Robust Real-time Visual Tracking

Liu Ting

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2017

Acknowledgments

First, I would like to express my sincerest gratitude to my supervisor Professor Jiang Xudong, who has supported me for my thesis preparation and helped me to improve the quality of this thesis.

Second, I would like to thank my former supervisor Professor Wang Gang, who has supported and encouraged me patiently during my Ph.D. studies in Nanyang Technological University (NTU). His endless passion, critical thinking and rigorous attitude in research have affected the way I behaved during my research work. This research was not possible without his direction, advice, encouragement, and constructive criticism.

Then, I would like to thank my Thesis Advisory Committee members, Prof. Wang Lipo, Prof. Chia Liang Tien and all team members under Prof. Wang: Mr. Bing Shuai, Dr. Li Wang, Dr. Rahul Rama Varuir, Dr. Bing Wang, Dr. Zhen Zuo, Mr. Wang Xingxing, Mr. Jianhan Mei, Mrs. Abrar Hamzeh Saad Abdul Nabi, Mr. Amir Shahroudy, Mr. Jiuxiang Gu, Mr. Jason Kuen, Dr. Lianyang Ma, Dr. Jinghua Wang for their useful advices and discussions on my work. I am also indebted to all my other friends for their support and encouragement during my PhD studies.

Finally, I would like to dedicate this thesis to my parents, my wife Mrs. Cheng Zhang and my in-laws who have given me constant encouragement, love and support.

Contents

Acknowledgments	i
Abstract	v
List of Figures	vii
List of Tables	xii
List of Abbreviations	xiv
1 Introduction	1
1.1 Background	1
1.2 Motivations	2
1.3 Major Contributions of the Thesis	5
1.4 Organization of the Thesis	6
2 Related Work	8
2.1 Generative Tracking	8
2.1.1 Template representation trackers	8
2.1.2 Subspace representation trackers	10
2.1.3 Sparse representation trackers	11
2.2 Discriminative Tracking	15
2.2.1 Online updating trackers	15
2.2.2 Metric learning trackers	18
2.2.3 Correlation filter trackers	19
2.2.4 Deep learning trackers	22
2.3 Hybrid Generative-Discriminative Tracking	23

3	Visual tracking via temporally smooth sparse coding	26
3.1	Introduction	26
3.2	Temporally smooth tracking based on sparse representation	29
3.2.1	The basic sequential inference model	29
3.2.2	Sparse representation in tracking	30
3.2.3	Temporally smooth tracking	31
3.3	Experiments	33
3.4	Conclusion	39
4	Part-based tracking via discriminative correlation filters	40
4.1	Introduction	40
4.2	Part-based tracker via correlation filter	45
4.2.1	Proposed tracking method	45
4.2.2	Learning to select parts	53
4.3	Implementation	54
4.4	Experiments	55
4.4.1	Overall evaluation on OTB benchmark	56
4.4.2	Attribute-based evaluation	56
4.4.3	Evaluation on VOT2014 dataset.	60
4.5	Discussion	63
4.6	Conclusion	63
5	Color feature learning for correlation filter based tracking method	67
5.1	Introduction	67
5.2	Learning color features	69
5.2.1	Training Dataset	70
5.2.2	Objective Formulation	70
5.2.3	Formulation Optimization	71
5.3	Experiments	72

5.4	Conclusion	73
6	Object Tracking via ImageNet Classification Scores	76
6.1	Introduction	76
6.2	Proposed tracking algorithm	79
6.2.1	Occlusion estimation via ImageNet classification scores	80
6.2.2	Multi-model tracking	84
6.2.3	Frame tracking via selective features	85
6.2.4	Target re-detection	86
6.3	Experiments	89
6.3.1	Implementation	89
6.3.2	OBT evaluation	89
6.3.3	VOT 2014 evaluation	93
6.4	Conclusion	97
7	Conclusions and Future Works	98
7.1	Conclusions	98
7.2	Future Works	100
	Publications	103
	References	105

Abstract

Robust visual tracking plays an important role in many applications such as security surveillance, human-computer interaction and video analytics. Given the position of a target in the first frame of a video clip, the objective is to track the target in following frames of this sequence. Although many promising trackers have been proposed and achieved fairly good performance in simple environment, it is still very challenging to efficiently track arbitrary objects in complicated situations, especially when appearance changes significantly and heavy occlusion occurs. The goal of this thesis is to present various tracking algorithms in different perspectives to handle the aforementioned challenges in visual tracking.

In the first part of the thesis, a sparse coding based tracking method is presented. Although sparse representation has been widely used in object tracking for its robustness and accuracy, due to the high computing complexity, it is difficult for the sparse based tracker to track the target quickly. In this work, we revisit a more efficient method to optimize the ℓ_1 norm problem, which can make our tracking method operating in real-time. Additionally, for most conventional sparse coding based trackers, the target candidates are considered independently between consecutive frames. This work shows that the temporal correlation of these frames can be exploited to improve the performance and makes the tracker more robust to noise.

The second piece of work aims to overcome the occlusion challenge of visual tracking. Dividing targets into multiple parts is an effective way to solve the partial occlusion issue in visual object tracking methods. However, due to the complicated training and tracking

models, most of these part-based trackers cannot run efficiently. Recently correlation filter based trackers become increasingly more popular because of the high efficiency. Because most of the existing correlation trackers treat the target as fixed scale and rotation, it is difficult to realize long-term tracking. In this work, we develop a novel tracking method which tracks object parts through multiple correlation filters. Meanwhile, the Bayesian inference framework and a structural constraint mask are also exploited to enhance our method to various appearance changes.

In the third part of this thesis, discriminative color features are learned for visual tracking projects. Color features are widely used in single object tracking tasks. Previously, most traditional color-based trackers either rely on luminance information or use simple color representations for image description. The perceived color of the target may change significantly because of various lighting conditions. In this work, the color features are learned offline from pixels sampled from images across different cameras or camera views. The pixel values of the same color with different lighting conditions are encoded with the same values. We learn the color feature space through jointly optimizing a linear transformation and a dictionary (to encode pixel values). The marginal regression is adopted to speedup the feature extraction procedures.

The last piece of work investigates visual tracking based on large scale classification scores. Although the tracker introduced in the second part of this thesis handles occlusion well, limited by the hand-crafted features, it is not robust enough when the appearance of targets changes significantly caused by other factors, such as illumination, pose changes and rotation. Hence we propose to turn help to the deep learning features for better performance. While, considering the high computational complexity, the previous multi-part framework cannot be simply applied to deep features to tackle occlusion. In this work, an effective occlusion detection method is proposed based on the high-level classification scores from deep convolutional neural network trained on ImageNet. By exploiting the occlusion estimation scheme, we propose a novel tracking method by holistically considering multiple previously trained tracking models. A linear motion model is adopted to effectively re-detect the lost target.

List of Figures

1.1	The pipeline of visual tracking algorithm.	2
1.2	Illustration of tracking with parts. The green bounding boxes mean correct tracking of parts. The red ones represent failure of tracking.	3
1.3	Illustration of feature representation of a target object using Histograms of Oriented Gradients (HOG). The green and red colors represent corresponding patches from cell division.	4
1.4	Illustration of Convolutional Neural Network (CNN).	5
3.1	Comparison of tracking performance obtained by trackers without and with temporal smoothness constraint. Because of the sparse coding noise, there is an apparent drift of the blue boxes that are yielded from method without temporal smoothness constraint. The red boxes are obtained from the proposed method. By considering the temporal correlation, the red features are more similar to neighbouring ones and red boxes track the target more accurately.	27
3.2	Comparison of our approach with state-of-the-art trackers in challenging situations.	34

4.1	(a) and (b) show the tracking of each part tracker independently and the response maps of corresponding parts (in corresponding red colors). (c) is the selection the parts through learning which is introduced in 4.2.2. Then separate response maps are combined by Eq. 4.1. We track the whole target based on the joint confidence map in the Bayesian inference framework as shown in (d). When computing the likelihood, sampling candidates are shown as yellow rectangles. The solid yellow rectangle shows the tracking result on the confidence map with the maximum likelihood. (Given the limitation of space, we only show nine parts as examples in (a) and (b).)	41
4.2	Tracking target from frame #27 and #28, in which two parts are marked in bright and dark red bounding boxes respectively. The corresponding confidence maps of two part trackers are shown at the right with corresponding color bounding boxes. For the bright red part, it is tracked correctly between the consecutive frames. For the dark red part, it drifts in (b) because of the occlusion. To better evaluate the joint confidence map, the response of drifting part should contribute less. If we only consider PSR to weight the confidence maps, bright red and dark red parts will contribute equally. Different from the dark red part, the confidence maps of the bright red part in (a) and (b) have the smooth property. Hence, we should consider the smooth constraint to make weight selection more robust.	48
4.3	(a) and (b) are the response maps of two part trackers. (c) is the summation of (a) and (b). (d) is the structural constraint mask to enforce the spatial layout of individual parts. (e) is the confidence map used for state parameter searching after convolving with the mask.	51

4.4	(a) The target is near the camera and caught with a larger bounding box. (b)The target is far away from the camera with a small bounding box. (c) The corresponding joint confidence map of (a). (d) The corresponding joint confidence map of (b). Because of the occlusion, the responses of bottom part are suppressed by the weighting method.	53
4.5	The spatial layout of initial parts distribution	55
4.6	Precision and Success Plots of tracking results from the top 10 trackers on all video sequences of the benchmark [1].	57
4.7	Comparison of our approach with state-of-the-art trackers in challenging situations (1).	60
4.8	Comparison of our approach with state-of-the-art trackers in challenging situations (2).	61
4.9	The accuracy-robustness ranking plots with respect to the two experi- ments. Tracker is better if it resides closer to the top-right corner of the plot.	65
4.10	The raw plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.	66
5.1	(a)Sampled pixels (blue box) from one tracking frame, in which the tar- get’s face (red box) is under bright lighting condition. (b) Encoding of a randomly sampled pixel from this bright patch. (c) Sampled pixels (blue box) from one frame, in which the target’s face (red box) is under dark lighting condition. (d) Encoding of a randomly sampled pixel from this dark patch. It can be seen from (a) and (c), the color for the corresponding sampled patches are not close to each other. (b) and (d) show the results of the encoding using our approach. Using our proposed algorithm, the obtained encoding values are very similar to each other for corresponding pixels in the tracking target.	68

5.2	Comparison of our approach with state-of-the-art trackers in challenging situations such as illumination variations. The example frames are from the Trellis, Matrix, Shaking, Coke and Skiing sequences respectively. The results of proposed tracker are indicated by yellow boxes.	75
6.1	Tracking results of traditional correlation filter based trackers in heavy occlusion (KCF in green, DSST in red).	77
6.2	Overview of our algorithm. The upper formulation presents the multi-model tracking scheme. The confidence map of t -th frame is calculated via I tracking models selected from frame index set $\mathcal{D}$ (6.2.2). The green part is the individual frame tracking via convolutional features (6.2.3). The bright green, hunter green and dark green (from left to right) represent the processes based on different features extracted from Conv3, Conv4 and Conv5 layers of VGG net respectively. The red part is the re-detection (6.2.4). (Best view in color)	80
6.3	(a) Tracking results and the corresponding occlusion estimation scores. The top 3 classification categories (corresponding scores in percentage) are listed at the right down corner of each frame. (b) shows the examples of significant appearance changes and corresponding occlusion estimation scores.	82
6.4	Tracking results with lower (Conv3) features (dark green) and deeper features (bright green). When the appearances of the object change significantly (such as rotation in the first row, severe pose change in the second row and fast moving in the third row), the deeper features can handle better.	87
6.5	Prediction of re-detection position. The solid lines represent the previous obtained velocity and positions. The dotted lines is the predicted velocity and positions.	88
6.6	Success and Precision Plots of our proposed methods comparing with baselines.	90

6.7	Success and Precision Plots of tracking results from the top 10 trackers on all video sequences of the benchmark [1].	94
6.8	The accuracy-robustness ranking plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.	95
6.9	The raw plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.	96

List of Tables

3.1	Average center location error (in pixels). The best three results are shown in red, blue, and green fonts.	36
3.2	Average overlap rate(%). The best three results are shown in red, blue, and green fonts.	37
3.3	Running speed comparison of several popular sparse coding based L1 trackers	39
4.1	Average precision on different attributes: low resolution (LR), background cluttered (BC), out-of-view (OV), in-plane rotation (IPR), fast motion (FM), motion blur (MB), deformation (Def), occlusion (Occ), scale variation (SV), out-of-plane rotation (OPR) and illumination variation (IV). The best and the second best results are in red and blue colors.	58
4.2	Success scores on different attributes: low resolution (LR), background cluttered (BC), out-of-view (OV), in-plane rotation (IPR), fast motion (FM), motion blur (MB), deformation (Def), occlusion (Occ), scale variation (SV), out-of-plane rotation (OPR) and illumination variation (IV). The best and the second best results are in red and blue colors.	59
4.3	Ranking results on VOT2014 dataset. The "overall" column displays a joined ranking for both experiments, which are also used to order the trackers.	64
5.1	Average center location error (in pixels). The bold font indicate the best performance.	73

5.2	Average overlap rate(%). The bold font indicate the best performance. .	74
6.1	Success score and precision on different attributes: illumination variation (IV), out-of-plane rotation (OPR), scale variation (SV), occlusion (Occ), deformation (Def), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-view (OV) and low resolution (LR). The best results are in red colors.	92
6.2	The overall success and precision scores comparison with Deep trackers. The best results are shown in red color.	93
6.3	Ranking results on VOT2014 dataset. The "overall" column displays a joint ranking for both experiments, which are also used to order the trackers. The best results are shown in red color.	97

List of Abbreviation

3DDCT	3D discrete cosine transform
APG	Accelerated Proximal Gradient
ASLA	Adaptive Structural Local Appearance
ASR	Average Success Rate
AUC	Area Under Curve
AVT	Attentional Visual Tracking
CNN	Convolutional Neural Network
CPT	Composite Patch-based Templates
CPU	Central Processing Unit
CT	Compressive Tracker
CXT	Context Tracker
DEF	Deformation
DIF	Data Independent Feature
DLT	Deep Learning Tracker
DNN	Deep Neural Networks
EM	Expectation maximisation

FFT	Fast Fourier Transform
FM	Fast Motion
FragT	Fragment Tracker
GPU	Graphics Processing Unit
GT	Ground Truth
HOG	Histograms of Oriented Gradients
IPCA	Incremental Principal Component Analysis
IPR	In-Plane Rotation
IV	Illumination Variation
IVT	Incremental Visual Tracker
KCF	Kernelized Correlation Filter
L1T	L1-tracker
LR	Low Resolution
LSST	Least Soft-threshold Squares Tracking
MAP	Maximum a Posterior
MB	Motion Blur
MEEM	Multiple Experts Entropy Minimization
MIL	Multiple Instance Learning
MOSSE	Minimum Output Sum of Squared Error
MTT	Multiple Task Tracker
NLSS	Non-Local Self-Similarity

OCC	Occlusion
OPE	One Pass Evaluation
OPR	Out-of-Plane Rotation
OV	Out-of-View
P-N	Positive-Negative
PASCAL	Pattern Analysis, Statistical modelling and Computational Learning
PCA	Principal Component Analysis
PSR	Peak-to-Sidelobe Ratio
RVM	Relevance Vector Machine
SCCM	Smooth Constraint of Confidence Maps
SCM	Sparsity-based Collaborative Model
SDC	Sparsity-based Discriminative Classifier
SGM	Sparsity-based Generative Model
SGMM	Sequential Gaussian Mixture Model
SPOT	Structure Preserving Object Tracking
SR	Success Rates
Struck	Structured output tracking with kernels
SV	Scale Variation
SVM	Support Vector Machine
TLD	Tracking-Learning-Detection
TGPR	Transfer Gaussian Process Regression

TMRT	Temporally smooth Tracker via Marginal Regression
TST	Two-stage Sparsity Tracking
VGG	Visual Geometry Group
VTD	Visual Tracking Decomposition
VTs	Visual Tracking Sampling

Chapter 1

Introduction

1.1 Background

Visual tracking is an important technique in computer vision with various applications such as security and surveillance, human computer interaction and auto-control systems. For decades, many methods have been proposed for object tracking. With the development of single object tracking method, most of the tracking tasks in simple environment can be solved well by current algorithms. However, in more complicated situations, such as heavy occlusion and significant illumination changes, the more robust tracking system is required to realize efficient and accurate tracking.

The scenario of visual tracking algorithm is illustrated in Fig. 1.1. In the first frame of a video, we are given the location of the target object. The visual tracking methods aim to continuously localize the target in the following sequence.

Visual tracking is generally classified in two categories: generative and discriminative [2]. Generative trackers usually represent the target based on an appearance model and track the object by searching for the candidate with minimal reconstruction error. On the other hand, discriminative tracking algorithms try to separate the target from the background using binary classifiers. In addition, feature representation is another important component of both generative and discriminative tracking methods. In this thesis, we will visit the above mentioned aspects to develop robust and efficient tracking methods.

Figure 1.1: The pipeline of visual tracking algorithm.

1.2 Motivations

The robustness of visual tracking methods is affected by many factors, such as occlusion, illumination and pose changes. Additionally, speed is another important factor that determines the availability of a tracker. Although, some of the existing tracking methods can achieve fairly good performance in most situations, due to the complicated learning or updating schemes, it is difficult for them to realize real-time tracking. The goal of this thesis is developing efficient and robust algorithms for visual tracking tasks. We realize this goal through several different aspects.

As generative tracking methods, sparse coding based trackers are proven to be robust to most appearance changes in tracking tasks. However, most of the traditional sparse tracking methods use Lasso to solve the ℓ_1 norm problem, which is computationally expensive. Consequently, it is difficult for these sparse trackers to run in real-time. Meanwhile, these conventional sparse coding based trackers search the candidates independently between consecutive frames, which ignore the information of neighbouring frames. Because the target objects usually change slightly between adjacent frames, the similarity between these neighbouring frames can be exploited to improve the robustness of tracking results. To address the above mentioned problem, in the first work of

this thesis, marginal regression is revisited to speedup the ℓ_1 norm. Furthermore, the proposed tracker utilizes information among consecutive frames to improve the tracking accuracy.

