the percentage of foreground in a bounding box is less than a threshold α, it means that during the most recent update the CF (see Section

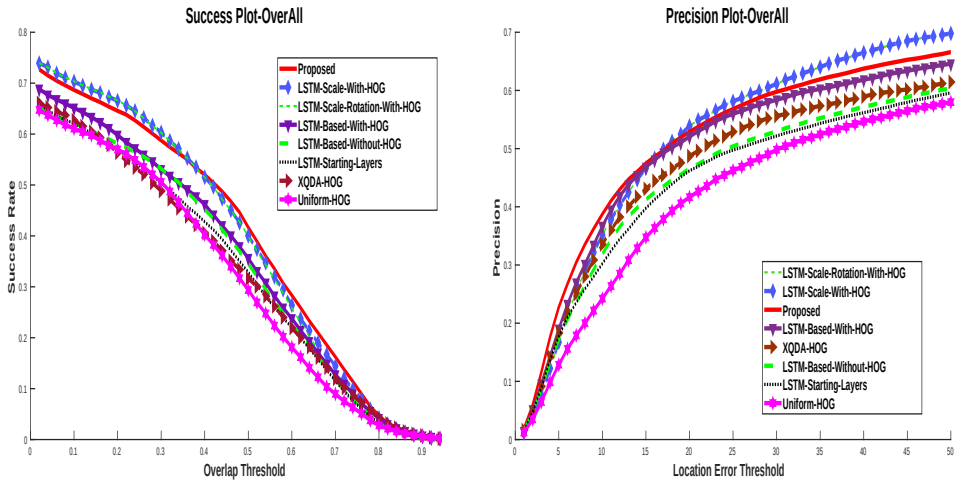


Figure 4.20: Success (left) and Precision (right) plots obtained for Ablation study over VOT-2016 Dataset. (Note: Zoom in for better view).

4.2.2) have been updated with features extracted from a bounding box that has more background. To compensate, in the current step *t*, filters must be trained with rich features that are extracted from the target's actual appearance (foreground). In such a situation, the appearance model pool is used to select a good image of the target in order to update the filters.

In the pool, the appearance that has the minimum Mahalanobis distance (De Maesschalck *et al.*, 2000) to the target predicted at time $(t-1)$ is used to train the filter at the current step *t*. Let, $box^{(t-1)}$ be the vectorized bounding box predicted at frame $(t-1)$, app_i be the vectorized i^{th} appearance sample in the model pool, where $i \in \{1,...,7\}$, and Π be the covariance matrix. Then, the Mahalanobis distance, $dist_i$, between the two can be computed as:

$$dist_i = \sqrt{\left(box^{(t-1)} - app_i\right)^\top \Pi^{-1} \left(box^{(t-1)} - app_i\right)} \tag{4.8}$$

The appearance sample that corresponds to $min(dist_i), i \in \{1,...,7\}$, is the appearance selected to update the correlation filters at frame t. This prevents the filter from drifting, resulting in prolonged and accurate tracking. On the other hand, if the percentage of foreground in a bounding box is more than the threshold α, the predicted box **b** itself is considered for updating the filters, without using the model pool.

4.2.7 Updating the Correlation Filters

As discussed above, before updating the filters from Section 4.2.2, an appropriate bounding box is selected that contains the recent best appearance of the target. Let, for a particular layer *u*, the features extracted from the selected box be $\mathbf{x}^u$. To update the filter, the output error over all tracked results so far should be minimized (Naresh Boddeti *et al.*, 2013). This requires solving a $K \times K$ linear system of equations per location at (*a,b*). But since the number of channels in the CNN features is usually large, the above approach will be computationally expensive. To avoid this, we use the following approximation (Ma *et al.*, 2015*a*) to update the filter $\hat{\mathbf{h}}_k$ in Equation 4.3 (ignoring the layer index):

$$\hat{\mathbf{m}}_k^{(t)} = (1-\eta)\hat{\mathbf{m}}_k^{(t-1)} + \eta\hat{\mathbf{y}} \bigodot conj(\hat{\mathbf{x}}_k^{(t)}); \tag{4.9}$$

$$\hat{\mathbf{n}}_k^{(t)} = (1-\eta)\hat{\mathbf{n}}_k^{(t-1)} + \eta \sum_{k=1}^{K} \hat{\mathbf{x}}_k^{(t)} \bigodot conj(\hat{\mathbf{x}}_k^{(t)}); \tag{4.10}$$

$$\hat{\mathbf{h}}_k^{(t)} = \frac{\hat{\mathbf{m}}_k^{(t)}}{\hat{\mathbf{n}}_k^{(t)} + \lambda} \tag{4.11}$$

where *k*={1,2,...,K}, *t* is the frame index and η is the learning rate. A similar method can be used to update the scale and rotation CFs, by substituting appropriate variables in Equation $4.9, 4.10$ and 4.11.

4.3 Evaluation of the LSTM Component

The Long Short-Term Memory (LSTM) network in Section 4.2.3 can be trained using either Histogram of Oriented Gradients (HOG) features or Convolutional Neural Network (CNN) features extracted from VGG-19-Net (Simonyan and Zisserman, 2014). In order to compare the VGG features with HOG features, we perform two experiments. In the first one, we train the LSTM using VGG features and in the second we train the LSTM using HOG features. The details of the experiments are given below:

4.3.1 Training Data Generation

A foreground training sample is generated by column-wise concatenation of feature vectors extracted from κ foreground patches to form a $T \times \kappa$ matrix. Similarly, a background training sample is generated by column-wise concatenation of feature vectors

extracted from κ background patches to form a $T \times \kappa$ matrix. By doing so, we get multiple foreground and background training samples (of size $T \times \kappa$ each), which are used to train the LSTM. An example of foreground and background patch generation is shown in Figure 4.5.

At test time, the trained LSTM is used to estimate the weights of the predicted bounding boxes corresponding to layer *conv5-4*, *conv4-4* and *conv3-4*. In order to do so, $T \times 1$ dimensional features are extracted corresponding to each predicted bounding box and it is column-wise concatenated with the matrix $\mathbf{A}$, that is, features extracted from the past $\kappa - 1$ foreground patches predicted by the tracker. The resultant $\mathbf{A} \in \mathcal{R}^{T \times \kappa}$ is fed to the trained LSTM, which classifies the κ^{th} feature vector as foreground or background. The procedure is shown in Figure 4.3.

4.3.2 CNN Features

We investigate the effect of size of the history used, and the features extracted from various fully connected layers of VGG during LSTM training. In this case, the foreground and background patches are re-sampled to 224×224 and κ = 4096. The details of each experiment are given below.

1. **LSTM1-FC1**: In this version, CNN features are extracted from the 1^{st} fully connected layer and κ = 1, hence, the size of the input to the LSTM is 4096×1.
2. **LSTM1-FC2**: This version is similar to version **LSTM1-FC1**. The only difference is that CNN features are extracted from the 2^{nd} fully connected layer, instead of the 1^{st}.
3. **LSTM4-FC1**: Here, the CNN features are extracted from the 1^{st} fully connected layer and κ = 4. Therefore, the input, $\mathbf{A}$, to the LSTM is of dimension 4096×4.
4. **LSTM4-FC2**: This version is similar to version **LSTM4-FC1**. The only difference is that CNN features are extracted from the 2^{nd} fully connected layer, instead of the 1^{st}.
5. **LSTM8-FC1**: Here, the CNN features are extracted from the 1^{st} fully connected layer and κ = 8. Therefore, the input, $\mathbf{A}$, to the LSTM is of dimension 4096×8.
6. **LSTM8-FC2**: This version is similar to version **LSTM8-FC1**. The only difference is that CNN features are extracted from the 2^{nd} fully connected layer, instead of the 1^{st}.
7. **LSTM1-PCA-FC1**: In this version, CNN features are extracted from the 1^{st} fully connected layer and κ = 1. The dimension of training and testing samples are reduced to 112×1 using Principal Component Analysis (PCA), reconstructing 99% information of the original samples. Therefore, the input to the LSTM is of dimension 112×1.

8. **LSTM1-PCA-FC2**: In this version, CNN features are extracted from the 2^{nd} fully connected layer and κ = 1. The dimension of training and testing samples are reduced to 45×1 using PCA, reconstructing 99% information of the original samples. Therefore, the input to the LSTM is of dimension 45×1.

9. **LSTM4-PCA-FC1**: Here, the CNN features are extracted from the 1^{st} fully connected layer and κ = 4. The dimension of training and testing samples are reduced to 112×4 using PCA. Therefore, the input to the LSTM is of dimension 112×4.

10. **LSTM4-PCA-FC2**: Here, the CNN features are extracted from the 2^{nd} fully connected layer and κ = 4. The dimension of training and testing samples are reduced to 45×4 using PCA. Therefore, the input to the LSTM is of dimension 45×4.

11. **LSTM8-PCA-FC1**: In this version, CNN features are extracted from the 1^{st} fully connected layer and κ = 8. The dimension of training and testing samples are reduced to 112×8 using PCA. Therefore, the input to the LSTM is of dimension 112×8.

12. **LSTM8-PCA-FC2**: In this version, CNN features are extracted from the 2^{nd} fully connected layer and κ = 8. The dimension of training and testing samples are reduced to 45×8 using PCA. Therefore, the input to the LSTM is of dimension 45×8.

4.3.3 HOG Features

In this case, the foreground and background patches are re-sampled to 32×32 and κ = 1764. Corresponding to each re-sampled patch, a 1764×1 dimensional HOG feature vector is computed. The experiments are done for different values of κ. Therefore, the dimension of training samples, testing samples and the LSTM input is equal to $1764 \times \kappa$. The details of each experiment are given below.

1. **LSTM1-HOG**: In this version, κ = 1, hence, size of the input to the LSTM is 1764×1.

2. **LSTM4-HOG**: Here, κ = 4. Therefore, the input, **A**, to the LSTM is of dimension 1764×4.

3. **LSTM8-HOG (Proposed)**: Here, κ = 8. Therefore, the input, **A**, to the LSTM is of dimension 1764×8.

4. **LSTM12-HOG**: Here, κ = 12. Therefore, the input, **A**, to the LSTM is of dimension 1764×12.

4.3.4 Training the LSTM

In each of the above cases, output of the LSTM is a two class classification (foreground or background). The number of units in the LSTM is 8 and the training is done using

approximately 300,000 samples for 10 epochs with a batch size of 32. The optimizer used is Adam, with learning rate 0.001, and a softmax loss function. To obtain the probability of each patch belonging to the foreground, we remove the softmax layer and used the output of layer just before the softmax layer. Each of the above LSTMs is used to evaluate the tracker performance on VOT2016 and VOT2017 datasets. Also, since the VGG network is trained to classify objects of size approximately 224×224, we separately evaluate the tracker on sequences that have objects with a minimum size 200×200 pixels and maximum size 250×250 pixels through out the video (we denote them as Selected Videos). This is done in order to analyze the effect of object scale on HOG features and the CNN features extracted using the VGG network.

Table 4.1 shows the Overlap Success Rate (OP) and Distance Precision Rate (DP) for each version of the LSTM over both the datasets. It is observed that when the performance is computed over the entire dataset (VOT2016 or VOT2017), the proposed tracker out-performs the trackers that use LSTMs trained using CNN features. However, trackers using a CNN feature based LSTM show better performance compared to proposed, when the performance is calculated over the selected videos. This is because VOT2016 and VOT2017 consist of several videos that have objects with size much smaller than 224×224. For such videos, due to excessive re-sampling of small objects to 224×224, the features extracted using VGG may not be suitable to encode the target appearance. In such cases, HOG features work better. Based on the observations from Table 4.1, we use HOG features with $\kappa = 8$ (version **LSTM8-HOG**) to train the LSTM in the proposed tracker.

4.3.5 Weight Estimation using the Trained LSTM

Figure 4.6 shows the weight estimation using the LSTM, version **LSTM8-HOG**, for the bounding boxes corresponding to layers *conv5-4*, *conv4-4* and *conv3-4*. Column 1 shows the bounding box predicted using the features extracted from layer *conv5-4* of VGG. Similarly, Column 2 and Column 3 show predictions corresponding to features from layers *conv4-4* and *conv3-4*, respectively. Column 4 shows the ground truth bounding box.

In Figure 4.6, Row 1 shows frame number 5 and Row 2 shows frame number 26 from the Sequence *'bag'*. Row 3 shows frame number 3 and Row 4 shows frame numner 27 from the Sequence *'bmx'*. Both the sequences are common to the VOT2016 and

VOT2017 datasets. It is observed that the LSTM successfully assigns the maximum weight to the predicted bounding box that contains maximum foreground or matches best with the ground truth. Similarly, the least weight is assigned to the box that matches least with the ground truth.

Figure 4.7 shows frame number 14 of the Sequence *'bag'* and frame number 33 of the Sequence *'bmx'*, where the LSTM fails to predict the weight of each patch accurately. We can see that in the first case, the LSTM assigns maximum weight to the bounding box that matches least with the ground truth. In the second case, it assigns varying weights despite the patches being very visually similar, where ideally we would expect the patches to be assigned almost identical weights. The probable reason for failure here is the severe deformation and rotation of the object over the past 7 frames in the sequence, that results in confusing the LSTM. To avoid tracker drift caused due to erroneous predictions of the LSTM, we use the appearance model pool, explained in Section 4.2.6.

4.4 Experiments

The proposed algorithm is tested over challenging videos from the OTB100 Dataset (Wu *et al.*, 2015), Visual Object Tracking (VOT) - 2016 Dataset (Kristan *et al.*, 2016*c*), VOT-2017 Dataset (Kristan *et al.*, 2017*a*), Tracking Dataset (Vojir *et al.*, 2014) and UAV123 Dataset (Mueller *et al.*, 2016). Out of over 70 trackers submitted in the VOT-2016 challenge, the comparison is done against the top ranked trackers (TRACA (Choi *et al.*, 2018), CCOT (Danelljan *et al.*, 2016*d*), MLDF (Kristan *et al.*, 2016*a*), SiamRN (Kristan *et al.*, 2016*a*), DNT (Chi *et al.*, 2017), RFD-CF2 (Kristan *et al.*, 2016*a*), staple (Bertinetto *et al.*, 2016*a*), staplep (Kristan *et al.*, 2016*a*)) and selected other recent trackers like MDNet (Nam and Han, 2016), ADNet (Yun *et al.*, 2017), ECO (Danelljan *et al.*, 2016*a*), Hedged (Qi *et al.*, 2016), CF2 (Ma *et al.*, 2015*a*), DSST (Danelljan *et al.*, 2017*b*), MEEM (Zhang *et al.*, 2014*a*), BVT (Yuan *et al.*, 2014), TGPR (Gao *et al.*, 2014), BACF (Kiani Galoogahi *et al.*, 2017), DCF-CA (Mueller *et al.*, 2017), MOSSE-CA (Mueller *et al.*, 2017), SAMF-CA (Mueller *et al.*, 2017) and STAPLE-CA (Mueller *et al.*, 2017). Among these trackers, CCOT (Danelljan *et al.*, 2016*d*) and ECO (Danelljan *et al.*, 2016*a*) are also the top performers of VOT-2017 and TRACA (Choi *et al.*, 2018) is from VOT-2018. For each dataset, the number of compared trackers vary

according to the availablity of results and implementations.

Performance of each tracker is evaluated using success plot, precision plot, average Center Location Error (CLE) and average Intersection Over Union (IOU). We also show a quantitative comparison of the Distance Precision (DP) rate at 20 pixels, and Overlap Success (OS) rate at an overlap threshold of 0.5 (Qi *et al.*, 2016).

4.4.1 Implementation Details

In our experiments, we use VGG-Net-19 (Simonyan and Zisserman, 2014), as it has shown superior performance to ResNet (He *et al.*, 2016) and GoogLeNet (Szegedy *et al.*, 2015) for object tracking tasks (Li *et al.*, 2017*b*). For feature extraction, fully connected layers of VGG-Net-19 are removed and outputs of the conv3-4, conv4-4 and conv5-4 convolutional layers are used as features. At frame t, if the size of the input search window is $W' \times H'$ (which is twice the size in frame $t-1$), the features from each convolutional layer are resized to a fixed spatial size of $W'/4 \times H'/4$. Since the tracker needs to retain the spatial resolution of each convolutional layer, outputs of the pooling layers are not used.

In Equation 4.1, the kernel width used for generating the Gaussian labels of position correlation filters is 0.1. For all position, scale and rotation correlation filters, the value of λ in Equation 4.2 is fixed to 10^{-4} during training. In Equation (4.10) (4.11) and (4.12), learning rate η is set to 0.01. In Section 4.2.6, the value of ζ is 0 for first 12 frames of each sequence. After 12 frames, it is equal to the average of weight q^1 obtained during the 8^{th} to 12^{th} frames. The value of α is fixed to 0.7. Scaling factors used in Section 4.2.4 are between 0.3 to 1.7, at an uniform interval of 0.043, and the rotation factors in Section 4.2.5 are [-8°, -6°, -4°, -2°, 0°, 2°, 4°, 6°, 8°]. Additionally, in order to remove the boundary discontinuities, the channels of the extracted features of each convolutional layer are weighted by a cosine window (Bolme *et al.*, 2010).

The proposed pipeline uses an LSTM with VGG-19 network. VGG has 140 million parameters and 15 billion floating-point operations per image during the forward pass. The LSTM has 4364 parameters and 2.87M floating point operations. All the modules of the proposed trackers are implemented in MATLAB 2017b. Only the LSTM is implemented in Keras. The tracking speed is nearly 1 frame per second.

Table 4.2: Comparisons with recent trackers over the OTB100 Dataset (Wu *et al.*, 2015) based on distance precision rate at a threshold of 20 pixels and overlap success rate at an overlap threshold of 0.5. Here, OV = Out-of-View, OPR = Our-of-Plane Rotation, OCC = Occlusion, MB = Motion Blur, IPR = In-Plane-Rotation, IV = Illumination Variation, FM = Fast Motion and OverAll is the evaluation over entire dataset. The first, second, third, fourth, fifth and sixth best trackers are highlighted in red, green, blue, cyan, magenta and purple (if shown) color, respectively. The same notation will be followed in Table 4.3 - 4.13.

	Overlap Success Rate (%)								Distance Precision Rate (%)							
	OV	OPR	OCC	MB	IPR	IV	FM	Overall	OV	OPR	OCC	MB	IPR	IV	FM	Overall
Hedged (Qi *et al.*, 2016)	71.2	85.0	82.5	83.9	88.6	84.9	87.5	89.0	46.3	57.6	58.3	56.0	62.2	65.1	53.5	63.5
CF2 (Ma *et al.*, 2015*a*)	71.1	84.8	81.5	85.5	89.1	84.8	87.9	87.7	45.0	57.4	57.8	59.1	63.0	64.7	58.1	62.8
Baseline-conv5 (Valmadre *et al.*, 2017*a*)	60.5	81.5	80.0	70.4	81.9	77.7	80.5	83.9	37.7	57.6	56.2	39.6	56.3	54.6	48.0	62.3
Baseline+CF-conv3 (Valmadre *et al.*, 2017*a*)	68.6	80.8	77.8	75.0	82.1	77.1	79.1	83.0	41.3	53.5	52.3	46.6	58.1	53.6	43.2	61.6
CFNet-conv5 (Valmadre *et al.*, 2017*a*)	71.6	81.1	77.7	79.6	85.0	79.7	81.7	82.2	44.1	55.7	51.5	53.0	58.9	59.6	55.5	59.7
CFNet-conv2 (Valmadre *et al.*, 2017*a*)	61.9	78.1	74.5	75.3	80.8	76.5	79.1	79.8	39.4	51.1	47.2	48.1	56.2	52.2	47.8	55.1
CFNet-conv1 (Valmadre *et al.*, 2017*a*)	55.4	74.8	70.0	72.4	78.4	73.7	71.4	77.0	22.0	46.2	44.4	43.2	51.2	46.0	33.0	46.8
SiamFC-3s (Valmadre *et al.*, 2017*a*)	74.3	80.9	78.5	76.8	79.8	77.5	81.1	82.1	50.1	55.1	53.1	49.5	54.1	53.7	53.1	58.5
SRDCF (Danelljan *et al.*, 2015*b*)	62.9	79.1	78.1	78.3	79.2	80.7	83.5	83.2	38.3	49.6	48.6	44.4	52.3	54.2	46.7	55.2
SAMF (Li and Zhu, 2014)	68.9	79.0	77.7	71.9	76.2	76.4	73.8	79.9	44.4	49.1	48.5	44.0	51.9	42.2	40.1	49.6
staple (Bertinetto *et al.*, 2016*a*)	70.0	76.4	75.3	72.2	78.7	80.3	74.6	81.1	44.8	48.6	48.4	44.2	52.2	46.9	43.7	51.4
LCT (Ma *et al.*, 2015*b*)	65.1	77.9	70.7	72.1	81.5	77.7	74.6	79.8	43.7	48.9	46.2	44.1	53.0	45.9	44.5	49.5
TGPR (Gao *et al.*, 2014)	58.5	74.5	68.0	68.6	74.9	68.9	74.3	74.0	30.6	40.0	32.8	32.2	44.4	39.8	42.9	40.3
DSST (Danelljan *et al.*, 2017*b*) T	51.9	68.7	63.1	61.9	73.6	76.9	62.6	72.3	28.0	26.5	24.3	21.1	43.4	45.8	20.0	21.3
KCF (Henriques *et al.*, 2014)	54.1	71.7	67.5	64.2	73.3	75.2	69.4	74.0	33.4	40.9	40.7	35.4	31.1	34.1	35.7	43.1
BVT (Yuan *et al.*, 2014)	59.5	66.9	68.1	60.6	63.9	57.8	62.3	67.5	33.7	34.7	39.2	25.8	17.1	18.1	28.5	37.5
SAMF-CA (Mueller *et al.*, 2017)	49.9	66.2	65.3	45.9	66.1	64.7	50.9	66.2	33.7	47.3	46..8	36.3	45.5	51.6	36.0	48.9
DCF-CA (Mueller *et al.*, 2017)	41.5	62.6	57.4	48.4	70.3	65.2	52.1	63.6	20.8	21.0	21.3	19.9	37.4	35.4	20.0	21.2
MOSSE-CA (Mueller *et al.*, 2017)	34.6	55.4	51.6	39.3	60.0	52.4	46.4	57.4	11.3	15.3	18.3	21.3	19.2	15.4	20.1	22.5
Proposed	71.8	82.7	79.8	80.3	82.8	78.4	81.1	82.3	46.3	58.3	54.1	53.9	58.2	58.5	50.0	60.7

4.4.2 Quantitative Evaluation

Evaluation on OTB100 Dataset

This section shows the comparison of the proposed tracker with existing trackers Hedged (Qi *et al.*, 2016), CF2 (Ma *et al.*, 2015*a*), Baseline-conv5 (Valmadre *et al.*, 2017*a*), Baseline+CF-conv3 (Valmadre *et al.*, 2017*a*), CFNet-conv5 (Valmadre *et al.*, 2017*a*), CFNet-conv2 (Valmadre *et al.*, 2017*a*), CFNet-conv1 (Valmadre *et al.*, 2017*a*), SiamFC-3s (Valmadre *et al.*, 2017*a*), SRDCF (Danelljan *et al.*, 2015*b*), SAMF (Li and Zhu, 2014), staple (Bertinetto *et al.*, 2016*a*), LCT (Ma *et al.*, 2015*b*), TGPR (Gao *et al.*, 2014), DSST (Danelljan *et al.*, 2017*b*), KCF (Henriques *et al.*, 2014), BVT (Yuan *et al.*, 2014), SAMF-CA (Mueller *et al.*, 2017), DCF-CA (Mueller *et al.*, 2017) and MOSSE-CA (Mueller *et al.*, 2017), for challenges including Object Out-of-View (OV), Out-of-Plane Rotation (OPR), Occlusion (OCC), Motion Blur (MB), In-Plane-Rotation (IPR), Illumination Variation (IV), Fast Motion (FM) and Overall performance.

Table 4.2 shows the Overlap Success Rate (OP) and Distance Precision Rate (DP) of the proposed tracker along with existing trackers, on the OTB100 dataset. It is observed that the proposed tracker is second best in OV, third best in OPR and MB, fourth in OCC and IPR, and fifth in FM. We also observe that the proposed tracker outperforms the CFNet-conv1, CFNet-conv2 and Baseline+CF-conv3 (Valmadre *et al.*, 2017*b*) in all

Table 4.3: Comparisons with recent trackers over the OTB100 Dataset (Wu *et al.*, 2015) based on average Intersection Over Union (IOU) ratio and Center Location Error (CLE). Following the same notation as Table 4.2.

	Average IOU (%)								Average CLE (Pixels)							
	OV	OPR	OCC	MB	IPR	IV	FM	Overall	OV	OPR	OCC	MB	IPR	IV	FM	Overall
Hedged (Qi *et al.*, 2016)	59.06	**76.50**	**77.44**	**76.60**	**82.50**	**80.34**	**79.95**	**81.65**	**59.59**	**45.12**	**49.18**	**44.25**	**36.57**	**46.28**	**42.46**	**39.64**
CF2 (Ma *et al.*, 2015*a*)	59.04	**76.53**	**76.25**	**78.30**	**83.23**	**80.51**	**80.41**	**80.51**	**54.52**	**45.70**	52.65	**41.57**	**35.95**	**46.03**	**41.74**	**41.82**
Baseline-conv5 (Valmadre *et al.*, 2017*a*)	56.73	**74.57**	**74.99**	66.74	77.32	73.54	75.99	**78.03**	82.57	70.49	74.18	93.74	54.39	56.59	57.51	55.65
Baseline+CF-conv3 (Valmadre *et al.*, 2017*a*)	**62.59**	73.67	72.63	70.81	76.90	72.54	74.32	76.70	82.29	54.02	59.27	64.41	50.26	56.35	56.08	46.97
CFNet-conv5 (Valmadre *et al.*, 2017*a*)	**66.10**	74.02	72.84	**75.44**	**79.86**	75.61	**76.99**	76.49	67.21	54.53	58.37	48.81	44.38	50.16	47.67	46.84
CFNet-conv2 (Valmadre *et al.*, 2017*a*)	56.19	71.51	69.91	71.43	75.93	72.95	74.66	74.55	82.40	59.15	62.46	61.10	49.08	54.28	53.25	51.57
CFNet-conv1 (Valmadre *et al.*, 2017*a*)	49.97	68.06	65.71	69.85	**79.86**	70.31	67.31	71.79	85.74	62.95	68.69	59.97	49.81	56.16	62.07	55.68
SiamFC-3s (Valmadre *et al.*, 2017*a*)	**67.97**	73.62	72.99	72.17	74.63	70.91	76.15	75.82	**59.42**	**49.03**	**50.18**	54.48	45.74	49.22	**46.74**	43.95
SRDCF (Danelljan *et al.*, 2015*b*)	53.26	70.88	73.35	71.53	73.66	**76.72**	75.74	76.77	79.38	56.59	60.50	59.08	54.69	48.20	49.14	48.83
SAMF (Li and Zhu, 2014)	57.53	70.87	72.83	65.20	71.20	72.04	66.84	73.26	62.06	49.45	53.31	57.84	50.33	52.56	58.99	50.02
Staple (Bertinetto *et al.*, 2016*a*)	59.59	69.15	70.99	66.94	74.05	**76.27**	69.06	74.98	61.29	49.65	50.22	54.39	**43.54**	**44.19**	51.11	**42.95**
LCT (Ma *et al.*, 2015*b*)	54.61	69.71	66.23	65.74	76.07	72.62	68.43	73.24	77.63	55.96	67.72	63.49	44.44	55.17	60.39	52.60
TGPR (Gao *et al.*, 2014)	52.51	67.01	63.37	63.55	69.09	64.25	68.76	67.83	85.16	63.25	70.01	61.88	58.38	62.97	56.73	58.48
DSST (Danelljan *et al.*, 2017*b*)	44.63	61.57	59.88	57.07	69.31	72.78	57.77	66.71	87.09	61.77	72.81	72.20	50.15	50.21	75.92	57.81
KCF (Henriques *et al.*, 2014)	46.23	64.50	63.41	59.31	69.04	70.95	63.95	68.17	77.56	61.55	67.76	72.35	55.09	63.63	64.13	60.58
BVT (Yuan *et al.*, 2014)	53.47	61.62	64.66	54.71	60.24	49.96	57.06	62.51	96.87	88.08	87.25	115.34	92.18	119.49	100.27	91.42
SAMF-CA (Mueller *et al.*, 2017)	48.02	61.07	61.90	44.40	62.74	61.84	48.47	62.05	95.29	77.19	76.04	105.14	78.75	77.13	101.43	75.76
DCF-CA (Mueller *et al.*, 2017)	38.33	57.50	53.41	46.11	66.04	61.99	49.02	59.51	130.04	83.77	91.92	108.02	68.74	84.48	106.84	86.46
MOSSE-CA (Mueller *et al.*, 2017)	33.74	50.80	49.36	38.73	56.46	44.32	44.23	53.35	120.05	88.74	96.53	107.75	73.15	92.78	107.47	86.73
Proposed	59.41	73.75	73.93	72.74	76.38	71.98	73.88	74.95	60.56	49.20	**50.20**	**45.14**	44.38	47.81	48.33	47.12

Table 4.4: Comparisons with recent trackers over the Tracking Dataset (Vojir *et al.*, 2014) based on distance precision rate at a threshold of 20 pixels and overlap success rate at an overlap threshold of 0.5. Here, CM = Camera Motion, IV = Illumination Variation, MC = Motion Change, OCC = Occlusion, R = Low Resolution, SC = Size Change and OverAll is the evaluation over entire dataset. Following the same notation as Table 4.2.

	Overlap Success Rate (%)								Distance Precision Rate (%)						
	CM	IV	MC	OCC	R	SC	OverAll		CM	IV	MC	OCC	R	SC	OverAll
ADNet (Yun *et al.*, 2017)	**75.2**	76.6	**67.8**	58.3	**70.5**	**68.2**	**69.8**		**98.8**	**98.3**	**88.9**	76.5	**88.0**	**88.5**	**89.2**
CF2 (Ma *et al.*, 2015*a*)	**72.1**	78.2	63.0	51.7	67.4	63.1	65.2		**97.6**	96.8	85.8	70.6	86.0	**84.7**	**85.6**
RFD-CF2 (Kristan *et al.*, 2016*a*)	59.8	**77.4**	56.6	51.5	62.4	**57.4**	59.8		**97.5**	**97.0**	**85.9**	71.1	84.2	83.4	84.4
staplep (Kristan *et al.*, 2016*a*)	**67.4**	**81.6**	**67.7**	**59.1**	**71.5**	**67.8**	**69.6**		**90.3**	**98.2**	**87.6**	**78.9**	**85.9**	**85.9**	**86.8**
staple (Bertinetto *et al.*, 2016*a*)	**64.2**	**81.2**	**68.7**	**63.5**	**72.8**	**68.2**	**70.0**		89.1	**98.0**	**87.3**	**78.8**	**87.7**	**87.1**	**87.9**
TGPR (Gao *et al.*, 2014)	3.7	5.1	17.3	36.7	11.3	16.1	15.9		9.9	1.4	25.5	49.5	12.8	21.8	20.6
MDNet (Nam and Han, 2016)	6.3	28.8	30.9	**62.5**	22.6	33.8	31.8		8.6	36.0	40.8	**80.7**	28.6	43.6	40.9
ECO (Danelljan *et al.*, 2016*a*)	2.1	28.4	12.7	26.5	14.8	18.2	18.3		6.4	37.1	18.4	36.8	15.2	22.7	21.7
BVT (Yuan *et al.*, 2014)	3.2	6.5	19.3	42.8	16.4	22.0	20.9		6.1	3.3	27.1	56.4	24.0	29.9	28.2
CCOT (Danelljan *et al.*, 2016*d*)	2.6	26.2	17.4	39.9	16.8	23.4	22.8		3.6	39.2	24.4	55.1	21.0	31.5	30.4
MEEM (Zhang *et al.*, 2014*a*)	1.4	2.7	14.3	36.4	18.7	19.7	18.8		2.7	1.4	20.5	47.3	23.7	26.7	25.2
DSST (Danelljan *et al.*, 2017*b*)	9.0	4.6	8.6	18.8	4.3	7.9	8.0		1.8	1.5	9.9	18.6	3.1	7.9	7.6
DCF-CA (Mueller *et al.*, 2017)	52.4	56.2	46.3	28.6	59.2	46.7	49.9		73.6	66.2	68.0	34.7	77.9	65.2	67.3
MOSSE-CA (Mueller *et al.*, 2017)	48.2	56.6	42.4	33.8	55.6	45.6	48.9		67.2	66.3	61.0	37.0	72.4	62.5	64.8
SAMF-CA (Mueller *et al.*, 2017)	61.5	55.1	54.5	32.7	65.1	54.6	57.0		81.2	66.3	73.7	42.4	81.0	71.7	73.5
Proposed	**72.8**	**81.7**	**64.8**	**55.6**	**70.3**	**63.7**	**65.7**		**97.8**	**98.8**	**91.2**	**82.3**	**90.8**	**87.2**	**88.0**

the challenges, and outperforms CFNet-conv5 (Valmadre *et al.*, 2017*b*) in OV, OPR, OCC, MB and Overall performance. It also outperforms the Baseline-conv5 (Valmadre *et al.*, 2017*b*) in OV, OPR, MB, IPR, IV and FM, where Baseline-conv5 is the winner of the VOT-17 real-time challenge (Kristan *et al.*, 2017*a*). The proposed tracker also outperforms Staple (Bertinetto *et al.*, 2016*a*) in all the challenges in terms OS and DP rate, except for Illumination Variation (IV) in OS rate.

Table 4.3 shows the Average Intersection Over Union (IOU) and Average Center Location Error (CLE) of the proposed tracker along with existing trackers, on the OTB100 dataset. It is observed that the proposed tracker is the best in OPR, second best in OV, third best in MB, fourth in OCC, IPR and IV, and fifth in FM and overall performance. We also observe that in terms of average IOU, the proposed tracker outperforms

Table 4.5: Comparisons with recent trackers over the Tracking Dataset (Vojir *et al.*, 2014) based on average Intersection Over Union (IOU) ratio and Center Location Error (CLE). Following the same notation as Table 4.2.

	Average IOU (%)							Average CLE (pixels)						
	CM	IV	MC	OCC	R	SC	OverAll	CM	IV	MC	OCC	R	SC	OverAll
ADNet (Yun *et al.*, 2017)	**68.39**	77.70	**68.71**	59.08	**71.49**	**69.20**	**70.80**	**13.56**	5.33	**14.07**	**26.66**	**13.01**	**12.80**	**12.05**
CF2 (Ma *et al.*, 2015*a*)	65.12	78.96	63.89	52.46	68.21	64.04	66.04	18.38	6.18	29.14	51.37	18.38	22.42	21.05
RFD-CF2 (Kristan *et al.*, 2016*a*)	58.80	78.12	57.48	52.25	63.24	58.32	60.67	18.91	5.86	25.81	50.68	18.96	22.80	21.41
staplep (Kristan *et al.*, 2016*a*)	**67.63**	**82.67**	**68.60**	**59.88**	**72.46**	**68.75**	**70.53**	24.70	**4.56**	24.51	48.30	24.94	22.35	21.01
staple (Bertinetto *et al.*, 2016*a*)	**68.51**	**82.26**	**69.68**	**64.49**	**73.82**	**69.20**	**70.95**	**14.56**	**4.81**	**12.77**	**21.18**	**13.64**	**13.60**	**12.81**
TGPR (Gao *et al.*, 2014)	14.10	5.35	17.65	37.41	11.66	16.43	16.33	118.45	157.73	122.67	79.42	122.13	116.31	113.53
MDNet (Nam and Han, 2016)	27.35	29.33	31.43	**63.41**	23.11	34.40	32.37	60.93	64.81	58.62	**17.96**	67.94	53.49	57.80
ECO (Danelljan *et al.*, 2016*a*)	13.28	29.07	12.95	26.94	15.27	18.65	18.71	114.75	49.84	125.71	141.21	103.09	105.75	102.99
BVT (Yuan *et al.*, 2014)	20.32	6.96	19.91	43.66	16.94	22.63	21.51	74.78	165.04	84.00	46.21	81.20	75.04	79.54
CCOT (Danelljan *et al.*, 2016*d*)	19.41	26.72	17.82	40.53	17.31	23.90	23.30	85.53	68.74	99.30	77.25	90.36	80.42	82.70
MEEM (Zhang *et al.*, 2014*a*)	19.38	3.01	14.59	37.07	19.09	20.08	19.24	90.11	144.76	108.19	69.50	88.29	91.83	91.45
DSST (Danelljan *et al.*, 2017*b*)	4.76	5.07	8.80	19.31	4.44	8.11	8.25	155.68	119.19	151.19	134.66	151.50	142.98	138.09
DCF-CA (Mueller *et al.*, 2017)	53.20	57.22	47.02	29.02	60.13	47.48	50.76	42.64	33.27	53.42	110.47	38.11	51.11	47.92
MOSSE-CA (Mueller *et al.*, 2017)	49.00	57.62	43.12	34.41	56.53	46.38	49.73	49.52	33.94	55.13	90.81	33.11	50.49	47.34
SAMF-CA (Mueller *et al.*, 2017)	62.40	55.82	55.34	33.22	65.96	55.44	57.81	36.60	34.46	51.43	108.89	38.12	46.02	47.34
Proposed	65.28	**82.38**	65.76	56.47	71.14	**64.63**	66.60	**15.31**	**3.99**	**21.18**	40.26	**9.80**	**19.43**	**18.28**

Table 4.6: Comparisons with recent trackers on VOT-2016 (Kristan *et al.*, 2016*c*) based on distance precision rate at a threshold of 20 pixels and overlap success rate at an overlap threshold of 0.5. Following the same notation as Table 4.2

	Overlap Success Rate (%)				Distance Precision Rate (%)			
	IV	MC	OCC	OverAll	IV	MC	OCC	OverAll
ADNet (Yun *et al.*, 2017)	45.0	34.7	37.7	38.0	**64.5**	56.4	61.4	**61.2**
SiamRN (Kristan *et al.*, 2016*a*)	45.0	36.6	**40.1**	**38.4**	55.6	54.4	**62.3**	57.5
Hedged (Qi *et al.*, 2016)	41.4	**35.4**	39.1	36.3	57.2	**55.8**	61.5	**57.7**
MLDF (Kristan *et al.*, 2016*a*)	**42.2**	36.3	36.1	**37.6**	49.5	55.9	58.8	58.2
CF2 (Ma *et al.*, 2015*a*)	44.3	36.3	37.0	37.5	55.6	55.8	57.5	57.3
DNT (Chi *et al.*, 2017)	41.2	**38.0**	34.6	37.7	57.2	**58.1**	49.6	58.1
RFD-CF2 (Kristan *et al.*, 2016*a*)	41.2	32.3	37.1	33.4	58.7	51.4	58.6	54.2
staplep (Kristan *et al.*, 2016*a*)	41.0	34.2	33.8	36.2	52.7	50.1	49.6	52.9
staple (Bertinetto *et al.*, 2016*a*)	**46.0**	35.1	31.7	36.2	**59.4**	53.5	49.2	54.7
TGPR (Gao *et al.*, 2014)	34.8	24.7	28.5	26.1	42.3	33.5	42.0	36.4
ECO (Danelljan *et al.*, 2016*a*)	**47.6**	**39.3**	**42.7**	**40.9**	**60.0**	**60.3**	**64.8**	**62.1**
BVT (Yuan *et al.*, 2014)	23.7	26.5	23.3	26.3	30.5	41.6	35.6	42.1
CCOT (Danelljan *et al.*, 2016*d*)	42.0	**38.1**	**41.8**	**39.8**	53.7	**59.9**	**65.2**	**61.7**
MEEM (Zhang *et al.*, 2014*a*)	36.2	29.2	34.4	31.2	49.5	42.9	55.3	46.9
DSST (Danelljan *et al.*, 2017*b*)	39.5	27.7	29.6	30.4	52.4	40.0	43.2	43.9
BACF (Kiani Galoogahi *et al.*, 2017)	41.7	30.0	30.9	29.3	**58.1**	47.1	49.1	45.4
DCF-CA (Mueller *et al.*, 2017)	20.6	15.4	17.1	14.7	33.2	25.1	29.2	24.4
MOSSE-CA (Mueller *et al.*, 2017)	19.9	14.9	18.3	15.5	29.7	23.1	26.6	24.8
SAMF-CA (Mueller *et al.*, 2017)	25.0	18.2	24.7	19.3	35.8	29.4	40.1	31.4
STAPLE-CA (Mueller *et al.*, 2017)	41.8	32.6	29.1	32.2	51.0	46.5	43.0	46.0
Proposed	**46.9**	**35.4**	**38.7**	36.5	**59.0**	52.0	**58.9**	53.7

CFNet-conv1 in all the challenges, except for IPR. It also outperform CFNet-conv2 in OV, OPR, OCC, MB, IPR, Overall; CFNet-conv5 in OCC; Baseline+CF-conv3 in OPR, OCC, MB and Baseline-conv5 in OV and MB. In terms of average CLE, the proposed tracker outperforms all the versions of Valmadre *et al.* (2017*b*) in almost all the challenges. As compared to Staple, the proposed tracker does better in OPR, OCC, MB, IPR and FM. In terms of average CLE, it outperforms Staple (Bertinetto *et al.*, 2016*a*) in OV, OPR, OCC, MB and FM.