Although sparse coding based tracker handles the appearance changes well in most situations, when the target is heavily occluded it cannot localize the object correctly. Hence, a more robust tracker which can tackle the occlusion problem is required. In previous researches, part-based models are popular in solving occlusion scenes. Since the target is divided into multiple parts, even though the target is partially occluded, we can still track the object by the visible parts. However, because of the complicated training and updating schemes, the speed of conventional part-based trackers is usually slow. In the second work of this thesis, benefiting from the highly efficient correlation filter tracker, we develop a real-time part-based visual tracking method. At the same time, combining with the generative tracking framework and a structural constraint mask, our method can also handle the other appearance changes in addition to occlusion.

Figure 1.2: Illustration of tracking with parts. The green bounding boxes mean correct tracking of parts. The red ones represent failure of tracking.

Feature representation is another important component of visual object tracking. In most of traditional tracking algorithms, raw pixel values or hand-crafted features are used

to represent targets. For example, in our second work, Histograms of Oriented Gradients (HOG) is used as the feature for correlation filter classifier (as shown in Fig. 1.3). However, when the appearance of targets change significantly, these hand-crafted features cannot handle well. Among these hand-crafted features, color as an important image descriptor is popularly used in tracking. Most conventional color-based trackers either rely on luminance information or use simple color representations for image description. Consequently, the perceived color of the target may change significantly because of the varying lighting conditions. Hence, in the third work of this thesis, we aim to learn color features offline from pixels sampled from images across different camera views.

Figure 1.3: Illustration of feature representation of a target object using Histograms of Oriented Gradients (HOG). The green and red colors represent corresponding patches from cell division.

Recently, Deep Neural Networks (DNNs), especially Convolutional Neural Networks (CNNs) have recently drawn a lot of attention due to their outstanding representation power in visual tracking. However, although these powerful features are used in trackers, the heavy occlusion is still a big challenge for the deep trackers. Without an effective

tracking performance evaluation method, most of existing trackers may be contaminated seriously when the target is heavily or totally occluded. Consequently they will lose the object in the following frames. Hence, an effective occlusion detection method is essential to evaluate the tracking status (the target is tracked or lost) and avoid the model pollution. Based on the semantically meaningful category responses of the pre-trained CNN classifiers, the fourth work of this thesis introduces an occlusion estimation method. Through adaptively updating and weighting the multiple tracking models, the tracker can be protected from contamination by the occlusion.

Figure 1.4: Illustration of Convolutional Neural Network (CNN).

1.3 Major Contributions of the Thesis

The major contributions of this thesis are summarized as follows.

- A real time temporally smooth visual tracker is developed with improved tracking performance. First, the information among consecutive frames is utilized to improve the tracking accuracy. Then, the marginal regression is employed to speedup the tracking speed significantly. Extensive experiments on real-world video sequences have been done to validate the high computational efficiency and better accuracy of the TMRT method.
- A real-time part-based object tracker is developed for heavy occlusion sequences. First, multiple correlation filter classifiers are used to build the independent part

trackers. Then, the weights for the response map of each part are adaptively learned by peak-to-sidelobe ratio and smoothness constraint. Finally, to enforce the spatial layout of tracking parts, a structural constraint mask is proposed to calculate the likelihood for joint confidence maps. Experimental results demonstrate that the proposed tracking algorithm with multi-part strategy can significantly improve the tracking performance

- A color feature learning framework is developed for object tracking. First, the generic features are learned from a person re-identification dataset. Then, the learned color patterns are applied to the correlation filter based trackers instead of traditional HOG. Finally, the marginal regression based sparse coding method is proposed to speedup the training and feature extraction procedure. Experimental results demonstrate that our learned color features are robust to complicated environment and are able to improve the performance of the baseline tracking system.
- A multi-model tracker which can detect the happening occlusion is developed. First, a novel occlusion estimation method is developed base on the high-level semantic descriptors, which is trained on the large ImageNet dataset. Then, a new multi-model tracking algorithm is proposed to avoid tracking model contamination and enhance the robustness of various appearance changes. A multi-classifier selection scheme is used to choose the robust trackers trained on different convolutional layers. Finally, a linear motion model is introduced to effectively predict the lost target and realize accurate re-detection. Extensive experiments have been done to prove the effectiveness of proposed tracker.

1.4 Organization of the Thesis

This thesis is organized as follows:

In Chapter 1, motivations and contributions of the thesis are introduced.

In Chapter 2, the related work in the literature is reviewed.

In Chapter 3, a temporally smooth visual tracking method is proposed to improve the robustness of sparse coding based tracking method. Based on marginal regression, we calculate the ℓ_1 norm efficiently. Experimental results are presented to demonstrate its effectiveness.

In Chapter 4, a part-based visual tracking method is proposed to solve the partial occlusion problem. To realize real-time visual tracking, the correlation filter is adopted to independently track the parts of targets. We develop an adaptive weighting scheme to combine the scores of responses of individual parts. Experiment results proofs that with our method, better tracking accuracy can be achieved.

In Chapter 5, a color feature learning method is proposed for object tracking. To capture the essential information of colors under different illumination, which is the key point of tracking problem, we learn the color patterns offline from pixels sampled from images across different camera views. Experimental results demonstrate that significant improvement can be achieved by using our learned color features, especially on the video sequences with complicated lighting conditions.

In Chapter 6, an occlusion detection method and a multi-model tracking method are developed for visual tracking. The classification scores pre-trained from the ImageNet dataset are used to evaluate the tracking performance. To further improve the robustness of proposed tracker, selective convolutional layers and multiple tracking models are used to train the tracker. A re-detection method is proposed to find the lost target when the appearance of target changes significantly.

Finally, conclusions and future works are given in Chapter 7.

Chapter 2

Related Work

In this chapter, a survey of related work in visual tracking is presented. In the following subsections, some of the popular tracking models are introduced. In details, sparse coding trackers, correlation filter trackers and deep learning trackers are specifically described following each subsection respectively.

A huge number of visual tracking methods have been reported in literature. The existing visual tracking methods are generally classified in two categories: generative and discriminative.

2.1 Generative Tracking

The generative tracking methods adopt an appearance model to express the target observations. They cast tracking as searching for the candidates/regions which are the most similar to the targeting objects. Most of the generative trackers usually online update the appearance model of the target to adapt to appearance variations. Below, we review some of the popular generative tracking methods based on different representation methods of appearance.

2.1.1 Template representation trackers

Templates representation is an important technique in visual tracking tasks. Expectation Maximisation (EM) algorithms are used in some trackers to renew the object

representation or update the tracking models. For example, in [3], the authors develop a wavelet-based mixture model for a motion-based tracker, in which the EM algorithm is used to update the object representation. [4] uses the EM method for the representation model in the particle filter based tracking framework. The main strength of EM algorithm is simple. However, due to the considerable number of iterations, it is pretty easy to achieve local optima. Meanwhile, sometimes the slow convergence will make the tracker lost the target in long term tracking.

[5] develops a level set based tracking method using incremental principal component analysis (IPCA). Although in the above IPCA based method, the tracker achieves the newest object appearance adaptively, only one sample can be considered during updating. Combining with IPCA, [6] uses the histograms of oriented gradients (HOG) to describe the images of hand.

Shape is another important information used in generative tracking tasks. In [7], the bootstrapper is used to extract shape information. [8] builds the tracker with templates of the shape samples. The shape information is learned by temporarily saving target samples in the high dimensional feature space. However, because of the complicated computational cost, the shape based trackers can hardly run in real-time. Meanwhile, like some other active-contour trackers, robust tracking performance is determined by an empirical selection of parameters. Hence the contribution of these models is limited to some degree.

[9] proposes an adaptive tracking algorithm, in which several types of features (flatness, texture, sketch, etc.) are used to develop a hybrid template. The hybrid template is learned by a simple and effective generative model from the batch of observations. The main shortcoming of this algorithm is that they cannot replace the invalid patches during tracker updating. To solve this problem, [10] proposes a composite template object tracking algorithm. The tracker learns and updates the Composite Patch-based Templates (CPT) of the tracking object. HOG, CS-LBP, and color histogram are exploited

to describe the partial distributions of flatness, texture and sketch of each composite template.

[11] proposes the visual tracking decomposition (VTD), in which previous observational model is divided into several basic appearance observations to model the tracking object. These basic observations are dynamically built by sparse PCA step by step. And they represent different aspects of the tracking object respectively. The final tracker is constructed by these basic models, which focus on the the specific change of targets or backgrounds separately. Limited by the fixed amount of basic models, VTD cannot tackle the significant changes between frames.

Different from VTD, [12] presents a model sampler to dynamically choose the proper models to make it robust to various appearance changes. However, because of the high computational complexity of the selection of appropriate templates, this algorithm cannot satisfy the requirements of real-world application, either.

2.1.2 Subspace representation trackers

Subspace representations are also widely used to compactly represent the tracking targets. Comparing with template based trackers, the probabilistic model with subspace algorithms usually take less computation cost. Using the probabilistic Markov model, [13] proposes a dynamic probabilistic tracker to renew the models of tracking objects. Considering the changing sample mean in [13], [14] updates the model with an incremental mean. To better handle the changes caused by intrinsic or extrinsic elements, the target is represented by a learned low-dimensional eigenspace. To emphasize the latest appearance model, [15] introduces a forgetting factor to renew the incremental subspaces. The appearance model is extracted from IPCA. Although this method can refresh the tracker more robustly comparing with traditional naive updating scheme, without evaluation of the updated models, the tracker may delete the good targets while update bad ones.

[16] proposes an online incremental tracker, in which a pre-trained appearance model of a specific category of objects is used to represent tracking targets. The representation of the aiming object of this class is learned from the video sequences, which include the aiming object instances. Benefiting from the complex strategy and the prior generic appearance, this algorithm yields the more accurate tracking performance. However, because this algorithm requires a generic prior model, it cannot track arbitrary objects without having the model of this specific object class.

Through combining retinex images with original ones, [17] creates a weighted tensor subspace to represent the target. The illumination changes and other appearance changes can be solved by incremental updating. [18] exploits the covariance matrices of image features to exploit both spatial and statistical information of the targeted object. An eigenspace representation for each mode of the object is incrementally learned; the tracking model is updated over time to adapt to various appearance changes. However, because of the complex calculation of covariance matrices, it is difficult to run this tracker in real-time.

[19] presents an efficient single object tracking method which incrementally updates the covariance model. Similar to the above mentioned forgetting weight, newer samples will be assigned with a larger weight to influence the estimation of representation more.

[20] develops an incremental 3D discrete cosine transform (3D-DCT) algorithm to reduce the computational complexity. The authors decompose the 3D-DCT into the 2D discrete cosine transform and 1D discrete cosine transform. The 3D-DCT is based on a set of cosine basis functions, which are computed through the 3D signal and they are independent from the input video.

2.1.3 Sparse representation trackers

The goal of sparse coding is to find succinct representations of stimuli. It aims to discover the basis vectors that are capable to obtain higher-level features in the data.

Recently, sparse representation based trackers are becoming more and more popular because of their robustness in long term tracking. When sparse constraint is applied, most of the information of object can be simply described by a few basis vectors. Then the tracker aims to looking for candidates with minimum reconstruction error. According to [21], the target objects are represented by the linear combination of the object and trivial templates. Subsequently, tracking is treated as a sparse approximation through solving the ℓ_1 norm optimization. Hence, this kind of tracking algorithm is called L1-tracker (L1T). In addition, the authors use the trivial templates to solve appearance variances such as partial occlusion. To speedup the tracking speed of L1T, [22] proposes compressive sensing tracking, which takes advantage of the information recovering capability of compressive sensing. The above mentioned trackers performs robustly in most simple environment. Nevertheless, it cannot handle significant pose change and heavy occlusion of the tracking object.

[23] proposes a two stage sparse optimization algorithm for visual tracking. The two stage sparsity tracking selects the sparse features which can minimize the object reconstruction loss and maximize the discriminatory. Because the object is modelled as a single entity, the tracker fails to tackle the heavy occlusion very well.

Different from previous sparse based trackers, the appearance of objects in [24] is represented and refreshed with the raw intensity values of the object parts. The authors exploit multiple subspaces to encode corresponding kinds of appearances. Then, through computing the similarity between the candidate and the learned appearance model to search for the target.

[25] formulates object tracking into a multi-task sparse learning based particle filter framework and learn the particle representations jointly by using the sparsity-inducing mixed norms. The tracker described in [26] represents the target as a linear combination of subspaces based on the learned dictionary. A set of learned eigenvalue and partitioned occlusion templates is used to conduct the dictionary. [27] creates a structural local

sparse appearance model for visual tracking. By using their proposed alignment-pooling method, the tracker can exploits both partial information and spatial information of the target. Pooling from certain spatial layout of the local patches helps the tracker to handle occlusion well. [28] proposes a non-local self-similarity (NLSS) based sparse coding tracker (NLSSC). It learns the sparse representations by considering the geometrical structure of the set of target candidates. The templates are refreshed online to tackle the appearance changes. Although the structural information based sparse trackers usually perform robustly, the low efficiency of these approaches often limit their practical application.

2.1.3.1 Marginal regression for sparse solution

The speed of sparse coding based trackers is usually slow. In the past, several methods have been proposed to improve the speed, however it is still too slow for practical applications. Recently the Lasso [29, 30] is a popular tool for ℓ_1 optimization. It has many available algorithms and has been proven success in many fields. However, even using the best available methods, the Lasso solutions are still computationally time consuming, especially when the number of covariates is significantly larger than that of data points. The typical complexity of Lasso algorithm requires $O(p^3 + np^2)$ operations.

In most of the existing sparse coding based trackers, Lasso is used to generate the sparse codes. However, the speed of these trackers is usually slow for real work applications. To overcome this limitation, in Chapter 3 and 5, we adopt the marginal regression to obtain sparse codes efficiently for visual tracking. In this section, we compare the difference between Lasso and marginal regression and introduce how marginal regression can be used to replace Lasso and other complicated operations.

Consider a regression model, $Y = D\beta + z$, where $Y = (Y_1, Y_2, \dots, Y_n)^T$, D is a $n \times p$ design matrix, coefficients $\beta = (\beta_1, \beta_2, \dots, \beta_p)^T$ and $z = (z_1, z_2, \dots, z_n)^T$ is the noise variable. For sparse coding, the main problem is variable selection: determining which components of β are non-zero.

Recently, Lasso has become one of the main algorithms to optimize sparse coding problem. When considering the regression model, Lasso is defined as:

$$\hat{\beta}_{lasso} = \arg \min_{\beta} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1 \quad (2.1)$$

where $\|\beta\|_1 = \sum_i |\beta_i|$ and λ is the tuning parameter. The gradient descent algorithms and LARS algorithm [31] for Lasso require $O(p^3 + np^2)$ operations. The former ones are usually faster than LARS in practical applications. Although as a convex optimization problem, lasso can solve the sparse coding problem accurately, the speed of it is too slow for visual tracking scenarios.

Marginal regression [32] (also called correlation learning, simple thresholding [33], and sure screening [34]) is an efficient method for variable selection in which the outcome variable is regressed on each covariate separately and the resulting coefficient estimates are screened. To compute the marginal regression estimates for sparse coding, we begin by computing the marginal regression coefficients, assuming D has been standardized, we calculate:

$$\hat{\alpha} = D^T Y \quad (2.2)$$

Then, we threshold $\hat{\alpha}$ using a parameter $\tau > 0$:

$$\hat{\beta}_i = \begin{cases} \hat{\alpha}_i & |\hat{\alpha}_i| \geq \tau \\ 0 & otherwise \end{cases} \quad (2.3)$$

We sort these coefficients in terms of their absolute values, and select the top largest coefficients whose ℓ_1 norm is bounded by τ . The thresholding parameter τ is selected by cross validation.

This procedure requires just $O(np)$ operations. When p is much larger than n , marginal regression provides two orders of magnitude speedup over Lasso. This is a significant advantage of marginal regression because it is now tractable for much larger problems. [34] and [35] introduce more details about the comparison of statistical properties of marginal regression and Lasso.

Since sparse representation plays an important role in generative visual tracking methods, in Chapter 3, we aim to improve the speed of traditional sparse coding based tracking methods through the help of marginal regression. Meanwhile, sparse coding is also adopted in Section 5 to improve the robustness of learned feature. Similarly the marginal regression is used to speedup the algorithm.

2.2 Discriminative Tracking

Different from generative trackers, discriminative tracking approaches treat object tracking as a classification problem. Discriminative tracking methods take advantages of the information of the target and surroundings at the same time. Then, they train a binary classifier to discriminate the object from backgrounds. The classifier is updated over time to adapt to appearance variances. The discriminative algorithms are also termed as tracking by detection. Some discriminative tracking methods are introduced in the following sections.

2.2.1 Online updating trackers

The below trackers are based on online learning methods. Based on the initial position of the tracking object in the current frame, the tracker aims to search for possible locations from a searching window.

[36] applies AdaBoost to construct a strong tracker via an ensemble of weak classifiers. The weak classifiers are learned from multiple training sets based on features like colour or orientation histogram. The strong tracker aims to discriminate targets from backgrounds by labeling corresponding pixels (from target or background). Benefiting from the simple tracking strategy, the tracker can run fast. However, because Adaboost is sensitive to noise, the performance of this tracker is not robust to complicated situations. In [37], the authors propose a tracker, which applies an online-AdaBoost-based classifier to select the most discriminative features from a feature pool. Similar to ensemble tracker, they also

build a strong classifier through the combination of multiple weak classifiers. And these weak classifiers are updated during tracking to solve appearance changes. In addition, the tracker can replace the bad weak classifier by a new one from the feature pool to improve the robustness of the final tracker. The above mentioned tracking approaches can effectively handle simple appearance changes and even partial occlusion. However, without evaluation of the updating models, the aggregation of wrongly updated models may cause drifting.