Figure 4.8 shows the Success and Precision plots for the proposed tracker along with existing trackers, on the OTB100 dataset.

Table 4.7: Comparisons with recent trackers on VOT-2016 (Kristan *et al.*, 2016*c*) based on average Intersection Over Union (IOU) ratio and Center Location Error (CLE). Following the same notation as Table 4.2

	Average IOU (%)				Average CLE (pixels)			
	IV	**MC**	**OCC**	**OverAll**	**IV**	**MC**	**OCC**	**OverAll**
ADNet (Yun *et al.*, 2017)	45.85	35.50	**38.50**	38.83	56.17	67.26	**61.09**	**60.04**
SiamRN (Kristan *et al.*, 2016*a*)	45.76	37.33	**40.86**	**39.11**	66.65	79.27	67.00	73.46
Hedged (Qi *et al.*, 2016)	42.31	36.15	39.87	37.06	**53.33**	**76.38**	68.54	**73.37**
MLDF (Kristan *et al.*, 2016*a*)	43.11	**37.11**	36.92	**38.45**	**54.50**	**60.26**	**61.64**	**57.98**
CF2 (Ma *et al.*, 2015*a*)	**45.21**	**37.02**	37.70	38.28	59.74	78.45	68.92	74.68
DNT (Chi *et al.*, 2017)	42.10	**38.87**	35.40	**38.47**	69.32	**61.09**	**65.68**	**61.54**
RFD-CF2 (Kristan *et al.*, 2016*a*)	42.03	33.03	37.86	34.14	**54.92**	78.35	62.83	74.26
staplep (Kristan *et al.*, 2016*a*)	41.74	34.92	34.44	36.91	90.74	83.79	90.98	82.18
staple (Bertinetto *et al.*, 2016*a*)	**46.86**	35.86	32.38	36.89	88.76	96.49	117.77	97.51
TGPR (Gao *et al.*, 2014)	35.63	25.28	29.16	26.76	78.62	117.09	95.03	109.80
ECO (Danelljan *et al.*, 2016*a*)	**48.50**	**40.10**	**43.53**	**41.70**	56.10	**65.91**	65.39	63.76
BVT (Yuan *et al.*, 2014)	24.25	27.18	23.93	27.02	115.83	85.07	109.02	89.65
CCOT (Danelljan *et al.*, 2016*d*)	42.88	**38.94**	**42.56**	**40.57**	79.45	69.38	**56.88**	64.74
MEEM (Zhang *et al.*, 2014*a*)	37.01	29.87	35.12	31.90	69.95	93.91	74.40	86.12
DSST (Danelljan *et al.*, 2017*b*)	40.31	28.32	30.27	31.06	83.56	122.54	111.28	112.16
BACF (Kiani Galoogahi *et al.*, 2017)	42.50	30.67	31.53	29.92	90.81	103.52	99.57	109.28
DCF-CA (Mueller *et al.*, 2017)	21.01	15.72	17.48	15.02	178.18	176.02	159.71	178.11
MOSSE-CA (Mueller *et al.*, 2017)	20.43	15.30	18.75	15.84	179.35	169.32	149.07	162.20
SAMF-CA (Mueller *et al.*, 2017)	25.51	18.60	25.16	19.69	181.64	168.26	135.99	159.40
STAPLE-CA (Mueller *et al.*, 2017)	42.55	33.23	29.65	32.88	98.48	103.48	103.06	109.84
Proposed	**47.81**	36.17	39.48	37.19	**57.04**	86.27	69.77	81.64

Evaluation on Tracking Dataset

Tracking Dataset, introduced by Tomas Vojir (Vojir *et al.*, 2014), consists of 77 sequences collected from the published literature (Kalal *et al.*, 2012; Babenko *et al.*, 2011). It includes video sequences from OTB50, OTB100 and various VOT datasets. Hence, the Tracking Dataset is full of challenging sequences at a scale similar to that of OTB and VOT datasets. Sequences in this dataset vary in length from dozens of frames to thousands and contain diverse object types like articulated and rigid objects. It also has different scene settings like static/moving cameras, indoor/outdoor and lightning conditions. Excluding the sequences used for training the LSTM in Section 4.2.3, we evaluate our tracker on 16 challenging and comparatively longer sequences (with a maximum of 2351, minimum of 76 and average of 785 frames) from this dataset. The performance is compared with 12 trackers including CCOT (Danelljan *et al.*, 2016*d*), RFD-CF2 (Kristan *et al.*, 2016*a*), staple (Bertinetto *et al.*, 2016*a*), staplep (Kristan *et al.*, 2016*a*), MDNet (Nam and Han, 2016), ADNet (Yun *et al.*, 2017), ECO (Danelljan *et al.*, 2016*a*), CF2 (Ma *et al.*, 2015*a*), DSST (Danelljan *et al.*, 2017*b*), MEEM (Zhang *et al.*, 2014*a*), BVT (Yuan *et al.*, 2014), TGPR (Gao *et al.*, 2014), DCF-CA (Mueller *et al.*, 2017), MOSSE-CA (Mueller *et al.*, 2017) and SAMF-CA (Mueller *et al.*, 2017).

Figure 4.9 shows the success and precision plots obtained for videos with challenges like camera motion, occlusion, illumination variation, low resolution, size change and

Table 4.8: Average OS and DP rate (%) Rate obtained over the Tracking Dataset (Vojir *et al.*, 2014) and VOT-2016 Dataset (Kristan *et al.*, 2016*c*). Following the same notation as Table 4.2

	Average Overlap Success Rate (%)				Average Distance Precision Rate (%)			
	IV	MC	OCC	OverAll	IV	MC	OCC	OverAll
ADNet (Yun *et al.*, 2017)	56.66	46.92	45.49	49.74	76.98	68.40	66.97	71.53
CF2 (Ma *et al.*, 2015*a*)	56.81	46.15	42.42	47.72	70.81	66.87	62.33	67.75
RFD-CF2 (Kristan *et al.*, 2016*a*)	54.56	41.27	42.41	43.14	72.84	64.13	63.21	65.35
staplep (Kristan *et al.*, 2016*a*)	55.99	46.57	43.14	48.53	69.50	63.94	60.41	65.41
staple (Bertinetto *et al.*, 2016*a*)	58.99	47.50	43.44	48.68	73.65	65.98	60.13	66.95
TGPR (Gao *et al.*, 2014)	23.83	21.96	31.52	22.33	27.19	30.54	44.76	30.56
ECO (Danelljan *et al.*, 2016*a*)	40.51	29.47	36.71	32.55	51.54	44.82	54.46	47.18
BVT (Yuan *et al.*, 2014)	17.34	23.84	30.50	24.30	20.45	36.24	43.28	36.96
CCOT (Danelljan *et al.*, 2016*d*)	36.16	30.45	41.09	33.52	48.34	46.79	61.47	50.14
MEEM (Zhang *et al.*, 2014*a*)	23.82	23.69	35.13	26.62	31.73	34.62	52.34	38.88
DSST (Danelljan *et al.*, 2017*b*)	26.61	20.64	25.61	22.12	33.60	28.88	34.11	30.49
DCF-CA (Mueller *et al.*, 2017)	33.74	26.81	21.34	27.69	45.38	41.23	31.23	40.24
MOSSE-CA (Mueller *et al.*, 2017)	33.45	25.05	24.02	27.83	43.21	37.09	30.44	39.57
SAMF-CA (Mueller *et al.*, 2017)	36.11	31.60	27.65	33.22	47.06	45.75	40.94	46.94
Proposed	59.75	46.25	44.94	47.28	73.69	66.47	67.54	66.36

Table 4.9: Comparisons with recent trackers over the UAV123 dataset (Mueller *et al.*, 2016) based on average IOU in %, average CLE in pixels, DP rate at a threshold of 20 pixels and OS rate at an overlap threshold of 0.5, in %. Following the same notation as Table 4.2

	OS (%)	IOU (%)	DP (%)	CLE
ADNet (Yun *et al.*, 2017)	36.4	37.05	51.6	173.21
Hedged (Qi *et al.*, 2016)	31.7	32.37	47.7	187.45
staplep (Kristan *et al.*, 2016*a*)	38.6	39.23	55.1	170.31
staple (Bertinetto *et al.*, 2016*a*)	35.7	36.28	50.8	178.68
TGPR (Gao *et al.*, 2014)	31.2	31.76	45.5	204.78
MDNet (Nam and Han, 2016)	43.0	43.73	59.8	152.90
ECO (Danelljan *et al.*, 2016*a*)	53.7	54.59	73.2	76.17
BVT (Yuan *et al.*, 2014)	28.7	29.30	44.6	205.95
CCOT (Danelljan *et al.*, 2016*d*)	19.9	20.26	26.1	321.17
MEEM (Zhang *et al.*, 2014*a*)	16.1	16,41	24.0	343.80
DSST (Danelljan *et al.*, 2017*b*)	34.5	35.12	46.8	188.24
DCF-CA (Mueller *et al.*, 2017)	22.5	22.86	34.8	261.47
MOSSE-CA (Mueller *et al.*, 2017)	22.2	22.67	35.8	329.88
SAMF-CA (Mueller *et al.*, 2017)	26.2	26.59	37.8	239.90
Proposed	35.4	36.87	49.4	185.66

motion changes.

Table 4.4 shows the OS rate at an overlap threshold of 0.5 and DP rate of 20 pixels. In terms of overall DP rate, the proposed tracker performs better than Staple (Bertinetto *et al.*, 2016*a*) in all the challenges. It is observed that during challenges like illumination variation, motion change, occlusion and low resolution, the proposed tracker outperforms the existing deep learning and correlation filter based trackers. It also shows second best performance during camera motion and scale change. Here, better performance during illumination variation, occlusion and low resolution is due to the appearance model pool that prevents faulty updates to the correlation filter during such challenges. Good performance during scale changes is due to the scale correlation filter, and the rotation correlation filter helps in tracking during motion changes.

Table 4.10: Comparisons with recent trackers over the VOT-2017 dataset (Kristan *et al.*, 2017*a*) based on distance precision rate at a threshold of 20 pixels and overlap success rate at an overlap threshold of 0.5. Following the same notation as Table 4.2

	Overlap Success Rate (%)						Distance Precision Rate (%)					
	CM	IV	MC	OCC	SC	OverAll	CM	IV	MC	OCC	SC	OverAll
TRACA (Choi *et al.*, 2018)	26.10	35.50	27.30	28.50	29.60	25.20	39.50	43.80	38.60	41.00	43.40	37.50
CF2 (Ma *et al.*, 2015*a*)	32.90	38.50	33.70	33.90	34.90	32.20	50.90	44.80	52.40	50.80	54.20	49.80
RFD-CF2 (Kristan *et al.*, 2016*a*)	30.30	38.90	30.50	33.70	30.10	29.10	48.80	46.80	47.90	52.30	48.90	47.40
staplep (Kristan *et al.*, 2016*a*)	28.70	32.00	31.90	29.20	29.20	29.30	42.80	38.60	46.90	42.00	43.20	43.40
staple (Bertinetto *et al.*, 2016*a*)	24.90	38.30	32.30	26.10	29.20	28.50	44.50	46.60	48.60	39.80	43.20	43.00
BACF (Kiani Galoogahi *et al.*, 2017)	27.40	34.50	26.20	26.80	24.10	25.90	42.60	48.80	41.30	41.20	35.50	40.00
ECO (Danelljan *et al.*, 2016*a*)	34.90	38.10	35.80	35.90	32.80	32.60	53.80	45.00	56.30	55.00	50.40	51.10
CCOT (Danelljan *et al.*, 2016*d*)	34.90	32.10	35.50	36.10	33.10	32.30	54.60	38.50	56.50	56.20	50.50	51.20
STAPLE-CA (Mueller *et al.*, 2017)	27.90	34.10	31.10	28.10	29.80	29.10	41.30	41.10	45.10	42.30	41.00	42.70
DCF-CA (Mueller *et al.*, 2017)	12.40	16.30	14.80	17.90	15.20	14.60	21.60	25.60	25.10	30.30	25.90	24.80
MOSSE-CA (Mueller *et al.*, 2017)	16.90	15.20	14.70	18.90	16.50	16.50	28.60	21.70	23.20	27.10	24.90	26.70
SAMF-CA (Mueller *et al.*, 2017)	17.80	18.10	17.90	23.80	19.30	19.40	28.10	24.50	29.00	37.90	29.40	30.70
Proposed	31.40	40.00	32.00	30.70	30.80	28.60	47.70	46.40	46.10	45.90	44.20	42.80

Table 4.11: Comparisons with recent trackers over the VOT-2017 dataset (Kristan *et al.*, 2017*a*) based on average Intersection Over Union (IOU) ratio and Center Location Error (CLE). Following the same notation as Table 4.2

	Average IOU (%)						Average CLE (Pixels)					
	CM	IV	MC	OCC	SC	OverAll	CM	IV	MC	OCC	SC	OverAll
TRACA (Choi *et al.*, 2018)	26.67	36.28	27.92	29.06	30.27	25.79	161.48	100.47	125.34	147.12	128.94	160.06
CF2 (Ma *et al.*, 2015*a*)	33.60	39.31	34.41	34.62	35.69	32.86	108.16	88.86	90.98	107.69	82.51	114.25
RFD-CF2 (Kristan *et al.*, 2016*a*)	30.92	39.71	31.14	34.40	30.83	29.77	108.45	90.21	91.15	111.37	91.08	118.08
staplep (Kristan *et al.*, 2016*a*)	29.30	32.66	32.60	29.77	29.81	29.89	149.71	138.19	94.29	138.44	119.33	139.10
staple (Bertinetto *et al.*, 2016*a*)	30.00	39.04	32.98	26.64	29.85	29.05	157.03	136.23	101.10	160.60	135.63	154.96
BACF (Kiani Galoogahi *et al.*, 2017)	27.98	35.21	26.79	27.37	24.65	26.48	163.20	134.07	121.40	163.26	145.46	166.32
ECO (Danelljan *et al.*, 2016*a*)	35.57	38.90	36.59	36.65	33.54	33.33	124.35	98.74	77.31	137.97	110.78	133.53
CCOT (Danelljan *et al.*, 2016*d*)	35.59	32.85	36.24	36.84	33.78	32.96	120.54	122.81	81.08	140.36	119.88	139.50
STAPLE-CA (Mueller *et al.*, 2017)	28.48	34.82	31.77	28.70	30.38	29.69	164.60	142.93	115.01	145.00	132.07	153.03
DCF-CA (Mueller *et al.*, 2017)	12.69	16.65	15.09	18.23	15.58	14.96	217.62	227.18	177.76	203.52	194.79	218.28
MOSSE-CA (Mueller *et al.*, 2017)	17.34	15.64	15.10	19.41	16.89	16.92	171.67	222.40	170.06	174.56	167.30	178.96
SAMF-CA (Mueller *et al.*, 2017)	18.17	18.43	18.22	24.28	19.69	19.74	186.99	236.57	173.13	184.62	189.10	192.67
Proposed	32.00	40.74	32.65	31.29	31.40	29.20	124.50	92.41	111.23	161.87	124.47	149.03

Table 4.5 shows the IOU and CLE averaged over all the test videos. Compared to Staple, our tracker does better under illumination variation, in terms of average IOU, and does better during illumination variation and low resolution, in terms of average CLE. As can be seen in Figure 4.9, Table 4.4 and Table 4.5, the proposed tracker is able to outperform most of the existing trackers in almost all of the challenges, demonstrating the robustness of the tracker. It also leads several trackers in overall performance.

Evaluation on VOT-2016 Dataset

The VOT-2016 Dataset (Kristan *et al.*, 2016*c*) consists of 60 challenging video sequences. The proposed tracker is evaluated over this dataset on the basis of success plots, precision plots, average IOU and average CLE.

Figure 4.10 shows the success and precision plots obtained from videos with different challenges (illumination variation, motion change and occlusion). Table 4.6 shows the OS rate at an overlap threshold of 0.5 and DP rate of 20 pixels. Table 4.7 shows

Table 4.12: Overlap Score comparisons with trackers submitted in VOT-2017 challenge for baseline experiments performed using the VOT toolkit on VOT-2017

	All
mcct (Wang *et al.*, 2018*b*)	**0.3551**
LSART (Sun *et al.*, 2018*b*)	**0.3464**
ECO (Danelljan *et al.*, 2016*a*)	**0.3078**
CCOT (Danelljan *et al.*, 2016*d*)	0.3040
Staple (Bertinetto *et al.*, 2016*a*)	0.2733
MCPF (Zhang *et al.*, 2017*b*)	0.2732
CRT (Chen and Tao, 2018)	0.2731
RCPF (Kristan *et al.*, 2017*b*)	0.2453
Proposed	0.2197
KFebT (Senna *et al.*, 2017)	0.2114
ATLAS (Kristan *et al.*, 2017*b*)	0.2038
SiamFC (Bertinetto *et al.*, 2016*b*)	0.2030
SAPKLTF (Velasco-Salido and Martınez, 2017)	0.2022
MEEM (Zhang *et al.*, 2014*a*)	0.1976
KCF (Henriques *et al.*, 2014)	0.1779
ANT (Čehovin *et al.*, 2016*a*)	0.1765
GMD (Kristan *et al.*, 2017*b*)	0.1728
SRDCF (Danelljan *et al.*, 2015*b*)	0.1622
LGT (Cehovin *et al.*, 2012)	0.1549
MIL (Babenko *et al.*, 2010)	0.1407
IVT (Ross *et al.*, 2008)	0.0979
DSST (Danelljan *et al.*, 2017*b*)	0.0976
L1APG (Bao *et al.*, 2012)	0.0960
FragTrack (Kristan *et al.*, 2017*b*)	0.0914

the IOU and CLE averaged over all the test videos. As can be seen in Figure 4.10, Table 4.6 and Table 4.7, performance of the proposed tracker is comparable to the top ranked trackers from the VOT-2016 challenge (CCOT (Danelljan *et al.*, 2016*d*), MLDF (Kristan *et al.*, 2016*a*), SiamRN (Kristan *et al.*, 2016*a*), DNT (Chi *et al.*, 2017), RFD-CF2 (Kristan *et al.*, 2016*a*), staple (Bertinetto *et al.*, 2016*a*), staplep (Kristan *et al.*, 2016*a*)). It outperforms Staple (Bertinetto *et al.*, 2016*a*) in terms of OS rate, DP rate, average IOU and avergae CLE during challenges such as illumination variation, motion change, occlusion; and overall performance. It also outperforms several recent trackers (MDNet (Nam and Han, 2016), ADNet (Yun *et al.*, 2017), ECO (Danelljan *et al.*, 2016*a*), Hedged (Qi *et al.*, 2016), CF2 (Ma *et al.*, 2015*a*), DSST (Danelljan *et al.*, 2017*b*), MEEM (Zhang *et al.*, 2014*a*), BVT (Yuan *et al.*, 2014), TGPR (Gao *et al.*, 2014), BACF (Kiani Galoogahi *et al.*, 2017), DCF-CA (Mueller *et al.*, 2017), MOSSE-CA (Mueller *et al.*, 2017), SAMF-CA (Mueller *et al.*, 2017) and STAPLE-CA (Mueller *et al.*, 2017)) in challenging scenarios as well as overall performance. Note that though ECO (Danelljan *et al.*, 2016*a*) performs very well on the VOT-2016 Dataset, the performance does not generalize to video sequences in the Tracking Dataset. Similarly, Staple performs better in terms of OS rate on the Tracking Dataset (Vojir *et al.*, 2014), but the

Table 4.13: Performance overview showing Overlap, Failures and EAO for baseline experiment and Overlap AUC for unsupervised experiment on the proposed tracker and trackers submitted in VOT-2017 challenge on VOT-2017.

	Baseline			Unsupervised
	AR Rank		EAO	Overlap
	Overlap	Failures	EAO	AUC
Proposed	0.4521	25.8801	0.2197	0.3168
ANT	0.4669	32.4457	0.1765	0.2493
ATLAS	0.5045	29.6347	0.2038	0.3215
CCOT	0.5059	15.2288	0.3040	0.3304
CRT	0.5084	15.2058	0.2731	0.3459
DSST	0.4005	63.0723	0.0976	0.1511
ECO	0.4979	**13.5112**	**0.3078**	0.3430
FragTrack	0.3898	68.2760	0.0914	0.1690
GMD	0.4606	32.4518	0.1728	0.2339
IVT	0.3991	66.2236	0.0979	0.1247
KCF	0.4721	30.1225	0.1779	0.2619
KFebT	0.4563	25.4966	0.2114	0.2744
L1APG	0.4282	81.0559	0.0960	0.1547
LGT	0.4066	32.2648	0.1549	0.1960
LSART	0.5233	**10.2036**	**0.3464**	0.3802
mcct	**0.5534**	**9.8882**	**0.3551**	**0.4007**
MCPF	**0.5245**	18.9594	0.2732	**0.3861**
MEEM	0.4765	27.6372	0.1976	0.3001
MIL	0.3922	46.0567	0.1407	0.1674
RCPF	0.5241	17.4621	0.2453	**0.3920**
SAPKLTF	0.4865	24.5059	0.2022	0.2770
SiamFC	0.5115	24.6681	0.2030	0.3110
SRDCF	0.4867	35.4262	0.1622	0.2320
Staple	**0.5406**	19.8836	0.2733	0.3159

performance does not generalize to the VOT-2016 dataset. We, therefore, compute a weighted average of the OS and DP rate of the trackers in both the cases to infer the combined performances.

Table 4.8 shows weighted average of OS and DP rate obtained over 16 sequences of the Tracking Dataset and 60 sequences of the VOT-2016 Dataset. We take the weighted average since number of frames in both the datasets are not equal. Let us denote weighted average OS rate by AOS and weighted average DP rate by ADP, then each value in the table is calculated as,

$$AOS = \left(\frac{\text{frames in VOT Dataset}}{\text{total frames}} * OS_{VOT-2016}\right) + \left(\frac{\text{frames in Tracking Dataset}}{\text{total frames}} * OS_{Tracking}\right) \quad (4.12)$$

$$ADP = \left(\frac{\text{frames in VOT Dataset}}{\text{total frames}} * DP_{VOT-2016}\right) + \left(\frac{\text{frames in Tracking Dataset}}{\text{total frames}} * DP_{Tracking}\right), \quad (4.13)$$

where number of frames in the VOT Dataset are 21,455, number of frames in the

Tracking Dataset are 12,561, and the total number of frames are 34,016. $OS_{VOT-2016}$ and $DP_{VOT-2016}$ are the OS and DP rate over VOT-2016 dataset and $OS_{Tracking}$ and $DP_{Tracking}$ are the OS and DP rate over Tracking dataset. It is observed that when considering the combined performance, the proposed tracker outperforms Staple (Bertinetto *et al.*, 2016*a*) during all the challenges, except for motion change in terms of average OS rate. It outperforms most existing trackers during illumination variation and occlusion (again due to the appearance model pool). It also outperforms many trackers during motion change (due to rotation correlation filter) and in its overall performance.

Evaluation on UAV123 Dataset

UAV123 (Mueller *et al.*, 2016) is the second largest object tracking dataset after ALOV 300++ (Smeulders *et al.*, 2014), containing sequences captured from an aerial viewpoint. It contains 123 sequences (with more than 110K frames), out of which, 20 sequences (UAV20L) are meant for long-term aerial tracking. The sequences in this dataset contain attributes like aspect ratio changes, background clutter, camera motion, fast motion, full occlusions, illumination variation, low resolution, out-of-view objects (i.e. the object of interest leaves the field of view), partial occlusions, similar objects in the scene, scale variations and viewpoint changes.

Figure 4.11 shows the success and precision plots obtained for this dataset in comparison to CCOT (Danelljan *et al.*, 2016*d*), RFD-CF2 (Kristan *et al.*, 2016*a*), staple (Bertinetto *et al.*, 2016*a*), staplep (Kristan *et al.*, 2016*a*), MDNet (Nam and Han, 2016), ADNet (Yun *et al.*, 2017), ECO (Danelljan *et al.*, 2016*a*), DSST (Danelljan *et al.*, 2017*b*), MEEM (Zhang *et al.*, 2014*a*), BVT (Yuan *et al.*, 2014), TGPR (Gao *et al.*, 2014), DCF-CA (Mueller *et al.*, 2017), MOSSE-CA (Mueller *et al.*, 2017) and SAMF-CA (Mueller *et al.*, 2017). We also show average IOU, average CLE, OS rate and DP rate for the same in Table 4.9. It shows that overall performance of the proposed tracker is comparable to other recent trackers and top submissions in the VOT-2016 challenge. A potential reason for under-performance of the proposed tracker is that this dataset contain videos with high background clutter and small target objects. In the presence of background clutter, the LSTM in Section 4.2.3 struggles to distinguish between the foreground and the background and with small target objects, HOG features for the target are not very expressive since most target information is lost during resizing the object to 32×32.

Evaluation on VOT-2017 Dataset

The VOT-2017 Dataset (Kristan *et al.*, 2017*a*) consists of 60 challenging video sequences. The proposed tracker is evaluated over this dataset on the basis of success plots, precision plots, OS rate, DP rate, average IOU and average CLE.

Figure 4.12 shows the success and precision plots obtained from videos with different challenges (camera motion, illumination variation, motion change, occlusion and size change). Table 4.10 shows the OS rate at an overlap threshold of 0.5 and DP rate at 20 pixels. Table 4.11 shows the IOU and CLE averaged over all the test videos. As can be seen in Figure 4.12, Table 4.10 and Table 4.11, performance of the proposed tracker is comparable to the top ranked trackers including the trackers from the VOT challenges (TRACA (Choi *et al.*, 2018), CCOT (Danelljan *et al.*, 2016*d*), RFD-CF2 (Kristan *et al.*, 2016*a*), staple (Bertinetto *et al.*, 2016*a*), staplep (Kristan *et al.*, 2016*a*), ECO (Danelljan *et al.*, 2016*a*), CF2 (Ma *et al.*, 2015*a*)).

The proposed tracker performs better than Staple (Bertinetto *et al.*, 2016*a*) during all the challenges in terms of OS rate and is better during camera motion, occlusion and size change in terms of DP rate. It also outperforms Staple during camera motion, illumination variation, size change and overall performance, in terms of average IOU and average CLE. It also outperforms several recent trackers (TRACA (Choi *et al.*, 2018), BACF (Kiani Galoogahi *et al.*, 2017), DCF-CA (Mueller *et al.*, 2017), MOSSE-CA (Mueller *et al.*, 2017), SAMF-CA (Mueller *et al.*, 2017) and STAPLE-CA (Mueller *et al.*, 2017)) in challenging scenarios as well as overall performance.

Evaluation on VOT-2017 Dataset using the VOT-Toolkit

To evaluate a tracker, the toolkit applies a reset-based methodology, where the performance is measured in terms of Accuracy and Robustness (Čehovin *et al.*, 2016*b*). In the reset-based methodology, a failure is detected whenever a tracker predicts a bounding box with zero overlap with the ground truth. As a result, the tracker is re-initialized five frames after the failure. The average overlap between the ground truth and the predicted bounding boxes during successful tracking periods is captured by the Accuracy measure, and the number of times the tracker loses the target (fails) during tracking determines the Robustness. To reduce the bias due to resets, ten frames after re-initialization are ignored in the accuracy measure (Kristan *et al.*, 2016*b*). Another evaluation measure is Expected Average Overlap (EAO), which estimates the average overlap a tracker

is expected to attain on a large collection of short term sequences with the same visual properties in the given dataset. This measure is used to address the increased variance and bias of the average overlap measure (Wu *et al.*, 2015), when evaluating over variable sequence lengths.

The results obtained for experiments (Kristan *et al.*, 2017*b*) on baseline (reset based experiment) and unsupervised (no-reset experiment) are given as follows.

- The Baseline Experiment:

  Table 4.12 shows the overlap score of the proposed tracker compared to the 22 existing trackers, submitted in the VOT-2017 challenge (Kristan *et al.*, 2017*b*). Table 4.13 shows the overlap and failures, that are used to calculate the tracker accuracy and robustness, along with the EAO for all the trackers. Figure 4.13 shows the Mean Accuracy-Robustness (AR) plots for the same. Figure 4.14 shows the tracker orderings for overlap and failures. The comparison is shown for Camera Motion (CM), Illumination Variation (IV), Motion Change (MC), Occlusion (OCC), Size Change (SC) and EMPTY tags along with the mean, weighted mean and pooled performances (in the case of the AR plot). The expected overlap curves and overlap scores are shown in Figure 4.15.

  It is observed that the proposed tracker outperforms trackers KFebT (Senna *et al.*, 2017), ATLAS (Kristan *et al.*, 2017*b*), SiamFC (Bertinetto *et al.*, 2016*b*), SAPKLTF (Velasco-Salido and Martınez, 2017), MEEM (Zhang *et al.*, 2014*a*), KCF (Henriques *et al.*, 2014), ANT (Čehovin *et al.*, 2016*a*), GMD (Kristan *et al.*, 2017*b*), SRDCF (Danelljan *et al.*, 2015*b*), LGT (Cehovin *et al.*, 2012), MIL (Babenko *et al.*, 2010), IVT (Ross *et al.*, 2008), DSST (Danelljan *et al.*, 2017*b*), L1APG (Bao *et al.*, 2012) and FragTrack (Kristan *et al.*, 2017*b*) in most of the challenges.

- The Unsupervised Experiment:

  Figure 4.16 shows the average overlap plot and Table 4.13 shows the Area Under Curve (AUC) of the average overlap plot for the proposed tracker compared to the 22 existing trackers, submitted in the VOT-2017 challenge (Kristan *et al.*, 2017*b*). The overlap ordering can be seen in Figure 4.17. The comparison is shown for Camera Motion (CM), Illumination Variation (IV), Motion Change (MC), Occlusion (OCC), Size Change (SC) and EMPTY tags along with average performance.

  It is observed that the proposed tracker outperforms trackers Staple (Bertinetto *et al.*, 2016*a*), KFebT (Senna *et al.*, 2017), SiamFC (Bertinetto *et al.*, 2016*b*), SAPKLTF (Velasco-Salido and Martınez, 2017), MEEM (Zhang *et al.*, 2014*a*), KCF (Henriques *et al.*, 2014), ANT (Čehovin *et al.*, 2016*a*), GMD (Kristan *et al.*, 2017*b*), SRDCF (Danelljan *et al.*, 2015*b*), LGT (Cehovin *et al.*, 2012), MIL (Babenko *et al.*, 2010), IVT (Ross *et al.*, 2008), DSST (Danelljan *et al.*, 2017*b*), L1APG (Bao *et al.*, 2012) and FragTrack (Kristan *et al.*, 2017*b*) in most of the challenges and overall performance.

4.4.3 Qualitative Evaluation

Figure 4.18 and 4.19 shows tracking in intermediate frames of the sequences from the Tracking Dataset (Vojir *et al.*, 2014) and the VOT-2016 (Kristan *et al.*, 2016*c*) Dataset. The performance is shown against Hedged (Qi *et al.*, 2016), CF2 (Ma *et al.*, 2015*a*), CCOT (Danelljan *et al.*, 2016*d*) and ECO (Danelljan *et al.*, 2016*a*) trackers. We include only few trackers in order to maintain clarity in the figures.

In Figure 4.18, the frames in first row are from the sequence Vid_A_ball of the Tracking Dataset. It shows accurate tracking by the proposed tracker during scale changes. The second row is from the $basketball$ sequence of the VOT-2016 Dataset and shows inclination of the bounding box with the change in the object's inclination. The third row is from the $girl\_mov$ sequence of the Tracking Dataset. It shows the trackers performance during occlusion. The fourth row is from the $shaking\_camera$ sequence of the same dataset and shows performance during a shaking camera. The last row is from Vid_H_panda sequence of the Tracking Dataset, showing tracking during illumination variation. These frames show that the proposed tracker is able to track during challenges where other recent trackers fail. Row 3 and 4 also demonstrate accurate tracking in long sequences.

Figure 4.19 shows tracking failure during different challenges. The frames in first row are from the $girl\_mov$ sequence of the Tracking Dataset. It shows tracking failure when the target is occluded by an object with similar appearance. Later, due to the appearance model pool, the tracking is restored to the actual target. The second row is from the $rabbit$ sequence of the VOT-2016 Dataset and shows tracking failure due to background clutter. The third row is from the $butterfly$ sequence of the VOT-2016 Dataset. It shows tracking failure when HOG features from the background in the bounding box dominate over HOG features of the foreground in the same bounding box. The last row is from the $soldier$ sequence of the same dataset and shows failure during background clutter. The above failures are mainly during scenarios when the background is similar to the foreground. This confuses the LSTM in Section 4.2.3, making it hard to distinguish between foreground and background, leading to poor tracking.

Table 4.14: Table Comparisons of the proposed tracker with its experimental versions over VOT-2016 Dataset. The table shows average IOU in %, average CLE in pixels, DP rate at a threshold of 20 pixels and OS rate at an overlap threshold of 0.5, in %. The best performance is shown by bold numbers.

	OS Rate	**Avg. IOU**	**DP Rate**	**Avg. CLE**	**Success AUC**
Versions of Proposed LSTM-AMP					
Proposed	**36.5**	**37.19**	53.7	81.64	**40.3**
LSTM-Scale-Rotation-With-HOG	35.4	37.16	54.9	74.66	40.2
LSTM-Scale-With-HOG	35.0	36.73	54.9	74.66	40.2
LSTM-Based-With-HOG	31.9	33.56	53.0	87.75	36.9
LSTM-Based-Without-HOG	30.7	32.41	47.3	92.00	35.6
LSTM-Starting-Layers	29.8	31.50	47.0	91.37	29.1
XQDA-HOG	30.6	31.26	49.6	96.22	35.6
Uniform-HOG	28.5	30.13	42.6	93.59	28.8
Versions of Baseline (Hedged) (Qi *et al.*, 2016) (Using CNN Features)					
With dynamic weights	36.3	37.06	**57.7**	**73.37**	38.1
With uniform weights	24.1	28.33	40.3	95.51	24.1

4.5 Ablation Study

This section shows how each module in the proposed tracker contributes to the overall performance. We discuss here the performance of 7 variants of the proposed tracker. Figure 4.20 shows the success and precision plots, and Table 4.14 shows the average IOU, average CLE, OS rate and DP rate obtained for each, when evaluated over the VOT-2016 dataset. Details of each variant are as follows.

- The version $Uniform - HOG$ has the location correlation filters learned over features extracted from convolutional layers conv2-2, conv3-2 and conv4-2. The weights q^u for the location predicted using each layer u are fixed to $1/3$. This version does not have the LSTM, scale correlation filter, rotation correlation filter and appearance model pool. As can be seen in Figure 4.20 and Table 4.14, the tracker shows poor performance when appropriate layers are not used and scale, rotation and filter updates are overlooked. Similar experiments are conducted with combination of many other shallow convolutional layers. They show poor performance as compared to our next method ($LSTM - Starting - Layer$).

- The $LSTM - Starting - Layer$ is an upgraded version of $Uniform - HOG$. In this, instead of keeping the weights fixed at $1/3$, adaptive weights are calculated using an LSTM trained similar to the one in Section 4.2.3. The only difference is that instead of using HOG features of the image patch, raw image patches are used in training. These image patches are reshaped to 32×32 and vectorized to 1024×8. In this case, the input to the LSTM will be of size 1024×8, unlike the 1764×1 input in Section 4.2.3. It can be observed that adaptive weights improve the OS rate by 1.3% and the DP rate by 4.4%, as compared to $Uniform - HOG$.

- A further improved approach is $LSTM - Based - Without - HOG$, which is similar to the $LSTM - Starting - Layer$, except that instead of using layers conv2-2, conv3-2 and conv4-2, layers conv3-4, conv4-4 and conv5-4 are used. This further improves the OS rate by 0.9% and DP rate by 0.3%, showing the role of choosing the right layers for feature learning.

- The next approach is $LSTM - Based - With - HOG$, which is similar to $LSTM-Based-Without-HOG$, where conv3-4, conv4-4 and conv5-4 layers are used. Additionally, the LSTM in this tracker is replaced with the LSTM in Section 4.2.3, which is trained using the HOG features. As can be seen in Figure 4.20 and Table 4.14, the LSTM trained over HOG features can classify the image patch in to foreground/background more efficiently, resulting in an improved OS rate by 1.2% and DP rate by 5.7% over $LSTM - Based - Without - HOG$.

- In the approach $LSTM - Scale - With - HOG$, the scale correlation filter is added on top of the last tracker, resulting in 3.1% improvement in OS rate and 1.9% in DP rate over $LSTM - Based - With - HOG$.

- $LSTM-Scale-With-HOG$ offers further improvement by adding the rotation correlation filter to the $LSTM - Scale - Rotation - With - HOG$ approach. Adding this correlation filters results in improved OS rate by 0.4%. In Table 4.14, it can be seen that these two approaches have the same average CLE and DP rate. This is because during rotation of the bounding box, the center of the box remains unchanged.

- Next is the $proposed$ tracker, which also contains an appearance model pool to prevent the correlation filter from drifting. This slightly compromises the DP (by 1.2%) but results in a better over all OS by 1.1% as compared to version $LSTM - Scale - Rotation - With - HOG$.

- In another version, $XQDA-HOG$, to compute weights in the $proposed$ tracker, we use Cross-view Quadratic Discriminant Analysis (XQDA), a metric learning algorithm proposed by Liao *et al.* (2015). In our implementation, we simultaneously learn a subspace $W \in R^{k\times r}$ to project the feature vectors and a metric to measure the similarity between the projected vectors. We generate the cross-view training set by dividing the foreground and background data in to two equal halves (having 150,000 samples each set). Once we learn the XQDA model, during the test time, for a given predicted bounding box, we compute its distance from foreground and background classes. Let at any frame *t*, $\mathbf{b}^1$, $\mathbf{b}^2$, $\mathbf{b}^3$ be the estimated bounding boxes corresponding to layers *conv5-4*, *conv4-4* and *conv3-4* respectively. And let $dist^1$, $dist^2$ and $dist^3$ be the distance of $\mathbf{b}^1$, $\mathbf{b}^2$, $\mathbf{b}^3$ from the background class, respectively. Then the weights q^1, q^2, q^3 corresponding to $\mathbf{b}^1$, $\mathbf{b}^2$, $\mathbf{b}^3$, respectively, can be computed as

$$q^u = \frac{dist^u}{max(dist^1, dist^2, dist^3)} \tag{4.14}$$

  where $u \in \{1, 2, 3\}$.

  This version shows poor performance as compared to the $proposed$ version in terms of OS rate, DP rate, IOU and CLE by a large difference.

4.6 Chapter summary

The chapter proposes a deep learning based object tracking algorithm with the aim to improve performance over current deep learning based approaches. The majority of

these trackers use hierarchical features learned from multiple layers of a deep network and face several issues related to aggregation of the hierarchical features from various layers, difficulties in estimating variations in the scale of the object being tracked, as well as challenges in effectively modelling the object's appearance over long time periods.

In this chapter, we tackle these issues by exploiting the rich CNN features extracted from three different convolutional layers of VGG-Net. The target location is computed based on the contribution from each of these layers. For this task, we introduce an LSTM network that determines the adaptive contribution of each layer in predicting the target's location. At every frame, target scale and rotation is computed using correlation filters learned over FHOG features of the target. To further improve the tracker performance, we introduce an appearance model pool based correction module that ensures accurate updates of the correlation filters learned over CNN features. We evaluate our tracker with sizeable and challenging datasets including videos captured from an aerial view-point. Our experimental results on multiple standard datasets show that the proposed tracker outperforms many popular trackers.

In this chapter, we investigated the importance of different CFs learned using the deep features extracted from various layers. However, deep features extracted from a pre-trained network contain multiple channels and each feature channel encodes different appearance information of the target. Therefore, treating all the feature channels equally may result in a CF with less discriminative power. In the next chapter, we investigate the importance of each feature channel that is used to train the CF, where each feature channel is assigned a scalar weight according to the importance of information it encodes.

CHAPTER 5

CGRCF: Channel Graph Regularized Correlation Filters for Visual Object Tracking

5.1 Introduction

Training Correlation Filters (CFs) with multi-channel Convolutional Neural Network (CNN) features is a challenging task. Each CNN feature channel encodes a different attribute of the target. Chapter 4 proposed a method that could dynamically weight individual CFs. While this can help prioritise some information (i.e. semantic or texture information) over others, all channels with a single CF are treated equality, which may be sub-optimal as each channel captures different information and the importance of each channel may change from one tracking step to the next. Furthermore, while some channels may offer more informative features for tracking, others with less useful information may degrade the tracking and eventually lead to tracker drift (Danelljan *et al.*, 2015*b*). To address this issue of channel importance, feature selection (Xu *et al.*, 2019*a*), adaptive importance maps (Li *et al.*, 2018*a*) and reliability learning (Sun *et al.*, 2018*a*) methods have been proposed. At the same time, it is important to learn the CF whilst preserving the similarity between different inputs that holds similar importance (Zhou *et al.*, 2016).

This chapter attempts to address the above issues and suggests that learning a weight for each feature channel and preserving the similarity of importance between different feature channels results in better tracking performance. For this, we propose a CF formulation which uses channel-graph regularization, where Channel Regularization (CR) is used to learn the channel weights and graph regularization is used to preserve the similarity of importance between different feature channels for more accurate and robust visual tracking. Thus, channels contributing more towards tracking will get similar higher weights, whereas, channels contributing less will get similar lower weights. We first demonstrate the effects of the proposed CR formulation on the existing CF based trackers: BACF (Kiani Galoogahi *et al.*, 2017) and STRCF (Li *et al.*, 2018*c*); after this we combine the CR with graph regularization to formulate the proposed channel-graph

regularization, which further improves tracker performance. Figure 5.1 shows a block diagram that describes the process flow of the proposed Channel-Graph Regularization based Correlation Filter (CGRCF) formulation for tracking.

Figure 5.2 shows a pictorial representation of the feature channels and the corresponding weights obtained using the proposed channel regularization and channel-graph regularization. Note that each feature channel appear slightly different than the other, indicating that each feature channel encode a different appearance attribute Danelljan *et al.* (2015*b*). The top two rows show features that produce similar filter responses, yet are assigned dissimilar weights using channel regularization. This is corrected using the channel-graph regularization that assigns similar weights to feature channels that produce similar response. Hence, preserving the similarity of importance between different feature channels. The third row shows a feature channel with poor target representation which is assigned a high weight using the channel regularization and a low weight using the channel-graph regularization. The fourth row shows a feature channel with rich target representation which is assigned a low weight using the channel regularization and a high weight using the channel-graph regularization.

The summarized contributions of this chapter are as follows:

1. We propose a Channel-Graph Regularized CF (CGRCF) formulation for tracking, where the Channel Regularization (CR) is used to learn channel weights and the graph regularization is used to preserve the similarity of importance between different feature channels.

2. We first demonstrate effects of the proposed CR formulation on the existing CF based trackers: BACF (Kiani Galoogahi *et al.*, 2017) and STRCF (Li *et al.*, 2018*c*); in order to show the positive effects of the CR formulation.

3. Next, we combine the CR with graph regularization to formulate the channel-graph regularization and show further improvement in the the tracker performance.

4. We present an extensive evaluation of the proposed tracker formulations on the publicly available tracking datasets: OTB100 (Wu *et al.*, 2015), TC128 (Liang *et al.*, 2015), VOT-2017 (Kristan *et al.*, 2017*b*), VOT-2019 (Kristan *et al.*, 2019), LaSOT (Fan *et al.*, 2019), UAV123 (Mueller *et al.*, 2016), and GOT-10k (Huang *et al.*, 2019*a*) . A comparative analysis shows that the proposed formulations results in a significant improvement over the baseline trackers and other recent trackers.

5. We also provide an analysis to show that our proposed channel weighing strategy is more effective than the Squeeze-and-Excitation Network's (SE-Net's) channel weights (Hu *et al.*, 2018) for object tracking.

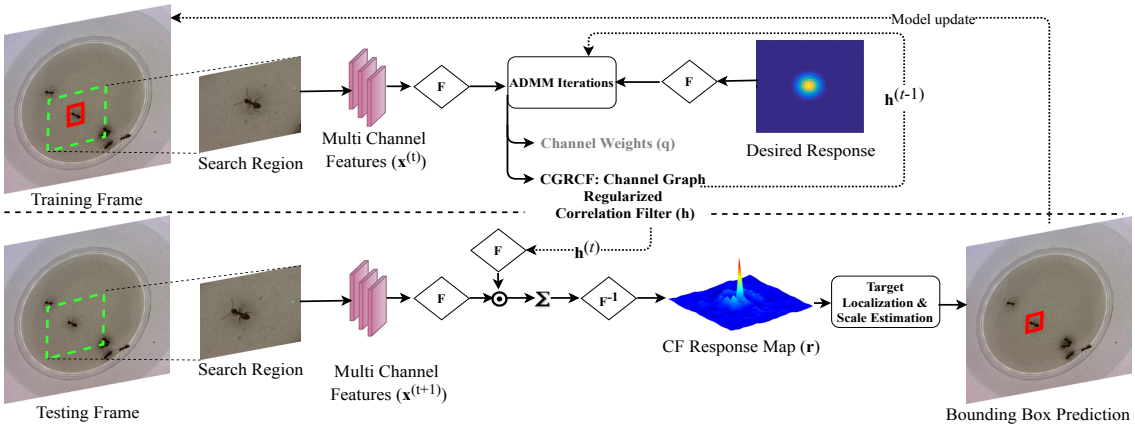


Figure 5.1: A Block diagram for the proposed CGRCF tracker. During training, $\mathbf{q}$ and $\mathbf{h}$ are learned via ADMM iterations. During testing, we extract an ensemble of deep and hand-crafted features from the search area. The target is localized using a response map obtained by the dot product of the Fourier transformed features and filters. For target scale estimation, we follow Dai *et al.* (2019). F and F^{-1} denotes Fourier and inverse Fourier transform operations respectively

5.2 Proposed Approach

In this section, we detail how the proposed Channel Regularization can be incorporated into two existing Correlation Filter (CF) based trackers, BACF (Kiani Galoogahi *et al.*, 2017) (see BACF-Channel Regularized in Section 5.2.1) and STRCF (Li *et al.*, 2018*c*) (see STRCF-Channel Regularized in Section 5.2.2). We then detail how channel regularization can be combined with graph regularization to formulate channel-graph regularization in Section 5.2.3. This is followed by detailed explanation of target localization.

5.2.1 BACF-Channel Regularized

The baseline Background Aware Correlation Filter Tracker (BACF) (Kiani Galoogahi *et al.*, 2017) formulation can be given by,

$$E(\mathbf{h}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k * (\mathbf{P}^\top \mathbf{h}_k)\right\|_2^2 + \frac{\lambda}{2}\sum_{k=1}^{K}\|\mathbf{h}_k\|_2^2, \tag{5.1}$$

where K is the total number of feature channels, $\mathbf{y} \in \mathbb{R}^{T\times 1}$ is the desired Gaussian shaped CF response. $\mathbf{x}_k \in \mathbb{R}^{T\times 1}$ is the vectorized feature and $\mathbf{h}_k \in \mathbb{R}^{T\times 1}$ is the vectorized filter for the k^{th} channel. λ is the regularization parameter and $*$ is the spatial correlation operator. $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$ is the matrix of filters from all K channels,

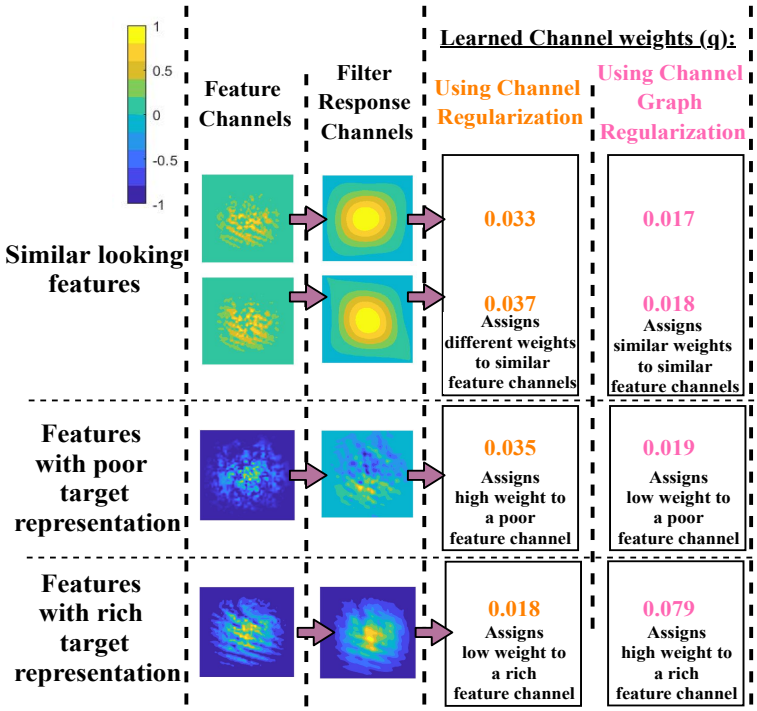


Figure 5.2: Block diagram showing the learned feature channel weights using the proposed Channel Regularized (CR) and Channel-Graph Regularized CF (CGRCF) formulations. The top two similar feature channels produces similar filter responses, to which different weights are assigned using the CR formulation and similar weights are assigned using the CGRCF. The third row shows a feature channel with poor target representation which is assigned a high weight using the CR and a low weight using the CGRCF. The fourth row shows a feature channel with rich target representation which is assigned a low weight using the CR and a high weight using the CGRCF.

$\mathbf{P} \in \mathbb{R}^{T \times T}$ represents a binary matrix that crops out the foreground region.

To learn appropriate weights for each of the K responses, $\mathbf{x}_k * (\mathbf{P}^T \mathbf{h}_k)$ in Equation 5.1, we propose the following BACF-Channel Regularized (BACF-CR) formulation,

$$E(\mathbf{h}, \mathbf{q}) = \frac{1}{2} \left\| \mathbf{y} - \sum_{k=1}^{K} q_k (\mathbf{x}_k * (\mathbf{P}^\top \mathbf{h}_k)) \right\|_2^2 + \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{h}_k\|_2^2 + \frac{\beta}{2} \|\mathbf{q}\|_2^2, \tag{5.2}$$

where q_k is a scalar weight for response channel k, β is the regularization parameter, $\mathbf{q} = \{q_1, q_2, ..., q_K\}$, and $\frac{\beta}{2}\|\mathbf{q}\|_2^2$ is a regularization term for channel weights. Inspired by many previous works to achieve computational efficiency, we use Parseval's theorem to express Equation 5.2 in the frequency domain (Dai *et al.*, 2019; Kiani Galoogahi *et al.*, 2017; Li *et al.*, 2018*c*; Danelljan *et al.*, 2015*b*), as follows,

$$E(\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}) = \frac{1}{2} \left\| \hat{\mathbf{y}} - \sum_{k=1}^{K} (\hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k) \right\|_2^2 + \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{h}_k\|_2^2 + \frac{\beta}{2} \|\mathbf{q}\|_2^2$$

$$s.t., \hat{\mathbf{g}}_k = \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k, \text{and} \hat{\mathbf{x}}_k = \sqrt{T}\mathbf{F}\mathbf{x}_k, k = 1, 2, ..., K, \tag{5.3}$$

where $\hat{}$ denotes the Discrete Fourier Transform (DFT) of a signal, such that $\hat{\mathbf{a}} = \sqrt{T}\mathbf{F}\mathbf{a}$, $\mathbf{a} \in \mathbb{R}^{T\times 1}$, $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$, $\mathbf{F}$ is a $T \times T$ orthonormal matrix of complex basis vectors that transforms any T dimensional vectorized signal into the Fourier domain and $\hat{\mathbf{G}} = [\hat{\mathbf{g}}_1, \hat{\mathbf{g}}_2, ..., \hat{\mathbf{g}}_K]$ is an auxiliary variable matrix in the Fourier domain.

Using Parseval's theorem, the energy can equivalently be represented in the time or frequency domain. Therefore, the last two terms in Equation 5.2 are equivalent to their Fourier counterparts and since they are used only to obtain the Fourier domain filter $\hat{\mathbf{g}}$, we do not convert them to the Fourier domain in Equation 5.3. Further, they can efficiently be solved in the time domain itself. We introduce an auxiliary variable, $\mathbf{G}$, to obtain the decomposition of $E(\mathbf{H}, \mathbf{q})$ in a form which can be efficiently solved using Alternating Direction Method of Multipliers (ADMM) iterations (Boyd *et al.*, 2011). It leads to decoupling between $\hat{\mathbf{G}}, \mathbf{H}$ and $\mathbf{q}$, where $\hat{\mathbf{G}}, \mathbf{H}$ and $\mathbf{q}$ can also be solved using ADMM iterations (Boyd *et al.*, 2011). The augmented Lagrangian form of Equation 5.3 can be written as,

$$E(\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}, \hat{\mathbf{S}}) = \frac{1}{2}\left\|\hat{\mathbf{y}} - \sum_{k=1}^{K} \hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k\right\|_2^2 + \frac{\lambda}{2}\sum_{k=1}^{K}\|\mathbf{h}_k\|_2^2 +$$

$$\frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2 + \frac{\beta}{2}\|\mathbf{q}\|_2^2, \tag{5.4}$$

where μ is the penalty factor and $\hat{\mathbf{S}} = [\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ..., \hat{\mathbf{s}}_K] \in \mathbb{R}^{T\times K}$ is the Fourier transform of the Lagrange multiplier. The above problem can be solved by using ADMM for the following sub-problems:

Solving for H

Given $\hat{\mathbf{G}}, \mathbf{q}$ and $\hat{\mathbf{S}}$ in Equation 5.4, the optimal solution for $\mathbf{H}^*$ can be obtained by,

$$\mathbf{h}_k^* = \underset{\mathbf{h}_k}{\operatorname{argmin}} \frac{\lambda}{2}\sum_{k=1}^{K}\|\mathbf{h}_k\|_2^2 + \frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2. \tag{5.5}$$

Computing the partial derivative with respect to $\mathbf{h}_k$ and equating to zero, we get,

$$\mathbf{h}_k^* = (\lambda \mathbf{I} + \mu T \mathbf{P}\mathbf{P}^\top q_k^2)^{-1} T q_k \mathbf{P}(\mu \mathbf{g}_k + \mathbf{s}_k), \tag{5.6}$$

where $\mathbf{I}$ is a $T \times T$ identity matrix and the inverse term is obtained by computing the reciprocal of each element.

Solving for $\hat{\mathbf{G}}$

Fixing $\mathbf{H}$, $\mathbf{q}$ and $\hat{\mathbf{S}}$ in Equation 5.4, the optimal $\hat{\mathbf{G}}^*$ can be obtained by solving,

$$\hat{\mathbf{G}}^* = \underset{\hat{\mathbf{G}}}{\operatorname{argmin}} \frac{1}{2} \left\| \hat{\mathbf{y}} - \sum_{k=1}^{K} \hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k \right\|_2^2 + \frac{\mu}{2} \sum_{k=1}^{K} \left\| \hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu} \right\|_2^2. \tag{5.7}$$

Due to high computational complexity, it is difficult to optimize Equation 5.7 (Dai *et al.*, 2019). Therefore, we process pixel-wise for all channels. The reformulated optimization problem in Equation 5.7 becomes,

$$\mathcal{V}_j^*(\hat{\mathbf{G}}) = \underset{\mathcal{V}_j(\mathbf{G})}{\operatorname{argmin}} \frac{1}{2} \left\| \hat{\mathbf{y}}_j - \mathcal{V}_j(\hat{\mathbf{X}})^\top \mathcal{V}_j(\hat{\mathbf{G}}) \right\|_2^2 + \frac{\mu}{2} \sum_{k=1}^{K} \left\| \mathcal{V}_j(\hat{\mathbf{G}}) + \mathcal{V}_j(\hat{\mathbf{M}}) \right\|_2^2, \tag{5.8}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_K]$ and $\hat{\mathbf{X}} = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, ..., \hat{\mathbf{x}}_K]$. $\mathcal{V}_j(\hat{\mathbf{X}}) = [\hat{\mathbf{x}}_{1j}, \hat{\mathbf{x}}_{2j} ..., \hat{\mathbf{x}}_{Kj}]^\top$ is a $K \times 1$ vector, picking the j^{th} element from each channel of $\hat{\mathbf{X}}$, i.e., $\mathcal{V}_1(\hat{\mathbf{X}}) = [\hat{\mathbf{x}}_{11}, \hat{\mathbf{x}}_{21}, ..., \hat{\mathbf{x}}_{K1}]^\top$ and $\mathcal{V}_j(\hat{\mathbf{G}}) = [\hat{\mathbf{g}}_{1j}, \hat{\mathbf{g}}_{2j}, ..., \hat{\mathbf{g}}_{Kj}]^\top$. Similarly, we form,

$$\mathcal{V}_j(\hat{\mathbf{M}}) = \mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) - \mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{H}\mathbf{q}), \tag{5.9}$$

where $\mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) = [\frac{\hat{\mathbf{s}}_{1j}}{\mu}, \frac{\hat{\mathbf{s}}_{2j}}{\mu}, ..., \frac{\hat{\mathbf{s}}_{kj}}{\mu}]^\top$. Solving Equation 5.9, we get,

$$\mathcal{V}_j^*(\hat{\mathbf{G}}) = \left(\mu \mathbf{I} + \mathcal{V}_j(\hat{\mathbf{X}}) \mathcal{V}_j(\hat{\mathbf{X}})^\top\right)^{-1} \left(\hat{\mathbf{y}}_j \mathcal{V}_j(\hat{\mathbf{X}}) - \mu \mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) + \mu \mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{H}\mathbf{q})\right), \tag{5.10}$$

where $\mathbf{q} = \{q_1, q_2, ..., q_K\}$ and $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$. Equation 5.10 can be efficiently computed using the Sherman-Morrison formula (Dai *et al.*, 2019) as follows.

$$\mathcal{V}_j^*(\hat{\mathbf{G}}) = \frac{1}{\mu}\left(\mathbf{I} - \frac{\mathcal{V}_j(\hat{\mathbf{X}}) \mathcal{V}_j(\hat{\mathbf{X}})^\top}{\mu + \mathcal{V}_j(\hat{\mathbf{X}})^\top \mathcal{V}_j(\hat{\mathbf{X}})}\right)(\hat{\mathbf{y}}_j \mathcal{V}_j(\hat{\mathbf{X}}) - \mu \mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) + \mu \mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{H}\mathbf{q})). \tag{5.11}$$

Solving for q_k

If $\hat{\mathbf{G}}$, $\mathbf{H}$ and $\hat{\mathbf{S}}$ are fixed in Equation 5.4, q_k can be computed as follows,

$$q_k^* = \underset{q_k}{\operatorname{argmin}} \frac{\mu}{2} \sum_{k=1}^{K} \left\| \hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu} \right\|_2^2 + \frac{\beta}{2} \|\mathbf{q}\|_2^2. \tag{5.12}$$

Solving Equation 5.12, we get,

$$q_k^* = \frac{\mu\sqrt{T}{\mathbf{h}_k}^\top \mathbf{P}\mathbf{g}_k + T{\mathbf{h}_k}^\top \mathbf{P}\mathbf{s_k}}{\mu\sqrt{T}{\mathbf{h}_k}^\top \mathbf{P}\mathbf{P}^\top \mathbf{h}_k + \beta}. \tag{5.13}$$

5.2.2 STRCF-Channel Regularized

The Spatial-Temporal Regularized Correlation Filters (STRCF) (Li *et al.*, 2018*c*) formulation can be given by,

$$E(\mathbf{h}, \mathbf{w}) = \frac{1}{2} \left\| \mathbf{y} - \sum_{k=1}^{K} \mathbf{x}_k * \mathbf{h}_k \right\|_2^2 + \frac{1}{2} \sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2 + \frac{\theta}{2} \left\| \mathbf{h}^{(t)} - \mathbf{h}^{(t-1)} \right\|_2^2, \tag{5.14}$$

where θ is the regularization parameter, $\mathbf{w}$ are the spatial weights, $\frac{\theta}{2} \left\| \mathbf{h}^{(t)} - \mathbf{h}^{(t-1)} \right\|_2^2$ is the temporal regularization term, $\frac{1}{2} \sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2$ is the spatial regularizer. $\mathbf{h}^{(t)}$ and $\mathbf{h}^{(t-1)}$ are the CFs used in the t^{th} and $(t-1)^{th}$ frames respectively.

For each of the K responses, $(\mathbf{x}_k * \mathbf{h}_k)$ in Equation 5.14, we learn an appropriate weight using the following STRCF-Channel Regularized (STRCF-CR) formulation,

$$E(\mathbf{h}, \mathbf{q}) = \frac{1}{2} \left\| \mathbf{y} - \sum_{k=1}^{K} q_k (\mathbf{x}_k * \mathbf{h}_k) \right\|_2^2 + \frac{1}{2} \sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2 +$$

$$\frac{\theta}{2} \sum_{k=1}^{K} \left\| \mathbf{h}_k^{(t)} - \mathbf{h}_k^{(t-1)} \right\|_2^2 + \frac{\beta}{2} \|\mathbf{q}\|_2^2. \tag{5.15}$$

Here, $\mathbf{w}$ is the spatial regularization matrix (Li *et al.*, 2018*c*). The augmented Lagrangian form of (14) can be written as,

$$E(\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}, \hat{\mathbf{S}}) = \frac{1}{2} \left\| \hat{\mathbf{y}} - \sum_{k=1}^{K} \hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k \right\|_2^2 + \frac{1}{2} \sum_{k=1}^{K} \|\mathbf{w} \odot \mathbf{h}_k\|_2^2 +$$

$$\frac{\theta}{2}\sum_{k=1}^{K}\left\|\mathbf{h}_k^{(t)}-\mathbf{h}_k^{(t-1)}\right\|_2^2+\frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}}_k-\sqrt{T}\mathbf{F}\mathbf{h}_k q_k+\frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2+\frac{\beta}{2}\|\mathbf{q}\|_2^2. \tag{5.16}$$

Similar to BACF-CR in Section 5.2.1, the following solution is obtained for Equation 5.16 using ADMM.

$$\mathbf{h}_k^*=(\lambda\mathbf{W}\mathbf{W}^T+\mu T q_k^2\mathbf{I}+\theta\mathbf{I})^{-1}(T q_k(\mu\mathbf{g}_k+\mathbf{s}_k)+\theta\mathbf{h}_k^{(t-1)}), \tag{5.17}$$

where $\mathbf{W}=diag(\mathbf{w})\in\mathbb{R}^{T\times T}$.

$$\mathcal{V}_j^*(\hat{\mathbf{G}})=\frac{1}{\mu}\left(\mathbf{I}-\frac{\mathcal{V}_j(\hat{\mathbf{X}})\mathcal{V}_j(\hat{\mathbf{X}})^\top}{\mu+\mathcal{V}_j(\hat{\mathbf{X}})^\top\mathcal{V}_j(\hat{\mathbf{X}})}\right)(\hat{\mathbf{y}}_j\mathcal{V}_j(\hat{\mathbf{X}})-\mu\mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right)+\mu\mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{H}\mathbf{q})). \tag{5.18}$$

$$q_k^*=\frac{\mu\sqrt{T}\mathbf{h}_k{}^\top\mathbf{g}_k+T\mathbf{h}_k{}^\top\mathbf{s_k}}{\mu\sqrt{T}\mathbf{h}_k{}^\top\mathbf{h}_k+\beta}. \tag{5.19}$$

5.2.3 Channel-Graph Regularized Correlation Filter

The channel regularized CF formulation in the previous sections is further improved by applying graph regularization. Graph regularization ensures that similar weights are learnt for feature channels that are associated with each other. This section introduces the proposed CF formulation that contains the channel-graph regularization. In the forthcoming sections the proposed Channel-Graph Regularized Correlation Filter will be denoted as CGRCF.

We define a nearest neighbor graph, G, with K vertices, where each vertex represents a feature channel. For graph G, let there be a weight matrix, $\mathbf{Z}$, that contains the distance between features $\mathbf{x}_k$ and $\mathbf{x}_j$. To compute $\mathbf{Z}$, any distance metric such as cosine distance or an appropriate distance from Minkowski family can be used. In our work, we use the heat kernel weighing scheme (Belkin and Niyogi, 2002; Zhou *et al.*, 2017) on the graph G. For every feature channel $\mathbf{x}_k$, an edge is placed between $\mathbf{x}_k$ and the other channels. The weight Z_{kj} can be given by,

$$Z_{kj}=e^{-\frac{\left\|\mathbf{x}_k-\mathbf{x}_j\right\|}{\sigma_k\sigma_j}}, \tag{5.20}$$

where σ_k and σ_j are used to adjust the decay speed of weight Z_{kj}. For each $\mathbf{x}_k$, its degree

can be defined as $d_k = \sum_{j=1}^{K} Z_{kj}$ and the degree matrix is $\mathbf{D} = diag(d_1, d_2, ..., d_\tau)$.

To obtain a set of representation coefficients using the weighted graph G, we minimize,

$$\frac{1}{2}\sum_{k=1}^{K}\sum_{j=1}^{K}(\mathbf{h}_k - \mathbf{h}_j)^2 Z_{kj} = \text{Tr}(\mathbf{HLH}^\top), \tag{5.21}$$

where Tr(.) is trace of a matrix. The Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{Z}$. The Laplacian regularizer in Equation 5.21 can be factorized as,

$$\text{Tr}(\mathbf{HLH}^\top) = Tr\left(\sum_{k,j=1}^{K} L_{kj}\mathbf{h}_k\mathbf{h}_j^\top\right) = \sum_{k,j=1}^{K} L_{kj}\mathbf{h}_j^\top\mathbf{h}_k, \tag{5.22}$$

where L_{kj} represents the k^{th} row and j^{th} column of $\mathbf{L}$. Similarly, the Laplacian regularizer for $\mathbf{q}$ can be obtained as $\sum_{k,j=1}^{K} L_{kj}q_j q_k$.

Using the above Laplacian regularizers to encode the association between the different feature channels and the channel regularization from Sections 5.2.1 and 5.2.2, we propose the channel-graph regularized CF based tracking problem as,

$$E(\mathbf{h}, \mathbf{q}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} q_k(\mathbf{x}_k * (\mathbf{P}^\top\mathbf{h}_k))\right\|_2^2 + \alpha\sum_{k,j=1}^{K} L_{kj}\mathbf{h}_j^\top\mathbf{h}_k + \beta\sum_{k,j=1}^{K} L_{kj}q_j q_k, \tag{5.23}$$

where α is the regularization parameter.

Each filter channel, $\mathbf{h}_k$, is updated individually, keeping the other channels $\{\mathbf{h}_j\}_{j\neq k}$ fixed. Thus, the optimization problem in Equation 5.23 can be re-written as,

$$E(\mathbf{h}, \mathbf{q}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} q_k(\mathbf{x}_k * (\mathbf{P}^\top\mathbf{h}_k))\right\|_2^2 + \alpha L_{kk}\mathbf{h}_k^\top\mathbf{h}_k +$$

$$2\alpha(\mathbf{h}_k)^\top\sum_{j\neq k}^{K} L_{kj}\mathbf{h}_j + \beta L_{kk}q_k^2 + 2\beta q_k\sum_{j\neq k}^{K} L_{kj}q_j. \tag{5.24}$$

Using Parseval's theorem, we can express Equation 5.24 in the frequency domain as follows,

$$E(\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}) = \frac{1}{2}\left\|\hat{\mathbf{y}} - \sum_{k=1}^{K} \hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k\right\|_2^2 + \alpha L_{kk}\mathbf{h}_k^\top\mathbf{h}_k +$$

$$2\alpha(\mathbf{h}_k)^\top\sum_{j\neq k}^{K} L_{kj}\mathbf{h}_j + \beta L_{kk}q_k^2 + 2\beta q_k\sum_{j\neq k}^{K} L_{kj}q_j,$$

$$s.t.,\ \hat{\mathbf{g}}_k = \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k \text{ and } \hat{\mathbf{x}}_k = \sqrt{T}\mathbf{F}\mathbf{x}_k, k = 1, 2, ..., K. \tag{5.25}$$

The optimal solution to the model in Equation 5.25 can be obtained using ADMM (Boyd *et al.*, 2011). The augmented Lagrangian form of Equation 5.25 can be given as,

$$E(\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}, \hat{\mathbf{S}}) = \frac{1}{2}\left\|\hat{\mathbf{y}} - \sum_{k=1}^{K} \hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k\right\|_2^2 + \frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2 +$$

$$\alpha L_{kk}\mathbf{h}_k^\top \mathbf{h}_k + 2\alpha(\mathbf{h}_k)^\top \sum_{j\neq k}^{K} L_{kj}\mathbf{h}_j + \beta L_{kk} q_k^2 + 2\beta q_k \sum_{j\neq k}^{K} L_{kj} q_j, \tag{5.26}$$

where $\hat{\mathbf{S}} = [\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ..., \hat{\mathbf{s}}_K] \in \mathbb{R}^{T\times K}$ is the Fourier transform of the Lagrange multiplier. The above problem can be solved by using ADMM for the following sub-problems:

Solving for H

Given $\hat{\mathbf{G}}$, $\mathbf{q}$ and $\hat{\mathbf{S}}$ in Equation 5.26, the optimal solution for $\mathbf{H}^*$ can be obtained by,

$$\mathbf{h}_k^* = \underset{\mathbf{h}_k}{\operatorname{argmin}} \frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2 +$$

$$\alpha L_{kk}\mathbf{h}_k^\top \mathbf{h}_k + 2\alpha(\mathbf{h}_k)^\top \sum_{j\neq k}^{K} L_{kj}\mathbf{h}_j. \tag{5.27}$$

Computing the partial derivative with respect to $\mathbf{h}_k$ and equating to zero, we get,

$$\mathbf{h}_k^* = (\mu T \mathbf{P}\mathbf{P}^\top q_k^2 + 2\alpha L_{kk})^{-1}(T q_k(\mu \mathbf{g}_k + \mathbf{s}_k) - 2\sum_{j\neq k}^{K} L_{kj}\mathbf{h}_j). \tag{5.28}$$

Solving for $\hat{\mathbf{G}}$

Fixing $\mathbf{H}$, $\mathbf{q}$ and $\hat{\mathbf{S}}$ in Equation 5.26, the optimal $\hat{\mathbf{G}}^*$ is obtained as per equation Equation 5.27.

Solving for q_k

If $\hat{\mathbf{G}}$, $\mathbf{H}$ and $\hat{\mathbf{S}}$ are fixed in Equation 5.26, q_k can be computed as follows,

$$q_k^* = \underset{q_k}{\operatorname{argmin}} \frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top \mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2 + \beta L_{kk} q_k^2 + 2\beta q_k \sum_{j\neq k}^{K} L_{kj} q_k. \tag{5.29}$$

Solving Equation 5.29, we get,

$$q_k^* = \frac{\mu\sqrt{T}\mathbf{h}_k{}^\top\mathbf{P}\mathbf{g}_k + T\mathbf{h}_k{}^\top\mathbf{P}\mathbf{s_k} - 2\beta\sum_{j\neq k}^{K} L_{kj}q_j}{\mu\sqrt{T}\mathbf{h}_k{}^\top\mathbf{P}\mathbf{P}^\top\mathbf{h}_k + 2\beta L_{kk}}. \tag{5.30}$$

5.2.4 Lagrangian Multiplier Update

The Lagrangian multipliers are updated using,

$$\hat{\mathbf{s}}_k^* = \underset{\hat{\mathbf{s}}_k}{\operatorname{argmin}} \frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^\top\mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2. \tag{5.31}$$

Computing the partial derivative with respect to $\hat{\mathbf{s}}_k$ and equating to zero, we get,

$$\hat{\mathbf{g}}_k - \sqrt{T}\mathbf{F}\mathbf{P}^\top\mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu} = 0. \tag{5.32}$$

Solving Equation 5.32, the Lagrangian variable is obtained as,

$$\hat{\mathbf{s}}_k^* = \mu(\sqrt{T}\mathbf{F}\mathbf{P}^\top\mathbf{h}_k q_k - \hat{\mathbf{g}}_k). \tag{5.33}$$

At frame t, the Lagrangian variable is updated using,

$$\hat{\mathbf{S}}^{(t)} = \hat{\mathbf{S}}^{(t-1)} + \hat{\mathbf{S}}^*, \tag{5.34}$$

where $\hat{\mathbf{S}} = [\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ..., \hat{\mathbf{s}}_K] \in \mathbb{R}^{T\times K}$ and $\hat{\mathbf{S}}^* = [\hat{\mathbf{s}}_1^*, \hat{\mathbf{s}}_2^*, ..., \hat{\mathbf{s}}_K^*] \in \mathbb{R}^{T\times K}$ is the current solution of $\hat{\mathbf{S}}$ obtained using $\hat{\mathbf{s}}_k^* = \mu^{(t)}(\sqrt{T}\mathbf{F}\mathbf{P}^\top\mathbf{h}_k^* q_k^* - \hat{\mathbf{g}}_k^*)$. Here h_k^*, q_k^* and $\hat{\mathbf{g}}_k^*$ are the current solutions of h_k, q_k and $\hat{\mathbf{g}}_k$. The penalty factor μ is updated using,

$$\mu^{(t)} = min(\mu_{max}, \delta\mu^{(t-1)}). \tag{5.35}$$

The above solution for **S** is common to all the formulations proposed in Sections 5.2.1, *B* and *C*. In Equation 5.31, the symbol $\mathbf{P} \in \mathbb{R}^{T\times T}$ represents a binary matrix that crops out the foreground region when solving for Section 5.2.1 and 5.2.2. When updating the Lagrangian multiplier for Section 5.2.2, the matrix **P** in Equation 5.31 can be replaced by a $T \times T$ identity matrix. The optimal filter $\mathbf{H}^*$ and feature channel weight q_k^* can be obtained by iteratively solving for **H**, **G**, q_k and **S**.

5.2.5 Target Localization

Target location is determined in the Fourier domain using,

$$\hat{\mathbf{r}} = \sum_{k=1}^{K} \hat{\mathbf{x}}_k \odot \hat{\mathbf{g}}_k, \tag{5.36}$$

where $\hat{}$ denotes the DFT of a signal, $\hat{\mathbf{r}}$ is the DFT of filter response map, $\mathbf{x}_k$ represents k^{th} feature channel and $\mathbf{g}_k$ is the k^{th} channel of the auxiliary variable derived using Equation 5.11 in the case of BACF-*Channel Regularized* and CGRCF. In the case of STRCF-*Channel Regularized*, $\hat{\mathbf{g}}_k$ is derived using Equation 5.18. The location at which the inverse Fourier transformed filter response, $\mathbf{r}$, shows a maximum value is used to estimate the target location. The target scale estimation strategy is adopted from Dai *et al.* (2019).

5.2.6 Model Update

To adjust to appearance variation, we use an online adaptive template scheme (Bolme *et al.*, 2010; Zhang and Suganthan, 2017; Bertinetto *et al.*, 2016*a*) to update the template model,

$$\hat{\mathbf{X}}^t_{model} = (1-\eta)\hat{\mathbf{X}}^{(t-1)}_{model} + \eta\hat{\mathbf{X}}^t, \tag{5.37}$$

where η is the online learning rate, $\hat{\mathbf{X}}^t$ is the current observation (features), $\hat{\mathbf{X}}^{(t-1)}_{model}$ is the old template model and $\hat{\mathbf{X}}^t_{model}$ is the updated template model. Based on this strategy, we use $\hat{\mathbf{X}}^t_{model}$ instead of $\hat{\mathbf{X}}$ in Equation 5.8 and 5.18. In Equation 5.37, we compute the current appearance model as a weighted sum of the current appearance observation and the past appearance model. This is done in order to preserve the past appearance of the target, so that tracker drift can be prevented during occlusion and target deformation.

5.3 Experiments

We evaluate the following proposed formulations:

- The **Proposed** (CGRCF) approach of Section 5.2.3 that incorporates channel-graph regularization, and uses an ensemble of features extracted from *Norm1* of VGG-M, *Conv4-3* of VGG-16 (Simonyan and Zisserman, 2014) and HOG features to represent and localize the target. We prefer VGG-Net (Simonyan and Zisserman, 2014), as it has shown superior performance to ResNet (He *et al.*, 2016), Squeeze-and-Excitation-ResNet50 (SE-ResNet50) (Hu *et al.*, 2018), and GoogLeNet (Szegedy *et al.*, 2015) for object tracking tasks (Li *et al.*, 2017*b*).

- **BACF-CR**, the formulation from Section 5.2.1, using the same features as **Proposed** (CGRCF).
- **STRCF-CR**, the formulation from Section 5.2.2, using the same features as **Proposed** (CGRCF).
- **BACF-CR-HOG**, the formulation from Section 5.2.1 using only HOG features.
- **STRCF-CR-HOG**, the formulation from Section 5.2.2 using only HOG features.

The final two formulations using only HOG features are included to enable fair comparison with the baselines BACF (Kiani Galoogahi *et al.*, 2017) and STRCF-HOG (Li *et al.*, 2018*c*).

All proposed formulations are implemented using MATLAB2019a with the MatConvNet toolbox. The regularization parameters λ in Equation 5.2, θ in Equation 5.15 and α in Equation 5.23 are set to 0.001. The parameter β in Equation 5.2, Equation 5.15 and Equation 5.23 is 0.01. The model learning rate, η, in Equation 5.37 is 0.0186. δ in Equation 5.35 is 10 and μ_{max} is 10^4. The initial value of the ADMM penalty factor, μ, is set to 1. The values of η, δ, μ_{max} and μ are common for all the proposed formulations described in Sections 5.2.1, 5.2.2 and 5.2.3. The ADMM steps are repeated for 3 iterations. All the above parameters are obtained empirically. The evaluation of the proposed formulations on OTB100 (Wu *et al.*, 2015), TC128 (Liang *et al.*, 2015), VOT-2017 (Kristan *et al.*, 2017*b*), VOT-2019 (Kristan *et al.*, 2019), LaSOT (Fan *et al.*, 2019), UAV123 (Mueller *et al.*, 2016), and GOT-10k (Huang *et al.*, 2019*a*) datasets is presented in the subsequent sections.