[38] develops an on-line semi-supervised boosting method for tracking applications. The authors formulate the update process in a semi-supervised boosting scheme by combining the decisions of prior and current classifiers. The tracking model is updated by an assistant tracker which is learned from the initial frame. Different from traditional supervised learning, [39] introduces a multiple instance learning (MIL) tracker. The MIL tracker updates the appearance model with a set of image patches. By associating multiple instance learning and semi-supervised learning, [40] proposes a coherent framework for visual tracking. The proposed tracker achieves more robust tracking performance than MIL. However, the computational complexity of MIL trackers is usually expensive because they have to consider the uncertainty of instance labels. Hence, [41] develops a batch mode adaptive MIL. They use multiple sets of bags as training bags and these bags are adopted separately. Then, the classifier is refreshed with the newest pre-learned ones. [42] presents an online boosting tracking algorithm, which updates the weak classifiers by themselves instead of using new ones. Since the parameters of the weak classifiers are modified based on the new data subset presented, the proposed tracker does not need to consider 1) how many weak classifiers to be replaced to capture the change in the data and 2) which search method to use to update the weak classifiers. [43] analyzes of the one class version of MIL method instead of the standard two-class problem which assumes both positive and negative bags are available. An SVM-based formulation is used to in this algorithm. A iterative heuristic algorithm is adopted to speedup the algorithm. In

[26], a tracker named Laplacian ranking support vector tracking is presented. The weakly learned model from the new frame are associated with the information achieved from first and latest frames. Through the proposed combination, the tracker is able to adapt to appearance changes. [44] proposes an online random forests algorithm. The authors adopt it to single object tracking using the traditional Haar features. By combining ideas from on-line bagging and extremely randomized forests, the authors propose an on-line decision tree growing procedure. A temporal weighting strategy is used to adaptively remove trees via the out-of-bag-error during certain frame intervals. The tracker described in [45] utilises compressive sensing theories to reduce the dimension of the feature space. The appearance model is based on features extracted from the multi-scale image feature space with data-independent basis. The non-adaptive random projections is applied to preserve the structural information of the target. A sparse measurement matrix is used to extract the features. Similarly, the tracking task is formulated as a binary classification via a naive Bayes classifier.

Because most of the above mentioned trackers use the online learning scheme, the previously learned candidates are easily ignored. However, in most tracking sequences, the first frame always contains more stable information of the target appearance. Thus, once the deviation occurs, the trackers may fail to relocate the target over time. In contrast to previous algorithms that handle tracking by employing semi-supervised or MIL, [52] introduces an on-line learning method with complementary tracking approaches. The proposed tracker uses a simple template model as a non-adaptive and thus stable component. An optical-flow based mean-shift tracker and an on-line random forests are used to learn the target appearance. [46] introduces an online learning tracker, which takes advantages of both new and old information using the weighted reservoir method. A principled approach is used to optimize the weighted risk using weighted reservoir sampling. Because the historical information is not completely forgotten, the association of recent and historical information helps the tracker to solve drifting problem. [47] builds

a set of observation models from various learning and tracking periods. The proposed observations are learned from multiple ranges of samples, with corresponding subspace features. An effective combination and temporal inference is then associated with a particle filter which includes different stages of important samples. In reference [48], the attentional visual tracking (AVT) algorithm is developed, considering the aspects of spatial selective attention. A set of attention parts are used to model the tracking object. Through learning of the historical data, the tracker can select a subset of these attention parts. [49] proposes a new tracking framework named Tracking-Learning-Detection (TLD) that explicitly decomposes the long-term tracking task into tracking, learning and detection. An adaptive short-term tracking algorithm is developed using the Kanade Lucas Tomasi algorithm. Additionally, in [50], a random forests based detector is developed. [51] proposes the positive negative learning approach to evaluate false positives and false negative. Moreover, it can generate the training candidates and renew the appearance model through the evaluation result. [52] proposes an online structured output support vector machine (SVM) to adaptively track the targets. The kernelized structured output support vector machine is learned online to provide adaptive tracking. Hence, the tracker is named as Struck. To speedup the tracking method, a budgeting mechanism is adopted to prevent the unbounded growth in the number of support vectors.

Although above mentioned trackers performs robustly in most tracking situations, there are still many challenges that need to be solved. For example, the TLD cannot tackle the out-of-plane rotation challenge well; and Struck cannot solve the scale changes well.

2.2.2 Metric learning trackers

Metric learning is another important technique that has been widely used in visual tracking tasks. Although there are many available distance metrics can be directly exploited (e.g. Bhattacharyya, Euclidean, relative entropy and Jeffries-Matusita algorithms),

the predefined metric cannot maintain that the final match should be the tracking object. Hence, metric learning approaches have been used to train metric dynamically through searching for the best representation of feature of targets. In [53], a new tracking algorithm is developed by using the Mahalanobis based distance. Through efficient gradient-based optimisation, the appearance representation and candidates matching are performed simultaneously. However, due to the complicated learning processes, the algorithm is time consuming, thus cannot be applied in real-time applications.

[54] presents a new discriminative tracking framework based on the neighbourhood component analysis. Training samples are collected over time from the video. The tracker automatically learns the optimal distance metric for accurate matching. [55] proposes a tracking method by introducing sparsity regularization for metric learning (or SRML). This regularization aims to find the lowest possible dimensionality of the projection and thereby determining the best order. The process can be treated as the minimum description length regularization in metric learning. Similarly, [56] combines the supervised distance metric learning with nearest-neighbour classification to solve single object tracking tasks. The appearance of object is represented via a spatial-pyramid structure, which considers both the shape and texture information of objects.

2.2.3 Correlation filter trackers

As discriminative models, correlation filter is widely used in visual tracking. Correlation filter is a supervised technique for learning a linear classifier or a linear regressor. In [57], Minimum Output Sum of Squared Error (MOSSE) filter is introduced for visual tracking. The grayscale samples are used to train the filter. Using an adaptive training scheme, MOSSE is considerably robust and efficient in tracking.

By exploiting the theory of Circulant matrices, [58] develops a fast learning and detection method with the help of Fast Fourier Transform (FFT). This can be done in the dual space of kernel machines as fast as with linear classifiers. They provide

closed-form solutions for training and detection with several selections of kernels (such as Gaussian and polynomial kernels).

To further improve the performance of [58], [59] proposes a new Kernelized Correlation Filter (KCF) tracking framework. The authors propose a fast multi-channel extension of linear correlation filters, via a linear kernel, which we call Dual Correlation Filter.

[60] exploits color attributes for better tracking in color videos. They extend the learning scheme for the tracker described in [58] to multichannel color features. Then, a low dimensional adaptive extension of color attributes is proposed to extract the color features.

To tackle the issue of the fixed template size in kernel correlation filter tracker, an effective scale adaptive scheme is proposed in [61]. Moreover, the powerful features including HoG and color-naming are integrated together to further boost the tracking performance.

In [62], the scale estimation is handled by learning discriminative correlation filters based on a scale pyramid representation. The tracker learns separate filters for translation and scale estimation.

However, all aforementioned correlation filter based trackers used the handcrafted features which are not robust enough to solve the drastic appearance changes.

2.2.3.1 Kernelized Correlation Filter tracker

As one of the most popular correlation filter based tracking algorithms, the Kernelized Correlation Filter tracker is widely used in visual tracking recently [59]. In Chapter 4, 5 and 6 of this thesis, the Kernelized Correlation Filter is used to develop the proposed tracking algorithms. In this section, we introduce the Kernelized Correlation Filter based tracking method in detail.

As states in the convolution theorem, the convolution of two patches can be computed by element-wise product in the Fourier domain, which will be much more efficient.

Through the help of Fast Fourier Transform, the correlation filter based trackers can calculate the correlation in the Fourier domain efficiently. While computing the tracking locations, these correlation responses can be transformed back into the spatial domain using the inverse Fast Fourier Transform.

The classifier (also called tracker in visual tracking) of KCF is trained using an image patch x with the size of $W \times H$. We can crop the training patches based on the center of previous tracked targets. Through exploiting the cyclic property and appropriate padding, the correlation filter trackers are able to cover all possible shifts $x_{w,h}$, $(w, h) \in \{0, \dots, W-1\} \times \{0, \dots, H-1\}$. And all these shifts can be seen as the training samples of the tracker. The tracker uses a Gaussian function to model the regression targets y . Then the value of a centered target is set to 1, and the values around the target smoothly decay to 0, ie. $y(w, h)$ is the label of $x_{w,h}$.

In order to find a function $f(z) = w^T z$ that minimizes the squared error over samples $x_{w,h}$ and their regression targets $y(w, h)$, we use the following function to train the models,

$$\min_w \sum_{w,h} |\langle \phi(x_{w,h}), w \rangle - y(w, h)|^2 + \lambda \|w\|^2, \quad (2.4)$$

where ϕ represents the mapping to the Hilbert space induced by the kernel κ . The inner product of x and x' is computed as $\langle \phi(x), \phi(x') \rangle = \kappa(x, x')$. λ is a parameter for the regularization term. Because the KCF transfers the linear problem to a non-linear feature-space $\phi(x)$, we can represent the solution w as $w = \sum_{w,h} \alpha(w, h) \phi(x_{w,h})$.

$$\alpha = \mathcal{F}^{-1} \left(\frac{\mathcal{F}(y)}{\mathcal{F}(k^x) + \lambda} \right), \quad (2.5)$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the Fourier transform and its inverse respectively; $(k^x) = \kappa(x_{w,h}, x)$. Here the vector α contains all the $\alpha(w, h)$ coefficients. As the target appearance, $\hat{x}$ will be updated during tracking. The models of correlation filter tracker consist of the learned target appearance $\hat{x}$ and the transformed classifier coefficients $\mathcal{F}(\alpha)$.

During tracking, a new patch z with the same size of x will be cropped out. Then the confidence score is computed using the following equation,

$$\hat{f}(z) = \mathcal{F}^{-1}(\mathcal{F}(k^z) \odot \mathcal{F}(\alpha)), \quad (2.6)$$

where $\odot$ denotes the element-wise product.

2.2.4 Deep learning trackers

With the development of computation power and requirement of large-scale visual datasets, Deep Neural Networks is becoming increasingly popular in computer vision. With the strong capabilities of learning feature representations, convolutional neural networks (CNN) have achieved record breaking performance in many computer vision fields, such as image classification [63], object detection [64], and saliency detection [65]. Different from traditional hand-crafted features, CNN features are usually learned from massive annotated visual data and considerable number of labelled categories (such as ImageNet [66]). Hence, these deep features can carry rich high-level semantic information and are better at distinguishing objects of different classes comparing with hand-crafted features. Recently many popular Deep Neural Network based tracking algorithms are developed due to the outstanding representation power of DNNs [67, 68]. Since most of existing deep learning based trackers are proposed based on the discriminative tracking frameworks, we give a brief review of these deep trackers in this subsection.

Fan et al. [69] propose a tracking algorithm based on a pre-trained network. They train a human-specific network for people tracking.

Li et al.[67] present a target-specific CNN for visual tracking, where the CNN is trained incrementally during tracking with new examples obtained online.

Another tracking algorithm based on a pre-trained network is developed by [70]. They train a stacked denoising autoencoder using a large number of images to learn generic image features.

[71] proposes an online visual tracking algorithm by learning discriminative saliency map using CNN, which is pre-trained on a large-scale image repository offline. The tracker takes outputs from hidden layers of a pre-trained CNN as feature descriptors. Then, the tracker uses Support Vector Machine (SVM) to learn discriminative target appearance models based on the CNN features. A target-specific saliency map by back projecting CNN features is adopted to search for the final tracking result in each frame.

[72] exploits features extracted from convolutional neural networks trained on object recognition datasets to improve tracking accuracy and robustness. The authors interpret the hierarchies of convolutional layers as a nonlinear counterpart of an image pyramid representation and exploit these multiple levels of abstraction for visual tracking. Then, the CNN features are used to train correlation filters to track the targets.

In [73], a new object tracking algorithm is proposed based on fully convolutional neural network. A principled feature map selection method is developed to select discriminative features and remove noisy and irrelevant feature maps.

Some prior works treated the CNN as a black-box feature extractor which ignore the important structure information contained in the middle convolutional layers. Some very recent papers used the pre-trained network trained from ImageNet; although they considered the hidden layers to better discriminate between the target and background, most of these algorithms failed to exploit the classification layer of the CNN, which includes more semantic information.

2.3 Hybrid Generative-Discriminative Tracking

Although the above mentioned trackers have achieved fairly good performance, the generative and discriminative methods have their own disadvantages in visual tracking. Because the generative tracking frameworks only pay attention to the appearance of tracking object, they are easy to lose the target with cluttered backgrounds. On the

other hand, the discriminative methods are sensitive to complicated scenarios and usually sustain the drifting because they are easy to be impacted by other objects with similar presentation of the target. Hence, to achieve more effective tracking approaches, some works are attempting to combine the generative and discriminative algorithms. The hybrid generative-discriminative model based methods aim to find a good trade-off between the generative and the discriminative appearance models. Some of the hybrid generative-discriminative model based methods are as follows.

[74] proposes a tracker using an online feature selection strategy. By referring to a localized generative appearance model, the tracker is able to solve occlusion. Discriminative classifiers based on feature extraction have prepared a fixed prior model by training offline, and adapted the parameters to appearance changes. According to the authors, the generative model exhibits the properties of non-negativity, localization and orthogonality.

In [75], a graph based discriminative tracker is developed based on ISL and Fisher discriminant analysis. The subspace of the tracked object and the topology structure of graphs are learned over time to distinguish targets from surroundings. The main limitation of this tracker is the error accumulation during updating.

[76] proposes a co-training framework to combine the generative tracker with the discriminate tracker. In their algorithm, the generative tracker builds a compact representation of the appearance of target through learning a set of subspaces. The discriminative one adopts the SVM to learn local appearance. The tracker has powerful reacquisition capability. Nevertheless, it cannot robustly handle sudden appearance changes or occlusion.

[77] presents a novel tracking algorithm by exploiting sparse coding and particle filters. The appearance of candidates is encoded by target and background templates. The additive construction error is used to make the tracker robust to various appearance changes. The templates of target and background are prompted based on competitive

orders to improve the capability of discrimination. Both types of templates are trained using a weight based scheme. An online basis learning method is applied to model the error basis sets.

Based on the framework of sequential Bayesian learning, [78] proposes a generative model-based tracker with sequential Gaussian mixture model (SGMM) and a discriminative model-based tracker with fast relevance vector machine (RVM) to solve the problem of tracking regions in many challenging situations. With a three-level hierarchy, the authors also investigate how to combine generative and discriminative models for visual tracking.

[79] proposes a new solution to kernel based visual tracking, which performs better than conventional single-view kernel trackers. A robust object representation model is trained from a large amount of data using SVM. Tracking is then viewed as a binary classification problem, and a discriminative classification rule is learned to distinguish the target from surroundings. The target is searched by maximizing the classification score.

[80] proposes a object tracking algorithm using a collaborative model. They adopt the appearance model that exploits both holistic templates and local representations. A sparsity-based discriminative classifier (SDC) and a sparsity-based generative model (SGM) are simultaneously used in their tracker. The update scheme considers both the latest observations and the original template to deal with appearance changes.

Although these hybrid trackers have the advantages of both generative and discriminative tracking methods, due to the complex framework structure and high computational cost, it is difficult for them to realize efficient tracking. To overcome the speed issue of hybrid trackers, in Chapter 4, by exploiting multiple correlation filters, we introduce an efficient hybrid tracking method. It firstly uses the discriminative filters to generate the confidence maps and adopts a generative framework to track the target based on the map. Extensive experiments have been done to show the capability of our proposed hybrid tracker.

Chapter 3

Visual tracking via temporally smooth sparse coding

Sparse representation has been popular in visual tracking recently for its robustness and accuracy. However, for most conventional sparse coding based trackers, the target candidates are considered independently between consecutive frames. In this chapter, a novel sparse coding based tracking method is developed, which exploits the temporal correlation of consecutive frames to improve the performance of tracking and to make the tracker more robust to noise. To improve the tracking speed, we revisit a more efficient method for ℓ_1 norm problem, marginal regression, which can solve the sparse coding problem more efficiently. Consequently we can realize real-time tracking based on the temporally smooth sparse representation. Extensive experiments have been done to demonstrate the effectiveness and efficiency of our method. This work is published in [81].

3.1 Introduction

Visual tracking plays an important role in computer society with various applications such as security surveillance, auto driving and auto control systems [1, 82, 83]. With the development of single object tracking method, most of the tracking tasks in simple environment with slow motion and slight occlusion can be solved well by current algorithms.

Figure 3.1: Comparison of tracking performance obtained by trackers without and with temporal smoothness constraint. Because of the sparse coding noise, there is an apparent drift of the blue boxes that are yielded from method without temporal smoothness constraint. The red boxes are obtained from the proposed method. By considering the temporal correlation, the red features are more similar to neighbouring ones and red boxes track the target more accurately.

However, in more complicated situations more robust and faster tracking methods are required to realize real-time and accurate tracking.

Inspired by the compressive sensing in signal processing [84, 85], sparse coding [86] recently has drawn pretty much attention in computer vision society. In [87], the authors apply sparse representation in face recognition successfully. Motivated by this success, [21] try to transfer sparse coding to visual tracking and obtain fairly good performance. As introduced in Section 2.1.3, the sparse representation has been approved to be robust against significant appearance changes in visual tracking. However, the slow tracking speed caused by ℓ_1 norm optimization is the main limitation of the state-of-the-art sparsity trackers in practice. Hence, developing an efficient solver for the ℓ_1 minimization has been a key to make the sparse coding based trackers more useful.

Currently most sparse coding based trackers [21, 25, 27, 80, 88, 89] are using the Lasso to solve the ℓ_1 norm related problem which is computationally expensive. The solver for the ℓ_1 norm minimizations used in [21] is based on the interior point method which turns out to be too slow for tracking. A minimal error bounding strategy is introduced in [88] to reduce the number of particles to equal to the number of the ℓ_1 norm minimizations for solving. A speedup by four to five times is reported in [88], but it is still far away from being real time. Hence, in this chapter, a more efficient method, marginal regression is adopted to solve the ℓ_1 norm problem to improve the tracking speed. Readers may refer to Section 2.1.3.1 for more details about marginal regression and the difference between Lasso and marginal regression.

Additionally, the conventional sparse coding based trackers treat the candidates independently between consecutive frames. Apparently, the information of neighbouring frames should be helpful to improve the stability of tracking results, because the targets usually change slightly between adjacent frames in the tracking sequences. We should take advantage of the temporal correlation to enhance the tracker. Thus, we aim to propose a temporally smooth sparse coding method to model temporal correlation for tracking.

Motivated by [27] and recent progress of marginal regression, we propose a novel tracking algorithm, called Temporally smooth Tracker via Marginal Regression (TMRT), which explicitly considers the relationship of neighbouring frames and take advantage of marginal regression to dramatically improve the tracking speed. The experimental results show that our tracker can achieve high speed and high accuracy.

3.2 Temporally smooth tracking based on sparse representation

In this work, we propose to develop an efficient sparse coding based tracking algorithm which also takes the information between consecutive frames into consideration. By taking advantages of the constraint term between neighbouring frames, our tracker performs robustly in noisy situations, especially in motion blur challenges. To improve the speed of our sparse coding tracker, marginal regression is used to optimize the ℓ_1 norm problem. Although marginal regression is one of the main contributions of this work, since we have introduced marginal regression in detail in Section 2.1.3.1, we will not repeat the similar content in this chapter. Readers may refer to the above mentioned section for more information. In the following parts of this section, we will introduce the basic sequential inference model which is used in our proposed tracker first. Then the sparse representation and proposed temporally smooth tracking method are introduced respectively.