5.3.1 Comparison with Baselines on OTB100 and TC128

Table 5.1 shows a comparison of the proposed BACF-CR and BACF-CR-HOG formulations with the baseline, BACF (Kiani Galoogahi *et al.*, 2017), in terms of Overlap Success (OS) rate and Distance Precision (DP) rate. The evaluation is done over 100 sequences from the OTB100 (Wu *et al.*, 2015) dataset including challenges like Size Change (SC), Out-of-View (OV), Out-of-Plane Rotation (OPR), Occlusion (OCC), Motion Blur (MB), Low Resolution (LR), In-Plane-Rotation (IPR), Illumination Variation (IV), Fast Motion (FM), Deformation (DEF) and Background Clutter (BC). In BACF-CR-HOG, the CFs are trained using only the HOG features, just like BACF (Kiani Galoogahi *et al.*, 2017). BACF-CR-HOG can be obtained by simply adding the proposed channel requlrization to BACF (Kiani Galoogahi *et al.*, 2017). It is observed that the

proposed BACF-CR-HOG has improved OS and DP rate as compared to the baseline for many challenges. The performance further improves when the ensemble of deep features (from *Norm1* of VGG-M and *Conv4-3* of VGG-16 (Simonyan and Zisserman, 2014)) and HOG features are used to train the CFs in BACF-CR. Table 5.2 shows a comparison of the proposed CR trackers with their baseline versions on TC128 (Liang *et al.*, 2015). The proposed methods outperform their respective baselines in terms of OS and DP rate and the proposed STRCF-CR-HOG outperforms the baseline STRCF-HOG in terms of DP rate.

In Table 5.1, it is observed that BACF-CR-HOG does not outperform the baseline BACF in many challenges in terms of overlap success rate. Where as, in terms of distance precision rate, BACF-CR-HOG outperforms the baseline BACF in all challenging scenarios. In Table 5.2, BACF-CR-HOG outperforms BACF in terms of overlap success rate and distance precision rate. STRCF-CR-HOG outperforms STRCF-HOG in terms of overlap success rate. The above analysis shows that the channel regularization results in performance improvement even when deep features are not used for learning the filters.

Table 5.1: Comparisons of BACF-Channel Regularized (CR) with the baseline BACF (Kiani Galoogahi *et al.*, 2017) trackers over the OTB100 dataset (Wu *et al.*, 2015) based on overlap success rate at an overlap threshold of 0.5 and distance precision rate at a threshold of 20 pixels. Here SC = Size Change, OV = Out of View, OPR = Out of Plane Rotation, OCC = Occlusion, MB = Motion Blur, LR = Low Resolution, IPR = In Plane Rotation, IV = Illumination Variation, FM = Fast Motion, DEF = Deformation and BC = Background Clutter. The best performance is shown in **bold**.

		Overlap Success Rate (%)											
	Features Used	**SC**	**OV**	**OPR**	**OCC**	**MB**	**LR**	**IPR**	**IV**	**FM**	**DEF**	**BC**	**Overall**
BACF (Kiani Galoogahi *et al.*, 2017)	HOG	82.9	79.5	82.8	80.1	78.7	78.6	83.6	85.3	86.1	82.4	85.1	86.5
BACF-CR-HOG	HOG	82.0	77.2	83.4	79.4	80.4	74.4	82.4	83.4	86.0	82.9	84.1	86.3
BACF-CR	HOG + Deep	**92.0**	**82.2**	**92.3**	**89.3**	**85.4**	**93.3**	**91.0**	**91.6**	**90.4**	**93.4**	**88.6**	**93.0**
		Distance Precision Rate (%)											
BACF (Kiani Galoogahi *et al.*, 2017)	HOG	47.9	51.4	54.0	54.3	45.3	65.7	55.7	68.8	51.9	54.8	64.3	62.4
BACF-CR-HOG	HOG	55.7	56.1	60.0	61.4	50.8	69.9	59.0	69.9	55.6	58.2	70.9	64.3
BACF-CR	HOG + Deep	**63.6**	**58.1**	**66.4**	**66.1**	**52.6**	**90.0**	**63.8**	**70.8**	**56.4**	**65.7**	**71.9**	**69.5**

5.3.2 Evaluation on TC128 Dataset

Figure 5.3 shows the success and precision plots comparing the proposed trackers with recent trackers: GFSDCF (Xu *et al.*, 2019*a*), ASRCF (Dai *et al.*, 2019), LDES (Li *et al.*, 2019*b*), SCSAtt (Rahman *et al.*, 2020), LADCF (Xu *et al.*, 2019*b*), ARCF (Huang *et al.*, 2019*b*), IBCCF (Li *et al.*, 2017*a*), AutoTrack (Li *et al.*, 2020*a*), SSRDCF (Guo *et al.*, 2019), KAOT (Li *et al.*, 2020*c*), STRCF (Li *et al.*, 2018*c*), BACF (Kiani Galoogahi

Table 5.2: Comparisons of the Baseline trackers with their Channel Regularized (CR) versions over TC128 (Liang *et al.*, 2015) based on Overlap Success (OS) rate at an overlap threshold of 0.5 and Distance Precision (DP) rate at a threshold of 20 pixels. The best performance is shown in **red**

	OS Rate (%)	**DP Rate (%)**
BACF (Kiani Galoogahi *et al.*, 2017)	48.45	63.81
BACF-CR-HOG	48.66	65.54
BACF-CR	**57.50**	**78.29**
STRCF-HOG (Li *et al.*, 2018*c*)	**49.24**	65.28
STRCF-CR-HOG	49.15	**66.45**
STRCF (Li *et al.*, 2018*c*)	50.64	67.58
STRCF-CR	**56.91**	**77.76**

et al., 2017), CFWCR (He *et al.*, 2017), HDT (Qi *et al.*, 2016), DRCF (Fu *et al.*, 2020), SITUP (Ma *et al.*, 2020), EnKCF (Uzkent and Seo, 2018), Struck (Hare *et al.*, 2015), fDSST (Danelljan *et al.*, 2016*b*), KCF (Henriques *et al.*, 2014), LCCF (Zhang *et al.*, 2017*a*) and SCT4 (Choi *et al.*, 2016).

The proposed Channel Regularized (CR) versions BACF-CR-HOG, STRCF-CR and KCF-CR-HOG outperforms the baselines BACF (Kiani Galoogahi *et al.*, 2017), STRCF (Li *et al.*, 2018*c*) and KCF (Henriques *et al.*, 2014) respectively, in terms of success and precision. STRCF-CR-HOG outperforms the baseline STRCF-HOG in terms of precision. It is observed that the proposed BACF-CR ranks second in terms of precision and success scores, and the proposed STRCF-CR performs third and fifth best in terms of precision and success, respectively. The **Proposed** (CGRCF) tracker performs comparably to the proposed BACF-CR in terms of success score and is fifth best in terms of precision score.

5.3.3 Evaluation on VOT-2017 Dataset

This section shows a comparison of the proposed trackers with recent existing trackers using the VOT toolkit. To evaluate a tracker, the toolkit applies a reset-based methodology, where the performance is measured in terms of Accuracy (A) and Robustness (R) (Čehovin *et al.*, 2016*b*). We evaluate the trackers using an Accuracy-Robustness (AR) plot in which a tracker is more accurate if it is higher along the vertical axis and is more robust if it further to the right on the horizontal axis (Čehovin *et al.*, 2014). An additional measure is Expected Average Overlap (EAO), which estimates the average

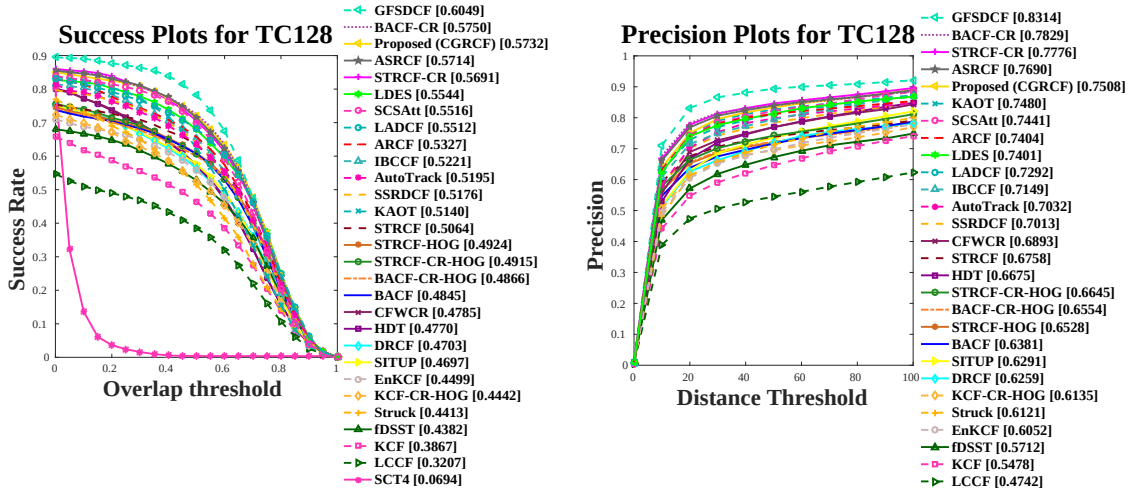


Figure 5.3: Success and Precision plots for TC128 (Liang *et al.*, 2015). The legend of success plots contains Area-Under-the-Curve scores and the legend of precision plots contains the precision scores at 20 pixels. The trackers are arranged in descending order of their performance

overlap a tracker is expected to attain on a large collection of short term sequences with the same visual properties in the given dataset. This measure is used to address the increased variance and bias of the average overlap measure (Wu *et al.*, 2015), when evaluating over variable sequence lengths. The results obtained for the baseline (reset based) and unsupervised (no-reset) experiments (Kristan *et al.*, 2017*b*) are as follows.

- The Baseline Experiment:

  Table 5.3 shows the EAO, Accuracy (A) and Robustness (R) of the proposed trackers (**Proposed** (CGRCF), BACF-CR, BACF-CR-HOG, STRCF-CR and STRCF-CF-HOG) compared to the existing trackers submitted in the VOT-2017 challenge (Kristan *et al.*, 2017*b*) and other recent trackers ARCF (Huang *et al.*, 2019*b*), LDES (Li *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), BACF (Kiani Galoogahi *et al.*, 2017) and STRCF-HOG (Li *et al.*, 2018*c*). It is observed that the **Proposed** (CGRCF) tracker is third best in terms of R and EAO and the proposed BACF-CR is third best in terms of A.

- The Unsupervised Experiment:

  Table 5.3 shows the Area Under Curve (AUC) of the average overlap plot and speed of the proposed trackers compared with existing trackers. It is observed that the **Proposed** (CGRCF) tracker ranks fourth in terms of Overlap Curve AUC and third in terms of normalized speed.

5.3.4 Evaluation on VOT-2019 Dataset

The VOT-2019 evaluation follows the format and metrics of the VOT-2017 evaluation described in Section 5.3.3. Results for the evaluation are as follows:

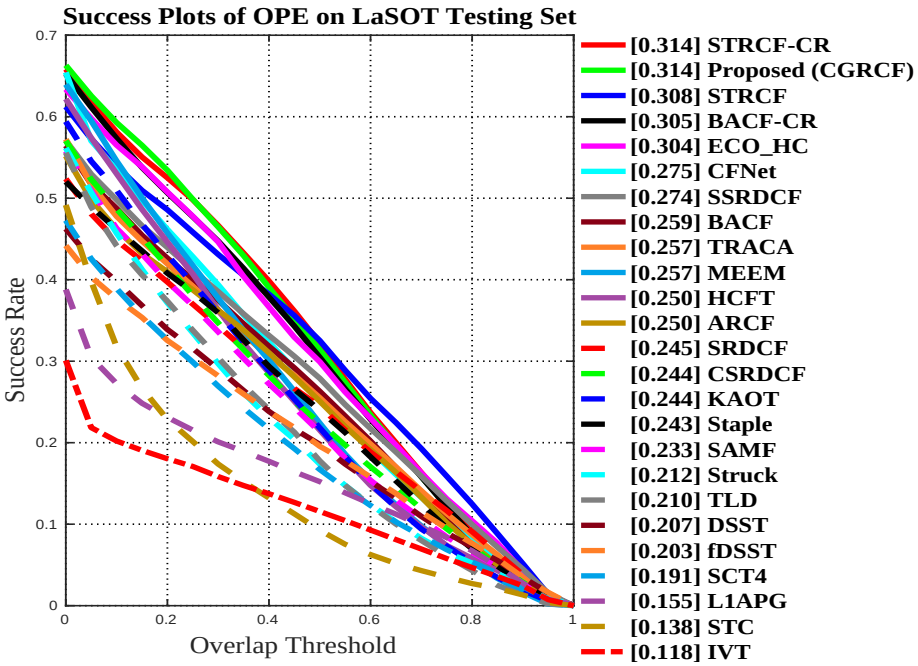


Figure 5.4: Success plots for LaSOT (Fan *et al.*, 2019). The legend of the success plots contains Area-Under-the-Curve scores. The trackers are arranged in descending order of their performance

- The Baseline Experiment:

  Table 5.4 shows the EAO, Accuracy (A) and Robustness (R) of the proposed trackers (**Proposed** (CGRCF), BACF-CR, BACF-CR-HOG, STRCF-CR and STRCF-CF-HOG) compared to trackers submitted to the VOT-2019 challenge and other recent trackers, ARCF (Huang *et al.*, 2019*b*), LDES (Li *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), BACF (Kiani Galoogahi *et al.*, 2017) and STRCF-HOG (Li *et al.*, 2018*c*). It is observed that the proposed BACF-CR performs second best in terms of A and EAO and third best in terms of R. The proposed STRCF-CR performs second best in terms R and third best in terms of A. The **Proposed** (CGRCF) tracker performs best in terms of R and is comparable to the proposed BACF-CR in terms of EAO.

- The Unsupervised Experiment:

  Table 5.4 shows the Area Under Curve (AUC) of the average overlap plots for the proposed trackers compared with the existing trackers. It is observed that the **Proposed** (CGRCF) tracker is best and STRCF-CR is second best in terms of the overlap.

5.3.5 Evaluation on LaSOT Dataset

Figure 5.4 show the success plots comparing the proposed trackers on LaSOT (Fan *et al.*, 2019) with recent trackers: STRCF (Li *et al.*, 2018*c*), ECO_HC (Danelljan *et al.*, 2017*a*), CFNet (Valmadre *et al.*, 2017*a*), SSRDCF (Guo *et al.*, 2019), BACF (Kiani Galoogahi *et al.*, 2017), TRACA (Choi *et al.*, 2018), MEEM (Zhang *et al.*, 2014*a*), HCFT

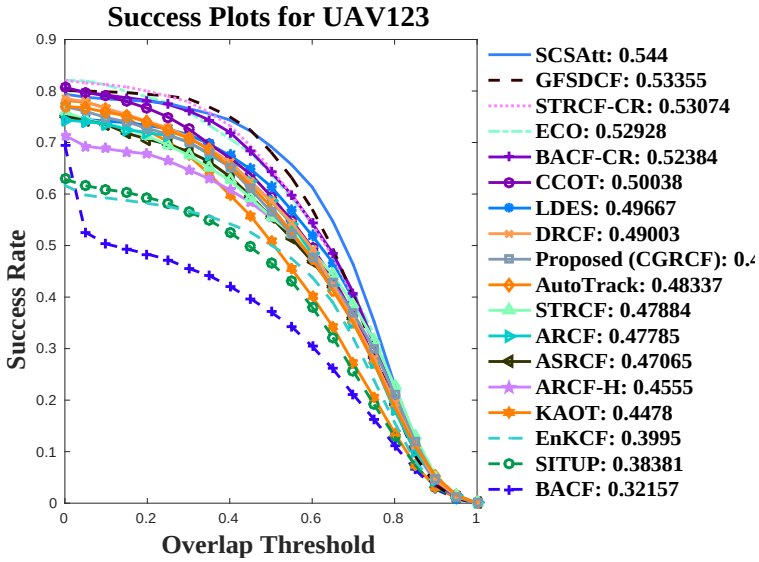


Figure 5.5: Success plots for UAV123 (Mueller *et al.*, 2016). The legend of the success plots contains Area-Under-the-Curve scores. The trackers are arranged in descending order of their performance

(Ma *et al.*, 2015*a*), ARCF (Huang *et al.*, 2019*b*), SRDCF (Danelljan *et al.*, 2015*b*), CSRDCF (Lukezic *et al.*, 2017), KAOT (Li *et al.*, 2020*c*), Staple (Bertinetto *et al.*, 2016*a*), SAMF (Li and Zhu, 2014), Struck (Hare *et al.*, 2015), TLD (Kalal *et al.*, 2011), DSST (Danelljan *et al.*, 2014*a*), fDSST (Danelljan *et al.*, 2016*b*), SCT4 (Choi *et al.*, 2016), L1APG (Bao *et al.*, 2012) and IVT (Ross *et al.*, 2008). It is observed that the proposed STRCF-CR ranks first, BACF-CR ranks fourth and the **Proposed** (CGRCF) ranks second in terms of success.

5.3.6 Evaluation on UAV123 Dataset

Figure 5.5 show the success plots comparing the proposed trackers on UAV123 (Mueller *et al.*, 2016) with recent trackers: GFSDCF (Xu *et al.*, 2019*a*), ASRCF (Dai *et al.*, 2019), LDES (Li *et al.*, 2019*b*), SCSAtt (Rahman *et al.*, 2020), ARCF (Huang *et al.*, 2019*b*), AutoTrack (Li *et al.*, 2020*a*), KAOT (Li *et al.*, 2020*c*), STRCF (Li *et al.*, 2018*c*), BACF (Kiani Galoogahi *et al.*, 2017), DRCF (Fu *et al.*, 2020), SITUP (Ma *et al.*, 2020), EnKCF (Uzkent and Seo, 2018), ECO (Danelljan *et al.*, 2017*a*) and CCOT (Danelljan *et al.*, 2016*d*). It is observed that the proposed STRCF-CR ranks third, the proposed BACF-CR ranks fifth, and the **Proposed** (CGRCF) outperforms many recent trackers but is not comparable to the proposed BACF-CR and STRCF-CR in terms of success. We argue that the small target size is the reason for the reduced performance, and this is discussed further in Section 5.3.9.

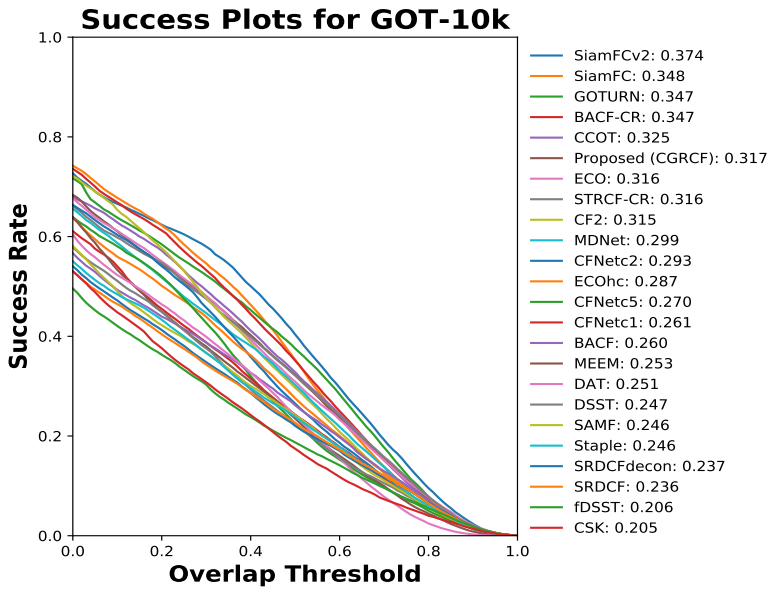


Figure 5.6: Success plots for GOT-10k (Huang *et al.*, 2019*a*). The legend of the success plots contains Area-Under-the-Curve scores. The trackers are arranged in descending order of their performance

5.3.7 Evaluation on GOT-10k Dataset

Figure 5.6 show the success plots comparing the proposed trackers on GOT-10k dataset (Huang *et al.*, 2019*a*) with recent trackers: SiamFCv2 (Valmadre *et al.*, 2017*a*), SiamFC (Bertinetto *et al.*, 2016*b*), GOTURN (Held *et al.*, 2016*b*), CCOT (Danelljan *et al.*, 2016*d*), ECO (Danelljan *et al.*, 2017*a*), CF2 (Ma *et al.*, 2015*a*), MDNet (Nam and Han, 2016), CFNetc2 (Valmadre *et al.*, 2017*a*), ECOhc (Danelljan *et al.*, 2017*a*), CFNetc5 (Valmadre *et al.*, 2017*a*), CFNetc1 (Valmadre *et al.*, 2017*a*), BACF (Kiani Galoogahi *et al.*, 2017), MEEM (Zhang *et al.*, 2014*a*), DAT (Pu *et al.*, 2018), DSST (Danelljan *et al.*, 2014*a*), SAMF (Li and Zhu, 2014), Staple (Bertinetto *et al.*, 2016*a*), SRDCFdecon (Danelljan *et al.*, 2016*c*), SRDCF (Danelljan *et al.*, 2015*b*), fDSST (Danelljan *et al.*, 2016*b*), and CSK (Henriques *et al.*, 2012). It is observed that the proposed BACF-CR ranks fourth, STRCF-CR ranks eighth, and the **Proposed** (CGRCF) ranks sixth in terms of success.

5.3.8 Evaluation using VGG-Net, ResNet50 and SE-ResNet50

Table 5.6 shows an evaluation of the proposed trackers (CGRCF, BACF-CR and STRCF-CR) using VGG-Net (Simonyan and Zisserman, 2014), ResNet50 (He *et al.*, 2016) and (Squeeze and Excitation) SE-ResNet50 (Hu *et al.*, 2018). In Table 5.6, the post-fix χ in the experiments represents that the channel weights learned using the proposed regular-

izations have been removed. This is done to analyze the tracker performance with and without the channel weights learned using the proposed regularizations and the channel weighing strategy of SE-Net (Hu *et al.*, 2018).

Our experiments demonstrate that the proposed CGRCF performs best with VGG-Net in terms of Overlap Success (OS) rate and with ResNet50 in terms of Distance Precision (DP) rate, outperforming the channel weighted SE-ResNet50. In case of BACF-CR, best performance is obtained with VGG-Net using the channel weights learned by the proposed Channel Regularization (CR). STRCF-CR performs best with ResNet50 in terms of OS rate and with VGG-Net in terms of DP rate.

The potential reason for VGG-Net outperforming ResNet-50 in most experiments can be explained using the analysis provided in Li *et al.* (2017*b*) that evaluates the performance of features extracted from VGG-Net (Simonyan and Zisserman, 2014), ResNet (He *et al.*, 2016), and GoogLeNet (Szegedy *et al.*, 2015) for visual tracking. For this evaluation, all settings in the tracker are fixed, except that the feature maps are replaced with feature maps of VGG-Net (Simonyan and Zisserman, 2014), ResNet (He *et al.*, 2016) and GoogLeNet (Szegedy *et al.*, 2015). It is observed that the VGG-Net feature based tracker achieves the best performance and significantly outperforms the ResNet and GoogLeNet based tracker in terms of Average Overlap Precision, Average Distance Precision and Average Center Location Error. It is observed that ResNet's high-level features and middle-level features are not optimal as general purpose features for other computer vision tasks (Li *et al.*, 2017*b*). The probable reason is that ResNet is an ensemble model and is not as suited to transfer learning because it needs to transfer several sub-models at the same time (Li *et al.*, 2017*b*). This may also lead the SE-ResNet (Hu *et al.*, 2018) to perform sub-optimally in object tracking. Also, SE-ResNet learns channel weights with respect to classification task which may not show superior performance compared to weights that are learned for object tracking. This is evident from the performance of the proposed trackers using Resnet50 (He *et al.*, 2016) and SE-ResNet50 (Hu *et al.*, 2018). Therefore, we use VGG-Net with the proposed channel and graph regularization.

5.3.9 Discussion

Section 5.3.1 shows that the proposed CR formulation can be conveniently applied to the HOG feature as well as deep feature based CF trackers to improve performance. Section 5.3.2, 5.3.3, 5.3.4, 5.3.5, 5.3.6, and 5.3.7 shows the impact of introducing the

Table 5.3: VOT toolkit report for VOT-2017 (Kristan *et al.*, 2017*b*) showing Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) for baseline experiment, and Overlap AUC and Speed for unsupervised experiment. The top three trackers are shown in **red**, **blue** and **green**

	Baseline			**Unsupervised**	
	AR-Rank				**Speed**
	A	**R**	**EAO**	**Overlap**	**Normalized**
Proposed (CGRCF)	0.47	**29.23**	**0.20**	0.36	**10.61**
BACF-CR	**0.48**	34.26	0.17	0.34	7.71
STRCF-CR	0.47	31.01	0.18	0.35	4.17
BACF-CR-HOG	0.47	69.72	0.11	0.24	3.75
STRCF-CR-HOG	0.47	60.77	0.11	0.24	2.80
Correlation Filter based and Hybrid Trackers					
BACF (Kiani Galoogahi *et al.*, 2017)	0.44	55.77	0.12	0.20	0.18
STRCF-HOG (Li *et al.*, 2018*c*)	0.45	61.33	0.11	0.30	7.80
ASRCF (Dai *et al.*, 2019)	0.46	30.97	0.18	0.34	2.57
ARCF (Huang *et al.*, 2019*b*)	0.46	41.41	0.15	0.28	**130.45**
LDES (Li *et al.*, 2019*b*)	**0.49**	39.64	0.18	0.32	**12.80**
CCOT (Danelljan *et al.*, 2016*d*)	0.48	**20.41**	**0.26**	**0.39**	0.05
ECO (Danelljan *et al.*, 2017*a*)	0.47	**17.66**	**0.28**	**0.40**	0.99
SRDCF (Danelljan *et al.*, 2015*b*)	0.47	64.11	0.11	0.24	1.25
UCT (Kristan *et al.*, 2017*b*)	0.48	29.79	0.20	**0.37**	3.08
ANT (Kristan *et al.*, 2017*b*)	0.45	40.15	0.16	0.27	1.89
BST (Kristan *et al.*, 2017*b*)	0.26	55.50	0.11	0.14	0.28
CGS (Kristan *et al.*, 2017*b*)	**0.50**	53.37	0.14	0.33	0.12
ATLAS (Kristan *et al.*, 2017*b*)	0.48	37.42	0.19	0.34	2.34
FSTC (Kristan *et al.*, 2017*b*)	0.47	31.95	0.18	0.33	0.19
GMD (Kristan *et al.*, 2017*b*)	0.44	54.73	0.12	0.24	2.15

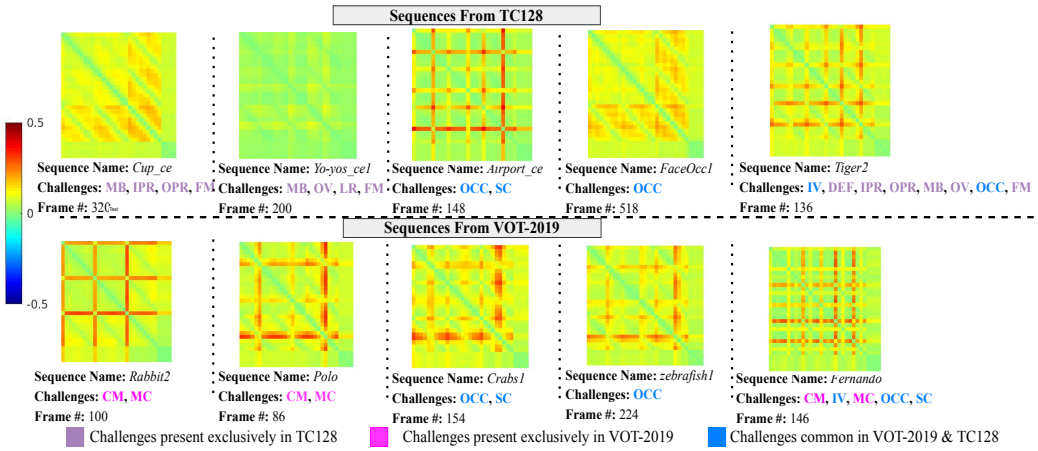


Figure 5.7: Comparison of the weight matrix ($\mathbf{Z}$) obtained for intermediate frames. Each plot shows a 31×31 weight matrix computed for 31 channels of HOG features. Here, the diagonal elements of $\mathbf{Z}$ are set to zero for better display.

Table 5.4: VOT toolkit report for VOT-2019 (Kristan *et al.*, 2019) showing Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) for baseline experiment and Overlap AUC for unsupervised experiment. The top three trackers are shown in **red**, **blue** and **green**

	Baseline			**Unsupervised**
		AR-Rank		
	EAO	**A**	**R**	**Overlap**
Proposed (CGRCF)	**0.1569**	0.4698	**42.1693**	**0.3368**
BACF-CR	**0.1570**	**0.4828**	**43.3683**	0.3184
STRCF-CR	0.1486	**0.4746**	**43.2568**	**0.3273**
BACF-CR-HOG	0.1071	0.4641	73.4347	0.2445
STRCF-CR-HOG	0.1152	0.4711	68.3897	0.2365
Correlation Filter based and Hybrid Trackers				
BACF (Kiani Galoogahi *et al.*, 2017)	0.1162	0.4476	65.7094	0.1959
STRCF-HOG (Li *et al.*, 2018*c*)	0.1141	0.4523	70.0246	0.2985
ASRCF (Dai *et al.*, 2019)	0.1451	0.4652	44.5818	**0.3230**
ARCF (Huang *et al.*, 2019*b*)	0.1351	0.4669	52.7181	0.2690
LDES (Li *et al.*, 2019*b*)	**0.1747**	**0.4882**	50.2721	0.2940
CISRDCF (Kristan *et al.*, 2019)	0.1533	0.4147	48.9861	0.2417
ANT (Kristan *et al.*, 2019)	0.1509	0.4518	53.0936	0.2390
LGT (Kristan *et al.*, 2019)	0.1308	0.3960	54.8683	0.2062

Table 5.5: Comparison of Properties of Various Tracking Benchmarks

Dataset	**Video**	**Test Video**	**Min Frame**	**Max Frame**	**Total Frames**	**Num. of Attributes**
OTB100 (Wu *et al.*, 2015)	100	-	71	3872	59k	11
TC128 (Liang *et al.*, 2015)	128	-	71	3872	55k	11
VOT-2017 (Kristan *et al.*, 2017*b*)	60	-	41	1500	21k	5
VOT-2019 (Kristan *et al.*, 2019)	60	-	41	1500	20k	5
LaSOT (Fan *et al.*, 2019)	1.4k	280	1k	11397	3.52M	14
UAV123 (Mueller *et al.*, 2016)	123	-	109	3085	113k	12
GOT-10k (Huang *et al.*, 2019*a*)	10k	180	-	-	1.5M	6

proposed graph regularization to the proposed CR formulation presented in Section 5.2.1. It is observed that though the proposed BACF-CR works better compared to the **Proposed** (CGRCF) on TC128 (Figure 5.3), and GOT-10k (Figure 5.4), UAV123 dataset (Figure 5.5), a comprehensive review using VOT-2017 (Table 5.3), VOT-2019 (Table 5.4) and LaSOT (Figure 5.4) shows that CGRCF works better for most evaluation metrics. Table 5.3 shows a comparison of the proposed formulations with existing state-of-the-art trackers on the VOT-2017 dataset. It is observed that the proposed CGRCF formulation outperforms the proposed BACF-CR formulation in terms of Robustness (R) and Expected Average Overlap (EAO) for the baseline experiments and in terms of overlap and speed for the unsupervised experiments. Table 5.4 shows a comparison of the proposed formulations with existing state-of-the-art trackers on VOT-2019 dataset.

Table 5.6: Evaluation of the Proposed Trackers, using VGG-Net (Simonyan and Zisserman, 2014), ResNet50 (He *et al.*, 2016) and SE-Resnet50 (Hu *et al.*, 2018), in terms of Overlap Success (OS) rate at an overlap threshold of 0.5 and Distance Precision (DP) rate at a threshold of 20 pixels on TC128 (Liang *et al.*, 2015). χ represents that the channel weights learned using the proposed regularizations have been removed. The best performance is shown in **red**.

Network Used	OS Rate (%)	DP Rate (%)
CGRCF		
VGG-Net	**57.32**	75.08
VGG-Net - χ	33.13	43.91
ResNet50	55.78	**77.49**
ResNet50 - χ	47.50	62.55
SE-ResNet50 - χ	45.31	60.02
BACF-CR		
VGG-Net	**57.50**	**78.29**
ResNet50	54.78	75.35
SE-ResNet50 - χ	54.93	75.19
STRCF-CR		
VGG-Net	56.91	**77.76**
ResNet50	**57.04**	77.37
SE-ResNet50 - χ	55.33	75.31

Figure 5.8: Intermediate Frames of Sequences from TC128 Showing Tracker's Performance During Various Challenges.

It is observed that the proposed CGRCF formulation outperforms the proposed BACF-CR formulation in terms of R for the baseline experiments and in terms of overlap for the unsupervised experiments. The performance of CGRCF is comparable to BACF-CR in terms of EAO for the baseline experiments. Figure 5.3 shows a comparison of the proposed formulations with the existing state-of-the-art trackers on TC128 dataset. It is observed that the proposed CGRCF formulation performs comparable to the proposed BACF-CR formulation in terms overlap success rate. Figure 5.4 shows evaluation on LaSOT dataset where the proposed CGRCF works better than the proposed BACF-CR. The computational cost and effect of the proposed graph regularization are discussed below.

Computational Complexity

The computational cost of each ADMM sub-problem that solves for $\mathbf{h}_k$ in Equations 5.6, 5.17 and 5.27 is $\mathcal{O}(T \log T)$. The cost of solving for $\hat{\mathbf{g}}$ in Equations 5.11 and 5.18 is $\mathcal{O}(KT)$, and the cost of solving for $\mathbf{q}$ using Equations 5.13, 5.19 and 5.30 is also $\mathcal{O}(KT)$. Section 5.3 shows a comparison of the proposed formulations with recent HOG and deep feature based CF trackers.

Effect of the Graph Regularization

Figure 5.7 shows a comparison of the weight matrix ($\mathbf{Z}$) for sequences with various challenges from TC128 and VOT-2019. Each plot shows a 31×31 weight matrix computed for 31 channels of the HOG features. For the sake of clarity, we exclude deep feature channels. In the first two columns, we compare $\mathbf{Z}$ obtained from sequences that contain challenges that are exclusive to the respective datasets; sequences *Cup_ce* and *Yo-yos_ce1* from TC128 and sequences *Rabbit2* and *Polo* from VOT-2019. It is observed that $\mathbf{Z}$ for the sequences from TC128 has lower values compared to $\mathbf{Z}$ for sequences from VOT-2019. It should be noted that that $\mathbf{Z}$ for the sequence *Yo-yos_ce1* from TC128 contains exceptionally low values. This is because *Yo-yos_ce1* is a Low Resolution (LR) sequence with a very small target object, which results in a poor feature representation that is not sufficient to capture the similarity between the channels.

The third and fourth columns compare the $\mathbf{Z}$ obtained from the sequences with common challenges in both datasets. It is observed that the $\mathbf{Z}$ obtained for sequences with similar challenges are similar. The last column compares one of the most challenging sequences from each dataset. Since VOT-2019 contains only 5 challenges and TC128 contains 11 challenges, as can be seen in Table 5.5, the most challenging sequences from TC128 are far more complex than those from VOT-2019. It is observed that in most cases, the $\mathbf{Z}$ obtained from TC128 sequences has lower values compared to those from VOT-2019 sequences. This is because TC128 is a more complex dataset and contains more challenges with longer sequences as compared to VOT-2019. An overview of all the tracking benchmarks used in this work is given in Table 5.5.

The lower values in $\mathbf{Z}$ implies that different feature channels of TC128 sequences have very low similarity between each other. Hence, sequences from TC128 do not benefit from the graph regularization which is meant to preserve similarity between different feature channels. This is why BACF-CR significantly outperforms CGRCF on

TC128 in terms of distance precision rate.

To demonstrate the effects of $\mathbf{Z}$ learned for different challenges on tracker's performance, we show intermediate frames of sequences from TC128 in Figure 5.8. The performance is shown against ECO (Danelljan *et al.*, 2017*a*) , ARCF (Huang *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), CCOT (Danelljan *et al.*, 2016*d*), LDES (Li *et al.*, 2019*b*), STRCF (Li *et al.*, 2018*c*), BACF (Kiani Galoogahi *et al.*, 2017), and the proposed CGRCF, BACF-CR and STRCF-CR trackers. We include only a few trackers in order to maintain clarity in the figures. It is observed that the proposed CGRCF successfully tracks the target during challenges like Occlusion (OCC), Fast Motion (FM), Out-of-Plane Rotation (OPR), Motion Blur (MB), Illumination Variation (IV), In-Plane Rotation (IPR) and Background Clutter (BC) where most existing trackers fail. The proposed trackers, along with the other trackers, fail to track during challenges like Out-of-View (OV), and Low Resolution (LR) (as can be seen in Figure 5.7). This is the reason why the proposed tracker delivers mediocre performance on UAV123 (Mueller *et al.*, 2016) in which most sequences contain a small, low resolution target object. The potential reason for BACF-CR outperforming the CGRCF on GOT-10k (Huang *et al.*, 2019*a*) can be the presence of low-resolution attribute in the dataset.

The analysis across multiple datasets shows that the proposed CGRCF formulation results in a better performance as compared to using the proposed CR formulation alone. The CGRCF formulation not only outperform the baseline trackers, but also outperform many recent CF based trackers. It is observed that performance of most of the trackers varies across the datasets, yet the CGRCF approach is consistent and performs well in all evaluation settings.

5.4 Chapter Summary

In this work, we explore channel attention and study regularization components as methods to advance the performance of baseline trackers. The introduced channel regularization component enables the CF to suppress features that may have an adverse affect on tracking. It also enhances the contribution of the feature maps that are most relevant to the current tracking step. The graph regularization component promotes learning similar weights for feature channels with high similarity. As a result a more discriminative and robust CF is trained, that achieves efficient object tracking during

multiple challenges. We use ADMM to efficiently provide an optimal solution to the proposed formulations. The positive effects of the proposed formulations are demonstrated on OTB100 (Wu *et al.*, 2015), TC128 (Liang *et al.*, 2015), VOT-2017 (Kristan *et al.*, 2017*b*), VOT-2019 (Kristan *et al.*, 2019), LaSOT (Fan *et al.*, 2019), UAV123 (Mueller *et al.*, 2016), and GOT-10k (Huang *et al.*, 2019*a*) datasets. A comparative analysis with recent top ranked tracker reveals that the proposed approach outperforms the state of the art trackers in most challenges.

In the method proposed in this chapter, the developed STRCF-CR formulation has fixed spatial weights. The temporal regularization imposes a constraint such that the current learned filter is similar to the previous filter. Other improved spatio-temporal regularization based trackers like ASRCF (Dai *et al.*, 2019) learn spatial weights that are similar to some reference weights. We argue that due to continuous temporal and spatial variations in a tracking sequence, the filter and spatial weights in a tracking step will not be identical to their reference counterparts. In the next chapter, along with assigning dynamic weights to the deep feature channels, we model the spatial and temporal variations explicitly using an Importance Guided Sparse Spatio-Temporal constrained optimization framework for tracking.