3.2.1 The basic sequential inference model

The visual tracking problem is usually carried out as an inference task in a Markov model with hidden state variables. Let x_t denote the state variable describing the parameters of an object at the time t (e.g. location or motion parameters) and define $Y_t = [y_1, y_2, \dots, y_t]$ as a set of observed video frames. The optimal state $\hat{x}_t$ is computed by the maximum a

posterior (MAP) estimation

$$\hat{x}_t = \arg \max_{x_t^i} p(x_t^i | Y_t) \quad (3.1)$$

where x_t^i is the state of the i -th frame. Using Bayes' theorem, we have

$$p(x_t | Y_t) \propto p(y_t | x_t) \int p(x_t | x_{t-1}) p(x_{t-1} | Y_{t-1}) dx_{t-1} \quad (3.2)$$

The $p(y_t | x_t)$ is the observation model.

The dynamics between states in this space is usually modelled by the Brownian motion. Each parameter in x_t is modelled independently by a Gaussian distribution given its counterpart in x_{t-1} .

$$p(x_t | x_{t-1}) = \mathcal{N}(x_t, x_{1:t-1}, \Psi) \quad (3.3)$$

where Ψ is a diagonal covariance matrix whose elements are the corresponding variances of affine parameters. The observation model $p(y_t | x_t)$ denotes the likelihood of the observation y_t at state x_t . $p(y_t | x_t)$ is proportional to the similarity between the candidate and the target.

3.2.2 Sparse representation in tracking

Our method is developed based on the sparse representation method [27], hence we first give it a brief review. For the dictionary construction, we have N target templates $T = [T_1, T_2, \dots, T_N]$. Then M overlapped local image patches are extracted from the target region of each template based on a spatial layout. These local patches consist the dictionary, $D = [d_1, d_2, \dots, d_{(N \times M)}] \in \mathbb{R}^{X \times (N \times M)}$. Here X means the dimension of the image patch vector. Each local patch represents one fixed part of the target object. The target candidates are also decomposed into a collection of local patches, $Y = [y_1, y_2, \dots, y_M] \in \mathbb{R}^{X \times M}$. The dictionary captures the commonality of different templates and is able to represent various forms of these parts. The target candidates are represented by the patches from templates using sparse codes. With the sparsity

assumption, the local patches can be represented as the linear combination of only a few basis elements of the dictionary by solving

$$\begin{aligned} \min_{b_i} & \|y_i - Db_i\|_2^2 + \lambda \|b_i\|_1, \\ \text{s.t.} \quad & b_i \geq 0 \end{aligned} \quad (3.4)$$

where $b_i \in \mathbb{R}^{(N \times M) \times 1}$ is the corresponding sparse code of that local patch y_i , and all the elements of b_i are nonnegative. Note $\beta = [b_1, b_2, \dots, b_M]$ represents the sparse codes of one candidate. The sparse coefficients of each local patch are divided into several segments, according to the template that each element of the vector corresponds to.

After obtaining the coefficients b_i from Eq. 3.4, these segmented coefficients are summed to obtain v_i for the i -th patch through $\sum_{k=1}^N b_i^{(k)}$. To improve the robustness of our tracker, we take advantage of the structural relationship of the local patches as in [27]. For the vector v_i , each local patch of the candidate is represented by the patches at the same positions of the templates. Hence the feature is represented as $f = \text{diag}(V)$. Based on 3.2.1, the observation model $p(y_t | x_t)$ is proportioned to $\sum_{k=1}^M f_k$. Refer to [27] for more details.

3.2.3 Temporally smooth tracking

As described above, the conventional sparse coding based tracking methods infer b_i independently. Obviously, in tracking videos, neighbouring frames are presumably more related to each other than frames that are farther apart. Hence, we propose a mechanism to incorporate such feature similarity and temporal information into the framework of sparse coding tracking, leading to a sparse representation with an improved tracking accuracy. We propose the formulation based on the framework of sparse coding tracking methods.

$$\min_{b_{ri}^t} \|y_{ri}^t - Db_{ri}^t\|_2^2 + \lambda \|b_{ri}^t\|_1 + \gamma \|b_{ri}^t - b_{r*i}^{t-1}\|_2^2, \quad (3.5)$$

where y_{ri}^t denotes the i -th local patch of the r -th candidate at the frame t ; b_{ri}^t is the corresponding sparse code at the frame t ; b_{r*i}^{t-1} is the selected result from the frame $t-1$.

The scalar γ controls the tradeoff between the temporal smoothness and sparse constraint. Traditional sparse coding trackers minimize the reconstruction error of the encoded samples. Our proposed method TMRT, on the other hand, minimizes the reconstruction of encoded samples as well as the difference between neighbouring frames. In the t -th frame, we actually select a candidate whose sparse code is similar to that of the previous frame's result and has low reconstruction error. As a result, our tracker can be more robust to noise at individual frames.

Compared with traditional sparse based tracking methods, we have an extra pairwise constraint term. Thus, it is difficult to minimize directly. We adopt a two-step algorithm to solve the optimization problem.

We first utilize marginal regression to solve the sparse coding for each frame (Section 2.1.3.1); then based on the obtained sparse codes, we optimize the smooth part of the objective function. The sparse codes b_{ri}^t for the t -th frame are calculated by comparing the similarity with previous tracking results. In the second step, in addition to the smooth constraint term $\|b_{ri}^t - b_{r^*i}^{t-1}\|_2^2$, we also consider the reconstruction error obtained from the first step. The reason is that in many situations, such as heavy occlusion, if we ignore the reconstruction error, the tracking results may drift to the occluded object which can optimize $\|b_{ri}^t - b_{r^*i}^{t-1}\|_2^2$. Hence, we also consider the impact of $\|y_{ri}^t - Db_{ri}^t\|_2^2$. In our experiments, when the value of γ is between 10 to 50, we can get similar results. In our experiments, we just simply set γ as 10.

Our method is illustrated with an example as shown in Fig. 3.1. There are two candidates (expressed as red and blue boxes) at frame 167 and 168. The feature codes of the candidates are shown on the right with corresponding colors. Without temporal smoothness constraint, the regions in blue will be considered as the target according to ASLA tracking method. However, it will result in "drifting" of the target. In fact, when we consider the temporal smoothness between consecutive frames, regions in red will be considered as the targets based on the Eq. 3.5. By considering the neighbouring

frames simultaneously, we can obtain the accurate target position through the previous and following frames even when the object is moving quickly.

Because the computation of step two is simple, the main time consuming part is solving sparse coding. As introduced in Section 2.1.3.1, to improve the speed of traditional sparse coding based trackers, the marginal regression is adopted. To avoid repetition, marginal regression will not be introduced again in this chapter. Readers may refer to Section 2.1.3.1 for more details. Hence, the function 3.4, which used to be optimized by Lasso, now is replaced by marginal regression. Through the help of this efficient sparse coding optimization method, the proposed tracker runs much faster compared with the traditional sparse coding based methods. In addition, a template update scheme is adopted in [15] to overcome pose and illumination changes.

3.3 Experiments

To evaluate our proposed trackers, we compile a set of 22 challenging tracking sequences. Four of the sequences are shown in Fig. 3.2. These videos are recorded in indoor and outdoor environments and have variations of rotation, illumination change, occlusion, etc. Besides the baseline tracker ASLA [27], we also test the MRT tracker, which uses marginal regression instead of Lasso for the baseline method. We compared the proposed algorithms TMRT with nineteen state-of-the-art visual trackers: Frag[90], BSBT [91], LOT [92], CT [45], SMS [93], KMS [94], CPF [95], DFT [96], ORIA [97], IVT [15], CSK [58], CXT [98], TLD [51], VTD [11], ℓ_1 APG [89], MTT [25], SCM [80], LSST [99], ASLA [27]. All our experiments are performed using MATLAB R2012b on a 3.2 GHZ Intel Core i5 PC with 16 GB RAM. We resize the target image patch to 32×32 pixels and extract 3×3 overlapped local patches within the target region. For all experiments, we set the number of particles as 600, the total number of target templates as 10. For fair comparison, we use the source codes provided by the authors. They are initialized using their default parameters. For the IVT, ℓ_1 APG, MTT, SCM, LSST, ASLA and our

Figure 3.2: Comparison of our approach with state-of-the-art trackers in challenging situations.

proposed methods, we use the same affine parameters (x, y translation, rotation angle, scale, etc.) for each sequence in candidates sampling.

Table 3.1: Average center location error (in pixels). The best three results are shown in red, blue, and green fonts.

Sequence	FragT	BSBT	LoT	CT	SMS	KMS	CPF	DFT	ORIA	IVT	CSK	CXT	TLD	VTD	ℓ_1 apg	MTT	SCM	LSST	ASLA	MRT	TMRT
Car4	179	57.7	165	229	140	52.9	38.7	61.9	237	2.9	19.1	49.5	18.8	12.3	16.4	37.2	3.8	2.9	4.3	4.5	2.7
Car11	63.9	8.1	30.8	78.0	88.9	30.3	43.8	21.6	26.8	2.1	3.2	15.9	25.1	27.1	1.7	1.8	1.8	1.6	2.0	2.1	1.3
David	76.7	20.9	23.9	12.8	25.1	20.3	26.7	27.5	23.3	3.6	17.6	6.0	9.7	13.6	10.8	13.4	3.4	4.3	3.6	3.7	3.4
Occ1	4.6	14.5	34.7	19.9	23.1	19.2	28.8	22.6	21.3	16.3	11.9	22.6	17.6	11.1	6.8	14.1	3.2	5.3	10.8	12.8	4.2
Occ2	15.5	15.6	14.9	12.8	29.3	28.9	21.0	7.9	5.5	10.2	5.9	6.3	18.6	10.4	6.3	9.2	4.8	3.1	3.8	4.3	4.0
Sylvester	15.0	14.7	11.3	8.6	17.8	18.1	12.8	44.9	9.3	34.2	9.9	14.8	10.5	19.6	26.2	7.5	8.0	69.6	14.6	13.1	7.2
Singer1	22.0	76.4	127	13.7	8.7	53.1	7.7	18.8	8.1	8.5	14.0	11.3	32.7	4.1	3.1	41.2	3.8	3.5	4.8	5.2	2.7
Skating1	144	57.9	110	150	230	89.7	118	168	69.6	19.3	7.8	129	96.0	9.3	101	219	16.3	17.8	6.0	5.9	5.6
Woman	111	33.2	117	113	97.6	13.9	83.8	9.5	212	167	207	72.5	47.9	118	128	137	7.9	118	2.8	3.6	2.5
Subway	15.8	81.0	4.7	11.1	139	117	80.1	13.5	120	130	159	129	79.9	141	145	157	3.5	117	63.5	68.9	3.5
Walking2	54.3	41.5	58.8	58.5	78.6	43.9	49.9	29.1	20.0	2.5	17.9	30.4	24.3	44.0	5.1	4.0	1.6	53.2	21.4	22.5	1.9
Caviar	94.2	79.6	43.9	65.5	11.1	19.3	20.9	89.1	76.6	66.2	69.1	72.6	53.0	60.9	68.6	67.5	2.2	3.1	2.3	2.2	2.0
Freeman4	47.3	46.7	38.6	93.0	105	36.0	62.0	57.5	54.5	43.0	64.7	65.2	39.2	60.8	22.1	23.5	37.7	72.3	45.5	46.7	3.5
Tiger1	74.0	64.0	111	29.9	45.8	69.6	37.3	19.0	87.0	93.2	50.4	45.4	49.5	102	58.4	59.1	75.4	86.3	55.9	52.2	18.0
Deer	50.4	27.5	65.2	10.5	78.6	43.8	79.6	98.7	149	127	5.0	6.7	25.7	11.9	38.4	9.2	36.8	10.0	8.0	7.9	4.2
Motorbike	196	41.8	24.9	187	136	54.4	211	20.2	7.1	7.7	6.5	159	195	9.8	8.3	7.1	10.6	132	9.7	10.2	4.9
Biker	76.4	96.6	65.2	21.1	14.1	95.2	74.4	122	83.5	17.6	27.3	83.0	78.3	13.2	71.8	29.2	99.8	47.3	19.0	20.1	13.1
Football	16.9	32.6	18.3	11.6	190	87.1	15.8	10.3	14.2	42.5	16.2	12.8	11.8	4.1	12.4	6.5	10.4	7.6	18.0	16.5	7.8
Jumping	58.4	33.5	5.6	53.0	37.4	84.2	56.1	65.6	76.1	36.8	49.1	7.0	3.6	63.0	8.8	19.2	3.9	4.8	39.1	42.3	4.5
Board	18.3	197	162	52.2	20.9	20.5	35.8	123	185	84.4	18.0	154	127	52.8	183	139	32.2	20.1	7.3	7.9	7.2
Surfer	22.8	40.0	14.8	17.2	15.5	19.8	25.2	42.2	48.7	84.3	157	14.9	25.5	33.9	56.8	30.2	62.1	146	51.9	50.6	9.2
Shaking	192	66.6	82.6	80.0	38.7	59.2	80.7	26.3	28.4	85.7	17.2	29.2	37.1	9.5	79.8	9.2	11.1	102	22.1	21.0	8.5

Table 3.2: Average overlap rate(%). The best three results are shown in red, blue, and green fonts.

Sequence	FragT	BSBT	LOT	CT	SMS	KMS	CPF	DFT	ORIA	IVT	CSK	CXT	TLD	VTD	ℓ_1 apg	MTT	SCM	LSST	ASLA	MRT	TMRT
Car4	22	21	4	28	5	27	19	25	23	92	47	31	64	73	70	53	89	92	89	89	92
Car11	9	43	42	23	2	36	8	38	38	81	76	57	38	43	83	58	79	84	81	82	85
David	19	39	27	56	24	38	14	30	43	72	41	65	60	53	63	53	75	75	79	79	82
Ooc1	90	77	41	74	58	72	53	69	64	85	79	63	65	77	87	79	93	89	83	82	92
Ooc2	60	64	46	68	8	46	42	77	72	59	78	74	49	59	70	72	82	86	82	81	83
Sylvester	58	57	57	68	7	47	56	38	65	52	63	60	67	62	40	65	69	28	59	60	65
Singer1	34	21	19	34	54	32	45	36	65	66	36	49	41	79	83	32	85	80	81	83	87
Skating1	13	16	26	9	4	31	19	14	22	34	50	14	19	53	10	10	47	34	42	45	57
Woman	15	19	9	13	6	57	7	76	15	19	19	20	13	14	16	16	66	78	78	79	82
Subway	46	17	56	57	18	15	12	73	17	17	19	18	18	16	16	7	72	15	19	21	67
Walking2	27	26	34	27	31	27	32	40	45	79	46	37	31	33	76	79	82	34	37	36	81
Caviar	19	14	25	33	56	42	32	14	19	21	19	19	21	19	13	14	87	85	84	85	89
Freeman4	14	15	16	0	2	3	5	17	20	15	13	17	22	16	34	22	26	13	13	36	62
Tiger	26	22	14	41	39	39	39	53	13	10	26	32	38	12	31	26	16	27	29	35	60
Deer	8	40	21	60	10	41	12	25	4	22	73	70	41	58	45	60	46	58	62	63	74
Motorbike	13	30	58	14	11	49	11	30	65	73	71	23	20	70	73	74	67	26	72	71	74
Biker	21	26	44	46	46	25	36	25	38	64	50	42	23	63	34	42	35	36	55	56	67
Football	57	29	65	46	2	8	64	65	51	55	55	54	56	81	68	71	69	69	57	51	70
Jumping	14	15	58	7	9	10	10	11	9	28	5	52	69	8	59	30	73	65	24	56	71
Board	67	10	19	45	52	65	55	31	13	33	67	17	20	47	16	19	63	40	74	73	82
Surfer	39	30	49	52	50	49	35	35	21	23	25	49	45	38	18	40	25	19	37	38	60
Shaking	8	11	13	20	25	23	12	63	44	30	57	12	39	70	28	69	68	42	46	50	75

To assess the performance of the proposed tracker, two criteria, the center location error as well as the overlap rate, are employed in our algorithm. A smaller average error or a bigger overlap rate means a more accurate result. Given the tracking result of each frame R_T and the corresponding ground truth R_G , we can get the overlap rate by the PASCAL VOC [100] criterion, $score = \frac{area(R_T \cap R_G)}{area(R_T \cup R_G)}$. Table 3.1 and 3.2 report the quantitative comparison results respectively.

We first compare the tracking accuracy of the proposed TMRT tracker to that of the same tracker without smoothing constraint (MRT). From Table 3.1 and 3.2, we can see that our tracker with temporal smoothing performs better than MRT and ASLA. It proves that the marginal regression can obtain similar convergence result for ℓ_1 norm problem compared with Lasso; and the temporal smoothing gains extra accuracy compared to conventional sparse coding based tracking methods. As shown in the tables, the proposed tracker yields favorable performance against other state-of-the-art methods.

We compare our method with other sparse coding based trackers on speed. The computational cost of these methods is huge due to the Lasso method or APG for ℓ_1 norm, even though they employ the C language to improve the solution speed as toolbox. Due to the inherent similarity between these sparse coding based L1 tracker and the proposed tracker (TMRT), we compare their average runtimes in Table 3.3. Based on the results, it is clear that our trackers TMRT is much more efficient than the other sparse coding based L1 trackers. Besides the mentioned sparse coding based L1 trackers, we also test the speed of IVT, which is the baseline of most particle filter based tracking methods. IVT can run at 16fps in our computer. Our proposed method is only 1fps slower than IVT, while getting much better performance with sparse feature and temporal smoothness constraint.

We also plot the results of IVT, TLD, L1APG, CXT, CSK, MTT, VTD, SCM, ASLA, LSST and TMRT trackers in visualization comparison. Due to the pages limitation, we only select four of 22 sequences for visualization and analysis.

Table 3.3: Running speed comparison of several popular sparse coding based L1 trackers

L1	ℓ_1 apg	MTT	SCM	ASLA	LSST	TMRT
0.1fps	1fps	2fps	0.5fps	1.5fps	5fps	15fps

In the david sequence, a moving face is tracked. The tracking results are shown in Fig. 3.2.a. LSST fails at frames 402. L1APG, MTT and CSK start to drift around frame 467. The other trackers track the moving face accurately.