CHAPTER 6

IGSSTRCF: Importance Guided Sparse Spatio-Temporal Regularized Correlation Filters For Tracking

6.1 Introduction

Multi-channel CNN feature encode different attributes of the target in each channel. Therefore, the importance of each channel may change from one tracking step to the next. Some channels may offer more informative features for tracking, while others with less useful information may degrade the tracking and eventually lead to tracker drift (Danelljan *et al.*, 2015*b*). To address this issue of channel importance, feature selection (Xu *et al.*, 2019*a*), adaptive importance maps (Li *et al.*, 2018*a*) and reliability learning (Sun *et al.*, 2018*a*) methods have been proposed. Inspired by the above work, we investigated a channel graph regularization based Correlation Filter (CF) tracker formulation in Chapter 5. We further improve the proposed formulation from Chapter 5 by introducing an efficient spatial and temporal regularizations, increasing awareness of previous and spatially adjacent observations.

The challenges with spatial regularization based CF trackers are that they either have fixed spatial weights (Danelljan *et al.*, 2015*b*), or learn spatial weights that are similar to some reference weights (Dai *et al.*, 2019). Likewise, temporal regularization based CF tracker imposes a constraint such that the current learned filter is similar to the previous filter (Li *et al.*, 2018*c*). However, due to continuous temporal and spatial variations in a tracking sequence, the filter and spatial weights in a tracking step will not be identical to their reference counterparts. The motivation for our proposed approach stems from the proposition that the above variations can be explicitly modelled using a constrained optimization framework. To model the spatial and temporal variations, we propose an Importance Guided Sparse Spatio-Temporal Regularization based CF (IGSSTRCF) tracker with following advances:

1. We introduce a Sparse Spatial Regularization (SSR) component that learns the spatial weights with the help of reference weights, and simultaneously models the

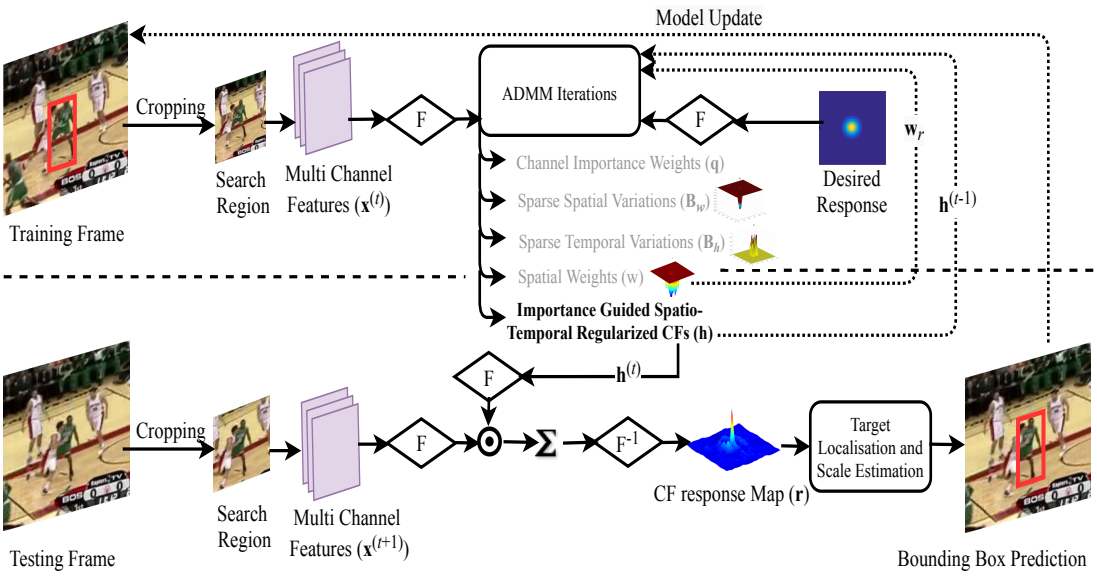


Figure 6.1: A Block diagram for the proposed IGSSTRCF tracker. During training, $\mathbf{q}, \mathbf{B}_w, \mathbf{B}_h, \mathbf{w}$ and $\mathbf{h}$ are learned via ADMM iterations. During testing, we extract an ensemble of deep and hand-crafted features from the search area. The target is localized using a response map obtained by the dot product of the Fourier transformed features and filters. For target scale estimation, we follow Dai *et al.* (2019). $\mathbf{F}$ and $\mathbf{F}^{-1}$ denotes Fourier and inverse Fourier transform operations respectively

sparse difference between the reference weights and the learned spatial weights. The filter coefficients belonging to the background region are assigned higher penalty weights. This suppresses the effect of unfavorable background information and boundary effects in the learned filter.

2. We introduce a Sparse Temporal Regularization (STR) component that learns a correlation filter by modelling the sparse difference between the previous and the current filter. As a result, the filter sparsely adapts to appearance changes, preventing drift.
3. We introduce a Channel Importance (CI) term that assigns higher weights to the feature channels that encode useful target information, and lowers weights to the less informative channels. As a result, less informative channels that may adversely effect training are suppressed.

Recently, in a concurrent work, GFSDCF (Xu *et al.*, 2019*a*), a group feature selection method for multi-channel image representations, reducing the dimensionality across both spatial and channel dimensions with a temporal smoothness regularisation term is introduced. Though our spatial regularization term is similar in spirit, the temporal regularization and overall formulation as an importance guided sparse spatio-temporal regularization is completely different from (Xu *et al.*, 2019*a*); The authors in (Xu *et al.*, 2019*a*) propose an L_1 channel regularization and does not focus on modeling the sparse temporal variations, Whereas, the proposed IGSSTRCF models the

sparse spatio-temporal variations and use L_2 channel regularization that learns non-zero weight for each feature channel.

Figure 6.1 shows a block diagram that describes the process flow of the proposed IGSSTRCF tracker. We evaluate the tracker on the benchmark datasets: TC128 (Liang *et al.*, 2015), UAV123 (Mueller *et al.*, 2016), VOT-2019 (Kristan *et al.*, 2019) and VOT-2017 (Kristan *et al.*, 2017*b*). A comparative analysis shows that the proposed formulation results in a significant improvement over the baselines (Dai *et al.*, 2019; Li *et al.*, 2018*c*) and other recent trackers. An ablation study is also presented that demonstrates the importance of each regularization term for tracker performance.

6.2 Proposed Approach

(Li *et al.*, 2018*c*) employs spatio-temporal constraints that utilize CFs learned in the previous frame to learn the CFs in the current frame. (Dai *et al.*, 2019) introduces an object aware spatial regularization that attempts to learn spatial weights that are similar to the reference spatial weights. The regularization terms in (Li *et al.*, 2018*c*) and (Dai *et al.*, 2019) make use of a reference to learn the CFs and spatial weights. However, the target appearance varies with every frame. Therefore, the spatial weights or CFs learned in consecutive frames should be constrained to be similar while still adapting to variations.

Besides the above methods, many CF based trackers focus on modeling channel importance as each feature channel can make a dynamic contribution during each tracking step (Ge *et al.*, 2019; Lu *et al.*, 2019*b*; Li *et al.*, 2018*a*; Sun *et al.*, 2018*a*; Zhou *et al.*, 2016). However, these channel importance based CF trackers do not employ spatial (Danelljan *et al.*, 2015*a*,*b*, 2016*c*; Kiani Galoogahi *et al.*, 2015) or temporal (Dai *et al.*, 2019; Li *et al.*, 2018*c*) regularization.

To combat the shortcomings of the above spatio-temporal regularization based (Dai *et al.*, 2019; Li *et al.*, 2018*c*) and channel importance based (Li *et al.*, 2018*a*; Sun *et al.*, 2018*a*; Zhou *et al.*, 2016) CF trackers, we propose an Importance Guided Sparse Spatio-Temporal Regularization based CF (IGSSTRCF) tracker which is formulated as follows,

$$E(\mathbf{h}, \mathbf{q}, \mathbf{w}, \mathbf{B}_w, \mathbf{B}_h) = \frac{1}{2}\left\| \mathbf{y} - \sum_{k=1}^{K} \overbrace{q_k}^{\text{CI}} (\mathbf{x}_k * (\mathbf{P}^T \mathbf{h}_k)) \right\|_2^2 +$$

$$\frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{w}\odot\mathbf{h}_k\|_2^2+\underbrace{\frac{\lambda_2}{2}\left\|\mathbf{w}-\mathbf{w}_r-\mathbf{B}_w\right\|_2^2+\zeta\|\mathbf{B}_w\|_1}_{\textbf{Sparse Spatial Regularization (SSR)}}+$$

$$\underbrace{\frac{\theta}{2}\left\|\mathbf{h}^{(t)}-\mathbf{h}^{(t-1)}-\mathbf{B}_h\right\|_2^2+\eta\|\mathbf{B}_h\|_1}_{\textbf{Sparse Temporal Regularization (STR)}}+\underbrace{\frac{\beta}{2}\|\mathbf{q}\|_2^2}_{\textbf{Channel Importance (CI)}}\quad,\tag{6.1}$$

where q_k is a scalar weight for response channel k, $\mathbf{q}=\{q_1,q_2,...,q_K\}$, and $\frac{\beta}{2}\|\mathbf{q}\|_2^2$ is a regularization term for the channel weights. $\frac{\lambda_2}{2}\|\mathbf{w}-\mathbf{w}_r-\mathbf{B}_w\|_2^2$ is the spatial regularization component and $\mathbf{B}_w$ is a sparse vector that learns the spatial changes between the current and reference spatial weights. $\frac{\theta}{2}\left\|\mathbf{h}^{(t)}-\mathbf{h}^{(t-1)}-\mathbf{B}_h\right\|_2^2$ is the temporal regularization term and $\mathbf{B}_h$ is a sparse vector that learns the temporal changes between the current and past filter. $\lambda_1,\lambda_2,\theta,\zeta,\eta$ and β are the regularization parameters. Using Parseval's theorem to express Equation 6.1 in frequency domain, the equality constrained optimization form is given by,

$$E(\hat{\mathbf{G}},\mathbf{H},\mathbf{q},\mathbf{w},\mathbf{B}_w,\mathbf{B}_h)=\frac{1}{2}\left\|\hat{\mathbf{y}}-\sum_{k=1}^{K}\hat{\mathbf{x}}_k\odot\hat{\mathbf{g}}_k\right\|_2^2+$$

$$\frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{w}\odot\mathbf{h}_k\|_2^2+\frac{\lambda_2}{2}\left\|\mathbf{w}-\mathbf{w}_r-\mathbf{B}_w\right\|_2^2+\zeta\|\mathbf{B}_w\|_1+$$

$$\frac{\theta}{2}\sum_{k=1}^{K}\left\|\mathbf{h}_k^{(t)}-\mathbf{h}_k^{(t-1)}-\mathbf{B}_h\right\|_2^2+\eta\|\mathbf{B}_h\|_1+\frac{\beta}{2}\|\mathbf{q}\|_2^2,$$

$$\textit{s.t.},\ \hat{\mathbf{g}}_k=\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{h}_k q_k,\tag{6.2}$$

where $\hat{}$ denotes the Discrete Fourier Transform (DFT) of a signal, such that $\hat{\mathbf{a}}=\sqrt{T}\mathbf{F}\mathbf{a}$, $\mathbf{a}\in\mathbb{R}^{T\times 1}$, $\mathbf{F}$ is a $T\times T$ orthonormal matrix of complex basis vectors that transforms any T dimensional vectorized signal into the Fourier domain and $\hat{\mathbf{G}}=[\hat{\mathbf{g}}_1,\hat{\mathbf{g}}_2,...,\hat{\mathbf{g}}_K]$ is an auxiliary variable matrix. The local optimal solution to the model in Equation 6.2 can be obtained using ADMM (Boyd *et al.*, 2011). The augmented Lagrangian form of Equation 6.2 is given by,

$$E(\hat{\mathbf{G}},\mathbf{H},\mathbf{q},\mathbf{w},\mathbf{B}_w,\mathbf{B}_h)=\frac{1}{2}\left\|\hat{\mathbf{y}}-\sum_{k=1}^{K}\hat{\mathbf{x}}_k\odot\hat{\mathbf{g}}_k\right\|_2^2+$$

$$\frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{w}\odot\mathbf{h}_k\|_2^2+\frac{\lambda_2}{2}\left\|\mathbf{w}-\mathbf{w}_r-\mathbf{B}_w\right\|_2^2+\zeta\|\mathbf{B}_w\|_1+$$

$$\frac{\theta}{2}\left\|\mathbf{h}^{(t)}-\mathbf{h}^{(t-1)}-\mathbf{B}_h\right\|_2^2+\eta\|\mathbf{B}_h\|_1+$$

$$\frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k}-\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{h}_k q_k+\frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2+\frac{\beta}{2}\|\mathbf{q}\|_2^2, \tag{6.3}$$

where μ is the penalty factor and $\hat{\mathbf{S}}=[\hat{\mathbf{s}}_1,\hat{\mathbf{s}}_2,...,\hat{\mathbf{s}}_K]\in\mathbb{R}^{T\times K}$ is the Fourier transform of the Lagrange multiplier. The above problem can be solved by using ADMM for the following sub-problems:

Solving for H

Given $\hat{\mathbf{G}},\mathbf{q},\mathbf{w},\mathbf{B}_w,\mathbf{B}_h$ in Equation 6.3, the optimal solution for $\mathbf{H}^*$ is obtained by,

$$\mathbf{h}_k^*=\underset{\mathbf{h}_k}{\operatorname{argmin}}\frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{w}\odot\mathbf{h}_k\|_2^2+\frac{\theta}{2}\sum_{k=1}^{K}\left\|\mathbf{h}_k^{(t)}-\mathbf{h}_k^{(t-1)}-\mathbf{B}_h\right\|_2^2+$$

$$\frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k}-\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{h}_k q_k+\frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2. \tag{6.4}$$

Solving Equation 6.4, we get,

$$\mathbf{h}_k^*=(\lambda_1\mathbf{W}\mathbf{W}^T+\mu T q_k^2\mathbf{I}+\theta\mathbf{I})^{-1}(Tq_k\mathbf{P}(\mu\mathbf{g}_k+\mathbf{s}_k)+\theta\mathbf{h}_k^{(t-1)}+\theta\mathbf{B}_h), \tag{6.5}$$

where $\mathbf{W}=diag(\mathbf{w})\in\mathbb{R}^{T\times T}$ and the inverse term can be conveniently obtained by computing the reciprocal of each element. $\mathbf{H}^*$ can be obtained using $\mathbf{H}^*=[\mathbf{h}_1^*,\mathbf{h}_2^*,...,\mathbf{h}_K^*]$.

Solving for $\hat{\mathbf{G}}$

Fixing $\mathbf{H},\mathbf{q},\mathbf{w},\mathbf{B}_w,\mathbf{B}_h$ in Equation 6.3, the optimal $\hat{\mathbf{G}}^*$ can be obtained by solving,

$$\hat{\mathbf{G}}^*=\underset{\hat{\mathbf{G}}}{\operatorname{argmin}}\frac{1}{2}\left\|\hat{\mathbf{y}}-\sum_{k=1}^{K}\hat{\mathbf{x}}_k\odot\hat{\mathbf{g}}_k\right\|_2^2+\frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k}-\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{h}_k q_k+\frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2. \tag{6.6}$$

However, due to high computational complexity, it is difficult to optimize Equation 6.6 (Dai *et al.*, 2019). Therefore, we proceed pixel-wise for all channels. The reformulated

optimization problem in Equation 6.6 is given by,

$$\mathcal{V}_j^*(\hat{\mathbf{G}}) = \underset{\mathcal{V}_j(\mathbf{G})}{\operatorname{argmin}} \frac{1}{2}\left\|\hat{\mathbf{y}}_j - \mathcal{V}_j(\hat{\mathbf{X}})^T \mathcal{V}_j(\hat{\mathbf{G}})\right\|_2^2 + \frac{\mu}{2}\left\|\mathcal{V}_j(\hat{\mathbf{G}}) + \mathcal{V}_j(\hat{\mathbf{M}})\right\|_2^2, \tag{6.7}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_K]$ and $\hat{\mathbf{X}} = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, ..., \hat{\mathbf{x}}_K]$. $\mathcal{V}_j(\hat{\mathbf{X}}) = [\hat{\mathbf{x}}_{1j}, \hat{\mathbf{x}}_{2j} ..., \hat{\mathbf{x}}_{Kj}]^T$ is a $K \times 1$ vector, picking the j^{th} element from each channel of $\hat{\mathbf{X}}$, i.e., $\mathcal{V}_1(\hat{\mathbf{X}}) = [\hat{\mathbf{x}}_{11}, \hat{\mathbf{x}}_{21}, ..., \hat{\mathbf{x}}_{K1}]^T$ and $\mathcal{V}_j(\hat{\mathbf{G}}) = [\hat{\mathbf{g}}_{1j}, \hat{\mathbf{g}}_{2j}, ..., \hat{\mathbf{g}}_{Kj}]^T$. Similarly, we form, $\mathcal{V}_j(\hat{\mathbf{M}}) = \mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) - \mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{H}\mathbf{q})$, where $\mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) = [\frac{\hat{\mathbf{s}}_{1j}}{\mu}, \frac{\hat{\mathbf{s}}_{2j}}{\mu}, ..., \frac{\hat{\mathbf{s}}_{kj}}{\mu}]^T$. Solving (9), we get,

$$\mathcal{V}_j^*(\hat{\mathbf{G}}) = (\mu\mathbf{I} + \mathcal{V}_j(\hat{\mathbf{X}})\mathcal{V}_j(\hat{\mathbf{X}})^T)^{-1}(\hat{\mathbf{y}}_j\mathcal{V}_j(\hat{\mathbf{X}}) - \mu\mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) + \mu\mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{H}\mathbf{q})). \tag{6.8}$$

Equation 6.8 can be efficiently computed using the Sherman-Morrison formula (Dai *et al.*, 2019) as follows.

$$\mathcal{V}_j^*(\hat{\mathbf{G}}) = \frac{1}{\mu}\left(\mathbf{I} - \frac{\mathcal{V}_j(\hat{\mathbf{X}})\mathcal{V}_j(\hat{\mathbf{X}})^T}{\mu + \mathcal{V}_j(\hat{\mathbf{X}})^T\mathcal{V}_j(\hat{\mathbf{X}})}\right)(\hat{\mathbf{y}}_j\mathcal{V}_j(\hat{\mathbf{X}}) - \mu\mathcal{V}_j\left(\frac{\hat{\mathbf{S}}}{\mu}\right) + \mu\mathcal{V}_j(\sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{H}\mathbf{q})). \tag{6.9}$$

Solving for q

If $\hat{\mathbf{G}}, \mathbf{H}, \mathbf{w}, \mathbf{B}_w, \mathbf{B}_h$ are fixed in Equation 6.3, q_k can be computed as follows,

$$q_k^* = \underset{q_k}{\operatorname{argmin}} \frac{\mu}{2}\sum_{k=1}^{K}\left\|\hat{\mathbf{g}_k} - \sqrt{T}\mathbf{F}\mathbf{P}^T\mathbf{h}_k q_k + \frac{\hat{\mathbf{s}}_k}{\mu}\right\|_2^2 + \frac{\beta}{2}\|\mathbf{q}\|_2^2. \tag{6.10}$$

Solving Equation 6.10, we get,

$$q_k^* = \frac{\mu\sqrt{T}\mathbf{h}_k{}^T\mathbf{P}\mathbf{g}_k + T\mathbf{h}_k{}^T\mathbf{P}\mathbf{s}_\mathbf{k}}{\mu\sqrt{T}\mathbf{h}_k{}^T\mathbf{P}\mathbf{P}^T\mathbf{h}_k + \beta}. \tag{6.11}$$

Solving for w

Fixing $\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}, \mathbf{B}_w$ and $\mathbf{B}_h$ in Equation 6.3, the closed-form solution for $\mathbf{w}$ is given by,

$$\mathbf{w}^* = \frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{N}_k\mathbf{w}\|_2^2 + \frac{\lambda_2}{2}\left\|\mathbf{w} - \mathbf{w}_r - \mathbf{B}_w\right\|_2^2, \tag{6.12}$$

$$= \left(\lambda_1\sum_{k=1}^{K}\mathbf{N}_k^T\mathbf{N}_k + \lambda_2\mathbf{I}\right)^{-1}\lambda_2(\mathbf{w}_r + \mathbf{B}_w), \tag{6.13}$$

$$= \frac{\lambda_2(\mathbf{w}_r + \mathbf{B}_w)}{\lambda_1 \sum_{k=1}^{K} \mathbf{h}_k \bigodot \mathbf{h}_k + \lambda_2}, \tag{6.14}$$

where $\mathbf{N}_k = \text{diag}(\mathbf{h}_k) \in \mathbb{R}^{T \times T}$.

Solving for $\mathbf{B}_w$

In ASRCF (Dai *et al.*, 2019), the authors attempt to learn the spatial weights $\mathbf{w}$ by incorporating the term $\|\mathbf{w} - \mathbf{w}_r\|_2^2$ in the CF formulation. This term attempts to make $\mathbf{w}$ similar to a reference weight $\mathbf{w}_r$. However, due to constant changes in the target appearance and background, $\mathbf{w}$ will not be exactly similar to $\mathbf{w}_r$. To capture the minor variation between $\mathbf{w}_r$ and $\mathbf{w}$, we propose to learn a sparse difference vector $\mathbf{B}_w$. Given $\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}, \mathbf{w}$ and $\mathbf{B}_h$, the solution for $\mathbf{B}_w$ can be obtained using,

$$\mathbf{B}_w^* = \underset{\mathbf{B}_w}{\text{argmin}} \frac{\lambda_2}{2} \left\| \mathbf{w} - \mathbf{w}_r - \mathbf{B}_w \right\|_2^2 + \zeta \|\mathbf{B}_w\|_1. \tag{6.15}$$

The solution for Equation 6.15 can obtained using the Iterative Soft Thresholding algorithm (IST) (Beck and Teboulle, 2009) by

$$\mathbf{B}_w^* = \mathcal{S}_{\frac{\zeta}{\lambda_2}}(\mathbf{w} - \mathbf{w}_r). \tag{6.16}$$

Here, $\mathcal{S}_\alpha(\mathbf{z}_i) = sign(\mathbf{z}_i)\text{max}(0, |\mathbf{z}_i| - \alpha)$, is the soft-thresholding operator for a vector $\mathbf{z}$. Figure 6.2 shows a pictorial representation of $\mathbf{B}_w$ learned for consecutive ADMM updates from the sequence *Mountainbike* in the TC128 dataset (Liang *et al.*, 2015). It can be seen that $\mathbf{B}_w$ has low penalty weights for the pixels corresponding to the target. Thus, when $\mathbf{B}_w$ is added to the reference $\mathbf{w}_r$, the resultant $\mathbf{w}$ has high penalty weights near the boundary region and low penalty weights near the target.

Solving for $\mathbf{B}_h$

In STRCF (Li *et al.*, 2018*c*), the temporal regularization term, $\left\| \mathbf{h}_k^{(t)} - \mathbf{h}_k^{(t-1)} \right\|$, is used to learn a filter $\mathbf{h}_k^{(t)}$ similar to $\mathbf{h}_k^{(t-1)}$, where t is the frame index and k represents the feature channel index. However, since the target appearance changes every frame, the filter $\mathbf{h}_k^{(t)}$ will be similar to $\mathbf{h}_k^{(t-1)}$, but should also adapt to the variations in the object appearance between consecutive frames. To capture the variation between $\mathbf{h}_k^{(t)}$ and $\mathbf{h}_k^{(t-1)}$, we learn a sparse difference vector $\mathbf{B}_h$. Given $\hat{\mathbf{G}}, \mathbf{H}, \mathbf{q}, \mathbf{w}$ and $\mathbf{B}_w$, the solution

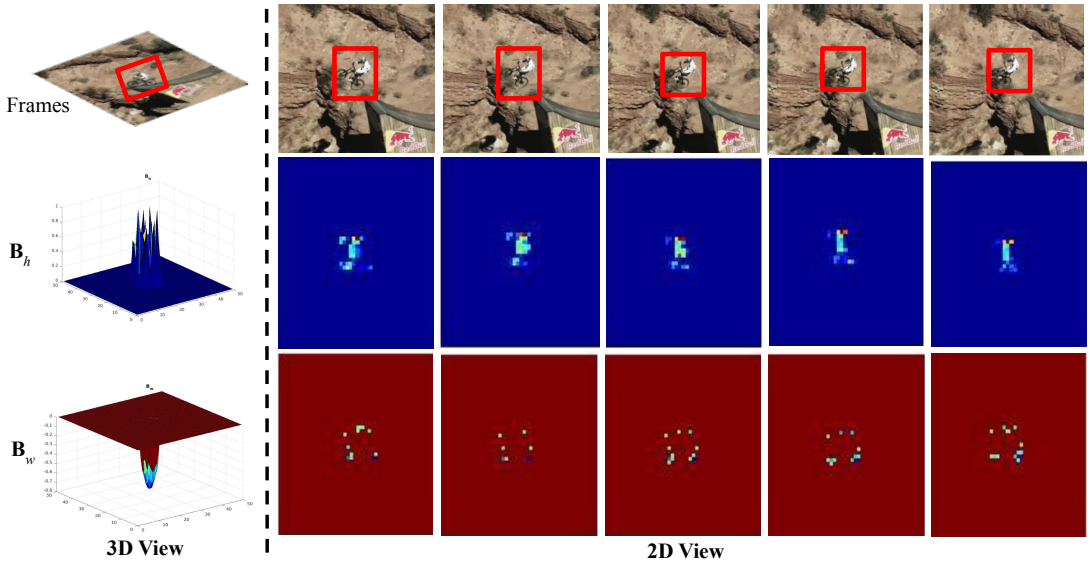


Figure 6.2: Pictorial representation of $\mathbf{B}_w$ and $\mathbf{B}_h$ learned for consecutive ADMM updates from the sequence *Mountainbike* (Frame# 46, 48, 50, 52 and 54) of the TC128 dataset (Liang *et al.*, 2015). $\mathbf{B}_h$ is normalized between [0 1] for display purpose

for $\mathbf{B}_h$ can be obtained using,

$$\mathbf{B}_h^* = \underset{\mathbf{B}_h}{\operatorname{argmin}} \frac{\theta}{2} \left\| \mathbf{h}_k^{(t)} - \mathbf{h}_k^{(t-1)} - \mathbf{B}_h \right\|_2^2 + \eta \|\mathbf{B}_h\|_1. \tag{6.17}$$

The solution for Equation 6.17 can obtained using IST (Beck and Teboulle, 2009) by

$$\mathbf{B}_h^* = \mathcal{S}_{\frac{\eta}{\theta}}(\mathbf{h}_k^{(t)} - \mathbf{h}_k^{(t-1)}). \tag{6.18}$$

Here, $\mathcal{S}$ is the soft-thresholding operator. Figure 6.2 shows a pictorial representation of $\mathbf{B}_h$ learned for consecutive ADMM updates. It can be seen that $\mathbf{B}_h$ is non-zero for the pixels corresponding to the target and zero for the background. Thus, when $\mathbf{B}_h$ is added to $\mathbf{h}^{(t-1)}$, the resultant $\mathbf{h}^{(t)}$ contains refined filter coefficients in the target region.

6.2.1 Lagrangian Multiplier Update

The Lagrangian multipliers are updated using,

$$\mu^{(t+1)} = min(\mu_{max}, \nu\mu^{(t)}), \tag{6.19}$$

$$\hat{\mathbf{S}}^{(t+1)} = \hat{\mathbf{S}}^{(t)} + \mu^{(t+1)}(\hat{\mathbf{G}}^{(t+1)} - \hat{\mathbf{H}}^{(t+1)}), \tag{6.20}$$

$$\lambda_2^{(t+1)} = \rho\lambda_2^{(t)}, \tag{6.21}$$

$$\theta^{(t+1)} = \rho\theta^{(t)}, \tag{6.22}$$

where $\rho > 1$, $\hat{\mathbf{H}}^{(t+1)}$ and $\hat{\mathbf{G}}^{(t+1)}$ are the current solutions to $\hat{\mathbf{H}}$ and $\hat{\mathbf{G}}$ respectively, and $\hat{\mathbf{S}}^{(t)}$ is the Fourier transform of the Lagrangian variable in the previous state. Thus, the optimal filter $\mathbf{H}^*$, feature channel weight q_k^*, spatial weight $\mathbf{w}^*$ and the sparse difference components, $\mathbf{B}_w^*$ and $\mathbf{B}_h^*$, can be obtained by iteratively solving for $\mathbf{H}$, $\mathbf{G}$, q_k, $\mathbf{w}$, $\mathbf{B}_w$ and $\mathbf{B}_h$ followed by the Lagrangian update, until convergence.

6.2.2 Target Localization

The target location is determined using,

$$\hat{\mathbf{r}} = \sum_{k=1}^{K} \hat{\mathbf{x}}_\mathbf{k} \odot \hat{\mathbf{g}}_\mathbf{k}, \tag{6.23}$$

where $\hat{\mathbf{r}}$ is the response map in the Fourier domain. The location at which $\hat{\mathbf{r}}$ shows the maximum value is used to estimate the target location. For target scale estimation, we follow the same strategy as Dai *et al.* (2019).

6.2.3 Model Update

In order to adjust to target appearance variations, we use an online adaptive template scheme (Bertinetto *et al.*, 2016*a*; Bolme *et al.*, 2010; Zhang and Suganthan, 2017) to update the template model,

$$\hat{\mathbf{X}}_{model}^{(t)} = (1-\omega)\hat{\mathbf{X}}_{model}^{(t-1)} + \omega\hat{\mathbf{X}}^{(t)}, \tag{6.24}$$

where ω is the online learning rate, $\hat{\mathbf{X}}^{(t)}$ is the current observation, $\hat{\mathbf{X}}_{model}^{(t-1)}$ is the old template model and $\hat{\mathbf{X}}_{model}^{(t)}$ is the updated template model. To introduce a reasonable prior for adaptive spatial regularization, the reference spatial weights are updated using $\mathbf{w}_r \leftarrow \mathbf{w}^*$. In the first frame, $\mathbf{w}_r$ is initialized with a negative Gaussian shape (Dai *et al.*, 2019; Danelljan *et al.*, 2015*b*). The above update schemes ensure our model is adaptive to target appearance variations during tracking.

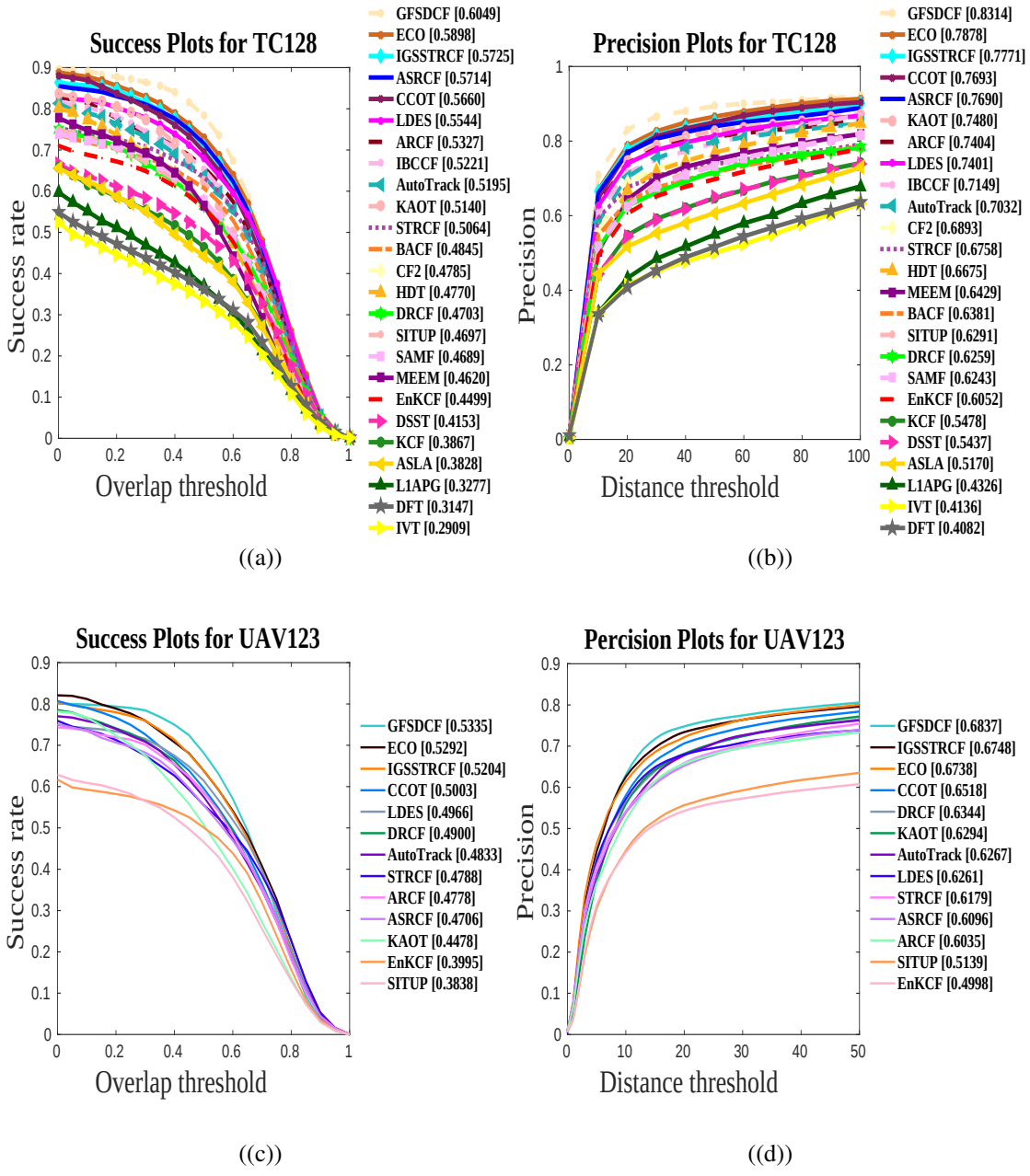


Figure 6.3: Success and Precision plots for TC128 (Liang *et al.*, 2015) ((a) and (b)) and UAV123 dataset (Mueller *et al.*, 2016) ((c) and (d)), with trackers arranged in descending order of their performance. The legend of the precision plots contains the scores at a threshold of 20 pixels and the legend of the success plots contains Area-Under-the-Curve scores for each tracker

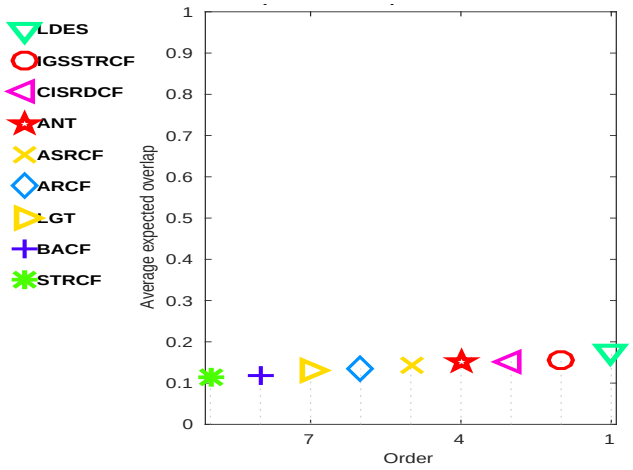


Figure 6.4: Expected overlap scores for the baseline experiments on VOT-2019 Kristan *et al.* (2019), showing the that proposed IGSSTRCF tracker performs the second best

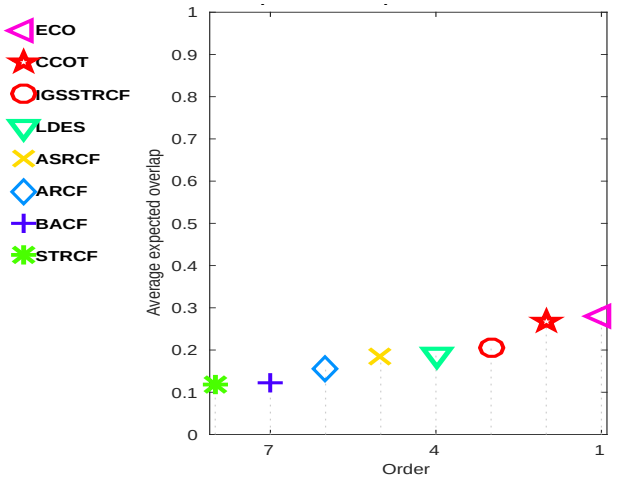


Figure 6.5: Expected overlap scores for the baseline experiments on VOT-2017 Kristan *et al.* (2017*b*), showing the that proposed IGSSTRCF tracker outperforms several state-of-the-art trackers

6.3 Experiments

This section provides implementation details and presents the performance analysis of the proposed tracker on four benchmark datasets: TC128 (Liang *et al.*, 2015), UAV123 (Mueller *et al.*, 2016), VOT-2019 (Kristan *et al.*, 2019) and VOT-2017 (Kristan *et al.*, 2017*b*), in comparison to state-of-the-art trackers. The section ends with an extensive ablation study examining the contribution of each regularization component of the proposed IGSSTRCF tracker.

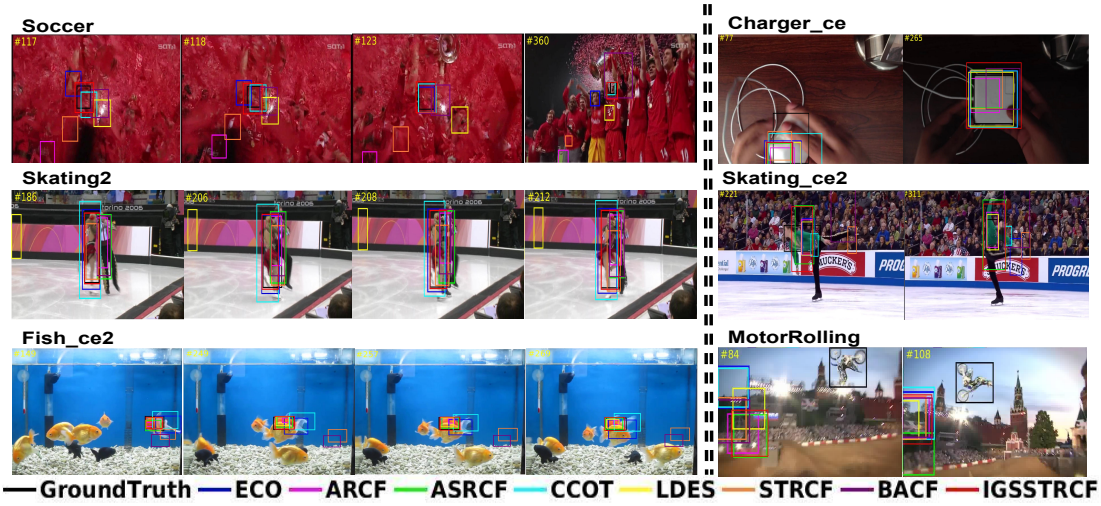


Figure 6.6: Intermediate frames showing examples of successfully tracked frames (left) and failure cases (right) in different sequences from the TC128 dataset (Liang *et al.*, 2015)

6.3.1 Implementation Details

All proposed formulations are implemented using MATLAB2019a with the MatConvNet toolbox. We use an ensemble of features extracted from *Norm1* of VGG-M, *Conv4-3* of VGG-16 (Simonyan and Zisserman, 2014) and HOG features to represent and localize the target. In Equation 6.1, the parameters θ and λ_2 are 1.2, λ_1 and β are 0.01, and η and ζ are 0.001. ρ in Equation 6.21 is 1.5 and the learning rate, ω, in Equation 6.24 is 0.0186. ν in Equation 6.19 is 10 and μ_{max} is 10^4. The initial value of the ADMM penality factor, μ, is set to 1. The ADMM is updated every 2 frames. The value of each parameter is selected empirically.