As shown in Fig. 3.2.b, a stuffed animal is being moved around on the Sylvser sequence. LSST, L1APG and IVT trackers fail around frame 450, 597 and 928 respectively. Our method track the target throughout the sequence.

In the Jumping sequence, the tracked object is subject to fast location changes when the man is skipping rope. Most methods fail to track the object when the man jump quickly. LSST and TMRT methods can track the object quite well.

In the Freeman4 sequence, the target person move around a classroom while the other students are waving papers ahead. Initially, SCM, TLD, L1APG, VTD and our tracker succeed to recover from the heavy occlusion. After frame 204, only our TMRT tracker tracks the man successfully.

3.4 Conclusion

In summary, based on the framework of the sparse based trackers [21, 27, 88], we develop a real time temporally smooth visual tracker with improved tracking performance. The proposed tracker utilizes information among consecutive frames to improve the tracking accuracy and employs marginal regression to speedup the tracking speed significantly. Extensive experiments on real-world video sequences have been done to validate the high computational efficiency and better accuracy of the TMRT method.

Chapter 4

Part-based tracking via discriminative correlation filters

In order to better deal with the partial occlusion issue, part-based trackers are widely used in visual object tracking recently. However, it is still difficult to realize fast and robust tracking, due to complicated online training and updating process. Correlation filters have been used in visual object tracking tasks recently because of their high efficiency. However, the traditional correlation filter based tracking methods do not deal with occlusion well. In this chapter, we propose a novel tracking method which tracks objects based on parts with multiple correlation filters. The Bayesian inference framework and a structural constraint mask are adopted to enable our tracker to be robust to various appearance changes. Additionally, a discriminative part selection scheme is adopted to further improve performance and accelerate our method. Experimental results demonstrate that our multiple part tracker can significantly improve tracking performance on benchmark datasets. The previous version of this work is published in [101].

4.1 Introduction

Although in simple environment with slow motion and slight occlusion, most of existing trackers work well, in more complicated situations, more robust tracking methods are expected to conduct accurate and efficient tracking. In this chapter, we propose a no-

Parts trackers represented with
bounding boxes independently
(a)

Confidence maps of
individual parts
(b)

Confidence maps after
part selection
(c)

Bayesian inference tracking based on
joint confidence map
(d)

Figure 4.1: (a) and (b) show the tracking of each part tracker independently and the response maps of corresponding parts (in corresponding red colors). (c) is the selection the parts through learning which is introduced in 4.2.2. Then separate response maps are combined by Eq. 4.1. We track the whole target based on the joint confidence map in the Bayesian inference framework as shown in (d). When computing the likelihood, sampling candidates are shown as yellow rectangles. The solid yellow rectangle shows the tracking result on the confidence map with the maximum likelihood. (Given the limitation of space, we only show nine parts as examples in (a) and (b).)

vel part-based tracking method which can effectively work in heavy occlusion and other challenging scenarios. As introduced in Section 2.2.3, correlation filter based discriminative trackers have significant achievements with fairly high speed and robust tracking performance recently. In the traditional correlation filter based trackers, the correlation filters are designed to search correlation peaks given the known target object, and produce relatively low responses to background. The target object is initialized based on a small tracking window centered on the object in the first frame. Then the object is tracked by correlating the filters over larger candidate windows in the following frames. These correlation filters will be updated online during tracking. Correlation filter based trackers have been proven to be competitive with far more complicated tracking methods [57, 59, 60, 102].

Meanwhile, part-based tracking methods [90, 103–105] have also been paid more attention by corresponding researchers, particularly due to their favorable property of robustness against partial occlusion. Because they model the object appearance based on multiple parts of the target, when the target is partially occluded, we can still use the visible parts for robust tracking. These kind of methods can be viewed as tracking by matching multiple parts of the object over time in a video sequence. In [106], the authors propose a coupled-layer visual model for object tracking. The model is a combination of both local and global layers, where the local layer consists of a set of local patches that geometrically constrain targets appearance changes. [107] similarly divides a targets into multiple regions. They present a tracking algorithm based upon modeling foreground and background regions with a mixture of Gaussians. In [108], foreground shape is modeled as a small number of rectangular blocks. The algorithm tracks objects by matching the intensity histograms, and updates the shape by adjusting small blocks of the tracked window. Kwon et al. [109] present a local patch-based appearance model which is updated online. In [110], Zhang et al. track targets by matching parts among multiple frames. Another work similar to ours is [111], in which the part-based tracking problem

is solved by using latent structured learning. However, the computational cost of these methods is usually high. Consequently, it is difficult for multiple part based trackers to run efficiently.

Hence, to improve the efficiency, we aim to develop our algorithm based on the efficient correlation filters. Different from the previous correlation filter based trackers which treating target objects as holistic appearance, we divide a tracking target into several parts. Subsequently, we can take advantage of the reliable tracking parts to realize correct tracking of the whole target when the object is partially occluded.

Additionally, because visual tracking can be considered as an estimation of the state for a time series state space model, the problem can also be formulated in probabilistic terms [112, 113]. Particle filtering is a technique for implementing a Bayesian inference filter by Monte Carlo simulation, which is popular in visual tracking. Similar to Section 3.2.1, it recursively constructs the posterior probability density function of the state space using Monte Carlo integration. In our method, the joint confidence maps are used in the Bayesian inference framework to infer the candidate with the maximum posterior probability. To better enforce the spatial layout of parts, a structural constraint mask is adopted to calculate the likelihood of the observation and state.

Another similar work is introduced in [101], where fixed parts spatial layout is used. By taking advantages of the multiple parts scheme, this method can effectively deal with most of tracking challenges. However, when the bounding boxes of targets include much background information, the fixed part spatial layout cannot solve the problem well. In this chapter, we use a discriminative part selection method to choose the most discriminative parts from a number of part candidates.

In summary, this chapter introduces an efficient part-based single object tracking algorithm which is able to handle partial occlusion accurately and effectively. Through using the correlation filters as part classifiers, the part evaluation of our proposed method can be very fast. Then, the main difficulty of multi part tracking is how to properly

combine the scores of individual part tracking. Once we mistakenly assign big weights to the occluded parts, there will be tracking errors. We develop new criteria to measure the performance of different parts, and assign proper weights to them. Specifically, we propose to use Smooth Constraint of Confidence Maps (SCCM) as a criterion to measure how likely a part is occluded. The experiments prove the effectiveness of the SCCM criterion. Besides, we develop a spatial layout constraint method to: 1) effectively suppress the noise caused by combining individual parts, 2) estimate the correct size of bounding box when the target is occluded. Additionally, a discriminative parts selection method is proposed to choose the most robust parts from a number of tracking part samples. Through using such selection method, our proposed algorithm can effectively focus on more robust tracking part.

The main contributions of this chapter are as follows:

- Based on correlation filter and Bayesian inference framework, we propose a real-time part-based tracking method which is robust to occlusion and various appearance changes.
- We adaptively learn the weight parameters for the response map of each part via the peak-to-sidelobe ratio and the smoothness constraint.
- To enhance the spatial layout of tracking parts, a structural constraint mask is proposed to calculate the likelihood for joint confidence maps.
- We propose a discriminative parts selection method to enhance the performance of our method.

Extensive experiments have been done to show the effectiveness of our tracker.

4.2 Part-based tracker via correlation filter

In this section, we provide the details of our proposed method. Firstly, cause the proposed algorithm is developed based on the Kernelized Correlation Filter (KCF) [59], readers may refer to Section 2.2.3.1 for more details about KCF. In Section 4.2.1, the adaptive weighting and updating schemes are provided respectively. Subsequently, we introduce the tracking algorithm using proposed confidence maps. In Section 4.2.2, the discriminative part selection method is described.

In this work, we aim to build an efficient part-based tracking algorithm which is robust to occlusion and deformation. However, since the high computational complexity, complicated tracking systems cannot be extended to multiple parts to solve the occlusion problem. Recently, the KCF has achieved very good performance with high speed (360 frames per second). To achieve competitive performance and efficiency, we develop our part-based method based on the efficient KCF tracker. Our key idea is to employ the fast KCF classifier as the part classifier, and develop a new method to adaptively combine the part classification scores and adaptively update the model. Furthermore, to effectively solve the scale, rotation, etc. problems, a Bayesian inference framework is employed to achieve more robust performance. Because the parts can be overlap patches, intuitively, the tracker should achieve better results when dividing the objects into more parts. However, it will significantly increase the time consuming of the tracker at the same time. To balance the accuracy and efficiency, we provide a novel discriminative part selection method to further improve the tracker’s performance.

4.2.1 Proposed tracking method

The partial occlusion is one of the main challenging factors that limits the performance of current visual trackers. As discussed above, we can simply divide the target into parts and track these parts independently; when target object is partially occluded, we can still locate the target correctly relying on the other visible parts (unoccluded parts). However,

because of the complicated training and updating processes, part-based trackers usually cannot track the object fast. Benefiting from the efficient KCF tracker, we propose to develop a new part-based tracker which can run fast and achieve accurate tracking results.

Fig. 4.1 illustrates the basic flow of our part-based tracking method. The algorithm proceeds as follow. The initial position of the target object is given in the first frame. Then the tracker is required to track the object in the subsequent frames of the video. As a part-based tracker, the target object is divided into multiple parts. For each part of the object, the independent KCF tracker will be used to produce the response map corresponding to each part. These confidence maps are needed to combine together to represent the entire target in the candidate searching window. Finally this confidence map is used in the Bayesian inference framework to finish the tracking of whole targets. During the proposed tracking process, there will be two main difficulties need to be solved: (1) combination of each part confidence map and (2) updating of the correlation filter models.

To effectively combine these confidence maps from different part trackers plays a key role in this method. In different frames, different parts of targets may suffer different appearance changes, illumination variation or occlusion. If we simply combine confidence maps with the same weight, the response of falsely tracked parts may be unfairly emphasized. Given the detection result of each part tracker, the contribution made for the joint confidence map should be different from each part, ie. the response of more reliable parts should be given larger weights. Through adaptively weighting each part response, the joint confidence map puts more emphasis on reliable parts and eliminates the clutters caused by drifting parts. For correlation filter based classifier, the peak-to-sidelobe ratio (PSR) (Eq. 4.3) can be used to quantify the sharpness of the correlation peak. The higher PSR value means more confident detection. In a tracking system, the higher value of PSR means the higher matching score between current frame and previous ones. Hence, the PSR can be adopted to assign variable weights to the confidence maps of parts

under different situations. Additionally, for tracking problem, the temporal smoothness property can be exploited to detect whether the target is occluded or not. Considering this observation, we argue that the smooth constraint of confidences maps should be considered for the weight parameters. It is also validated from our experiments. From Fig. 4.2 we can see that the responses of occluded parts should be combined with smaller weights. The smooth constraint of confidence maps (SCCM) is defined in Eq. 4.4.

We define the confidence map at the t -th frame as:

$$C^t = \sum_{i=1}^N w_i^t \hat{f}_{p(i)}^t, \quad (4.1)$$

where $\hat{f}_{p(i)}^t$ is the confidence map (Eq. 2.6) of the i -th part at time t . $p(i)$ denotes the relative position of part response in the joint confidence map C^t ; it is determined by the maximum value of the part confidence map. N is the number of parts used to divide the target. w_i^t is the weight parameter of corresponding part.

$$w_i^t = PSR_i + \eta \cdot \frac{1}{SCCM_i}, \quad (4.2)$$

$$PSR_i = \frac{\max(\hat{f}_{p(i)}^t) - \mu_i}{\sigma_i}, \quad (4.3)$$

$$SCCM_i = \left\| \hat{f}_{p(i)}^t - \hat{f}_{p(i)}^{t-1} \oplus \Delta \right\|_2^2, \quad (4.4)$$

where, μ_i and σ_i are the mean and the standard deviation of the i -th confidence map respectively. η is the trade-off between correlation sharpness and smoothness of confidence maps; in our experiments, it is simply set as 1. $\oplus$ means a shift operation of the confidence map, and Δ denotes the corresponding shift of maximum value in confidence maps from frame $t-1$ to t . $\hat{f}_{p(i)}^{t-1}$ and $\hat{f}_{p(i)}^t$ denote the individual response maps of part i , as shown at the right of Fig. 4.2.a and 4.2.b. Because the locations of a part may shift in corresponding response maps, Δ is considered for calculating the smoothness between consecutive frames. As shown in Fig. 4.2.b, when the target part is occluded by the leaf,

the PSR value is similar to that of the correctly tracked part. In this case, the term of SCCM is required to suppress the contribution of responses from the occluded parts.

4.2.a:

4.2.b:

Figure 4.2: Tracking target from frame #27 and #28, in which two parts are marked in bright and dark red bounding boxes respectively. The corresponding confidence maps of two part trackers are shown at the right with corresponding color bounding boxes. For the bright red part, it is tracked correctly between the consecutive frames. For the dark red part, it drifts in (b) because of the occlusion. To better evaluate the joint confidence map, the response of drifting part should contribute less. If we only consider PSR to weight the confidence maps, bright red and dark red parts will contribute equally. Different from the dark red part, the confidence maps of the bright red part in (a) and (b) have the smooth property. Hence, we should consider the smooth constraint to make weight selection more robust.

Besides effective combination of the individual confidence maps, model updating is another difficulty need to be solved for this multiple part tracking method. During tracking of the target, its appearance varies due to various factors such as illumination and pose changes. Hence, the trackers need to update their models to adapt to appearance changes. However, for the traditional single tracking systems, if the model is updated too frequently, small errors will be introduced and accumulated at each update. Consequently, the tracker may drift away from the target. What is more, for the part-based tracking method, we also need to consider the model updating of each part tracker.

In the conventional KCF tracker, the model consists of the learned target appearance and the transformed classifier coefficients. They are computed by only taking the current appearance into account. The tracker then employs an ad-hoc method to update the classifier coefficients by simple linear interpolation: $\mathcal{F}(\alpha)^t = (1 - \gamma)\mathcal{F}(\alpha)^{t-1} + \gamma\mathcal{F}(\alpha)$, where $\mathcal{F}(\alpha)$ is the classifier coefficients (please see Section 2.2.3.1) and γ is a learning rate parameter. Because KCF used a fix learning rate, the appearance and correlation filter will be updated without adaptation to specific video frames. Thus, if the tracker loses the target, the whole model will be contaminated in the remaining frames. We use the following adaptive updating scheme to handle this problem.

Intuitively, we should ignore the occluded parts during object tracking. Hence in order to avoid introducing errors to the models, the occlusion part should not be updated. Similar to the weight parameters discussed in Section 4.2.1, a threshold is used to adaptively update each part tracker separately. The learning rate for the model is set proportional to the weight value (Eq. 4.2), ie. the part trackers with higher detection scores should be updated more because they are seen as more reliable tracking parts. Therefore the update scheme is defined as:

$$\mathcal{F}(\alpha)_i^t = \begin{cases} (1 - \beta w_i^t)\mathcal{F}(\alpha)_i^{t-1} + \beta w_i^t \mathcal{F}(\alpha)_i & \text{if } w_i^t > \text{threshold} \\ \mathcal{F}(\alpha)_i^{t-1} & \text{else} \end{cases} \quad (4.5)$$

$$\hat{x}_i^t = \begin{cases} (1 - \beta w_i^t)\hat{x}_i^{t-1} + \beta w_i^t x_i & \text{if } w_i^t > \text{threshold} \\ \hat{x}_i^{t-1} & \text{else} \end{cases} \quad (4.6)$$

In contrast to the traditional correlation filter based trackers, due to the adaptive updating scheme, even when all the parts of a target are occluded at one frame, our method can still maintain the accuracy of the classifier by using the classifiers of previous frames. Hence, it is able to relocate the occluded target when it appears in the following frames.

Algorithm 1 Multiple part tracking

- 1: **Inputs:** t -th frame F_t , the previous tracking state s^{t-1} ;
 - 2: Calculate the confidence map of each part;
 - 3: **if** $t = 10$ **then**
 - 4: Select the discriminative parts via Sec. 4.2.2;
 - 5: **end if**
 - 6: Combine the confidence maps based on Eq. (4.1) using selected parts;
 - 7: Apply the affine transformation on s^{t-1} to obtain a number of tracking states s_j^t ;
 - 8: Calculate the posterior probability $p(s_j^t | O^t)$ according to Eq. (4.8) and Eq. (4.9);
 - 9: Predict the tracking state by $\hat{s}^t = \arg \max_{s_j^t} p(s_j^t | O^t)$;
 - 10: **if** $w_i^t > \text{threshold Eq. (4.5, 4.6)}$ **then**
 - 11: Update the model;
 - 12: **else**
 - 13: Keep the model unchanged;
 - 14: **end if**
 - 15: **Output:**
 - 16: **Find the location of each part** $p(i)$.
 - 17: **Find the target location** $\hat{s}^t$ **on confidence map and the corresponding position on the video frame;**
-

When tracking an object via multiple parts, the trackers can determine the position of the whole target through computing the geometric center based on the tracking results of all the parts. However, due to the heavy occlusion and shape deformation, some of the part trackers may drift and the final bounding box may not be able to locate the target correctly. In [39, 52], dense sampling methods have been adopted to search for the state of the target objects. But dense sampling requires high computational load and reduces the speed of the whole tracking system. Therefore, in our tracking method, the tracking problem is solved by the Bayesian inference framework.

Similar to Section 3.2.1, maximum a posterior (MAP) estimation is used to compute the optimal state $\hat{s}^t$. In Algorithm 1, we use s^t to denote the state variable, describing the affine motion parameters of an object at the time t and use $O^t = [o^1, o^2, \dots, o^t]$ to represent the observations with respect to the combined response maps.

$$\hat{s}^t = \arg \max_{s_j^t} p(s_j^t | O^t), \quad (4.7)$$

Figure 4.3: (a) and (b) are the response maps of two part trackers. (c) is the summation of (a) and (b). (d) is the structural constraint mask to enforce the spatial layout of individual parts. (e) is the confidence map used for state parameter searching after convolving with the mask.

where s_j^t is the state of the j -th sample at the time t . Same as Section 3.2.1, the posterior probability is computed as,

$$p(s^t|O^t) \propto p(o^t|s^t) \int p(s^t|s^{t-1})p(s^{t-1}|O^{t-1})ds^{t-1}, \quad (4.8)$$

The parameters of s^t is modelled independently by Eq. 3.3. To differentiate the state variable and learned appearance, we used s and O instead of x and Y in Section 3.2.1.

We use the Bayesian inference based tracking framework to calculate the likelihood in this chapter. In the traditional Bayesian inference based tracking methods, because more complicated features are adopted to represent the objects, a set of basis vectors or templates need to be built to calculate the likelihood. Because we are applying the Bayesian inference framework into confidence maps, the likelihood can be easily computed through the summation of confidence scores. Considering the noise caused by

accumulation of multiple response maps, we should enforce the spatial constraint that the response summation between neighbouring part should not influence the likelihood calculation. Given this assumption, we introduce a spatial layout constraint mask as shown in Fig. 4.3(d). By applying this mask, we explore the structural relationships among parts. In our method, the observation model is constructed by

$$p(o^t|s^t) = \frac{1}{|M^t|} \sum C^t(s^t) \odot M^t, \quad (4.9)$$

where $\odot$ is the element-wise production. M^t is a spatial layout constraint mask built by N cosine windows which gradually reduce the pixel values near the edge to zero. The relative positions of these cosine windows are determined by the maximum value of corresponding part response maps. The size of cosine window is determined by the size of tracking part. $|M^t|$ is the number of pixels within the mask. $C^t(s^t)$ means a candidate sample in the joint confidence map (the yellow rectangles in Fig. 4.1(d)). The summation is based on all the pixels within the candidate window. Besides the effect on spatial layout enforcement, by using such a structural constraint mask, we can make our method robust to the response noise of correlation filters.

The baseline method KCF used a fixed size of bounding box, which might lead to drifting when the scale of the object changed significantly. Benefiting from the multi-part scheme and Bayesian framework, our tracker is capable of solving the challenges of scale changes of targets. Fig. 4.4 shows one example of scale changes. Although the size of individual part trackers is fixed, the size of the whole target is determined through the Bayesian framework which is able to solve the challenges such as scale change. Ideally, the individual parts should stay close to each other to cover the entire target. However, in some challenging situations, such as significant appearance and illumination changes, some part trackers may move far away from the target. When this happens, we relocate the drifting parts using the correctly tracked parts based on spatial layout.

Figure 4.4: (a) The target is near the camera and caught with a larger bounding box. (b) The target is far away from the camera with a small bounding box. (c) The corresponding joint confidence map of (a). (d) The corresponding joint confidence map of (b). Because of the occlusion, the responses of bottom part are suppressed by the weighting method.

4.2.2 Learning to select parts

If we can successfully divide the tracking targets into proper parts, it will be easier to learn the corresponding appearances of the objects and realize more robust tracking performance. However, without pre-knowledge, it is difficult to recognize the individual parts of an entire object in tracking tasks. In most of the other part-based trackers and our previous method, the spatial layout of the parts are predefined with local patches. Consequently, the useless background information maybe covered by these parts. In addition, without considering the computational complexity, dividing the targets into

more parts should lead more robust tracking performance. In this section, we propose a discriminative part selection method to balance the tracking accuracy and efficiency.

In this chapter, different from the previous method [101] dividing the tracking object into fixed number of parts, we densely sample n parts in the first frame. Based on the results of following several frames, we aim to learn the proper parts and discard the low score parts to enhance and accelerate our tracker. Instead of using all the predefined parts, we aim to learn the most discriminative ones which can make the tracker more efficient as well as robust to different shapes and appearances (as shown in Fig. 4.1(c)). A minimum patch size is set to solve the low resolution challenging when the target is too small to contain enough information. In the first frame, the targeting object is divided into n parts. Through learning the confident parts in the subsequent frames, we reduce the number of part trackers to N . Since overlapped patches are used as the tracking parts, a threshold is set to avoid replicated parts tracking. We sort the confidence scores w_i (based on Eq. 4.2, 4.3 and 4.4) of all parts in a decreasing order and select the first N largest as the final tracking parts. If the overlapping rate of any of the two selected parts is larger than $1/2$, we discard the part with smaller confidence score and skip to the next part-based on the decreasing order. The N denotes an integer smaller than n which is the number of sampled parts in the first frame. This is a straightforward way to shrink the full model with all parts down to a submodel $\mathcal{M}_\gamma$ with the size of $d = N < n$.

It is difficult to exclude background information from rectangular bounding boxes used in most trackers. Most of these background information is useless and reduces the efficiency of these methods. Through using the above mentioned method, we can effectively find the most discriminative parts, which contain more foreground information.

4.3 Implementation

All of our experiments are conducted by using MATLAB R2012b on a 3.2 GHz Intel Core i5 PC with 16 GB RAM. In the first frame, we artificially divide the target into

5×5 parts as shown in Fig. 4.5. In the following 10 frames, we use the tracking results to learn the most discriminative parts as discussed in Section 4.2.2. The part size is selected by experimental validation on a number of sequences. We find when the part size is between $1/4$ and $1/6$ of the object size, the results do not change much. However, when it is too big or too small, the results can be worse. This fact shows that a part should be at the right size to not only contain enough information but also be robust to occlusion.

Figure 4.5: The spatial layout of initial parts distribution

For the individual part tracker, when the target is tracked under normal conditions, the PSR value typically ranges between 8.0 and 15.0, which indicates very strong peaks. We find that when PSR value drops to around 5.0, it is an indication that the detection of corresponding part is not reliable. When the value of SCCM is larger than 0.5, it means that the parts may suffer heavy occlusion. We set the threshold in Eq. 4.5 and Eq. 4.6 as 7. The other learning rate β is fixed as 0.01.

4.4 Experiments

The results of our experiments are presented in this section. Firstly, we introduce both quantitative and attribute-based comparisons with state-of-the-art trackers and our pre-

vious version method on the benchmark [1]. Then, we compare our method with some other trackers on VOT2014 data set [114].

4.4.1 Overall evaluation on OTB benchmark

To assess the performance of the proposed tracker, we evaluate our tracker on the OTB benchmark [1] containing 50 video sequences. The benchmark uses two measurements: i) Precision vs. Location error threshold (the percentage of the frames in which distances between tracking results and ground truth is below certain thresholds.) and ii) Success rate vs. Overlap threshold (the percentage of the frames in which overlapping scores of tracking results against ground truth are higher than certain thresholds.) We compare our tracker with the top 8 trackers (ASLA [27], CT [45], MIL [39], MTT [25], SCM [80], Struck [52], TLD [51], VTS [12]) of [1], and KCF [59]. In Fig. 4.6, “Ours_pre” presents the method of previous version [101]. “Ours” denotes our tracker using five individual parts learned from Section 4.2.2; The other trackers in Fig. 4.6 are [12, 27, 39, 51, 52, 80, 98, 115]. From Fig. 4.6, we can see clearly that our proposed method is comparable to other state-of-the-art trackers, especially when there are significant occlusion. We owe this success to the part-based scheme which can significantly enhance tracking performance by using reliable parts and adaptive weighting.

4.4.2 Attribute-based evaluation

Table 4.1 and Table 4.2 show the average precision and success scores of tracking results from the top 11 trackers on benchmark video sequences in terms of 11 categories: background clutter, deformation, fast motion, in-plane rotation, illumination variation, low resolution, motion blur, occlusion, out-of-plane rotation, out-of-view and scale variation. We can find that our trackers outperform other tracking methods in most of the categories. It demonstrates that our proposed tracking algorithm can handle these typical tracking challenges well.

Figure 4.6: Precision and Success Plots of tracking results from the top 10 trackers on all video sequences of the benchmark [1].

Table 4.1: Average precision on different attributes: low resolution (LR), background cluttered (BC), out-of-view (OV), in-plane rotation (IPR), fast motion (FM), motion blur (MB), deformation (Def), occlusion (Occ), scale variation (SV), out-of-plane rotation (OPR) and illumination variation (IV). The best and the second best results are in red and blue colors.

	Ours	Ours_pre	KCF	ASLA	CT	MIL	SCM	Struck	TLD	VTS	MTT
IV	0.801	0.777	0.728	0.517	0.359	0.349	0.594	0.558	0.537	0.573	0.351
OPR	0.783	0.763	0.729	0.518	0.394	0.466	0.618	0.597	0.596	0.604	0.473
SV	0.772	0.730	0.679	0.552	0.448	0.471	0.672	0.639	0.606	0.582	0.461
Occ	0.812	0.795	0.749	0.460	0.412	0.427	0.640	0.564	0.563	0.534	0.426
Def	0.773	0.787	0.740	0.445	0.435	0.455	0.586	0.521	0.512	0.487	0.332
MB	0.785	0.826	0.650	0.278	0.306	0.357	0.339	0.551	0.518	0.375	0.308
FM	0.726	0.724	0.602	0.253	0.323	0.396	0.333	0.604	0.551	0.353	0.401
IPR	0.804	0.780	0.725	0.511	0.356	0.453	0.597	0.617	0.584	0.579	0.522
OV	0.818	0.863	0.650	0.333	0.336	0.393	0.429	0.539	0.576	0.455	0.374
BC	0.792	0.759	0.753	0.496	0.339	0.456	0.578	0.585	0.428	0.578	0.424
LR	0.575	0.583	0.381	0.156	0.152	0.171	0.305	0.545	0.349	0.187	0.510
OverAll	0.803	0.775	0.740	0.532	0.406	0.475	0.649	0.656	0.608	0.575	0.475

Table 4.2: Success scores on different attributes: low resolution (LR), background cluttered (BC), out-of-view (OV), in-plane rotation (IPR), fast motion (FM), motion blur (MB), deformation (Def), occlusion (Occ), scale variation (SV), out-of-plane rotation (OPR) and illumination variation (IV). The best and the second best results are in red and blue colors.

	Ours	Ours_pre	KCF	ASLA	CT	MIL	SCM	Struck	TLD	VTS	MTT
IV	0.511	0.524	0.493	0.429	0.295	0.311	0.473	0.428	0.399	0.429	0.305
OPR	0.538	0.508	0.495	0.422	0.297	0.350	0.470	0.432	0.420	0.425	0.362
SV	0.484	0.448	0.427	0.452	0.302	0.335	0.518	0.425	0.421	0.400	0.348
Occ	0.549	0.544	0.514	0.376	0.321	0.335	0.487	0.413	0.402	0.398	0.342
Def	0.521	0.551	0.534	0.372	0.345	0.369	0.448	0.393	0.378	0.368	0.280
MB	0.528	0.563	0.497	0.258	0.269	0.282	0.298	0.433	0.404	0.304	0.274
FM	0.516	0.514	0.459	0.247	0.298	0.326	0.296	0.462	0.417	0.300	0.333
IPR	0.544	0.515	0.497	0.425	0.282	0.340	0.458	0.444	0.416	0.416	0.395
OV	0.470	0.630	0.550	0.312	0.359	0.382	0.361	0.459	0.457	0.443	0.342
BC	0.529	0.527	0.535	0.408	0.273	0.373	0.450	0.458	0.345	0.428	0.337
LR	0.371	0.368	0.312	0.157	0.120	0.153	0.279	0.372	0.309	0.168	0.389
OverAll	0.562	0.530	0.514	0.434	0.306	0.359	0.499	0.474	0.437	0.416	0.376

Figure 4.7: Comparison of our approach with state-of-the-art trackers in challenging situations (1).

Due to the page limitation, we show some qualitative results of several sequences on dataset [1] in Fig. 4.7 and 4.8.

4.4.3 Evaluation on VOT2014 dataset.

In this section, we report the performance of our proposed tracker on the standard VOT2014 dataset. The VOT2014 Challenge Benchmark [114] provides an evaluation kit

Figure 4.8: Comparison of our approach with state-of-the-art trackers in challenging situations (2).

and the dataset with 25 sequences which are manually selected such that the various visual phenomena are represented well within the selection.

The tracking performance in the VOT2014 Challenge Benchmark is primarily evaluated based on two independent metrics: accuracy and robustness. The accuracy is measured in terms of the average of the overlap ratios between the prediction and ground truth of each sequence. The robustness is measured according to the average number of failures which happens when the overlap rate drops to zero. The tracker will be re-initialized to the correct position to continue tracking when such failure occurs.

We follow the evaluation protocol to test our method. The main comparison results can be found in Table 4.3 and Fig. 4.9, 4.10. In this highly competitive challenge, our proposed method performed favourably compared to other state-of-the-art trackers and finished third place in the competition of 39.

Fig. 4.9 and 4.10 show the qualitative performance comparison of different trackers in the challenge. It is consistent with the scores reported in Table 4.3. From the results, we can observe that our tracker achieves close performance in overall ranking comparing with other state-of-the-art trackers. The first two trackers use the classifier based on a binary feature vector constructed from color, grayscale and gradient information. An online sparse structural SVM and a probabilistic object-background segmentation mask is used to weight the features. PLT14 is an improved version of PLT13 with a size adaptation algorithm. DSST trains a scale filter in addition to the traditional correlation filter which is used to locate the target. The DGT tracker [116] uses superpixel features and the tracking problem is formulated as a matching problem between the target graph and the candidate graph. As a part-based tracker, the location of the target is voted by the successfully matched parts according to their matching scores.

It should be noted that the VOT challenge does not focus on long-term tracking or heavy occlusion, which our algorithm is designed to deal with. But the results also show that our method remains comparable to other trackers. Different from the measuring

criterion in the OTB benchmark, the scores listed in Table 4.3 and the plots in Fig. 4.9 and 4.10 are rank-based. There are many evaluation methods for visual tracking methods, and different evaluation criteria provide different points of view for accessing the tracking method. In VOT2014, a fixed overlap rate is used to evaluate the performance and the tracker would be re-initialized several frames when the overlap ratio keeps to be zero. That means the challenge mainly focus on short-term tracking situations. In this chapter, we offer both results on OTB2013 and VOT2014 benchmark to justify the effectiveness of our proposed method.

4.5 Discussion

One limitation of our proposed method is its weak ability to handle low resolution videos. It is because that when we divide the low resolution target into several parts, the individual part cannot contain enough information to train a robust correlation filter. Though overlapped patches are used for these individual parts, in some low resolution sequences, the multiple parts scheme cannot be exploited to solve the heavy occlusion problem.

4.6 Conclusion

By combining correlation filter tracker and Bayesian inference framework, we develop a novel part-based tracker to achieve robust tracking performance. To effectively combine the individual response maps of each part, we adopt an adaptive weighting scheme, which can also be used to evaluate the part tracking performance. Additionally, through adaptive updating and structural masking methods, our tracker is further improved to adapt to appearance changes. Through discriminative learning of parts selection, our method outperforms the previous tracker based on predefined spatial layout at the expense of modest time consuming. Extensive experiments have been done to show the capability of our proposed method on OTB and VOT2014 datasets.

Table 4.3: Ranking results on VOT2014 dataset. The "overall" column displays a joined ranking for both experiments, which are also used to order the trackers.

	Baseline		Region noise		overall		
	Acc.	Rob.	Acc.	Rob.	Acc.	Rob.	Rank
PLT_14	7.28	1.96	5.68	1.6	6.48	1.78	8.26
PLT_13	8.4	1.24	7.72	1.96	8.06	1.6	9.66
Ours	4.28	7.24	4.8	6.76	4.54	7	11.54
DSST	5.2	8.16	4.24	7.28	4.72	7.72	12.44
SAMF	4.68	8.52	4.36	8.36	4.52	8.44	12.96
KCF	3.76	8.52	4.96	8.96	4.36	8.74	13.1
DGT	9.04	5.68	6.28	6.84	7.66	6.26	13.92
HMMTxD	6.08	11.2	4.72	11.16	5.4	11.18	16.58
MCT	10.24	8.36	8.32	7.64	9.28	8	17.28
eASMS	9.16	9.2	7.12	9.28	8.14	9.24	17.38
MatFlow	12.8	5.24	10.24	9.52	11.52	7.38	18.9
ACAT	8.6	11.12	9.08	9.56	8.84	10.34	19.18
ABS	12.12	10.32	9.84	8.64	10.98	9.48	20.46
ACT	10.32	11	10.28	10.8	10.3	10.9	21.2
VTDMG	10.24	12.84	9.28	10.24	9.76	11.54	21.3
qwsEDFT	9.12	12.2	9.6	14.28	9.36	13.24	22.6
LGTv1	15.6	8.96	14.2	6.76	14.9	7.86	22.76
DynMS	12.52	11.4	12.12	12.52	12.32	11.96	24.28
BDF	12.92	12.44	12.84	11.2	12.88	11.82	24.7
Struck	10.84	15.12	11.08	13	10.96	14.06	25.02
ThunderStruck	11.72	15.96	11.12	12.84	11.42	14.4	25.82
aStruck	13.2	12.92	13.32	14.76	13.26	13.84	27.1
SIR_PF	12.88	14.28	10.68	16.48	11.78	15.38	27.16
EDFT	11.12	15.6	11.48	17.08	11.3	16.34	27.64
Matrioska	13.64	13.08	11.92	16.72	12.78	14.9	27.68
OGT	10.44	18.2	10.32	18.4	10.38	18.3	28.68
IPRT	15.6	14.48	14.96	14.48	15.28	14.48	29.76
LT_FLO	10.68	20.96	10.28	19.68	10.48	20.32	30.8
PTp	21.04	12	16.8	11.96	18.92	11.98	30.9
CMT	13.96	16.64	17.04	16.28	15.5	16.46	31.96
FoT	12.52	19.32	13.52	21.64	13.02	20.48	33.5
IIVTv2	15.28	18.28	16.96	16.64	16.12	17.46	33.58
FSDT	14.92	23	14.84	19.04	14.88	21.02	35.9
IVT	14.76	20.84	18.52	18.88	16.64	19.86	36.5
CT	17.64	19.56	18.52	18.4	18.08	18.98	37.06
IMPNCC	16.08	22.92	19.72	21.04	17.9	21.98	39.88
MIL	22.88	16.08	28.76	16.84	25.82	16.46	42.28
FRT	14.16	26.96	17.2	26.32	15.68	26.64	42.32
NCC	12.68	30.56	12.72	30.76	12.7	30.66	43.36

Figure 4.9: The accuracy-robustness ranking plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.

Figure 4.10: The raw plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.

Chapter 5

Color feature learning for correlation filter based tracking method

Color features have been popularly used in visual tracking. However, most conventional color-based trackers either rely on luminance information or use simple color representations for image description. During the tracking sequences, the perceived color of the target may change because of the varying lighting conditions. In this chapter, we learn the color patterns offline from pixels sampled from images across different camera views. The pixel values of the same color from different views (with different illumination) should be encoded with the same values. The color feature space is learned by jointly learning a linear transformation and a dictionary to encode pixel values. To speedup the feature extraction, we use the marginal regression to calculate the sparse feature codes. Experimental results demonstrate that significant improvement can be achieved by using our learned color features, especially on the video sequences with complicated lighting conditions. This work is published in [117].