6.3.2 Performance Analysis

We present an extensive evaluation of IGSSTRCF on four challenging tracking benchmarks with a tracking speed of 8 frames per second. For TC128 (Liang *et al.*, 2015) and UAV123 (Mueller *et al.*, 2016) datasets, we report a one-pass evaluation with distance precision and overlap success plots. For the VOT datasets (Kristan *et al.*, 2017*b*, 2019), we use the benchmark protocol to evaluate the tracker in terms of Expected Average Overlap (EAO), Accuracy (A) and Robustness (R) for the baseline experiments, and overlap Area-Under-the-Curve (AUC) for the unsupervised experiments (Kristan *et al.*, 2017*b*). For VOT datasets, the expected overlap curves, scores, unsupervised overlap AUC, and A-R analysis for individual challenges are included in the supplementary material.

Evaluation on TC128 Dataset

The proposed tracker is evaluated on TC128 (Liang *et al.*, 2015). Figure 6.3 (a) and (b) shows the success and precision plots comparing the proposed IGSSTRCF tracker with recent trackers: GFSDCF (Xu *et al.*, 2019*a*), ECO (Danelljan *et al.*, 2016*a*), AS-RCF (Dai *et al.*, 2019), IBCCF (Li *et al.*, 2017*a*), AutoTrack (Li *et al.*, 2020*b*), CCOT (Danelljan *et al.*, 2016*d*), LDES (Li *et al.*, 2019*b*), ARCF (Huang *et al.*, 2019*b*), STRCF (Li *et al.*, 2018*c*), BACF (Kiani Galoogahi *et al.*, 2017), KAOT (Li *et al.*, 2020*c*), CF2 (Ma *et al.*, 2015*a*), HDT (Qi *et al.*, 2016), DRCF (Fu *et al.*, 2020), SITUP (Ma *et al.*, 2020), SAMF (Li and Zhu, 2014), MEEM (Zhang *et al.*, 2014*a*), EnKCF (Uzkent and Seo, 2018), DSST (Danelljan *et al.*, 2014*a*), KCF (Henriques *et al.*, 2014), ASLA (Jia *et al.*, 2012), L1APG (Bao *et al.*, 2012), DFT (Sevilla-Lara and Learned-Miller, 2012) and IVT (Ross *et al.*, 2008). The proposed IGSSTRCF outperforms all the compared trackers in terms of overlap success and distance precision, except for GFSDCF (Xu *et al.*, 2019*a*) and ECO (Danelljan *et al.*, 2016*a*).

Evaluation on UAV123 Dataset

The proposed tracker is evaluated on UAV123 dataset (Mueller *et al.*, 2016). Figure 6.3 (c) and (d) shows the success and precision plots comparing the proposed IGSSTRCF tracker with recent trackers: GFSDCF (Xu *et al.*, 2019*a*), ECO (Danelljan *et al.*, 2016*a*), CCOT (Danelljan *et al.*, 2016*d*), DRCF (Fu *et al.*, 2020), KAOT (Li *et al.*, 2020*c*), AutoTrack (Li *et al.*, 2020*b*), LDES (Li *et al.*, 2019*b*), STRCF (Li *et al.*, 2018*c*) ARCF (Huang *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), SITUP (Ma *et al.*, 2020) and EnKCF (Uzkent and Seo, 2018). The proposed IGSSTRCF tracker is second best in terms of precision and third best in terms of success.

Evaluation on VOT-2019 Dataset

The proposed IGSSTRCF tracker is evaluated using the VOT toolkit on VOT-2019 (Kristan *et al.*, 2019). A comparison is shown with the recent state of the art trackers: ASRCF (Dai *et al.*, 2019), STRCF (Li *et al.*, 2018*c*), LDES (Li *et al.*, 2019*b*), ARCF (Huang *et al.*, 2019*b*), BACF (Kiani Galoogahi *et al.*, 2017), CISRDCF (Kristan *et al.*, 2019), ANT (Kristan *et al.*, 2019), LGT (Kristan *et al.*, 2019), FoT (Kristan *et al.*, 2019), MIL (Kristan *et al.*, 2019), KCF (Kristan *et al.*, 2019), Struck (Kristan *et al.*, 2019), IVT (Kristan *et al.*, 2019) and, L1APG (Kristan *et al.*, 2019). Table 6.1

Table 6.1: VOT toolkit report for VOT-2019 showing Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) for the baseline experiment and Overlap AUC for the unsupervised experiment. The top three trackers are shown in **red**, **blue** and **green**

	Baseline			Unsupervised
	EAO	**A**	**R**	**AUC**
IGSSTRCF	**0.1559**	**0.4730**	**39.0094**	**0.3286**
ASRCF (Dai *et al.*, 2019)	0.1451	0.4652	**44.5818**	**0.3230**
STRCF (Li *et al.*, 2018*c*)	0.1140	0.4520	70.02	**0.2980**
LDES (Li *et al.*, 2019*b*)	**0.1747**	**0.4882**	50.2721	0.2940
ARCF (Huang *et al.*, 2019*b*)	0.1351	**0.4669**	52.7181	0.2690
BACF (Kiani Galoogahi *et al.*, 2017)	0.1162	0.4476	65.7094	0.1959
CISRDCF (Kristan *et al.*, 2019)	**0.1533**	0.4147	**48.9861**	0.2417
ANT (Kristan *et al.*, 2019)	0.1509	0.4518	53.0936	0.2390
LGT (Kristan *et al.*, 2019)	0.1308	0.3960	54.8683	0.2062
FoT (Kristan *et al.*, 2019)	0.1290	0.3621	70.4328	0.1354
MIL (Kristan *et al.*, 2019)	0.1179	0.3847	73.6540	0.1664
KCF (Kristan *et al.*, 2019)	0.1103	0.4348	73.0953	0.2059
Struck (Kristan *et al.*, 2019)	0.0944	0.4103	96.3228	0.1743
IVT (Kristan *et al.*, 2019)	0.0869	0.3811	117.7786	0.1095
L1APG (Kristan *et al.*, 2019)	0.0774	0.3901	147.7737	0.1224

shows the accuracy, robustness and EAO for the baseline experiments (Kristan *et al.*, 2017*b*). It is observed that IGSSTRCF performs best in terms of robustness, and second best in terms of EAO and accuracy. The expected overlap scores for the baseline experiments are shown in Figure 6.4. For the unsupervised experiments (Kristan *et al.*, 2017*b*), Table 6.1 shows the overlap AUC. It is observed that IGSSTRCF performs best in the overlap criterion.

Evaluation on VOT-2017 Dataset

The proposed tracker is evaluated using the VOT toolkit on VOT-2017 (Kristan *et al.*, 2017*b*). A comparison is shown with recent trackers: ASRCF (Dai *et al.*, 2019), STRCF (Li *et al.*, 2018*c*), LDES (Li *et al.*, 2019*b*), ARCF (Huang *et al.*, 2019*b*), BACF (Kiani Galoogahi *et al.*, 2017), ECO (Danelljan *et al.*, 2017*a*), CCOT (Danelljan *et al.*, 2016*d*), SRDCF (Danelljan *et al.*, 2015*b*), ANT (Kristan *et al.*, 2017*b*), BST (Kristan *et al.*, 2017*b*), CGS (Kristan *et al.*, 2017*b*), ATLAS (Kristan *et al.*, 2017*b*), and GMD (Kristan *et al.*, 2017*b*). Table 6.2 shows the accuracy, robustness and EAO for baseline experiments (Kristan *et al.*, 2017*b*). It is observed that IGSSTRCF is third best in terms of robustness and EAO. The expected overlap scores for the baseline experiments are shown in Figure 6.5. For the unsupervised experiments, Table 6.2 shows the overlap AUC. It is observed that the proposed tracker is third best in the overlap criterion.

Table 6.2: VOT toolkit report for VOT-2017 showing Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) for the baseline experiment and Overlap AUC for the unsupervised experiment. The top three trackers are shown in **red**, **blue** and **green**

	Baseline			Unsupervised
	A	**R**	**EAO**	**AUC**
IGSSTRCF	0.4761	**26.2841**	**0.2040**	**0.3539**
ASRCF (Dai *et al.*, 2019)	0.4654	30.9708	0.1851	0.3411
STRCF (Li *et al.*, 2018*c*)	0.4510	61.3300	0.1180	0.3000
LDES (Li *et al.*, 2019*b*)	**0.4929**	39.6484	0.1875	0.3237
ARCF (Huang *et al.*, 2019*b*)	0.4615	41.4174	0.1547	0.2824
BACF (Kiani Galoogahi *et al.*, 2017)	0.4476	55.7769	0.1235	0.2083
ECO (Danelljan *et al.*, 2016*a*)	0.4762	**17.6628**	**0.2809**	**0.4025**
CCOT (Danelljan *et al.*, 2016*d*)	**0.4851**	**20.4138**	**0.2674**	**0.3909**
SRDCF (Danelljan *et al.*, 2015*b*)	0.4767	64.1136	0.1179	0.2445
ANT (Kristan *et al.*, 2017*b*)	0.4540	40.1593	0.1676	0.2770
BST (Kristan *et al.*, 2017*b*)	0.2627	55.5033	0.1150	0.1458
CGS (Kristan *et al.*, 2017*b*)	**0.4979**	53.3758	0.1406	0.3386
ATLAS (Kristan *et al.*, 2017*b*)	0.4835	37.4268	0.1969	0.3431
GMD (Kristan *et al.*, 2017*b*)	0.4422	54.7325	0.1295	0.2492

Table 6.3: Ablation analysis showing contribution of each regularization component of the proposed IGSSTRCF tracker in terms of Overlap Score (OP), Success (S) and Precision (P). The best performance is shown in **red**

	VOT-2019	VOT-2017	TC128	
	OS	**OS**	**S**	**P**
IGSSTRCF	**0.155**	**0.204**	**57.25**	**77.71**
IGSSTRCF - SSR	0.152	0.195	56.84	77.35
IGSSTRCF - CI	0.127	0.150	55.85	76.04
IGSSTRCF - STR	0.151	0.178	55.78	76.78
IGSSTRCF - SSR - CI	0.152	0.196	56.35	76.51
IGSSTRCF - STR - CI	0.153	0.193	56.68	77.30
IGSSTRCF - STR - SSR	0.149	0.171	54.25	72.74

Discussion and Qualitative Evaluation

Evaluation across multiple tracking datasets shows that the proposed IGSSTRCF tracker outperforms most of the recent trackers on TC128, UAV123, VOT-2017 and VOT-2019. It is observed that the proposed IGSSTRCF performs comparable or worse than GFS-DCF and ECO on TC128 and UAV123. This is because, as explained in Section 5.3.9, TC128 and UAV123 are complex datasets and contains more challenges with longer sequences that includes sequences with a small, low resolution target object as compared to VOT-2019.

To demonstrate the performance qualitatively, we present examples of success and failure for some tracking sequences of TC128 dataset (Liang *et al.*, 2015). The proposed

IGSSTRCF tracker is compared with ECO (Danelljan *et al.*, 2017*a*), ARCF (Huang *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), CCOT (Danelljan *et al.*, 2016*d*), LDES (Li *et al.*, 2019*b*), STRCF (Li *et al.*, 2018*c*) and BACF (Kiani Galoogahi *et al.*, 2017). Figure 6.6 (left) shows frames from the sequences *soccer*, *Skating2*, and *Fish_ce2*, where the IGSSTRCF tracks successfully during background clutter, similar object presence, and multi-object presence, while most other trackers fail. Figure 6.6 (right) shows frames from the sequences *Charger_ce*, *skating_ce2*, and *MotorRolling*, where most of the trackers, including IGSSTRCF, fail during scale change, object deformation, and in-plane-rotation.

6.3.3 Ablation Study

Evaluation of Individual Regularization Component

To demonstrate the contribution of each regularization component of the proposed IGSSTRCF, we remove the regularization terms from the IGSSTRCF formulation one at a time, and evaluate the performance. Table 6.3 shows a comparison of the complete IGSSTRCF formulation with formulations without the regularizations. In Table 6.3, column 1, "IGSSTRCF - *regularization term*" denotes the IGSSTRCF tracker without the stated *regularization term*. The comparison is shown on VOT-2019 (Kristan *et al.*, 2019) and VOT-2017 (Kristan *et al.*, 2017*b*) in terms of overlap score for baseline experiments, and on TC128 (Liang *et al.*, 2015) in terms of success plot AUC and precision score at a threshold of 20 pixels. It is observed that the proposed IGSSTRCF works best with the SSR, STR and CI terms all included.

Table 6.4: Ablation analysis on VOT-2019 Kristan *et al.* (2019), VOT-2017 Kristan *et al.* (2017*b*), TC128 Liang *et al.* (2015) and UAV123 Mueller *et al.* (2016) to demonstrate comparison of the regularized version with its respective baselines. For VOT datasets, we report overlap score for the baseline experiments. For TC128 and UAV123, we report overlap success plot AUC and distance precision score at a threshold of 20 pixel. The best performing method is shown in red color

	VOT-2019	VOT-2017	TC128		UAV123	
	Overlap Score	Overlap Score	Success	Precision	Success	Precision
BACF Kiani Galoogahi *et al.* (2017)	0.12	0.12	48.45	63.81	40.01	45.20
BACF Kiani Galoogahi *et al.* (2017) + CI	0.16	0.18	57.50	78.29	52.38	67.44
STRCF Li *et al.* (2018*c*)	0.11	0.12	50.64	67.58	47.88	61.79
STRCF Li *et al.* (2018*c*) + STR	0.13	0.15	56.11	76.46	51.30	65.79
ASRCF Dai *et al.* (2019)	0.14	0.18	57.14	76.90	47.06	60.96
ASRCF Dai *et al.* (2019) + SSR	0.15	0.19	56.68	77.30	52.05	67.60

Comparison with the Baselines

To demonstrate the performance of individual regularization terms independently, we add the regularization to the baseline tracker that is closest to the regularized version. The details are given as follows.

The Channel importance (CI) term is added to the baseline BACF Kiani Galoogahi *et al.* (2017). Equation 6.25 shows the BACF Kiani Galoogahi *et al.* (2017) with the CI term (denoted 'BACF Kiani Galoogahi *et al.* (2017) + CI'). The black text in Equation 6.25 is the original BACF Kiani Galoogahi *et al.* (2017) formulation.

$$E(\mathbf{h}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} \overbrace{q_k}^{\text{CI}} (\mathbf{x}_k * (\mathbf{P}^T \mathbf{h}_k))\right\|_2^2 + \frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{h}_k\|_2^2 + \underbrace{\frac{\beta}{2}\|\mathbf{q}\|_2^2}_{\text{CI}}, \tag{6.25}$$

The Sparse Temporal Regularization (STR) is added to the baseline STRCF Li *et al.* (2018*c*). Equation 6.3.3 shows the STRCF Li *et al.* (2018*c*) with the STR term (denoted 'STRCF Li *et al.* (2018*c*) + STR'). The black text in 6.3.3 is the original STRCF Li *et al.* (2018*c*) formulation.

$$E(\mathbf{h}, \mathbf{w}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} (\mathbf{x}_k * \mathbf{h}_k)\right\|_2^2 + \frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{w} \odot \mathbf{h}_k\|_2^2 + \underbrace{\frac{\theta}{2}\left\|\mathbf{h}^{(t)} - \mathbf{h}^{(t-1)} - \mathbf{B}_h\right\|_2^2 + \eta\|\mathbf{B}_h\|_1}_{\text{STR}}, \tag{6.26}$$

Similarly, the Sparse Spatial Regularization (SSR) is added to the baseline ASRCF Dai *et al.* (2019). Equation shows the ASRCF Dai *et al.* (2019) with the SSR term (denoted 'ASRCF Dai *et al.* (2019) + SSR'). The black text in Equation 6.3.3 is the original ASRCF Dai *et al.* (2019) formulation.

$$E(\mathbf{h}, \mathbf{w}) = \frac{1}{2}\left\|\mathbf{y} - \sum_{k=1}^{K} (\mathbf{x}_k * (\mathbf{P}^T \mathbf{h}_k))\right\|_2^2 + \frac{\lambda_1}{2}\sum_{k=1}^{K}\|\mathbf{w} \odot \mathbf{h}_k\|_2^2 + \underbrace{\frac{\lambda_2}{2}\left\|\mathbf{w} - \mathbf{w}_r - \mathbf{B}_w\right\|_2^2 + \zeta\|\mathbf{B}_w\|_1}_{\text{SSR}}, \tag{6.27}$$

Table 6.4 shows comparison of the regularized version with their respective baselines on VOT-2019 Kristan *et al.* (2019), VOT-2017 Kristan *et al.* (2017*b*), TC128 Liang *et al.* (2015) and UAV123 Mueller *et al.* (2016). It is observed that adding the regularization

results in improving the baseline performance in all the cases, except for success in TC128 dataset.

6.4 Chapter Summary

In this work, we propose a novel importance guided sparse spatio-temporal regularization based CF tracker. The sparse spatial regularization learns the spatial weights by modelling the sparse difference between the current spatial weights and the reference spatial weights. The learned spatial weights are used to penalize the filter coefficients near the boundary region to prevent boundary effects. The sparse temporal regularization models the sparse difference between the current filter and the past reference filter. The sparse difference term helps filter out irrelevant information from the previous filter, while learning the current filter. This prevents irrelevant appearance information from persisting through further tracking steps. We also propose to learn the adaptive importance weights for each feature channel during training. This helps in suppressing the contribution of adverse feature channels and enhancing the contribution of useful feature channels. As a result of the proposed regularizations, a more discriminative and robust CF is trained, that achieves efficient object tracking during multiple challenges. We use ADMM to efficiently obtain an optimal solution for the proposed formulation. The positive effects of the proposed formulation are demonstrated on the TC128 (Liang *et al.*, 2015), UAV123 (Mueller *et al.*, 2016), VOT-2019 (Kristan *et al.*, 2019) and VOT-2017 (Kristan *et al.*, 2017*b*) datasets. A comparative analysis with recent top ranked trackers reveals that the proposed approach outperforms many state-of-the-art trackers.

In the next chapter, we explore kernel tricks to further improve the tracker performance in terms of speed as well as tracking accuracy. The regularized CF based trackers introduced in Chapters 5 and 6 are not used in combination with the kernel model (Henriques *et al.*, 2014) as the constraints may break the circulant matrix structure, making it computationally expensive. In Chapter 7, we propose the Temporally Regularized Multi-Kernel Correlation Filter (TRM-KCF) formulation for tracking. We also explore how different kernels enhance different aspects of the target appearance, and demonstrate the performance advantages of using multiple-kernels.

CHAPTER 7

TRM-KCF: Temporally Regularized Multi-Kernel Correlation Filters For Visual Tracking

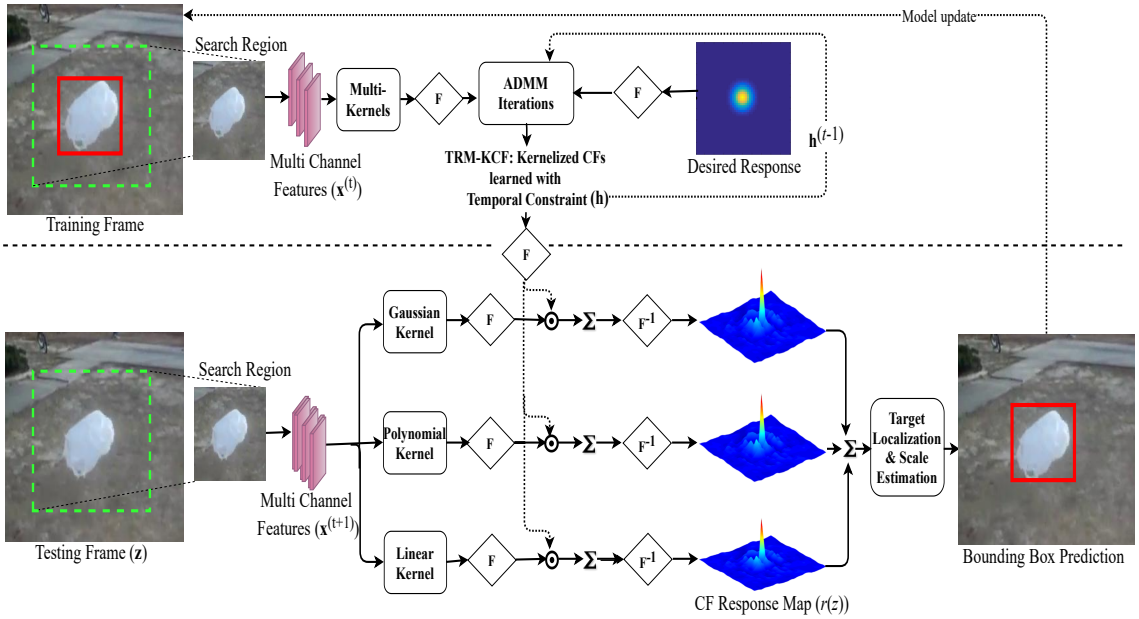


Figure 7.1: A block diagram for the proposed TRM-KCF tracker. During training, the filter $\mathbf{h}$ is learned using ADMM iterations (Boyd *et al.*, 2011). During testing, we extract the deep features from the search area. The target is localized using the average of response maps obtained by multi-kernel filters. For target scale estimation, we follow Dai *et al.* (2019). $\mathbf{F}$ and $\mathbf{F}^{-1}$ denotes Fourier and inverse Fourier transform operations respectively

7.1 Introduction

In the recent research, Correlation Filter (CF) trackers with spatial and temporal constraints have been proposed (Danelljan *et al.*, 2015*b*; Kiani Galoogahi *et al.*, 2017; Dai *et al.*, 2019; Li *et al.*, 2018*c*), including the trackers investigated in Chapters 5 and 6. The limitation of such methods is that they are not used in combination with the kernel model (Henriques *et al.*, 2014) as the constraints may break the circulant matrix structure, making it computationally expensive.

Recently, kernel tricks have gained popularity with correlation filter trackers. Henriques *et al.* (2012) uses high dimensional features with kernels and act as a baseline

for many kernel based trackers. It is further improved by using HOG features in the Kernelized Correlation Filter (KCF) (Henriques *et al.*, 2014).To improve the discriminative ability of the KCFs, Danelljan *et al.* (2014*b*) use color attributes to learn adaptive CFs for tracking and maps the multi-channel features into a Gaussian kernel space. Scale adaptive KCF trackers are introduced in Li and Zhu (2014) (SAMF) and Bibi and Ghanem (2015) (KCFMSTA), and an ensemble of KCF tracker is proposed in Uzkent and Seo (2018) (EnKCF). A kernel based structured output correlation tracker is proposed in Hare *et al.* (2015) (Struck). Zhang *et al.* (2016) introduces an output constraint transfer for the KCF (OCT-KCF). Tang and Feng (2015) extends KCF (Henriques *et al.*, 2014) to multiple kernels that enhance the model's distinguishability with the help of complementary features. A further improvement to this work is proposed in Tang *et al.* (2018) by taking advantage of the discriminative power spectrums of different features (MKCFup).

Although computationally efficient, KCF based trackers are not derived with spatial and temporal constraints as the constraints may not allow the straight forward use of the circulant matrix structure (Huang *et al.*, 2020) (CMKCF). To this end, Huang *et al.* (2020) introduces a multi-kernel CF with spatial constraints that handles occlusion efficiently. Inspired by the above trackers, this chapter extends the KCF (Henriques *et al.*, 2014) formulation to introduce the Temporally Regularized Multi-Kernel Correlation Filter (TRM-KCF) formulation for tracking. We argue that different kernels enhance different aspects of the target appearance, and demonstrate the performance advantages of using multiple-kernels.

This chapter extends the KCF (Henriques *et al.*, 2014) formulation to introduce the Temporally Regularized Multi-Kernel Correlation Filter (TRM-KCF) formulation for tracking. We argue that different kernels enhance different aspects of the target appearance, and demonstrate the performance advantages of using multiple-kernels. Figure 7.1 shows a block diagram that describes the process flow of the proposed TRM-KCF for tracking.

The summarized contributions of the chapter are as follows:

1. We derive a Temporally Regularized Multi-Kernel Correlation Filter (TRM-KCF) formulation for object tracking.

2. We demonstrate the performance of the proposed CF formulation with multiple kernels (Gaussian, Polynomial and Linear) as well as a single kernel (TR-KCF) to show the advantages of using the proposed multi-kernel (TRM-KCF) formulation.

3. We present an extensive evaluation of the proposed tracker on the publicly available tracking datasets: OTB100 (Wu *et al.*, 2015), TC128 (Liang *et al.*, 2015), VOT-2017 (Kristan *et al.*, 2017*b*), LaSOT (Fan *et al.*, 2019), UAV123 (Mueller *et al.*, 2016), and GOT-10k (Huang *et al.*, 2019*a*). A comparative analysis shows that the proposed formulation results in a significant improvement over the baseline tracker and other recent trackers.

7.2 Proposed Approach

Although computationally efficient, Kernelized Correlation Filte (KCF) based trackers are not derived with spatial and temporal constraints as the constraints may not allow the straight forward use of the circulant matrix structure (Huang *et al.*, 2020). To this end, Huang *et al.* (2020) introduces a multi-kernel correlation filter with spatial constraints that handles occlusion efficiently. Inspired by the above trackers, we derive a Temporally Regularized Multi-Kernel Correlation Filter (TRM-KCF) for tracking. The temporal constraint helps in obtaining more reliable filter coefficients for improved tracking and ensures that the tracker adapts to large appearance variations, preventing drift. Despite the constraints, we can kernelize and exploit the circulant matrix properties to speed up computations. In this section, we detail how temporal regularization can be incorporated into the KCF (Henriques *et al.*, 2014) formulation. The baseline KCF formulation is already discussed in Chapter 2, Section 2.3. The proposed TRM-KCF formulation is given by,

$$E(\mathbf{h}) = \frac{1}{2}\sum_{m=1}^{T} \left\| y_m - \mathbf{h}^\top \mathbf{x}_m \right\|_2^2 + \frac{\lambda}{2}\|\mathbf{h}\|_2^2 + \frac{\theta}{2}\left\| \mathbf{h}^{(t)} - \mathbf{h}^{(t-1)} \right\|_2^2, \tag{7.1}$$

where $\mathbf{x} \in \mathbb{R}^{T\times 1}$ are the vectorized features, $\mathbf{x}_m \in \mathbb{R}^{T\times 1}$ is the sample obtained after m cyclic shifts of $\mathbf{x}$, T denotes the size of the training sample $\mathbf{x}$, $\mathbf{y} \in \mathbb{R}^{T\times 1}$ is the desired Gaussian shaped correlation filter response, y_i is the *i*-th element of $\mathbf{y}$, $\mathbf{h} \in \mathbb{R}^{T\times 1}$ is the vectorized filter, $\frac{\theta}{2}\left\| \mathbf{h}^{(t)} - \mathbf{h}^{(t-1)} \right\|_2^2$ is the temporal regularization term, $\mathbf{h}^{(t)}$ and $\mathbf{h}^{(t-1)} \in \mathbb{R}^{T\times 1}$ are the vectorized filters at frame t and $(t-1)$, respectively. λ and θ are the regularization parameters. We can express each solution of $\mathbf{h}$ as the linear combination of the inputs, $\mathbf{h} = \sum_{m=1}^{T} \boldsymbol{\psi}_m \phi(\mathbf{x}_m)$. Therefore $\mathbf{h}^\top \mathbf{x}_n$ can be expressed as,

$$\mathbf{h}^\top \mathbf{x}_n = \sum_{m=1}^{T} \boldsymbol{\psi}_m \phi(\mathbf{x}_m)^\top \phi(\mathbf{x}_n). \tag{7.2}$$

Using Equation 7.2, Equation 7.1 can be re-written as,

$$E(\boldsymbol{\psi}) = \frac{1}{2}\left\|\mathbf{y} - \boldsymbol{\Omega}\boldsymbol{\psi}^{(t)}\right\|_2^2 + \frac{\lambda}{2}\left\|\sum_{m=1}^{T}\boldsymbol{\psi}_m^{(t)}\phi(\mathbf{x}_m^{(t)})\right\|_2^2 +$$

$$\frac{\theta}{2}\left\|\sum_{m=1}^{T}\boldsymbol{\psi}_m^{(t)}\phi(\mathbf{x}_m^{(t)}) - \sum_{m=1}^{T}\boldsymbol{\psi}_m^{(t-1)}\phi(\mathbf{x}_m^{(t-1)})\right\|_2^2, \tag{7.3}$$

where $\boldsymbol{\Omega}$ is given by,

$$\boldsymbol{\Omega} = \begin{bmatrix} (\phi(\mathbf{x}_1))^\top\phi(\mathbf{x}_1) & \cdots & (\phi(\mathbf{x}_1))^\top\phi(\mathbf{x}_n) \\ \vdots & \ddots & \vdots \\ (\phi(\mathbf{x}_m))^\top\phi(\mathbf{x}_1) & \cdots & (\phi(\mathbf{x}_m))^\top\phi(\mathbf{x}_n) \end{bmatrix} \tag{7.4}$$

Equation 7.3 can be expanded as,

$$E(\boldsymbol{\psi}) = \frac{1}{2}(\mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\boldsymbol{\Omega}\boldsymbol{\psi}^{(t)} - (\boldsymbol{\psi}^{(t)})^\top\boldsymbol{\Omega}^\top\mathbf{y} + (\boldsymbol{\psi}^{(t)})^\top\boldsymbol{\Omega}^\top\boldsymbol{\Omega}(\boldsymbol{\psi}^{(t)}))+$$

$$\frac{\lambda}{2}(\boldsymbol{\psi}^{(t)})^\top\boldsymbol{\Omega}(\boldsymbol{\psi}^{(t)}) + \frac{\theta}{2}((\boldsymbol{\psi}^{(t)})^\top\boldsymbol{\Omega}(\boldsymbol{\psi}^{(t)}) - 2(\boldsymbol{\psi}^{(t)})^\top\widetilde{\boldsymbol{\Omega}}\boldsymbol{\psi}^{(t-1)}+$$

$$\boldsymbol{\psi}^{(t-1)^\top}\boldsymbol{\Omega}^{(t-1)}\boldsymbol{\psi}^{(t-1)}), \tag{7.5}$$

where $\widetilde{\boldsymbol{\Omega}}$ is the kernel matrix formed using $\mathbf{x}^{(t)}$ and $\mathbf{x}^{(t-1)}$.

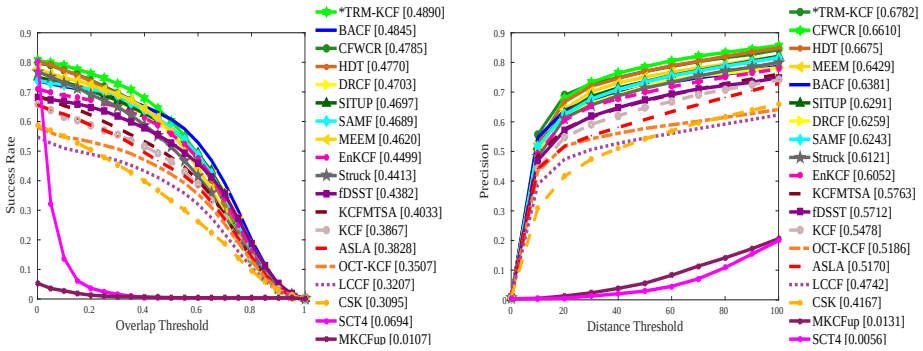


Figure 7.2: Success and Precision plots for TC128. The legend of the success plots contains Area-Under-the-Curve scores and the legend of the precision plots contains the precision scores at 20 pixels. The trackers are arranged in descending order of their performance

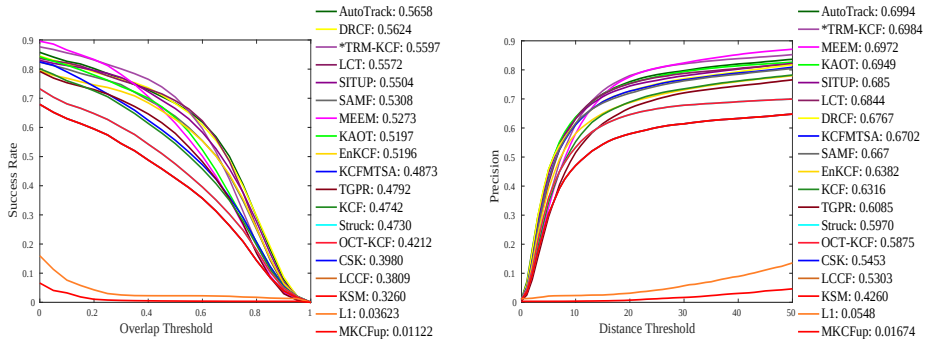


Figure 7.3: Success and Precision plots for OTB100. The legend of the success plots contains Area-Under-the-Curve scores and the legend of the precision plots contains the precision scores at 20 pixels. The trackers are arranged in descending order of their performance

Differentiating Equation 7.5 with respect to $\boldsymbol{\psi}^{(t)}$ and equating to zero, we obtain,

$$(\boldsymbol{\Omega}^\top \boldsymbol{\Omega}\boldsymbol{\psi}^{(t)} - \boldsymbol{\Omega}^\top \mathbf{y}) + \lambda \boldsymbol{\Omega}\boldsymbol{\psi}^{(t)} + \theta(\boldsymbol{\Omega}\boldsymbol{\psi}^{(t)} + \widetilde{\boldsymbol{\Omega}}^\top \boldsymbol{\psi}^{(t-1)}) = 0. \tag{7.6}$$

The solution for $\boldsymbol{\psi}^{(t)}$ can be obtained as,

$$\boldsymbol{\psi}^{(t)} = (\boldsymbol{\Omega} + (\lambda + \theta)\mathbf{I})^{-1}(\mathbf{y} - \theta \boldsymbol{\Omega}^{-1}\widetilde{\boldsymbol{\Omega}}^\top \boldsymbol{\psi}^{(t-1)}). \tag{7.7}$$

Here, kernel matrices $\boldsymbol{\Omega}$ and $\widetilde{\boldsymbol{\Omega}}$ are circulant, therefore $\boldsymbol{\Delta} = (\boldsymbol{\Omega}^{-1}\widetilde{\boldsymbol{\Omega}}^\top)$ is also circulant. Equation 7.7 can be re-written as,

$$\boldsymbol{\psi}^{(t)} = (C(\boldsymbol{\Omega}_1) + (\lambda + \theta)\mathbf{I})^{-1}(\mathbf{y} - \theta C(\boldsymbol{\Delta}_1)\boldsymbol{\psi}^{(t-1)}), \tag{7.8}$$

where $C(\mathbf{u})$ is a circulant matrix generated using circularly shifted samples of $\mathbf{u}$, $\boldsymbol{\Delta}_1$ denotes the first row of the circulant matrix $\boldsymbol{\Delta}$, and $\boldsymbol{\Omega}_1$ is the first row of $\boldsymbol{\Omega}$. Using the theorem $C(\mathbf{u})\mathbf{v} = \mathbf{F}^{-1}(\mathbf{F}^*(\mathbf{u}) \circ \mathbf{F}(\mathbf{v}))$ (Huang *et al.*, 2020), Equation 7.8 becomes,

$$\boldsymbol{\psi}^{(t)} = (C(\boldsymbol{\Omega}_1) + (\lambda + \theta)\mathbf{I})^{-1}(\mathbf{y} - \theta \mathbf{F}^{-1}(\mathbf{F}^*(\boldsymbol{\Delta}_1) \circ \mathbf{F}(\boldsymbol{\psi}^{(t-1)})). \tag{7.9}$$

Equation 7.9 can be re-written as,

$$\boldsymbol{\psi}^{(t)} = (C(\boldsymbol{\Omega}_1) + (\lambda + \theta)\mathbf{I})^{-1}\mathbf{F}^{-1}(\mathbf{F}(\mathbf{y}) - \theta(\mathbf{F}^*(\boldsymbol{\Delta}_1) \circ \mathbf{F}(\boldsymbol{\psi}^{(t-1)})), \tag{7.10}$$

which can be solved by,

$$\boldsymbol{\psi}^{(t)} = \mathbf{F}^{-1}\left(\frac{\mathbf{F}(\mathbf{y}) - \theta(\mathbf{F}^{*}(\boldsymbol{\Delta}_1) \circ \mathbf{F}(\boldsymbol{\psi}^{(t-1)}))}{F(\boldsymbol{\Omega}_1) + (\lambda + \theta)\mathbf{I}}\right). \tag{7.11}$$

The computational cost of solving Equation 7.11 is $\mathcal{O}(TlogT + T)$. Using Equation 7.11, we compute $\boldsymbol{\psi}$ for three different kernels $\boldsymbol{\psi}_{Gaussian}$, $\boldsymbol{\psi}_{Polynomial}$ and $\boldsymbol{\psi}_{Linear}$ for Gaussian, Polynomial and Linear kernels respectively. The filter response map corresponding to each kernel type is obtained using Equation 2.12 (see Chapter 2, Section 2.3) and are denoted $\mathbf{r}_{Gaussian}(\mathbf{z})$, $\mathbf{r}_{Polynomial}(\mathbf{z})$ and $\mathbf{r}_{Linear}(\mathbf{z})$. The final response map for the proposed TRM-KCF is obtained using,

$$\mathbf{r}(\mathbf{z}) = \frac{\mathbf{r}_{Gaussian}(\mathbf{z}) + \mathbf{r}_{Polynomial}(\mathbf{z}) + \mathbf{r}_{Linear}(\mathbf{z})}{3}. \tag{7.12}$$

The response map for TR-KCF is obtained using $\mathbf{r}_{Gaussian}(\mathbf{z})$ only. The location at which the final response map shows the maximum value is used to estimate the target location. The model is updated using,

$$\mathbf{x}^{(t)}_{model} = (1 - \eta)\mathbf{x}^{(t-1)}_{model} + \eta\mathbf{x}^{(t)}, \tag{7.13}$$

$$\mathbf{F}(\boldsymbol{\psi}^{(t)}_{Gaussian}) = (1 - \eta)\mathbf{F}(\boldsymbol{\psi}^{(t-1)}_{Gaussian}) + \eta\mathbf{F}(\boldsymbol{\psi}_{Gaussian}), \tag{7.14}$$

$$\mathbf{F}(\boldsymbol{\psi}^{(t)}_{Polynomial}) = (1 - \eta)\mathbf{F}(\boldsymbol{\psi}^{(t-1)}_{Polynomial}) + \eta\mathbf{F}(\boldsymbol{\psi}_{Polynomial}), \tag{7.15}$$

$$\mathbf{F}(\boldsymbol{\psi}^{(t)}_{Linear}) = (1 - \eta)\mathbf{F}(\boldsymbol{\psi}^{(t-1)}_{Linear}) + \eta\mathbf{F}(\boldsymbol{\psi}_{Linear}). \tag{7.16}$$

7.3 Experiments

The proposed formulation is implemented using MATLAB2019a with the MatConvNet toolbox. The deep features are extracted from the *conv*5-4 convolutional layer of VGG-Net-19 (Simonyan and Zisserman, 2014) and are used to represent and localize the target. The parameter λ in Equation 7.11 is set to 0.5, and the parameter θ is 0.01. The

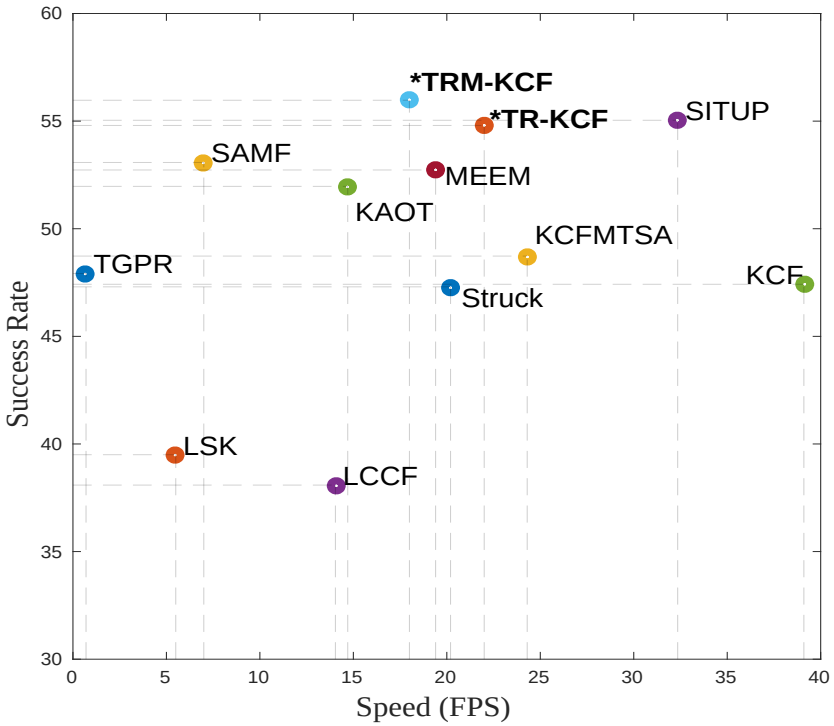


Figure 7.4: Success versus Speed plots for OTB100.

model learning rate, η, is 0.0186. All parameters are obtained empirically. We achieve the tracking speed of 18 frames per second for TRM-KCF and 22 frames per second for TR-KCF. The evaluation of the proposed formulations on the TC128 (Liang *et al.*, 2015), OTB100 (Wu *et al.*, 2015), VOT-2017 (Kristan *et al.*, 2017*b*), LaSOT (Fan *et al.*, 2019), UAV123 (Mueller *et al.*, 2016), and GOT-10k (Huang *et al.*, 2019*a*) datasets are presented in the subsequent sections.

7.3.1 Evaluation on TC128

Figure 7.2 shows the success and precision plots comparing the proposed trackers with recent trackers: BACF (Kiani Galoogahi *et al.*, 2017), CFWCR (He *et al.*, 2017), HDT (Qi *et al.*, 2016), DRCF (Fu *et al.*, 2020), SITUP (Ma *et al.*, 2020), SAMF (Li and Zhu, 2014), MEEM (Zhang *et al.*, 2014*a*), EnKCF (Uzkent and Seo, 2018), Struck (Hare *et al.*, 2015), fDSST (Danelljan *et al.*, 2016*b*), KCFMTSA (Bibi and Ghanem, 2015), KCF (Henriques *et al.*, 2014), ASLA (Jia *et al.*, 2012), OCT-KCF (Zhang *et al.*, 2016), LCCF (Zhang *et al.*, 2017*a*), CSK (Henriques *et al.*, 2012), SCT4 (Choi *et al.*, 2016), and MKCFup (Tang *et al.*, 2018). Among the compared trackers, TRM-KCF ranks first in terms of success and precision scores.

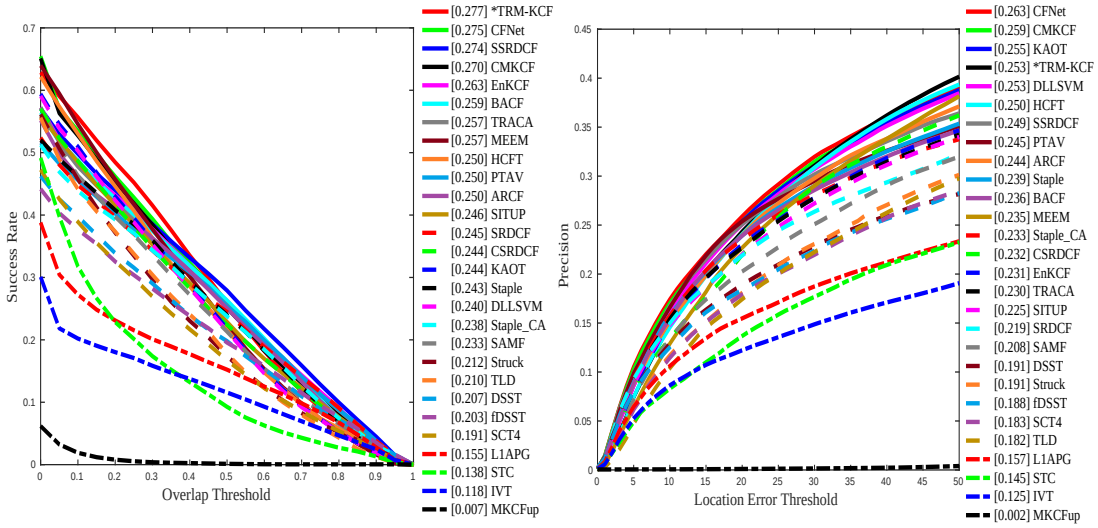


Figure 7.5: Success and Precision plots for LaSOT. The legend of the success plots contains Area-Under-the-Curve scores and the legend of the precision plots contains the precision scores at 20 pixels. The trackers are arranged in descending order of their performance

7.3.2 Evaluation on OTB100

Figure 7.3 shows the success and precision plots comparing the proposed trackers with recent trackers: AutoTrack (Li *et al.*, 2020*a*), DRCF (Fu *et al.*, 2020), LCT (Ma *et al.*, 2015*b*), SITUP (Ma *et al.*, 2020), SAMF (Li and Zhu, 2014), MEEM (Zhang *et al.*, 2014*a*), KAOT (Li *et al.*, 2020*c*), EnKCF (Uzkent and Seo, 2018), KCFMTSA (Bibi and Ghanem, 2015), TGPR (Gao *et al.*, 2014), KCF (Henriques *et al.*, 2014), Struck (Hare *et al.*, 2015), OCT-KCF (Zhang *et al.*, 2016), CSK (Henriques *et al.*, 2012), LCCF (Zhang *et al.*, 2017*a*), KSM (Comaniciu *et al.*, 2003), L1 (Bao *et al.*, 2012), and MKCFup (Tang *et al.*, 2018).

Among the compared trackers, TRM-KCF ranks third in terms of success score and second in terms of precision score. Figure 7.4 shows a success versus speed plot where closer the tracker to the the top right corner, the better is its performance. The comparison between SITUP (Ma *et al.*, 2020), SAMF (Li and Zhu, 2014), MEEM (Zhang *et al.*, 2014*a*), KAOT (Li *et al.*, 2020*c*), KCFMTSA (Bibi and Ghanem, 2015), TGPR (Gao *et al.*, 2014), KCF (Henriques *et al.*, 2014), Struck (Hare *et al.*, 2015), LCCF (Zhang *et al.*, 2017*a*), KSM (Comaniciu *et al.*, 2003), LSK (Liu *et al.*, 2011) and the proposed tracker shows that TRM-KCF has higher success rate and is faster than many recent trackers.

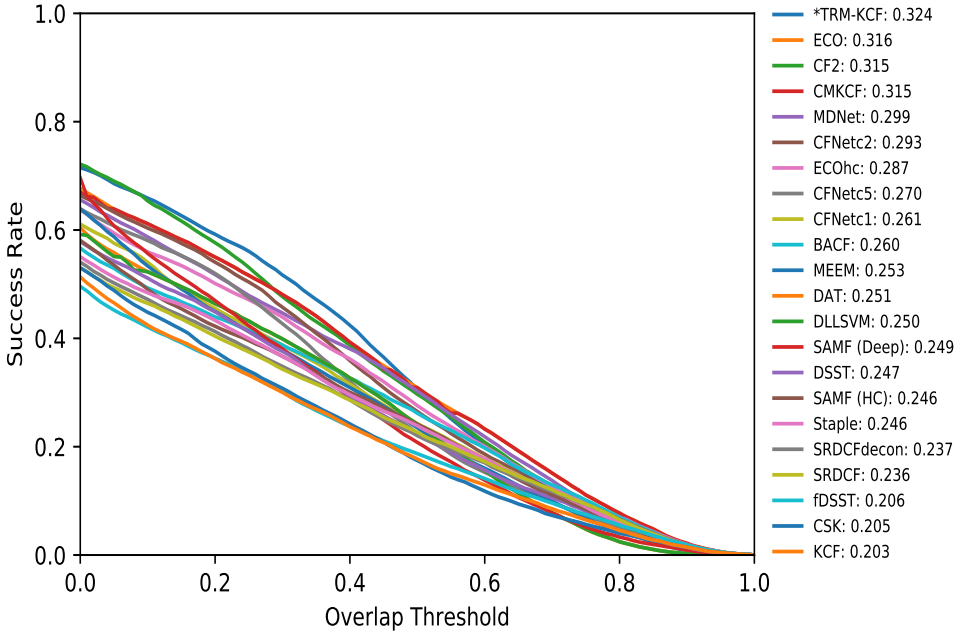


Figure 7.6: Success plots for GOT-10k. The legend of the success plots contains Area-Under-the-Curve scores. The trackers are arranged in descending order of their performance

7.3.3 Evaluation on LaSOT

Figure 7.5 shows the success and precision plots comparing the proposed trackers on LaSOT (Fan *et al.*, 2019) with recent trackers: CFNet (Valmadre *et al.*, 2017*a*), SS-RDCF (Guo *et al.*, 2019), EnKCF (Uzkent and Seo, 2018), BACF (Kiani Galoogahi *et al.*, 2017), TRACA (Choi *et al.*, 2018), MEEM (Zhang *et al.*, 2014*a*), HCFT (Qi *et al.*, 2016), PTAV (Fan and Ling, 2017*a*), ARCF (Huang *et al.*, 2019*b*), SITUP (Ma *et al.*, 2020), SRDCF (Danelljan *et al.*, 2015*b*), CSRDCF (Lukezic *et al.*, 2017), KAOT (Li *et al.*, 2020*c*), Staple (Bertinetto *et al.*, 2016*a*), Staple_CA (Mueller *et al.*, 2017), SAMF (Li and Zhu, 2014), Struck (Hare *et al.*, 2015), TLD (Kalal *et al.*, 2011), DSST (Danelljan *et al.*, 2014*a*), fDSST (Danelljan *et al.*, 2016*b*), SCT4 (Choi *et al.*, 2016), L1APG (Bao *et al.*, 2012), STC (Zhang *et al.*, 2014*b*), IVT (Ross *et al.*, 2008), and MKCFup (Tang *et al.*, 2018). It is observed that the proposed TRM-KCF ranks first in terms of success and third in terms of precision score.

7.3.4 Evaluation on GOT-10k

Figure 7.6 shows the success plots comparing the proposed trackers on GOT-10k (Huang *et al.*, 2019*a*) with recent trackers: ECO (Danelljan *et al.*, 2017*a*), CF2 (Ma *et al.*, 2015*a*), MDNet (Nam and Han, 2016), CFNet (Valmadre *et al.*, 2017*a*), ECO_hc (Danelljan *et al.*, 2017*a*), BACF (Kiani Galoogahi *et al.*, 2017), MEEM (Zhang *et al.*, 2014*a*),

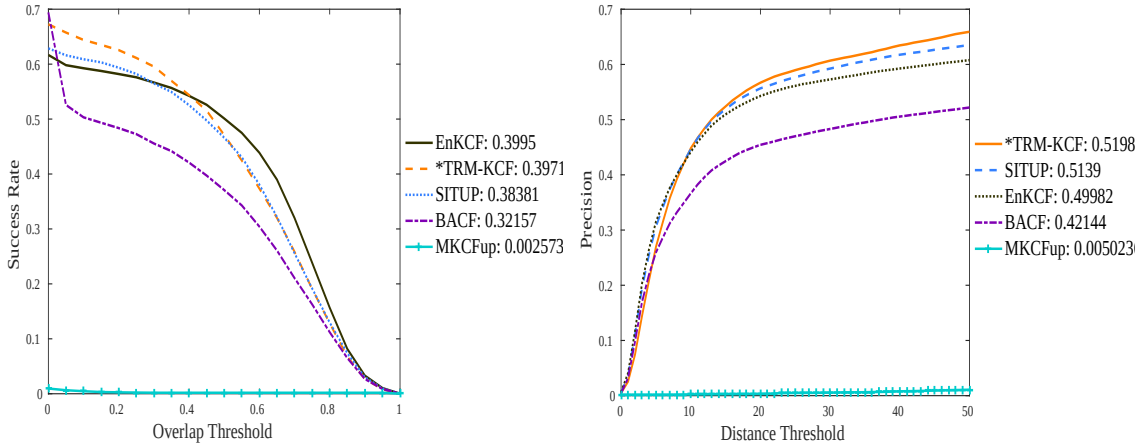


Figure 7.7: Success and Precision plots for UAV123. The legend of the success plots contains Area-Under-the-Curve scores and the legend of the precision plots contains the precision scores at 20 pixels. The trackers are arranged in descending order of their performance

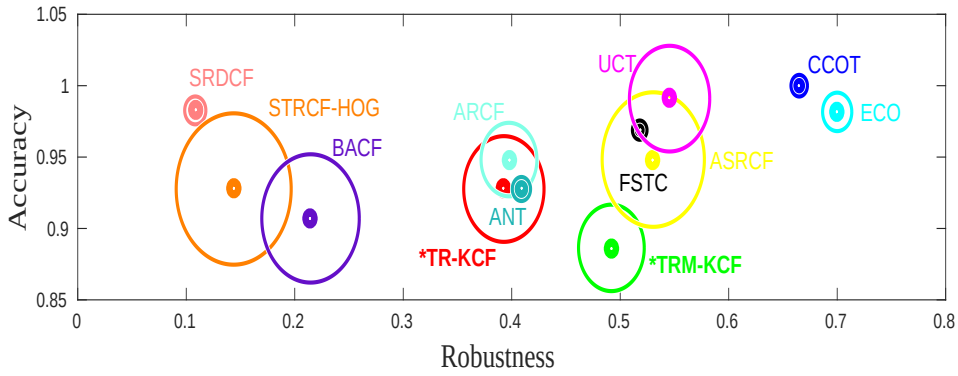


Figure 7.8: Accuracy versus Robustness versus Speed plots for VOT-2017. Here the size of each circle is directly proportional to the speed of tracker.

DAT (Pu *et al.*, 2018), SAMF (Li and Zhu, 2014), DSST (Danelljan *et al.*, 2014*a*), Staple (Bertinetto *et al.*, 2016*a*), SRDCFdecon (Danelljan *et al.*, 2016*c*), SRDCF (Danelljan *et al.*, 2015*b*), fDSST (Danelljan *et al.*, 2016*b*), CSK (Henriques *et al.*, 2012), and KCF(Henriques *et al.*, 2014). The proposed TRM-KCF ranks first in terms of success score.

7.3.5 Evaluation on UAV123

Figure 7.7 shows the success and precision plots comparing the proposed trackers on UAV123 (Mueller *et al.*, 2016) with recent trackers: BACF (Kiani Galoogahi *et al.*, 2017), SITUP (Ma *et al.*, 2020), EnKCF (Uzkent and Seo, 2018), and MKCFup (Tang *et al.*, 2018). It is observed that the proposed TRM-KCF ranks second in terms of success and ranks first in terms of precision score.

Table 7.1: VOT toolkit report for VOT-2017 Kristan *et al.* (2017*b*) showing Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) for baseline experiment, and Overlap AUC for unsupervised experiment. The top three trackers are shown in **red**, **blue** and **green**

	Baseline			**Unsupervised**
	AR-Rank			
	A	**R**	**EAO**	**Overlap**
TRM-KCF	0.43	**34.00**	**0.16**	0.31
BACF Kiani Galoogahi *et al.* (2017)	0.44	55.77	0.12	0.20
STRCF-HOG Li *et al.* (2018*c*)	0.45	61.33	0.11	0.30
ASRCF Dai *et al.* (2019)	0.46	**30.97**	**0.18**	**0.34**
ARCF Huang *et al.* (2019*b*)	0.46	41.41	0.15	0.28
LDES Li *et al.* (2019*b*)	**0.49**	39.64	**0.18**	**0.32**
SRDCF Danelljan *et al.* (2015*b*)	**0.47**	64.11	0.11	0.24
ANT Kristan *et al.* (2017*b*)	0.45	40.15	**0.16**	0.27
BST Kristan *et al.* (2017*b*)	0.26	55.50	0.11	0.14
GMD Kristan *et al.* (2017*b*)	0.44	54.73	0.12	0.24
DSST Kristan *et al.* (2017*b*)	0.40	63.07	0.10	0.15
FragTrack Kristan *et al.* (2017*b*)	0.38	68.27	0.09	0.16
IVT Kristan *et al.* (2017*b*)	0.39	66.22	0.10	0.12
L1APG Kristan *et al.* (2017*b*)	0.42	81.05	0.10	0.15
MIL Kristan *et al.* (2017*b*)	0.39	46.05	0.14	0.16
CMKCF Kristan *et al.* (2017*b*)	**0.50**	**23.18**	**0.24**	**0.40**
DLLSVM Kristan *et al.* (2017*b*)	0.37	63.01	0.11	0.14

7.3.6 Evaluation on VOT-2017

In this section we present a comparison of the proposed trackers with recent existing trackers using the VOT official toolkit. To evaluate a tracker, the toolkit applies a reset-based methodology, where the performance is measured in terms of Accuracy (A) and Robustness (R) (Čehovin *et al.*, 2016*b*). We evaluate the trackers using an Accuracy-Robustness (AR) plot in which a tracker is more accurate if it is higher along the vertical axis and is more robust if it is further to the right on the horizontal axis (Čehovin *et al.*, 2014). An additional evaluation measure is Expected Average Overlap (EAO), which estimates the average overlap a tracker is expected to attain on a large collection of short term sequences with the same visual properties in the given dataset. This measure is used to address the increased variance and bias of the average overlap measure (Wu *et al.*, 2015), when evaluating over variable sequence lengths.

The results obtained for the baseline (reset based) and unsupervised (no-reset) experiments (Kristan *et al.*, 2017*b*) are given as follows.

- The Baseline Experiment:

  Table 7.1 shows the EAO, Accuracy (A) and Robustness (R) of the proposed TRM-KCF compared to the existing trackers submitted to the VOT-2017 challenge (Kristan *et al.*, 2017*b*) and other recent trackers ARCF (Huang *et al.*, 2019*b*), LDES (Li *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), BACF (Kiani Galoogahi *et al.*, 2017) and STRCF-HOG (Li *et al.*, 2018*c*). It is observed that the proposed tracker

Table 7.2: Robustness comparison of the proposed tracker with recent trackers for the baseline experiment on VOT-2017 dataset. Here CM = Camera Motion, EMP = Empty Tag, IV= Illumination Variation, MC = Motion Change, OCC = Occlusion and SV = Size Variation. The top three trackers are shown in **red**, **blue** and **green**.

	CM	EMP	IV	MC	OCC	SV	Mean	Weighted Mean	Average Pooled
TRM-KCF	**53.00**	**30.00**	**3.00**	**29.00**	**27.00**	**16.00**	**26.33**	**34.00**	**123.00**
BACF (Kiani Galoogahi *et al.*, 2017)	77.00	61.00	6.00	43.00	31.00	33.00	41.83	55.77	189.00
STRCF-HOG (Li *et al.*, 2018*c*)	85.00	68.00	9.00	**46.00**	39.00	31.00	46.33	61.33	198.00
ASRCF (Dai *et al.*, 2019)	**45.00**	**26.00**	6.00	**29.00**	**23.00**	**23.00**	**25.33**	**30.97**	**105.00**
ARCF (Huang *et al.*, 2019*b*)	**53.00**	49.00	8.00	**29.00**	34.00	**20.00**	32.16	41.41	141.00
LDES (Li *et al.*, 2019*b*)	**47.00**	46.00	10.00	**31.00**	31.00	27.00	**32.00**	**39.64**	**133.00**
SRDCF (Danelljan *et al.*, 2015*b*)	76.00	86.00	9.00	49.00	32.00	29.00	46.83	64.11	208.00
ANT (Kristan *et al.*, 2017*b*)	64.00	**26.00**	8.00	45.00	27.00	30.00	33.33	40.15	135.00
BST (Kristan *et al.*, 2017*b*)	74.73	66.93	**4.86**	41.93	**24.66**	26.73	39.97	55.50	188.60
GMD (Kristan *et al.*, 2017*b*)	86.06	50.33	**5.40**	47.66	30.73	26.20	41.06	54.73	187.46

is second best in terms of R and EAO. Table 7.2 shows the robustness comparison for challenges like Camera Motion (CM), Illumination Variation (IV), Motion Change (MC), Occlusion (OCC) and Scale Variation (SV). It is observed that the proposed tracker lies in top three for all the challenges. Figure 7.8 shows an accuracy versus robustness versus speed plot where closer the tracker to the the top right corner, the better is its performance. The size of the of each circle indicates the tracking speed of respective tracker. The comparison between trackers submitted in the VOT-2017 challenge (Kristan *et al.*, 2017*b*) and other recent trackers ECO (Danelljan *et al.*, 2017*a*), CCOT (Danelljan *et al.*, 2016*d*), ARCF (Huang *et al.*, 2019*b*), LDES (Li *et al.*, 2019*b*), ASRCF (Dai *et al.*, 2019), BACF (Kiani Galoogahi *et al.*, 2017), STRCF-HOG (Li *et al.*, 2018*c*) and the proposed tracker shows that TRM-KCF is faster and better than many recent trackers.

- The Unsupervised Experiment:

  Table 7.1 shows the Area Under Curve (AUC) of the average overlap plot of the proposed trackers compared with existing trackers. It is observed that the Proposed TRM-KCF tracker ranks third in terms of Overlap Curve AUC. Table 7.3 show overlap comparison for various challenges. It is observed that the proposed tracker ranks second in CM, IV and OCC, and ranks third in EMP, MC and overall performance.

7.3.7 Discussion and Qualitative Evaluation

The proposed formulations is evaluated on the TC128 (Liang *et al.*, 2015), OTB100 (Wu *et al.*, 2015), VOT-2017 (Kristan *et al.*, 2017*b*), LaSOT (Fan *et al.*, 2019), UAV123 (Mueller *et al.*, 2016), and GOT-10k (Huang *et al.*, 2019*a*) datasets. It is observed that the proposed TRM-KCF outperforms all the compared trackers on the fairly complex TC128, LaSOT and GOT-10k datasets. The tracker also demonstrate competitive performance on OTB100, UAV123 and VOT-2017 with tracking speed of 22 frames per second.

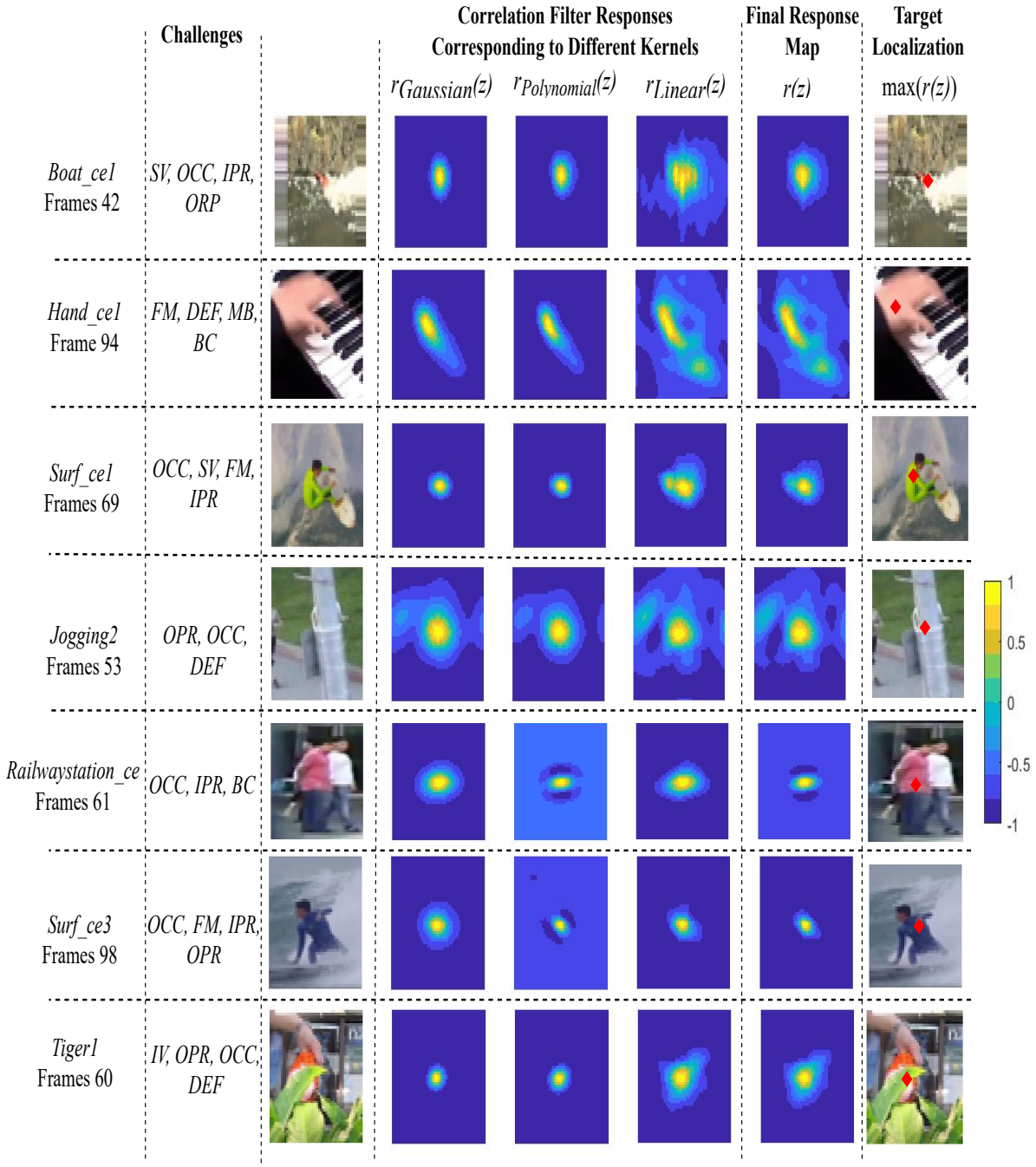


Figure 7.9: Examples of correlation filter responses obtained corresponding different type of kernels for intermediate frames of TC128 with various challenges. Each row shows the responses obtained corresponding to Gaussian, Polynomial and Linear kernel along with the final response obtained using the mean of all the responses. Here SV = Scale Variation, OCC = Occlusion, IPR = In Plane Rotation, OPR = Out of Plane Rotation, FM = Fast Motion, DEF = Deformation, MB = Motion Blur, BC = Background Clutter, and IV = Illumination Variation.

Table 7.3: Overlap comparison of the proposed tracker with recent trackers for the unsupervised experiment on VOT-2017 dataset. Here CM = Camera Motion, EMP = Empty Tag, IV= Illumination Variation, MC = Motion Change, OCC = Occlusion and SV = Size Variation. The top three trackers are shown in **red**, **blue** and **green**.

	CM	EMP	IV	MC	OCC	SV	Overall
TRM-KCF	**0.3379**	**0.3249**	**0.3272**	**0.2863**	**0.2295**	0.2402	**0.3099**
BACF (Kiani Galoogahi *et al.*, 2017)	0.2196	0.2204	0.2649	0.2021	0.1809	0.1472	0.2083
STRCF-HOG (Li *et al.*, 2018*c*)	0.3046	0.3183	0.2971	0.2708	0.1884	**0.3142**	0.3006
ASRCF (Dai *et al.*, 2019)	**0.3583**	**0.3419**	**0.3083**	**0.3452**	**0.2511**	**0.3401**	**0.3411**
ARCF (Huang *et al.*, 2019*b*)	0.3055	0.2735	0.2781	0.2836	0.1989	**0.3046**	0.2824
LDES (Li *et al.*, 2019*b*)	**0.3302**	**0.3405**	**0.3648**	**0.2988**	0.1985	0.2929	**0.3237**
SRDCF (Danelljan *et al.*, 2015*b*)	0.2606	0.2230	0.2901	0.2417	0.1990	0.2534	0.2445
ANT (Kristan *et al.*, 2017*b*)	0.3036	0.2681	0.1910	0.2822	**0.2047**	0.2909	0.2770
BST (Kristan *et al.*, 2017*b*)	0.1411	0.1678	0.1720	0.1067	0.1194	0.1391	0.1458
GMD (Kristan *et al.*, 2017*b*)	0.2707	0.2486	0.1932	0.2262	0.1683	0.2886	0.2492

Table 7.4: Ablation study showing the performance of the proposed tracker with multiple-kernels (TRM-KCF) and with single kernel (TR-KCF)

		TR-KCF	**TRM-KCF**	KCF (Henriques *et al.*, 2014)
TC128	Success(%)	0.4740	**0.4890**	0.3867
	Precision(%)	0.6719	**0.6782**	0.5478
OTB100	Success(%)	0.5480	**0.5597**	0.4742
	Precision(%)	0.6916	**0.6984**	0.6316
UAV123	Success(%)	**0.3980**	0.3971	0.2651
	Precision(%)	**0.5223**	0.5198	0.4052
VOT-2017	Accuracy	**0.45**	0.43	0.42
	Robustness	41.78	**34.00**	77.30
	EAO	0.14	**0.16**	0.13
	Overlap	**0.32**	0.31	0.26

Figure 7.9 shows examples of the correlation filter responses obtained corresponding to different types of kernels for intermediate frames in sequences from TC128 (Liang *et al.*, 2015). Each row shows the responses obtained corresponding to Gaussian, Polynomial and Linear kernels along with the final response obtained using the mean of all the responses. It is observed that different response maps are obtained for different types of kernel and individual response maps may also contain high values in background regions. Such response maps, when used individually may result in tracker drift during target localization. Therefore, in our proposed multi-kernel approach, we localize the target using the response map obtained after combining the response maps from all kernels. Figure 7.10 shows intermediate tracking frames of sequences from TC128 (Liang *et al.*, 2015). The top two rows show the successfully tracked frames in the sequences *Eagle_ce*, *Fish_ce1*, *Toyplane_ce*, *Skiing*, *Ball_ce2*, *Hand_ce1*, *Microphone_ce1*, and *Hand_ce2*, that contain challenges like Scale Variation (SV), Occlusion

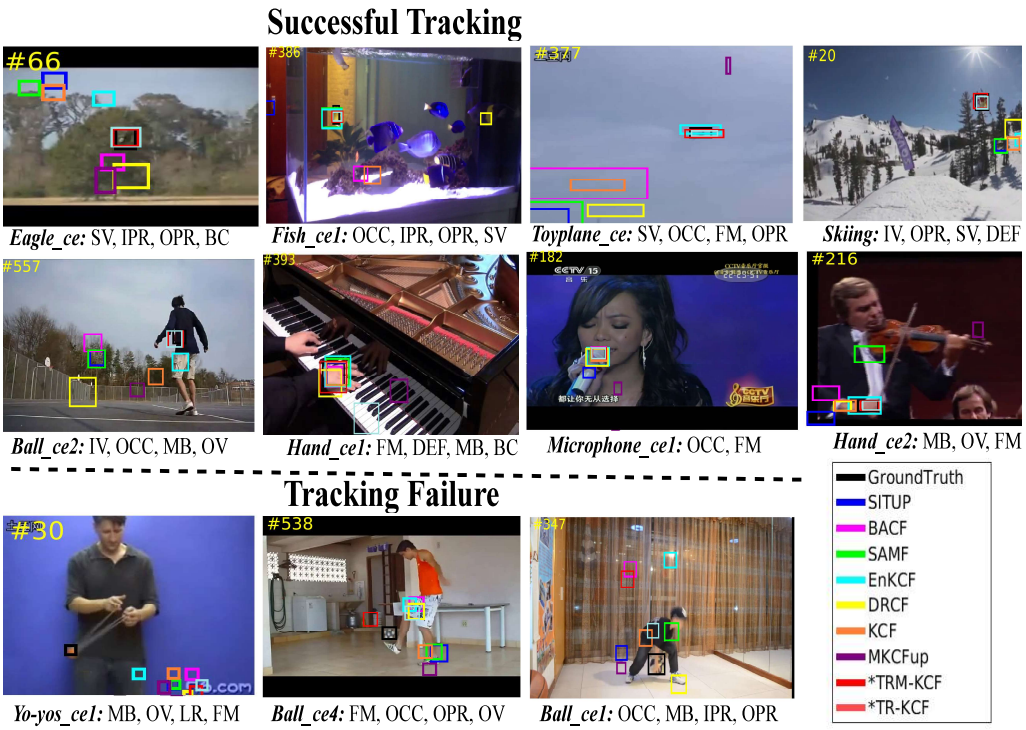


Figure 7.10: Intermediate Frames of Sequences from TC128 Showing Tracker's Performance. Here SV = Scale Variation, OCC = Occlusion, IPR = In Plane Rotation, OPR = Out of Plane Rotation, FM = Fast Motion, DEF = Deformation, MB = Motion Blur, BC = Background Clutter, IV = Illumination Variation, OV = Out of View, and LR = Low Resolution.

(OCC), In Plane Rotation (IPR), Out of Plane Rotation (OPR), Fast Motion (FM), Deformation (DEF), Motion Blur (MB), Background Clutter (BC), and Illumination Variation (IV). Row 3 shows the frames from sequences *Yo-yos_ce1*, *Ball_ce4*, and *Ball_ce1*. The proposed tracker drifts in *Yo-yos_ce1* due to the small target size where the deep features may not be able to encode all the necessary appearance cues resulting in a less discriminative filter. This is the potential reason for sub-standard performance of the proposed TRM-KCF on UAV123 as most sequences in UAV123 contain a small target. The drift in *Ball_ce4* and *Ball_ce1* occurs due to a combination of challenges like FM, OCC, OPR, OV, MB, and IPR.

7.3.8 Ablation Study

In this section, we compare the performance of the proposed tracker using multiple-kernels (TRM-KCF) with the proposed tracker using a single Gaussian kernel (TR-KCF). Table 7.4 shows a comparison of the proposed TRM-KCF with TR-KCF and KCF (Henriques *et al.*, 2014) on TC128, OTB100, UAV123 and VOT-2017. It is observed that use of multiple kernels results in better performance compared to a single

kernel on TC128 and OTB100. The TRM-KCF outperforms TR-KCF in terms of Robustness and EAO on VOT-2017. On UAV123, the performance of TRM-KCF is comparable to TR-KCF in terms of success score. The analysis across multiple datasets shows that using multiple kernels results in better performance compared to using a single kernel. The proposed TRM-KCF formulation not only outperforms the proposed TR-KCF, but also outperforms many recent CF based trackers. It is observed that the performance of most trackers varies across the datasets, yet the TRM-KCF approach is consistent and performs well in all evaluation settings.

7.4 Chapter Summary

In this work, we derive a Temporally Regularized Multi-Kernel Correlation Filter formulation for robust visual tracking. The introduced temporal regularization component enables the KCF to adapt to large appearance variations that may have an adverse affect on tracking: While methods proposed in Chapters 5 and 6 also add similar regularisation methods, they do so outside of the kernelized CF framework, and thus are substantially slower and less robust. In this chapter, the target localization is done using response maps obtained corresponding to multiple kernel emphasizes the response in the foreground region. This prevents tracker drift when the response map contains high values in the background region. As a result a more discriminative and robust KCF is trained, that achieves efficient object tracking through multiple challenges. The positive effects of the proposed formulations are demonstrated on OTB100, TC128, VOT-2017, LaSOT, UAV123, and GOT-10k datasets. A comparative analysis with recent top ranked tracker reveals that the proposed approach outperforms the state of the art trackers for most challenges.

CHAPTER 8

Thesis Summary and Future Directions

8.1 Introduction

From video surveillance to human-computer interaction, visual object tracking is one of the most fundamental applications within computer vision. Correlation Filter (CF) based trackers have been widely used for a number of years, in part due to their computational efficiency. While end-to-end deep learning based methods have also emerged in recent years, CF trackers remain relevant due to their high speed and competitive accuracy. Recently, CF based strategies have demonstrated significant improvement in tracking performance when trained with multi-channel deep features. Many spatio-temporal regularization based CF trackers offer computationally inexpensive real time tracking with performance similar to the end-to-end deep learning based trackers. In this research, we investigate various methods to improve the performance of visual object trackers using CFs.

8.2 Summary

In this thesis, four novel CF based tracking approaches have been proposed, namely LSTM-AMP, CGRCF, IGSSTRCF, and TRM-KCF. In the first approach, we proposed an LSTM based ensemble CF tracker (LSTM-AMP in Chapter 4). CF trackers are combined with an appearance model pool to avoid faulty filter updates. The tracker performance is further improved by using dedicated filters to compute the scale and rotation of the target.

The CFs learned in Chapter 4 are trained using multi-channel deep features. However, each feature channel encodes different appearance information and these may not be equally important for tracking at different times. This motivated us to introduce channel regularized CF tracker that learned adaptive channel importance for each feature channel (CGRCF in Chapter 5). Thus, feature channels that encodes useful appearance cues are assigned higher weights and the less important feature channels are

suppressed through lower weights.

To further improve tracking accuracy, we investigated spatially and temporally regularized CFs. The spatial regularization suppressed the effects of unfavorable background information and boundary effects in the learned filter. The temporal regularization helped the filter adapt to appearance changes, preventing drift. We also modelled sparse spatial and sparse temporal variations to enhance the discriminative ability of the learned filters (IGSSTRCF in Chapter 6).

We further introduced the kernel trick to the temporally regularized CF tracker that is learned using multiple kernels (TRM-KCF in Chapter 7). The proposed tracker captured the non-linearity in the appearance features while retaining the circulant matrix structure.

Table 8.1: VOT toolkit report for VOT-2017 (Kristan *et al.*, 2017*b*) showing Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) for baseline experiment, and Overlap AUC for unsupervised experiment. The top three trackers are shown in **red**, **blue** and **green**

	Baseline			**Unsupervised**	**Speed**
	AR-Rank				
	A	**R**	**EAO**	**Overlap**	
LSTM-AMP (Chapter 4)	**0.45**	25.88	**0.21**	**0.31**	$\leq$5 FPS
CGRCF (Chapter 5)	**0.47**	**29.23**	**0.20**	**0.36**	$\leq$5 FPS
IGSSTRCF (Chapter 6)	**0.47**	**26.28**	**0.20**	**0.35**	$\leq$5 FPS
TRM-KCF (Chapter 7)	**0.43**	**34.00**	**0.16**	**0.31**	**22 FPS**

In Table 8.1, we compare all trackers proposed in this thesis (LSTM-AMP, CGRCF, IGSSTRCF, and TRM-KCF) using the VOT-2017 official toolkit. The comparison is done in terms of Accuracy (A), Robustness (R), and Expected Average Overlap (EAO) for the baseline experiments, overlap for the unsupervised experiments and tracking speed in Frames Per Second (FPS). It is observed that LSTM-AMP ranks first in terms of Expected Average Overlap (EAO). To improve the LSTM-AMP, spatio-temporal and channel regularizations are introduced in CGRCF and IGSSTRCF formulations. This results in improved Accuracy (A), Robustness (R) and overlap without a significant a drop in Expected Average Overlap (EAO). In further experiments, we use kernel tricks with temporal regularization (TRM-KCF), which results in significantly improved Robustness (R) with comparable performance in terms of Accuracy (A) and Overlap. It also results in significant improvement in tracking speed. TRM-KCF does not perform as well in terms of Expected Average Overlap (EAO) as it contains only the temporal

regularization, unlike CGRCF and IGSSTRCF that incorporate spatio-temporal regularization along with channel regularization. However, TRM-KCF formulation is substantially faster and real-time capable. It can be concluded that LSTM-AMP is preferable when Expected Average Overlap (EAO) needs to be higher, whereas CGRCF and IGSSTRCF can be used where high tracking Accuracy (A) is desired. For high speed robust tracking TRM-KCF will be most suitable.