5.1 Introduction

Color feature plays an important role in computer vision fields. In [118], the authors investigate the importance of color features by analysing the weights of the most discriminative features. And they find that over 75 percentages of the classifier weights are

Figure 5.1: (a) Sampled pixels (blue box) from one tracking frame, in which the target's face (red box) is under bright lighting condition. (b) Encoding of a randomly sampled pixel from this bright patch. (c) Sampled pixels (blue box) from one frame, in which the target's face (red box) is under dark lighting condition. (d) Encoding of a randomly sampled pixel from this dark patch. It can be seen from (a) and (c), the color for the corresponding sampled patches are not close to each other. (b) and (d) show the results of the encoding using our approach. Using our proposed algorithm, the obtained encoding values are very similar to each other for corresponding pixels in the tracking target.

assigned to the color related features. This work supports that color features should be paid more attention than other ones such as those extracted from shape or texture information.

Similarly, color-based features have also been proven to be an important cue for visual tracking. However, color measurements can vary significantly over a tracking video due to variations in illuminant, shadows, shading, specularities, camera and object geometry, consequently the perceived color of same parts for a target appear to be different.

Considering this observation and as validated from our experiments, using color features such as RGB, HSV or even more complicated color attributes representation may not be adequate to achieve an robust color representation[60, 95, 119]. A recent work [120] proposes a framework to address the illumination variation in person re-identification. Hence we propose a data driven framework that learns rich color features from raw pixel values for visual tracking.

In this work, we aim to learn new color features for videos under various lighting environment for visual tracking. In the learned color space, the codes of the pixels are robust to different angles of view, shadows, shading, etc. An auto-encoder based framework is used to transform the RGB pixel to a higher dimensional space. Then the transformed pixel values are encoded using marginal regression technique to attain more discriminative sparse codes. The idea behind the encoding is that the encoded features of corresponding pixels should be the same or similar to each other. Fig. 5.1 shows the difference of original RGB pixels and the corresponding encoding with proposed method in a tracking sequence.

5.2 Learning color features

We aim to learn color features to enhance the state-of-the-art tracking methods. Recently, the tracking system with the Kernelized Correlation Filter (KCF) [59] achieves very good performance. Hence, we integrate our feature learning method into this system. Since in Section 2.2.3.1 the Kernelized Correlation Filter has been introduced in detail, readers may refer to that part for more information about KCF.

As shown in Fig. 5.1, in the practical tracking tasks, the appearance of color changes during the sequences due to the lightning. It can be seen that the patches sampled appear to be of different colors. Previous color feature based tracking methods usually use RGB, HSV, color attributes or other color handcrafted features to represent the target objects. However, such features cannot capture the essential color information, when the lighting

environment changes significantly. Hence, we aim to learn a novel color space to solve the challenging problems in visual tracking.

5.2.1 Training Dataset

Since the color of an object could change significantly due to different views and illumination, a good tracker desires features which are robust in tracking. Therefore, we employ the joint learning model to learn color features from selected images to handle illumination changes of objects in visual tracking. Note that this is performed offline.

To train a new color space, we need to collect training data under different lighting environment. We use the CAVIAR4REID dataset [121] to train the system. It is a person re-id evaluation dataset which is extracted from the caviar dataset for evaluation of people tracking and detection algorithms. It is a dataset which contains a total of 72 pedestrians: 50 of them with two camera views and 22 with one. We manually extract the patches, which should be the same color, however, because of different illumination, they appear to be different appearances. The patches are carefully selected so that sampled patches are distributed among different colors under various lighting conditions.

5.2.2 Objective Formulation

To learn the new color space, the feature codes of colors should be able to capture the essential color information which is robust through the tracking videos.

We use an auto-encoder to discover the stable structures and patterns in the data and project them into a rich color space.

Recently, sparse coding [122][123] has been found to improve the performance in classification, detection, tracking, etc. Hence, to further improve the performance, a sparse encoding is applied to the descriptors. We propose a joint learning framework to learn the transformation and dictionary simultaneously.

$$\min_{W,D,\alpha} \left\| [W^T(WX + b_1) + b_2] - X \right\|_2^2 + \lambda \|W\|_2^2 + \eta \|(WX + b_1) - D\alpha\|_2^2 + \nu \|\alpha\|_1 \quad (5.1)$$

The first row of the formulation is the auto-encoder loss function; the second row is the loss function of sparse coding. In the formulation, α is the encoding of the same color under different lighting conditions; X denotes the input RGB values of pixels from patches extracted from images of CAVIAR4REID; W is the linear transformation matrix that transforms each of the pixels into a new space; D is the sparse coding Dictionary to encode these pixels in a more discriminative space; the parameter η controls the trade-off between the two terms. b_1 and b_2 are the bias term. Because the training work is based on 3-dimensional RGB pixel values, the input X is constituted of sampled pixels. And the output α is the encoding of corresponding pixel.

Due to the complexity of the tracking problem, the desired feature extraction procedure should be computationally efficient. The training time should be as short as possible to improve the efficiency of the whole system. In both of the training and feature extraction procedure, sparse coding is the main time consuming operation. Hence, as discussed in 2.1.3.1, the marginal regression is adopted in this algorithm to improve the speed of the whole system.

5.2.3 Formulation Optimization

We alternatively optimize the objective function between α , D , W , b_1 and b_2 . The L-BFGS gradient based optimization procedure is used to update these values. Initially, α is updated based on their gradients while keeping D , W , b_1 and b_2 fixed. Because it is difficult to solve the L1 norm in gradient based optimization. Instead of calculating $\|\alpha\|_1$, we use the marginal regression to constrain updated results of α . Given the pages limitation, we only show the partial derivative of the loss function w.r.t α :

$$\frac{\partial L}{\partial \alpha} = -2\eta \times D^T \times (WX + b_1 + D\alpha) \quad (5.2)$$

where L is Eq. 5.1. When we calculate the α from the above function, Eq. 2.3 is considered as the principle to select coefficients. Then D is updated based on the similar gradients of itself and keeping α , W , b_1 and b_2 fixed. Finally, keeping α and D fixed, W , b_1 and b_2 are updated together.

During tracking, for each pixel of the candidate window, we first use W and b_1 to transform it into the higher dimensional space and use the learned dictionary D to compute the sparse codes by marginal regression. When the size of a tracking candidate is $M \times N$, the dimensionality of the features for the candidate will be $M \times N \times d$ where d is the number of learned sparse coding dictionaries. Fig. 5.1(b) and 5.1(d) show the representations obtained by using the joint learning framework. It can be seen that final encoding is very close to each other for corresponding pixels in the tracking target.

5.3 Experiments

We evaluate our learned color features to demonstrate its robustness to complicated lighting conditions on five challenging tracking sequences. These videos are recorded in indoor and outdoor environments and have variations of illumination change, occlusion, etc. Besides the baseline tracker KCF [59], we also compared the proposed algorithm with another nine state-of-the-art visual trackers: Frag[90], SMS [93], MIL [39], IVT [15], ASL [27], TLD [51], VTS [12], ADA [60]. All our experiments are performed using MATLAB R2012b on a 3.2 GHZ Intel Core i5 PC with 16 GB RAM. For fair comparison, we use the source codes provided by the authors. They are initialized using their default parameters.

To assess the performance of the proposed tracker, two criteria, the center location error as well as the overlap rate, are employed in our approach. A smaller average error or a bigger overlap rate means a more accurate result. Given the tracking result of each frame R_T and the corresponding ground truth R_G , we can get the overlap rate by

the PASCAL VOC [100] criterion, $score = \frac{area(R_T \cap R_G)}{area(R_T \cup R_G)}$. Table 5.1 and 5.2 report the quantitative comparison results respectively.

From Table 5.1 and 5.2, we can see clearly that in most situations, our method performs better than other state-of-the-art methods. It proves that the learned color features make the tracker robust to different lighting conditions. The results of Matrix and Shaking sequences of VTS are directly quoted from reference [12].

We also plot the results of Frag, SMS, MIL, IVT, ASL, TLD, VTS, ACA and KCF trackers in visualization comparison. The results of visualization are shown in Fig. 5.2. It can be seen that our proposed method performs very well on all these challenging sequences. When the appearance of target in Trellis changes significantly at frame 399, most of other trackers lose the face. In Matrix, because of the heavy illumination changes, only the proposed method tracks the actor’s face correctly through the whole sequence. Except for our method, VTS, ADA and KCF also propose well in sequence Shaking. For the Coke sequence in Fig. 5.2, after the heavy occlusion at frame 269, our method still tracks the can robustly. Although the skiing man moves quite fast in Skiing, our method succeed to catch the essential color information of the target.

Table 5.1: Average center location error (in pixels). The bold font indicate the best performance.

Sequence	Frag	SMS	MIL	IVT	ASL	TLD	VTS	ACA	KCF	Ours
Matrix	78.5	59.9	55.0	64.4	65.1	57.2	12.0	79.2	63.6	10.5
Shaking	39.1	81.7	24.0	85.7	63.5	37.1	5.0	14.7	17.1	5.7
Trellis	59.5	36.8	21.5	29.7	7.5	31.0	24.3	20.6	18.8	7.2
Coke	54.8	81.9	46.7	82.9	60.1	25.0	62.4	30.7	18.6	10.9
Skiing	101.0	95.5	66.9	152.3	66.6	142.8	72.0	74.4	47.5	6.0

5.4 Conclusion

In this chapter, we propose a framework for learning new color features for KCF tracker. We learn the generic features from a person re-identification dataset and apply the

Table 5.2: Average overlap rate(%). The bold font indicate the best performance.

Sequence	Frag	SMS	MIL	IVT	ASL	TLD	VTs	ACA	KCF	Ours
Matrix	16	17	20	32	42	16	60	12	13	65
Shaking	18	21	45	23	21	39	70	59	58	72
Trellis	29	21	30	35	80	48	49	59	58	86
Coke	34	28	25	12	17	40	18	42	55	62
Skiing	13	16	16	28	19	27	20	18	16	52

learned features to challenging visual object tracking. Moreover, we propose a marginal regression based sparse coding method to speedup the training and feature extraction procedure. Experimental results demonstrate that our learned color features are robust to complicated environment and are able to improve the performance of the baseline tracking system. In future, for each tracking sequence, we can online learn color features by incorporating the discriminative information between object and background.

Figure 5.2: Comparison of our approach with state-of-the-art trackers in challenging situations such as illumination variations. The example frames are from the Trellis, Matrix, Shaking, Coke and Skiing sequences respectively. The results of proposed tracker are indicated by yellow boxes.

Chapter 6

Object Tracking via ImageNet Classification Scores

As introduced in our previous two works, correlation filter based trackers are widely used in single object tracking because of their efficiency. However, most of these trackers cannot handle the occlusion well. In this chapter, an effective occlusion detection method is proposed based on the high-level classification scores from deep convolutional neural network trained on ImageNet. Benefiting from the occlusion estimation algorithm, we propose a novel tracking method by holistically considering multiple previously trained tracking models. For each frame, multiple correlation filters are trained via the hierarchical convolutional features and they are progressively selected based on the tracking quality (status) of the tracker. A linear motion model is adopted to effectively re-detect the lost target. Extensive experiments have been done to prove the effectiveness of our method.

6.1 Introduction

Although the conventional correlation filter based trackers perform fairly well in some environments (as introduced in Section 2.2.3 and Chapter 4), they cannot well handle the significant appearance changes. One important reason is that the traditional handcrafted features that are used in these trackers are not very robust in challenging situations.

Figure 6.1: Tracking results of traditional correlation filter based trackers in heavy occlusion (KCF in green, DSST in red).

Deep Neural Networks (DNNs), especially Convolutional Neural Networks (CNNs) [124] have recently drawn a lot of attention due to their outstanding representation power in various computer vision tasks. And as introduced in Section 2.2.4, some works have tried to apply DNN features to visual tracking. In several prior works [67, 69], DNN is used as a black-box feature extractor. Hence they fail to exploit the structure information of the intermediate convolutional layers. Some most recent works [72, 73] consider the hidden layers to better discriminate objects from backgrounds. They use the pre-trained network trained from ImageNet, however, they ignore the classification layer of this pre-trained network. Actually these category results include pretty much semantic information, which can help the tracker to recognize the targets during tracking.

In addition, even though more powerful features are used in these above mentioned deep trackers, the heavy occlusion is still a big challenge for them. As the tracking model is updated incrementally in the correlation filter based methods, the tracker may be contaminated seriously when the target is heavily or totally occluded. Consequently they will lose the object in the following frames. From Fig. 6.1 we can clearly see that the trackers falsely track the obstruction objects instead after severe occlusion occurs.

Hence, an effective occlusion detection method is essential to evaluate the tracking status (the target is tracked or lost) and avoid the model pollution. Additionally, in the CNN based trackers, if the tracking status is estimated correctly, we can pertinently select the features from different CNN layers to track the object. On the other side, re-detection is another important issue in tracking projects. However, without a reliable occlusion estimation method, we cannot promptly and accurately re-detect the target.

Hence, the main differences of our work comparing with the previous DNN based trackers is that by taking advantage of semantically meaningful category scores trained from ImageNet, we propose an occlusion estimation method to evaluate the tracking status and adaptively select the features extracted from different convolutional layers.

In this chapter, we propose an occlusion estimation method based on the semantically meaningful category responses of the pre-trained CNN classifiers. In order to predict whether the tracked object is the right target or not, we argue that a high-level semantic descriptor should be more helpful comparing with the sophisticated low-level features.

The responses of known category classifiers are used as high-level semantic features in object and action recognition [125–127] to recognize novel categories or actions. With the development of recent large-scale CNN classifiers trained on the ImageNet, the similar high-level semantic features (i.e., classification scores) are easy to access to encode objects. The pre-trained classifiers have limited category numbers, but our intention is not to encode an explicit decomposition of the target. When we apply the CNN classifiers to our tracker, we accept that the existing object classifiers will not encode the target object with a specific category, but the ancillary image properties. By considering combination of these ancillary characteristics, we can represent an object unrelated to the semantic meanings of pre-trained categories. Hence these high-level semantic category scores empower the tracker to potentially handle arbitrary objects.

For most of the correlation filter based discriminative tracking methods, one tracker is trained and updated through the whole sequence. Once this tracker is contaminated

by the occlusion severely, it would be difficult to recover by itself. Thus, we develop a new discriminative tracker based on multiple tracking models trained from preceding frames. Through adaptively updating and weighting the multiple tracking models, our tracker can be protected from contamination by the occlusion. For each frame tracking, features extracted from middle convolutional layers of a CNN are used to train the correlation filters. Instead of using all of the convolutional layers features, we distinctively select layers to track the target based on the occlusion estimation scores. Additionally, benefiting from the occlusion estimation scheme, a re-detection method based on a linear motion model is adopted to track the lost target and improve the robustness of our tracker, especially when tracking among the same class distracters. The contributions of this work are summarized below:

- We develop a novel occlusion estimation method base on the high-level semantic descriptors, which is trained on the large ImageNet data set.
- We propose a new multi-model tracking algorithm to avoid tracking model contamination and enhance the robustness of various appearance changes.
- We introduce a linear motion model to effectively predict the lost target and realize accurate re-detection.

6.2 Proposed tracking algorithm

We develop our proposed algorithm based on the correlation filter based tracker. By exploiting the high-level semantic descriptors, an effective occlusion detection method is developed. In our proposed tracker, we build a multi-model tracking algorithm based on the models trained from multiple preceding frames. During each frame's tracking, we pertinently select features of a CNN network to track the target. To better predict the lost target, a motion model is adopted to reduce the size of searching region and avoid

distraction from the same class. The overview of our proposed algorithm is shown in Fig. 6.2.

In the following sections of this chapter, given the input image z and a tracker $Tr(\alpha, x^l)$, we define the confidence map as $f(Tr, z)$. The framework of each tracker (also called filter) is based on KCF which is introduced in Section 2.2.3.1. Because we need to represent multiple trackers in this chapter, we use α to denote the classification coefficients and x^l as the learned target appearance.

Figure 6.2: Overview of our algorithm. The upper formulation presents the multi-model tracking scheme. The confidence map of t -th frame is calculated via I tracking models selected from frame index set $\mathcal{D}$ (6.2.2). The green part is the individual frame tracking via convolutional features (6.2.3). The bright green, hunter green and dark green (from left to right) represent the processes based on different features extracted from Conv3, Conv4 and Conv5 layers of VGG net respectively. The red part is the re-detection (6.2.4). (Best view in color)

6.2.1 Occlusion estimation via ImageNet classification scores

Because of the high efficiency and good performance, correlation filters are widely used in visual tracking applications [57, 59]. By adopting the powerful CNN features, recent trackers [71–73] are able to effectively address the challenges incurred by appearance, pose and illumination changes. However, occlusion is still a challenging issue which

limits the performance of correlation filter trackers. Without an effective occlusion detection scheme, the trackers do not know whether the targets are tracked correctly or not. Consequently the model may be contaminated by the obstruction object.

To effectively detect when the occlusion occurs, we need to recognize what it is inside the tracked bounding box. Inspired by the object recognition works in [125, 126], we find that the high-level semantic information should work better than the low-level features to describe the content inside the bounding box. With huge numbers of image samples, the large scale CNN classifiers trained on ImageNet have achieved good performance in classification and detection recently [64, 128]. The category responses of these CNN classifiers can be treated as the high-level semantically meaningful features. Though the pre-trained CNN classifiers have limited number of categories, we do not expect to assign a specific semantic meaning to each tracking target. On the contrary, we aim to represent the object with a combination of existing categories characteristics.

When applying the CNN classifiers to the tracking target, we find that the target's category responses in different frames tend to be similar if the target is tracked correctly. In the case where the target is occluded, the category scores will be different from those of the previous tracked objects. Hence, these category responses can be used to evaluate the tracking quality (i.e., whether the target is tracked correctly or not).

Examples are shown in Fig. 6.3(a), in which we present the top 3 category responses. We can find that when the target is tracked correctly, the top 3 scores (which are obtained from the output of last layer of the VGG network [128]) correspond to similar class categories (shown in corresponding colors in Fig. 6.3(a)). In the case that the target is severely occluded, the predicted categories are totally different from those of the previous correctly tracked results.