This research has resulted in the following three accepted publications and one paper under review.

1. Jain, Monika, A. Venkata Subramanyam, Simon Denman, Sridha Sridharan, and Clinton Fookes. "LSTM guided ensemble correlation filter tracking with appearance model pool." Computer Vision and Image Understanding 195 (2020): 102935.
2. Jain, Monika, A. V. Subramanyam, Simon Denman, Sridha Sridharan, and Clinton Fookes. "IGSSTRCF: Importance Guided Sparse Spatio-Temporal Regularized Correlation Filters for Tracking." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2775-2784. 2021.
3. Jain, Monika, Arjun Tyagi, A. V. Subramanyam, Simon Denman, Sridha Sridharan, and Clinton Fookes. "Channel Graph Regularized Correlation Filters for Visual Object Tracking." IEEE Transactions on Circuits and Systems for Video Technology (2021).
4. TRM-KCF: Temporally Regularized Multi-Kernel CF For Visual Tracking, submitted to Pattern Recognition, Elsevier. 2021 (Under Review)

8.3 Future Work

We limit the scope of this thesis to Single Object Tracking (SOT). In future, Correlation Filter (CF) based visual object tracking can be explored in the following directions.

8.3.1 Long-Term Object Tracking

Due to fast computations in frequency domain, CFs can be explored for long-term tracking. One vital challenge in long term visual tracking is the absence of the target. If the target partially or fully disappears from the view, conventional CFs can be easily distracted by irrelevant objects because they do not contain a long-term component. Therefore, introducing a target re-detection module in the CF based trackers can significantly improve long-term CF based tracking methods. Additionally, investigation of long-term CF trackers with channel importance and spatio-temporal constraints has received little attention and can be explored in future.

8.3.2 Multi-Object Tracking

Multiple Object Tracking (MOT) is a computer vision task that aims to analyze videos in order to identify and track objects belonging to one or more categories, such as pedestrians, cars, animals and inanimate objects, without any prior knowledge about the appearance and number of targets. Different from object detection algorithms, whose output is a collection of rectangular bounding boxes identified by their coordinates, height and width, MOT algorithms also associate a target ID to each box (known as a detection), in order to distinguish among intra-class objects. MOT task plays an important role in computer vision: from video surveillance to autonomous cars, from action recognition to crowd behaviour analysis, many of these problems would benefit from a high-quality tracking algorithm.

While in SOT the appearance of the target is known a priori, in MOT a detection step is necessary to identify the targets, that can leave or enter the scene. The main difficulty in tracking multiple targets simultaneously stems from the various occlusions and interactions between objects, that can sometimes also have similar appearance. Thus, simply applying SOT models directly to a MOT problem leads to poor results, and often results in target drift and numerous ID switch errors, as such models usually struggle in distinguishing between similar looking intra-class objects. Though a series of algorithms specifically tuned to MOT have already been developed in recent years to address the above issues, most of these algorithms are end-to-end deep learning based. Therefore, MOT still remains a challenging task and hold significant scope for improvement by incorporating ideas from Correlation Filter based tracking.

LIST OF PAPERS BASED ON THESIS

1. Jain, Monika, A. Venkata Subramanyam, Simon Denman, Sridha Sridharan, and Clinton Fookes. "LSTM guided ensemble correlation filter tracking with appearance model pool." Computer Vision and Image Understanding 195 (2020): 102935.

2. Jain, Monika, A. V. Subramanyam, Simon Denman, Sridha Sridharan, and Clinton Fookes. "IGSSTRCF: Importance Guided Sparse Spatio-Temporal Regularized Correlation Filters for Tracking." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2775-2784. 2021.

3. Jain, Monika, Arjun Tyagi, A. V. Subramanyam, Simon Denman, Sridha Sridharan, and Clinton Fookes. "Channel Graph Regularized Correlation Filters for Visual Object Tracking." IEEE Transactions on Circuits and Systems for Video Technology (2021).

4. TRM-KCF: Temporally Regularized Multi-Kernel CF For Visual Tracking, submitted to Pattern Recognition, Elsevier. 2021 (Under Review)

REFERENCES

1. **Adam, A.**, **E. Rivlin**, and **I. Shimshoni**, Robust fragments-based tracking using the integral histogram. *In Proceedings of the IEEE Conference on Computer vision and pattern recognition*, volume 1. 2006.

2. **Babenko, B.**, **M.-H. Yang**, and **S. Belongie** (2010). Robust object tracking with online multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, **33**(8), 1619–1632.

3. **Babenko, B.**, **M.-H. Yang**, and **S. Belongie** (2011). Robust object tracking with online multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, **33**(8), 1619–1632.

4. **Bao, C.**, **Y. Wu**, **H. Ling**, and **H. Ji**, Real time robust l1 tracker using accelerated proximal gradient approach. *In IEEE Conference on Computer Vision and Pattern Recognition*. 2012.

5. **Beck, A.** and **M. Teboulle** (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, **2**(1), 183–202.

6. **Belkin, M.** and **P. Niyogi**, Laplacian eigenmaps and spectral techniques for embedding and clustering. *In Advances in neural information processing systems*. 2002.

7. **Bertinetto, L.**, **J. Valmadre**, **S. Golodetz**, **O. Miksik**, and **P. H. Torr**, Staple: Complementary learners for real-time tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016*a*.

8. **Bertinetto, L.**, **J. Valmadre**, **J. F. Henriques**, **A. Vedaldi**, and **P. H. Torr**, Fully-convolutional siamese networks for object tracking. *In European conference on computer vision*. Springer, 2016*b*.

9. **Bhat, G.**, **J. Johnander**, **M. Danelljan**, **F. S. Khan**, and **M. Felsberg**, Unveiling the power of deep tracking. *In Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

10. **Bibi, A.** and **B. Ghanem**, Multi-template scale-adaptive kernelized correlation filters. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2015.

11. **Bolme, D. S.**, **J. R. Beveridge**, **B. A. Draper**, and **Y. M. Lui**, Visual object tracking using adaptive correlation filters. *In IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010.

12. **Bouguet, J.-Y.** *et al.* (2001). Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel corporation*, **5**(1-10), 4.

13. **Boyd, S.**, **N. Parikh**, **E. Chu**, **B. Peleato**, **J. Eckstein**, *et al.* (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, **3**(1), 1–122.

14. **Bradski, G. R.** (1998). Computer vision face tracking for use in a perceptual user interface.

15. **Cehovin, L.**, **M. Kristan**, and **A. Leonardis** (2012). Robust visual tracking using an adaptive coupled-layer visual model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35**(4), 941–953.

16. **Čehovin, L.**, **M. Kristan**, and **A. Leonardis**, Is my new tracker really better than yours? *In IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2014.

17. **Čehovin, L.**, **A. Leonardis**, and **M. Kristan**, Robust visual tracking using template anchors. *In 2016 IEEE winter conference on applications of computer vision (WACV)*. IEEE, 2016*a*.

18. **Čehovin, L.**, **A. Leonardis**, and **M. Kristan** (2016*b*). Visual object tracking performance measures revisited. *IEEE Transactions on Image Processing*, **25**(3), 1261–1274.

19. **Chen, K.** and **W. Tao** (2018). Convolutional regression for visual tracking. *IEEE Transactions on Image Processing*, **27**(7), 3611–3620.

20. **Chi, Z.**, **H. Li**, **H. Lu**, and **M.-H. Yang** (2017). Dual deep network for visual tracking. *IEEE Transactions on Image Processing*, **26**(4), 2005–2015.

21. **Choi, J.**, **H. J. Chang**, **T. Fischer**, **S. Yun**, **K. Lee**, **J. Jeong**, **Y. Demiris**, and **J. Y. Choi**, Context-aware deep feature compression for high-speed visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

22. **Choi, J.**, **H. Jin Chang**, **J. Jeong**, **Y. Demiris**, and **J. Young Choi**, Visual tracking using attention-modulated disintegration and integration. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

23. **Comaniciu, D.** and **P. Meer** (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, **24**(5), 603–619.

24. **Comaniciu, D.**, **V. Ramesh**, and **P. Meer** (2003). Kernel-based object tracking. *IEEE Transactions on pattern analysis and machine intelligence*, **25**(5), 564–577.

25. **Cui, Z.**, **S. Xiao**, **J. Feng**, and **S. Yan**, Recurrently target-attending tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

26. **Dai, K.**, **D. Wang**, **H. Lu**, **C. Sun**, and **J. Li**, Visual tracking via adaptive spatially-regularized correlation filters. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

27. **Dalal, N.** and **B. Triggs**, Histograms of oriented gradients for human detection. 2005.

28. **Danelljan, M.**, **G. Bhat**, **F. S. Khan**, and **M. Felsberg** (2016*a*). Eco: Efficient convolution operators for tracking. *arXiv preprint arXiv:1611.09224*.

29. **Danelljan, M.**, **G. Bhat**, **F. Shahbaz Khan**, and **M. Felsberg**, Eco: Efficient convolution operators for tracking. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017*a*.

30. **Danelljan, M.**, **G. Häger**, **F. Khan**, and **M. Felsberg**, Accurate scale estimation for robust visual tracking. *In British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014*a*.

31. **Danelljan, M.**, **G. Häger**, **F. S. Khan**, and **M. Felsberg** (2016*b*). Discriminative scale space tracking. *IEEE transactions on pattern analysis and machine intelligence*, **39**(8), 1561–1575.

32. **Danelljan, M.**, **G. Häger**, **F. S. Khan**, and **M. Felsberg** (2017*b*). Discriminative scale space tracking. *IEEE transactions on pattern analysis and machine intelligence*, **39**(8), 1561–1575.

33. **Danelljan, M.**, **G. Hager**, **F. Shahbaz Khan**, and **M. Felsberg**, Convolutional features for correlation filter based visual tracking. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2015*a*.

34. **Danelljan, M.**, **G. Hager**, **F. Shahbaz Khan**, and **M. Felsberg**, Learning spatially regularized correlation filters for visual tracking. *In Proceedings of the IEEE international conference on computer vision*. 2015*b*.

35. **Danelljan, M.**, **G. Hager**, **F. Shahbaz Khan**, and **M. Felsberg**, Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016*c*.

36. **Danelljan, M.**, **A. Robinson**, **F. S. Khan**, and **M. Felsberg**, Beyond correlation filters: Learning continuous convolution operators for visual tracking. *In European Conference on Computer Vision*. Springer, 2016*d*.

37. **Danelljan, M.**, **F. Shahbaz Khan**, **M. Felsberg**, and **J. Van de Weijer**, Adaptive color attributes for real-time visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014*b*.

38. **De Maesschalck, R.**, **D. Jouan-Rimbaud**, and **D. L. Massart** (2000). The mahalanobis distance. *Chemometrics and intelligent laboratory systems*, **50**(1), 1–18.

39. **Dong, X.**, **J. Shen**, **L. Shao**, and **F. Porikli**, Clnet: A compact latent network for fast adjusting siamese trackers. *In* **A. Vedaldi**, **H. Bischof**, **T. Brox**, and **J.-M. Frahm** (eds.), *Computer Vision – ECCV 2020*. Springer International Publishing, Cham, 2020. ISBN 978-3-030-58565-5.

40. **Dong, X.**, **J. Shen**, **W. Wang**, **Y. Liu**, **L. Shao**, and **F. Porikli**, Hyperparameter optimization for tracking with continuous deep q-learning. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

41. **Dong, X.**, **J. Shen**, **W. Wang**, **L. Shao**, **H. Ling**, and **F. Porikli** (2021). Dynamical hyperparameter optimization via deep reinforcement learning in tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **43**(5), 1515–1529.

42. **Dong, X.**, **J. Shen**, **D. Yu**, **W. Wang**, **J. Liu**, and **H. Huang** (2016). Occlusion-aware real-time object tracking. *IEEE Transactions on Multimedia*, **19**(4), 763–771.

43. **Du, F.**, **P. Liu**, **W. Zhao**, and **X. Tang** (2019). Joint channel reliability and correlation filters learning for visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*.

44. **Fan, H.**, **L. Lin**, **F. Yang**, **P. Chu**, **G. Deng**, **S. Yu**, **H. Bai**, **Y. Xu**, **C. Liao**, and **H. Ling**, Lasot: A high-quality benchmark for large-scale single object tracking. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2019.

45. **Fan, H.** and **H. Ling**, Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking. *In Proceedings of the IEEE International Conference on Computer Vision*. 2017*a*.

46. **Fan, H.** and **H. Ling**, Sanet: Structure-aware network for visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2017*b*.

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

48. **Felzenszwalb, P.**, **R. Girshick**, **D. McAllester**, and **D. Ramanan** (2010). Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(9), 1627–1645.

49. **Fu, C.**, **J. Xu**, **F. Lin**, **F. Guo**, **T. Liu**, and **Z. Zhang** (2020). Object saliency-aware dual regularized correlation filter for real-time aerial tracking. *IEEE Transactions on Geoscience and Remote Sensing*.

50. **Gao, J.**, **H. Ling**, **W. Hu**, and **J. Xing**, Transfer learning based visual tracking with gaussian processes regression. *In European Conference on Computer Vision*. Springer, 2014.

51. **Gao, P.**, **R. Yuan**, **F. Wang**, **L. Xiao**, **H. Fujita**, and **Y. Zhang** (2020). Siamese attentional keypoint network for high performance visual tracking. *Knowledge-Based Systems*, **193**, 105448. ISSN 0950-7051. URL `https://www.sciencedirect.com/science/article/pii/S0950705119306665`.

52. **Ge, S.**, **Z. Luo**, **C. Zhang**, **Y. Hua**, and **D. Tao** (2019). Distilling channels for efficient deep tracking. *IEEE Transactions on Image Processing*.

53. **Girshick, R.**, **J. Donahue**, **T. Darrell**, and **J. Malik**, Rich feature hierarchies for accurate object detection and semantic segmentation. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.

54. **Grabner, H.**, **C. Leistner**, and **H. Bischof**, Semi-supervised on-line boosting for robust tracking. *In European conference on computer vision*. Springer, 2008.

55. **Guo, Q.**, **R. Han**, **W. Feng**, **Z. Chen**, and **L. Wan** (2019). Selective spatial regularization by reinforcement learned decision making for object tracking. *IEEE Transactions on Image Processing*, **29**, 2999–3013.

56. **Hare, S.**, **S. Golodetz**, **A. Saffari**, **V. Vineet**, **M.-M. Cheng**, **S. L. Hicks**, and **P. H. Torr** (2015). Struck: Structured output tracking with kernels. *IEEE transactions on pattern analysis and machine intelligence*, **38**(10), 2096–2109.

57. **Hare, S.**, **S. Golodetz**, **A. Saffari**, **V. Vineet**, **M.-M. Cheng**, **S. L. Hicks**, and **P. H. Torr** (2016). Struck: Structured output tracking with kernels. *IEEE transactions on pattern analysis and machine intelligence*, **38**(10), 2096–2109.

58. **He, K.**, **X. Zhang**, **S. Ren**, and **J. Sun**, Deep residual learning for image recognition. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

59. **He, Z.**, **Y. Fan**, **J. Zhuang**, **Y. Dong**, and **H. Bai**, Correlation filters with weighted convolution responses. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017.

60. **Held, D.**, **S. Thrun**, and **S. Savarese**, Learning to track at 100 fps with deep regression networks. *In* **B. Leibe**, **J. Matas**, **N. Sebe**, and **M. Welling** (eds.), *Computer Vision – ECCV 2016*. Springer International Publishing, Cham, 2016*a*. ISBN 978-3-319-46448-0.

61. **Held, D.**, **S. Thrun**, and **S. Savarese**, Learning to track at 100 fps with deep regression networks. *In European Conference on Computer Vision*. Springer, 2016*b*.

62. **Henriques, J. F.**, **R. Caseiro**, **P. Martins**, and **J. Batista**, Exploiting the circulant structure of tracking-by-detection with kernels. *In European conference on computer vision*. Springer, 2012.

63. **Henriques, J. F.**, **R. Caseiro**, **P. Martins**, and **J. Batista** (2014). High-speed tracking with kernelized correlation filters. *IEEE transactions on pattern analysis and machine intelligence*, **37**(3), 583–596.

64. **Henriques, J. F.**, **R. Caseiro**, **P. Martins**, and **J. Batista** (2015). High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**(3), 583–596.

65. **Hong, S.**, **T. You**, **S. Kwak**, and **B. Han**, Online tracking by learning discriminative saliency map with convolutional neural network. *In International Conference on Machine Learning*. 2015.

66. **Horn, B. K.** and **B. G. Schunck** (1981). Determining optical flow. *Artificial intelligence*, **17**(1-3), 185–203.

67. **Hu, J.**, **L. Shen**, and **G. Sun**, Squeeze-and-excitation networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

68. **Huang, B.**, **T. Xu**, **S. Jiang**, **Y. Chen**, and **Y. Bai** (2020). Robust visual tracking via constrained multi-kernel correlation filters. *IEEE Transactions on Multimedia*, **22**(11), 2820–2832.

69. **Huang, L.**, **X. Zhao**, and **K. Huang** (2019*a*). Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1. ISSN 1939-3539. URL `http://dx.doi.org/10.1109/TPAMI.2019.2957464`.

70. **Huang, Z.**, **C. Fu**, **Y. Li**, **F. Lin**, and **P. Lu**, Learning aberrance repressed correlation filters for real-time uav tracking. *In Proceedings of the IEEE International Conference on Computer Vision*. 2019*b*.

71. **Javed, S.**, **A. Mahmood**, **J. Dias**, **L. Seneviratne**, and **N. Werghi** (2021). Hierarchical spatiotemporal graph regularized discriminative correlation filter for visual object tracking. *IEEE Transactions on Cybernetics*.

72. **Jia, X.**, **H. Lu**, and **M.-H. Yang**, Visual tracking via adaptive structural local sparse appearance model. *In IEEE Conference on computer vision and pattern recognition*. 2012.

73. **Jia, Y.**, **E. Shelhamer**, **J. Donahue**, **S. Karayev**, **J. Long**, **R. Girshick**, **S. Guadarrama**, and **T. Darrell**, Caffe: Convolutional architecture for fast feature embedding. *In Proceedings of the 22nd ACM international conference on Multimedia*. 2014.

74. **Jr., S. B.** and **A. Belangour**, Multi object tracking: a survey. *In* **X. Jiang** and **H. Fujita** (eds.), *Thirteenth International Conference on Digital Image Processing (ICDIP 2021)*, volume 11878. International Society for Optics and Photonics, SPIE, 2021. URL `https://doi.org/10.1117/12.2602901`.

75. **Julier, S. J.** and **J. K. Uhlmann** (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, **92**(3), 401–422.

76. **Kalal, Z.**, **K. Mikolajczyk**, and **J. Matas** (2011). Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, **34**(7), 1409–1422.

77. **Kalal, Z.**, **K. Mikolajczyk**, and **J. Matas** (2012). Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, **34**(7), 1409–1422.

78. **Kiani Galoogahi, H.**, **A. Fagg**, and **S. Lucey**, Learning background-aware correlation filters for visual tracking. *In Proceedings of the IEEE International Conference on Computer Vision*. 2017.

79. **Kiani Galoogahi, H.**, **T. Sim**, and **S. Lucey**, Multi-channel correlation filters. *In Proceedings of the IEEE international conference on computer vision*. 2013.

80. **Kiani Galoogahi, H.**, **T. Sim**, and **S. Lucey**, Correlation filters with limited boundaries. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

81. **Kristan, M.**, **A. Leonardis**, **J. Matas**, **M. Felsberg**, **R. Pflugfelder**, **L. Čehovin**, **T. Vojir**, **G. Häger**, **A. Lukežič**, and **G. Fernandez** (2016*a*). The visual object tracking vot2016 challenge results. Springer. URL `http://www.springer.com/gp/book/9783319488806`.

82. **Kristan, M.**, **A. Leonardis**, **J. Matas**, **M. Felsberg**, **R. Pflugfelder**, **L. Čehovin Zajc**, **T. Vojir**, **G. Häger**, **A. Lukežič**, **A. Eldesokey**, and **G. Fernandez** (2017*a*). The visual object tracking vot2017 challenge results. URL `http://openaccess.thecvf.com/content_ICCV_2017_workshops/papers/w28/Kristan_The_Visual_Object_ICCV_2017_paper.pdf`.

83. **Kristan, M.**, **A. Leonardis**, **J. Matas**, **M. Felsberg**, **R. Pflugfelder**, **L. ˇCehovin Zajc**, **T. Vojir**, **G. Bhat**, **A. Lukezic**, **A. Eldesokey**, *et al.*, The sixth visual object tracking vot2018 challenge results. *In Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018.

84. **Kristan, M.**, **A. Leonardis**, **J. Matas**, **M. Felsberg**, *et al.*, The visual object tracking vot2017 challenge results. *In Proceedings of the IEEE International Conference on Computer Vision*. 2017*b*.

85. **Kristan, M.**, **J. Matas**, **A. Leonardis**, **M. Felsberg**, *et al.*, The seventh visual object tracking vot2019 challenge results. *In Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019.

86. **Kristan, M.**, **J. Matas**, **A. Leonardis**, **T. Vojíř**, **R. Pflugfelder**, **G. Fernandez**, **G. Nebehay**, **F. Porikli**, and **L. Čehovin** (2016*b*). A novel performance evaluation methodology for single-target trackers. *IEEE transactions on pattern analysis and machine intelligence*, **38**(11), 2137–2155.

87. **Kristan, M.**, **J. Matas**, **A. Leonardis**, **T. Vojir**, **R. Pflugfelder**, **G. Fernandez**, **G. Nebehay**, **F. Porikli**, and **L. Čehovin** (2016*c*). A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **38**(11), 2137–2155.

88. **Krizhevsky, A.**, **I. Sutskever**, and **G. E. Hinton**, Imagenet classification with deep convolutional neural networks. *In Advances in neural information processing systems*. 2012.

89. **Kumar, B. V.**, **A. Mahalanobis**, and **R. D. Juday**, *Correlation pattern recognition*. Cambridge University Press, 2005.

90. **La Scala, B. F.** and **R. R. Bitmead** (1996). Design of an extended kalman filter frequency tracker. *IEEE Transactions on Signal Processing*, **44**(3), 739–742.

91. **Lai, Z.**, **W. K. Wong**, **Y. Xu**, **J. Yang**, and **D. Zhang** (2015). Approximate orthogonal sparse embedding for dimensionality reduction. *IEEE transactions on neural networks and learning systems*, **27**(4), 723–735.

92. **Li, A.**, **M. Yang**, and **W. Yang**, Feature integration with adaptive importance maps for visual tracking. *In Proceedings of the 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 2018*a*.

93. **Li, B.**, **W. Wu**, **Q. Wang**, **F. Zhang**, **J. Xing**, and **J. Yan**, Siamrpn++: Evolution of siamese visual tracking with very deep networks. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019*a*.

94. **Li, B.**, **J. Yan**, **W. Wu**, **Z. Zhu**, and **X. Hu**, High performance visual tracking with siamese region proposal network. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018*b*.

95. **Li, F.**, **C. Tian**, **W. Zuo**, **L. Zhang**, and **M.-H. Yang**, Learning spatial-temporal regularized correlation filters for visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018*c*.

96. **Li, F.**, **Y. Yao**, **P. Li**, **D. Zhang**, **W. Zuo**, and **M.-H. Yang**, Integrating boundary and center correlation filters for visual tracking with aspect ratio variation. *In Proceedings of the IEEE International Conference on Computer Vision*. 2017*a*.

97. **Li, H.**, **Y. Li**, and **F. Porikli** (2016). Convolutional neural net bagging for online visual tracking. *Computer Vision and Image Understanding*, **153**, 120 – 129. ISSN 1077-3142. Special issue on Visual Tracking.

98. **Li, H.**, **Y. Li**, **F. Porikli**, *et al.*, Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. *In BMVC*, volume 1. 2014.

99. **Li, Y.**, **C. Fu**, **F. Ding**, **Z. Huang**, and **G. Lu** (2020*a*). Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization. *arXiv preprint arXiv:2003.12949*.

100. **Li, Y.**, **C. Fu**, **F. Ding**, **Z. Huang**, and **G. Lu**, Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization. *In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020*b*.

101. **Li, Y.**, **C. Fu**, **Z. Huang**, **Y. Zhang**, and **J. Pan** (2020*c*). Keyfilter-aware real-time uav object tracking. *arXiv preprint arXiv:2003.05218*.

102. **Li, Y.**, **Y. Zhang**, **Y. Xu**, **Z. Miao**, and **H. Li**, Does resnet learn good general purpose features? *In Proceedings of the 2017 International Conference on Artificial Intelligence, Automation and Control Technologies*, AIACT '17. ACM, New York, NY, USA, 2017*b*. ISBN 978-1-4503-5231-4. URL `http://doi.acm.org/10.1145/3080845.3080864`.

103. **Li, Y.** and **J. Zhu**, A scale adaptive kernel correlation filter tracker with feature integration. *In European conference on computer vision*. Springer, 2014.

104. **Li, Y.**, **J. Zhu**, **S. C. Hoi**, **W. Song**, **Z. Wang**, and **H. Liu**, Robust estimation of similarity transformation for visual object tracking. *In Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33. 2019*b*.

105. **Liang, P.**, **E. Blasch**, and **H. Ling** (2015). Encoding color information for visual tracking: Algorithms and benchmark. *IEEE Transactions on Image Processing*, **24**(12), 5630–5644.

106. **Liao, S.**, **Y. Hu**, **X. Zhu**, and **S. Z. Li**, Person re-identification by local maximal occurrence representation and metric learning. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

107. **Liu, B.**, **J. Huang**, **L. Yang**, and **C. Kulikowsk**, Robust tracking using local sparse appearance model and k-selection. *In CVPR 2011*. IEEE, 2011.

108. **Liu, S.**, **T. Zhang**, **X. Cao**, and **C. Xu**, Structural correlation filter for robust visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

109. **Liu, T.**, **G. Wang**, and **Q. Yang**, Real-time part-based visual tracking via adaptive correlation filters. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

110. **Liu, Z.**, **Z. Lai**, **W. Ou**, **K. Zhang**, and **R. Zheng** (2020). Structured optimal graph based sparse feature extraction for semi-supervised learning. *Signal Processing*, 107456.

111. **Liu, Z.**, **J. Wang**, **G. Liu**, and **L. Zhang** (2019). Discriminative low-rank preserving projection for dimensionality reduction. *Applied Soft Computing*, **85**, 105768.

112. **Lowe, D. G.**, Object recognition from local scale-invariant features. *In proceedings of the IEEE International Conference on Computer vision*, volume 2. Ieee, 1999.

113. **Lu, X.**, **J. Li**, **Z. He**, **W. Wang**, and **H. Wang** (2019*a*). Distracter-aware tracking via correlation filter. *Neurocomputing*, **348**, 134–144.

114. **Lu, X.**, **C. Ma**, **B. Ni**, and **X. Yang** (2019*b*). Adaptive region proposal with channel regularization for robust object tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 1–1.

115. **Lu, X.**, **C. Ma**, **B. Ni**, **X. Yang**, **I. Reid**, and **M.-H. Yang**, Deep regression tracking with shrinkage loss. *In Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

116. **Lucas, B. D.**, **T. Kanade**, *et al.*, An iterative image registration technique with an application to stereo vision. Vancouver, 1981.

117. **Lukezic, A.**, **T. Vojir**, **L. ˇCehovin Zajc**, **J. Matas**, and **M. Kristan**, Discriminative correlation filter with channel and spatial reliability. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

118. **Ma, C.**, **J.-B. Huang**, **X. Yang**, and **M.-H. Yang**, Hierarchical convolutional features for visual tracking. *In Proceedings of the IEEE international conference on computer vision*. 2015*a*.

119. **Ma, C.**, **X. Yang**, **C. Zhang**, and **M.-H. Yang**, Long-term correlation tracking. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015*b*.

120. **Ma, H.**, **S. T. Acton**, and **Z. Lin** (2020). Situp: Scale invariant tracking using average peak-to-correlation energy. *IEEE Transactions on Image Processing*, **29**, 3546–3557.

121. **Marvasti-Zadeh, S. M.**, **L. Cheng**, **H. Ghanei-Yakhdan**, and **S. Kasaei** (2021). Deep learning for visual tracking: A comprehensive survey. *IEEE Transactions on Intelligent Transportation Systems*, 1–26.

122. **Maybeck, P. S.**, The kalman filter: An introduction to concepts. *In Autonomous robot vehicles*. Springer, 1990, 194–204.

123. **Mueller, M.**, **N. Smith**, and **B. Ghanem**, A benchmark and simulator for uav tracking. *In European Conference on Computer Vision*. Springer, 2016.

124. **Mueller, M.**, **N. Smith**, and **B. Ghanem**, Context-aware correlation filter tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

125. **Nam, H.**, **M. Baek**, and **B. Han** (2016). Modeling and propagating cnns in a tree structure for visual tracking. *arXiv preprint arXiv:1608.07242*.

126. **Nam, H.** and **B. Han**, Learning multi-domain convolutional neural networks for visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

127. **Naresh Boddeti, V.**, **T. Kanade**, and **B. Vijaya Kumar**, Correlation filters for object alignment. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013.

128. **Ning, G.**, **Z. Zhang**, **C. Huang**, **X. Ren**, **H. Wang**, **C. Cai**, and **Z. He**, Spatially supervised recurrent convolutional neural networks for visual object tracking. *In 2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2017.

129. **Nummiaro, K.**, **E. Koller-Meier**, and **L. Van Gool** (2003). An adaptive color-based particle filter. *Image and vision computing*, **21**(1), 99–110.

130. **Pu, S.**, **Y. Song**, **C. Ma**, **H. Zhang**, and **M.-H. Yang**, Deep attentive tracking via reciprocative learning. *In Advances in Neural Information Processing Systems*. 2018.

131. **Qi, Y.**, **S. Zhang**, **L. Qin**, **H. Yao**, **Q. Huang**, **J. Lim**, and **M.-H. Yang**, Hedged deep tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

132. **Qian, X.**, **L. Han**, **Y. Wang**, and **M. Ding** (2018). Deep learning assisted robust visual tracking with adaptive particle filtering. *Signal Processing: Image Communication*, **60**, 183 – 192. ISSN 0923-5965.

133. **Rahman, M. M.**, **M. Fiaz**, and **S. K. Jung** (2020). Efficient visual tracking with stacked channel-spatial attention learning. *IEEE Access*.

134. **Redmon, J.**, **S. Divvala**, **R. Girshick**, and **A. Farhadi**, You only look once: Unified, real-time object detection. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

135. **Ross, D. A.**, **J. Lim**, **R.-S. Lin**, and **M.-H. Yang** (2008). Incremental learning for robust visual tracking. *International journal of computer vision*, **77**(1-3), 125–141.

136. **Senna, P.**, **I. N. Drummond**, and **G. S. Bastos**, Real-time ensemble-based tracker with kalman filter. *In 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2017.

137. **Sevilla-Lara, L.** and **E. Learned-Miller**, Distribution fields for tracking. *In IEEE Conference on computer vision and pattern recognition*. 2012.

138. **Shen, J.**, **X. Tang**, **X. Dong**, and **L. Shao** (2020). Visual object tracking by hierarchical attention siamese network. *IEEE Transactions on Cybernetics*, **50**(7), 3068–3080.

139. **Simonyan, K.** and **A. Zisserman** (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

140. **Smeulders, A. W.**, **D. M. Chu**, **R. Cucchiara**, **S. Calderara**, **A. Dehghan**, and **M. Shah** (2014). Visual tracking: An experimental survey. *IEEE transactions on pattern analysis and machine intelligence*, **36**(7), 1442–1468.

141. **Song, Y.**, **C. Ma**, **X. Wu**, **L. Gong**, **L. Bao**, **W. Zuo**, **C. Shen**, **R. Lau**, and **M.-H. Yang** (2018). Vital: Visual tracking via adversarial learning. *arXiv preprint arXiv:1804.04273*.

142. **Sui, Y.**, **G. Wang**, and **L. Zhang** (2018). Joint correlation filtering for visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*.

143. **Sun, C.**, **D. Wang**, **H. Lu**, and **M.-H. Yang**, Correlation tracking via joint discrimination and reliability learning. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018*a*.

144. **Sun, C.**, **D. Wang**, **H. Lu**, and **M.-H. Yang**, Learning spatial-aware regressions for visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018*b*.

145. **Szegedy, C.**, **W. Liu**, **Y. Jia**, **P. Sermanet**, **S. Reed**, **D. Anguelov**, **D. Erhan**, **V. Vanhoucke**, and **A. Rabinovich**, Going deeper with convolutions. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

146. **Tang, F.** and **Q. Ling** (2019). Spatial-aware correlation filters with adaptive weight maps for visual tracking. *Neurocomputing*, **358**, 369–384.

147. **Tang, M.** and **J. Feng**, Multi-kernel correlation filter for visual tracking. *In Proceedings of the IEEE international conference on computer vision*. 2015.

148. **Tang, M.**, **B. Yu**, **F. Zhang**, and **J. Wang**, High-speed tracking with multi-kernel correlation filters. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

149. **Tao, R.**, **E. Gavves**, and **A. W. Smeulders**, Siamese instance search for tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

150. **Tian, G.**, **R. Hu**, **Z. Wang**, and **Y. Fu** (2009). Improved object tracking algorithm based on new hsv color probability model. *Advances in Neural Networks–ISNN 2009*, 1145–1151.

151. **Tibshirani, R.** (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, **58**(1), 267–288.

152. **Uzkent, B.** and **Y. Seo**, Enkcf: Ensemble of kernelized correlation filters for high-speed object tracking. *In IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2018.

153. **Valmadre, J.**, **L. Bertinetto**, **J. Henriques**, **A. Vedaldi**, and **P. H. Torr**, End-to-end representation learning for correlation filter based tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017*a*.

154. **Valmadre, J.**, **L. Bertinetto**, **J. Henriques**, **A. Vedaldi**, and **P. H. S. Torr**, End-to-end representation learning for correlation filter based tracking. *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017*b*.

155. **Velasco-Salido, E.** and **J. Martınez** (2017). Scale adaptive point-based kanade lukas tomasi colour-filter tracker. *Under Review*.

156. **Vojir, T.**, **J. Noskova**, and **J. Matas** (2014). Robust scale-adaptive mean-shift for tracking. *Pattern Recognition Letters*, **49**, 250–258.

157. **Wang, J.**, **W. Liu**, **W. Xing**, and **S. Zhang** (2018*a*). Visual object tracking with multi-scale superpixels and color-feature guided kernelized correlation filters. *Signal Processing: Image Communication*, **63**, 44 – 62. ISSN 0923-5965.

158. **Wang, L.**, **W. Ouyang**, **X. Wang**, and **H. Lu**, Visual tracking with fully convolutional networks. *In Proceedings of the IEEE international conference on computer vision*. 2015*a*.

159. **Wang, N.**, **S. Li**, **A. Gupta**, and **D.-Y. Yeung** (2015*b*). Transferring rich feature hierarchies for robust visual tracking. *arXiv preprint arXiv:1501.04587*.

160. **Wang, N.** and **D.-Y. Yeung**, Learning a deep compact image representation for visual tracking. *In Advances in neural information processing systems*. 2013.

161. **Wang, N.**, **W. Zhou**, **Q. Tian**, **R. Hong**, **M. Wang**, and **H. Li**, Multi-cue correlation filters for robust visual tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018*b*.

162. **Wang, Q.**, **J. Gao**, **J. Xing**, **M. Zhang**, and **W. Hu** (2017*a*). Dcfnet: Discriminant correlation filters network for visual tracking. URL `https://arxiv.org/abs/1704.04057`.

163. **Wang, X.**, **A. Shrivastava**, and **A. Gupta**, A-fast-rcnn: Hard positive generation via adversary for object detection. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017*b*.

164. **Wu, Y.**, **J. Lim**, and **M.-H. Yang** (2015). Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**(9), 1834–1848.

165. **Xu, T.**, **Z.-H. Feng**, **X.-J. Wu**, and **J. Kittler** (2019*a*). Joint group feature selection and discriminative filter learning for robust visual object tracking. *arXiv preprint arXiv:1907.13242*.

166. **Xu, T.**, **Z.-H. Feng**, **X.-J. Wu**, and **J. Kittler** (2019*b*). Learning low-rank and sparse discriminative correlation filters for coarse-to-fine visual object tracking. *IEEE Transactions on Circuits and Systems for Video Technology*.

167. **Xu, Y.**, **Z. Wang**, **Z. Li**, **Y. Yuan**, and **G. Yu** (2020). Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. *Proceedings of the AAAI Conference on Artificial Intelligence*, **34**(07), 12549–12556. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6944`.

168. **Yuan, Y.**, **S. Emmanuel**, **Y. Fang**, and **W. Lin** (2014). Visual object tracking based on backward model validation. *IEEE Transactions on Circuits and Systems for Video Technology*, **24**(11), 1898–1910.

169. **Yun, S.**, **J. Choi**, **Y. Yoo**, **K. Yun**, and **J. Young Choi**, Action-decision networks for visual tracking with deep reinforcement learning. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

170. **Zhang, B.**, **Z. Li**, **X. Cao**, **Q. Ye**, **C. Chen**, **L. Shen**, **A. Perina**, and **R. Jill** (2016). Output constraint transfer for kernelized correlation filter in tracking. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, **47**(4), 693–703.

171. **Zhang, B.**, **S. Luan**, **C. Chen**, **J. Han**, **W. Wang**, **A. Perina**, and **L. Shao** (2017*a*). Latent constrained correlation filter. *IEEE Transactions on Image Processing*, **27**(3), 1038–1048.

172. **Zhang, J.**, **S. Ma**, and **S. Sclaroff**, Meem: robust tracking via multiple experts using entropy minimization. *In European Conference on Computer Vision*. Springer, 2014*a*.

173. **Zhang, K.**, **L. Zhang**, **Q. Liu**, **D. Zhang**, and **M.-H. Yang**, Fast visual tracking via dense spatio-temporal context learning. *In European Conference on Computer Vision*. Springer, 2014*b*.

174. **Zhang, L.** and **P. N. Suganthan** (2017). Robust visual tracking via co-trained kernelized correlation filters. *Pattern Recognition*, **69**, 82 – 93. ISSN 0031-3203.

175. **Zhang, T.**, **C. Xu**, and **M.-H. Yang**, Multi-task correlation particle filter for robust object tracking. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017*b*.

176. **Zhou, T.**, **H. Bhaskar**, **F. Liu**, and **J. Yang** (2016). Graph regularized and locality-constrained coding for robust visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, **27**(10), 2153–2164.

177. **Zhou, T.**, **H. Bhaskar**, **F. Liu**, and **J. Yang** (2017). Graph regularized and locality-constrained coding for robust visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, **27**(10), 2153–2164. ISSN 1558-2205.

178. **Zhu, G.**, **J. Wang**, and **H. Lu** (2016). Clustering based ensemble correlation tracking. *Computer Vision and Image Understanding*, **153**, 55 – 63. ISSN 1077-3142. Special issue on Visual Tracking.

179. **Zhu, Z.**, **Q. Wang**, **B. Li**, **W. Wu**, **J. Yan**, and **W. Hu**, Distractor-aware siamese networks for visual object tracking. *In Proceedings of the European Conference on Computer Vision (ECCV)*. 2018*a*.

180. **Zhu, Z.**, **W. Wu**, **W. Zou**, and **J. Yan**, End-to-end flow correlation tracking with spatial-temporal attention. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018*b*.

181. **Zuo, W.**, **X. Wu**, **L. Lin**, **L. Zhang**, and **M.-H. Yang** (2018). Learning support correlation filters for visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, **41**(5), 1158–1172.