However, if the category scores of the pre-trained CNN classifiers are simply applied to calculate the similarity of consecutive frames, there will be much noise information caused by the large number of irrelevant classes. Therefore we propose to reduce the

Figure 6.3: (a) Tracking results and the corresponding occlusion estimation scores. The top 3 classification categories (corresponding scores in percentage) are listed at the right down corner of each frame. (b) shows the examples of significant appearance changes and corresponding occlusion estimation scores.

noise information by normalizing the scores as follows:

$$c_t = \frac{O^\rho}{\|O^\rho\|_1} \quad (6.1)$$

where $\rho > 1$ is a parameter of the normalization. O is the classification response vector (output of last layer of the VGG network) for the tracked bounding box (red bounding boxes in Fig. 6.3). $\|\cdot\|_1$ means the L1 norm. Through the above formulation, we can reduce the noisy information of the category responses, i.e. suppress the responses with small values and enhance the large values. c_t is the classification scores after noise reduction. In experiments, we find that $\rho = 3$ is a reasonable value and this value is fixed throughout our experiments. Then we use the following functions to calculate the occlusion estimation score (os):

$$os_t = \sum_e \min(c_t\{e\}, \hat{c}_{t-1}\{e\}) \quad (6.2)$$

$$\hat{c}_t = \begin{cases} \left(\sum_{i=1}^{t-1} \hat{c}_i \right) / (t-1) & \text{if } os_t \leq \tau \\ \frac{1}{2}c_t + \frac{1}{2} \left(\sum_{i=1}^{t-1} \hat{c}_i \right) / (t-1) & \text{if } os_t > \tau \end{cases} \quad (6.3)$$

here e is the index number of the elements of c_t . $\hat{c}$ is the template computed based on the previous frames. τ is the threshold used to determine occlusion or tracking failure. When the occlusion score os_t is equal or smaller than the threshold τ , it means the tracker fails to track the object; the template $\hat{c}_t$ is the average category scores of all the previous tracking results. Otherwise, we update the template by averaging the current frame's with the previous ones.

In Eq. 6.2, we use $\hat{c}_{t-1}$ instead of c_{t-1} to compare with c_t , because $\hat{c}_{t-1}$ contains information from previous frames, which makes it more robust to appearance change. By using the above template, the occlusion score can accurately estimate the occlusion and avoid the noise caused by the gradual appearance change of occlusion object.

From the examples shown in Fig. 6.3(b), we can see that even when the appearance of the object changes drastically, our proposed evaluation scores (os) can still obtain reasonable values, which means that the trackers are tracking the correct targets. However, when the target is occluded or the bounding box of tracking result fails to cover the correct target, the corresponding score (os) will be relatively low (last figure of Fig. 6.3(a)).

6.2.2 Multi-model tracking

In a tracking video, the object appearance changes constantly. Conventional correlation filter based tracking methods train one tracker through the whole sequence. The tracking model is updated with a learning rate frame by frame incrementally. In this scenario, only one tracker is learnt over the entire sequence. However, if the tracker lost the target, the tracker may be contaminated, so it will be difficult to relocate the object. To address this issue, we build a discriminative tracker which holistically leverages the multiple preceding trained models. Inspired by generative trackers, in which dictionary of previous frames are built to track the target, we develop a model dictionary containing multiple tracking models of previous frames. Apparently, some models may track the object better when the object appears with certain aspect at certain frames. For example, when a target object spins around in a video, trackers trained using the corresponding aspect of the object will handle the tracking better when it spins around to the same aspect. Mathematically, the summed confidence map of multi-model is achieved by the following equation:

$$f_t(z) = \sum_{i=1}^I \mathbf{w}_i f(Tr_{\mathcal{D}(i)}, z), \quad i = \{1, 2, \dots, I\} \quad (6.4)$$

where $\mathbf{w}$ are the weights for pervious frames' trackers $Tr_{\mathcal{D}(i)}$. We will introduce how to obtain each frame's tracker Tr in Section 6.2.3. I is the number of tracking models of previous frames. $\mathcal{D}$ denotes a set of I frame indexes (shown in yellow font in Fig. 6.2).

In our experiments, I is set as 5, which means that the model dictionary consists of 5 tracking models, which are trained from the frames indexed by $\mathcal{D}$.

In most tracking tasks, the earlier tracking results are more robust. Subsequently the trackers trained from earlier frames should be given larger weights to keep accurate tracking and avoid drifting. Hence, the frame index set $\mathcal{D}$ is updated based the method proposed in [27]. A probability sequence $P_s = \{0, \frac{1}{2^{I-1}-1}, \frac{3}{2^{I-1}-1}, \dots, 1\}$ is created. In each frame tracking, a random number on the unit interval $[0, 1]$ is generated. Through checking the section where this random number belongs to in P_s , we select the corresponding frame index to update. This proposed updating method aims to update the old models slowly and recent models quickly.

In addition, although the occlusion estimation score is initially designed for obstruction detection, the score can also reflect the tracking quality. Hence, we further create a weighting scheme based on the occlusion estimation scores (os) introduced in Section 6.2.1:

$$\mathbf{w}_i = \frac{os_{\mathcal{D}(i)} + \gamma}{\sum_{i=1}^I (os_{\mathcal{D}(i)} + \gamma)} \quad (6.5)$$

The estimation score os is calculated based on Eq. 6.2 and 6.3. γ is a small base weighting which is used to avoid zero denominator.

By using the proposed model dictionary and adaptive weighting method, our tracker can improve the robustness of the tracking models and accuracy of combined confidence maps.

6.2.3 Frame tracking via selective features

In this section, we introduce the tracking method of each frame based on convolutional layer features. We apply the VGG network to extract the middle level convolutional features to train the correlation filters. The input image patch is mapped to the feature maps of hidden convolutional layers.

Although the response maps created by higher convolutional layers tend to perform better in some cases (some examples are shown in Fig. 6.4), the low level filter is usually robust enough in most situations. In addition, the spatial resolution of the high convolutional layer sometimes is too coarse for accurate tracking. Hence, we do not use the high level features to track until the tracking result is not correct. In other words, we resort to the higher layers only when the low level features cannot locate the object accurately.

The filters trained on feature maps of Conv3, Conv4 and Conv5 layers are denoted as Tr^3 , Tr^4 and Tr^5 respectively. Hence, for each element inside the tracking model dictionary, there are three models represented as $Tr_{\mathcal{D}(i)}^3$, $Tr_{\mathcal{D}(i)}^4$ and $Tr_{\mathcal{D}(i)}^5$. In the first frame, we train the model based on Conv3, Conv4 and Conv5 layer features respectively. During the t -th frame tracking, we firstly use $Tr_{\mathcal{D}}^3$ to calculate the position of the target based on the confidence map obtained via Eq. 6.4. Then we calculate the corresponding os_t score based on the tracking result. Through checking the value of os_t , we can detect the tracking failure and subsequently select the hierarchical correlation filters. When the current model fails to track the target, we use the higher level features to track again and save the corresponding layers' models; otherwise we update the filters of other layers based on models of previous frame and continue the tracking on the next frame. The last os score obtained during each frame tracking is defined as the os score of this frame, which will be used in Eq. 6.5 for tracking via multi-model. When the os_t is smaller than the threshold in continuous k frames, the re-detection will be activated (Section 6.2.4).

6.2.4 Target re-detection

In most previous tracking methods, the trackers do not know whether the targeting object is lost or not, and they are not aware when the trackers fail. Therefore, re-detection is not accurate or prompt without an exact method to evaluate the tracking status. However, with the availability of effective occlusion detection that explicitly evaluates the tracking quality, our method is able to detect the happening of the occlusion. Once objects are

Figure 6.4: Tracking results with lower (Conv3) features (dark green) and deeper features (bright green). When the appearances of the object change significantly (such as rotation in the first row, severe pose change in the second row and fast moving in the third row), the deeper features can handle better.

occluded for a certain continuous period, we re-detect them using the tracking models selected from $\mathcal{D}$.

We use a constant velocity motion model to predict the potential positions of an object in the future frames. This kind of linear motion models are widely used in multiple people tracking [113, 129, 130]. We propose the following motion model to reduce the size of searching area and avoid distracters of the same class.

$$\begin{aligned} (x, y)_t &= (x, y)_{t-\Delta} + \bar{v} \bullet \Delta \\ \bar{v} &= \frac{\sum_{i=t-\Delta-m}^{t-\Delta} (v_i)}{m+1} \end{aligned} \quad (6.6)$$

$$v_t = \begin{cases} (x, y)_t - (x, y)_{t-1} & \text{if } os_t > \tau \\ v_{t-1} & \text{else} \end{cases} \quad (6.7)$$

where (x, y) and v represent the 2D image position and the velocity respectively, $(m + 1)$ is the number of frames that we aim to calculate the average velocity, Δ is the time period after the target is lost. To better save computation time, the re-detection is not activated until evaluation scores os_t is smaller than the threshold in continuous k frames, where $k \leq \Delta$. As the examples shown in Fig. 6.1, the tracker may stick to the occlusion, we need to update the velocity selectively based on the evaluation scores (as shown in Eq. 6.7). To balance the tradeoff between computation cost and performance, k is empirically set to 5 in our experiments and m is set to 4, meaning we consider the previous five frames to calculate the average velocity.

The red bounding boxes are two of the re-detection regions shown in Fig. 6.5. The positions of the re-detection regions are decided by moving the predicted position $(x, y)_t$ to up, down, left and right for a half of the size of the corresponding width and height of the object. The confidence maps of re-detection are calculated via: $f_{re}(z_{re}) = \sum_i \mathbf{w}_i f(Tr_{D(i)}, z_{re})$, where z_{re} are the re-searching regions.

Figure 6.5: Prediction of re-detection position. The solid lines represent the previous obtained velocity and positions. The dotted lines is the predicted velocity and positions.

By using the proposed motion model, we can also prevent the tracker from tracking the distracter of the same object class, which is one major limitation of the learned

features.

6.3 Experiments

In this section, we present the results of our experiments on the object tracking benchmark [1] and VOT2014 [131].

6.3.1 Implementation

All of our experiments are conducted by using MATLAB R2015b on a 3.2 GHz CPU. We extract the feature maps of Conv3_4, Conv4_4 and Conv5_4 layers of 19-layer VGG [128] to train the correlation filters respectively. The threshold of occlusion estimation score is set to $\tau = 0.05$ in the comparison with other trackers. The parameter is not sensitive to experiment results, when it is set larger than 0.05 and smaller than 0.10. Similar to [62], a pyramid scale model is trained using HOG features to solve the scale changes. These HOG features are concatenated with the CNN features during tracking. Without GPU acceleration and code optimization, our tracker runs at around 1 fps.

6.3.2 OBT evaluation

We evaluate our tracker on the OTB benchmark [1] containing 50 video sequences. The benchmark uses two measurements: i) Success rate vs. Overlap threshold, the percentage of the frames in which overlapping scores of tracking results against ground truth are higher than certain thresholds. ii) Precision vs. Location error threshold, the percentage of the frames in which distances between tracking results and ground truth is below certain thresholds.

6.3.2.1 Baseline comparison

We evaluate the impact of the proposed multi-model (MM), weighting (W), re-detection (R) and multi-layer (ML) tracking schemes here. We investigate the consequence of

Figure 6.6: Success and Precision Plots of our proposed methods comparing with base-lines.

simply using one tracking model (denoted as oneM+ML in Fig. 6.6). Then we test the performance of our proposed method with trackers trained from only the Conv3 layer features (denoted as MM+oneL). "oneM+oneL_w/oR" represent the proposed one-model one-layer trackers without re-detection. To better prove the effectiveness of our proposed weighting method introduced in Eq. 6.5, we also present the results of our method without the proposed weighting scheme (MM+ML_w/oW).

Fig. 6.6 shows the Success and Precision Plots on the OTB benchmark. From the experiments, we can see that even when only one convolutional layer is used, our proposed multi-model method (MM+oneL) outperforms the single model tracker with multiple layers' features (oneM+ML). Additionally, the tracker without the proposed weighting method (MM+ML_w/oW) performs worse than our proposed method MM+ML; the reason is probably that without judging the reliability of the models, the model dictionary may include some wrongly updated tracking models. In contrast, our proposed weighting scheme can effectively evaluate the tracking quality of the models and consequently ensure the accuracy of the model dictionary. Through the comparison between "oneM+oneL_w/oR" and "oneM+oneL", we can prove the effectiveness of our proposed re-detection method.

6.3.2.2 Overall evaluation

Here, we evaluate the overall performance in object tracking benchmark. In addition to the popular correlation filter trackers KCF [59] and DSST [62], we also compare our proposed tracker with recent trackers TGPR [132] and MEEM [133]. The other trackers appear in Fig. 6.7 are [11, 12, 27, 49, 52, 80, 98].

From Fig. 6.7, we can clearly see that our proposed method is comparable to other state-of-the-art trackers, especially when there are significant appearance changes and occlusion. We attribute this success to the occlusion estimation scheme which significantly enhance tracking performance through holistically considering multiple tracking models and robust re-detection.

Table 6.1: Success score and precision on different attributes: illumination variation (IV), out-of-plane rotation (OPR), scale variation (SV), occlusion (Occ), deformation (Def), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-view (OV) and low resolution (LR). The best results are in red colors.

	Suc.	MEEM	TGPR	KCF	Struck	DSST	Ours	Prc.	MEEM	TGPR	KCF	Struck	DSST	Ours
All	0.572	0.529	0.514	0.474	0.554	0.621		All	0.840	0.766	0.740	0.656	0.737	0.897
IV	0.548	0.486	0.493	0.428	0.563	0.573		IV	0.778	0.687	0.728	0.558	0.735	0.835
OPI	0.569	0.507	0.495	0.432	0.535	0.602		OPI	0.853	0.741	0.729	0.597	0.733	0.879
SV	0.510	0.443	0.427	0.425	0.541	0.541		SV	0.808	0.703	0.679	0.639	0.730	0.888
Occ	0.563	0.494	0.514	0.413	0.534	0.613		Occ	0.814	0.708	0.749	0.564	0.716	0.879
Def	0.582	0.556	0.534	0.393	0.510	0.637		Def	0.859	0.768	0.740	0.521	0.660	0.886
MB	0.565	0.440	0.497	0.433	0.464	0.627		MB	0.740	0.578	0.650	0.551	0.547	0.857
FM	0.568	0.441	0.459	0.462	0.435	0.580		FM	0.757	0.575	0.602	0.604	0.517	0.799
IPR	0.535	0.487	0.497	0.444	0.560	0.596		IPR	0.809	0.706	0.725	0.617	0.765	0.881
OV	0.597	0.431	0.550	0.459	0.459	0.600		OV	0.730	0.495	0.650	0.539	0.515	0.728
BC	0.578	0.543	0.535	0.458	0.517	0.636		BC	0.808	0.761	0.753	0.585	0.694	0.885
LR	0.367	0.351	0.312	0.372	0.409	0.549		LR	0.494	0.539	0.381	0.545	0.497	0.875

6.3.2.3 Attribute-based evaluation

Table 6.1 shows the average success scores and precision of tracking results from the top 6 trackers on benchmark video sequences in terms of 11 categories. We can find that our tracker outperforms other tracking methods in most of the categories. It demonstrates that our proposed tracking algorithm can handle these typical tracking challenges well.

6.3.2.4 Comparison with Deep trackers

We also provide the comparison of our method and some other deep learning based trackers. Except for the results of [70], which are obtained by running the source code, the other results are cited from the corresponding references directly. From Table 6.2 we can find that our proposed method performs comparably well. Furthermore, our proposed occlusion estimation method is complementary to the other trackers to detect the occlusion and monitor the tracking quality.

Table 6.2: The overall success and precision scores comparison with Deep trackers. The best results are shown in red color.

	Ours	[134]	[70]	[73]	[135]	[71]	[72]
Suc.	0.621	0.545	0.436	0.599	0.602	0.597	0.605
Pre.	0.897	0.723	0.587	0.856	0.819	0.852	0.891

6.3.3 VOT 2014 evaluation

In this section, we report the performance of our proposed tracker on the standard VOT2014 dataset. The VOT2014 Challenge Benchmark [131] provides an evaluation kit and the dataset with 25 sequences which are manually selected such that the various visual phenomena are represented well within the selection.

The tracking performance in the VOT2014 Challenge Benchmark is primarily evaluated based on two independent metrics: accuracy and robustness. The accuracy is measured in terms of the average of the overlap ratios between the prediction and ground truth

Figure 6.7: Success and Precision Plots of tracking results from the top 10 trackers on all video sequences of the benchmark [1].

Figure 6.8: The accuracy-robustness ranking plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.

Figure 6.9: The raw plots with respect to the two experiments. Tracker is better if it resides closer to the top-right corner of the plot.

Table 6.3: Ranking results on VOT2014 dataset. The "overall" column displays a joint ranking for both experiments, which are also used to order the trackers. The best results are shown in red color.

	baseline		region_noise		overall
	Acc. Rank	Rob. Rank	Acc. Rank	Rob. Rank	Rank
Ours	2.68	2.58	2.76	2.72	2.69
HCFT	3.42	2.66	3.44	2.58	3
DSST	3.02	3.04	2.86	3.16	3.02
SAMF	3	3.28	2.92	3.2	3.1
KCF	2.88	3.44	3.02	3.42	3.19

of each sequence. The robustness is measured according to the average number of failures which happens when the overlap rate drops to zero. The tracker will be re-initialized to the correct position to continue tracking when such failure occurs.

We follow the evaluation protocol to test our method. In addition to the top 3 trackers (DSST, SAMF and KCF) of VOT 2014, we also compare our method with the recent Hierarchical Convolutional Features Tracker (HCFT) [72]. The main comparison results can be found in Table 6.3, Fig. 6.8 and Fig. 6.9. In this highly competitive challenge, our proposed method performed favourably comparable to other state-of-the-art trackers.

6.4 Conclusion

Based on the high-level semantic category responses and middle convolutional layers, which are trained on ImageNet, we develop a multi-model correlation tracker. Benefiting from the occlusion estimation method, our tracker can adaptively combine the multiple models and re-detect the lost target. Extensive experiments have been done to verify the reliability of our proposed method.

Chapter 7

Conclusions and Future Works

7.1 Conclusions

In this thesis, we propose several algorithms to improve the robustness and efficiency for visual tracking.

First, based on the framework of the sparse based trackers, a real-time temporal smooth visual tracker is developed, which achieves significant improvement. The proposed algorithm exploits the information among consecutive frames to improve the tracking accuracy. To improve the computational efficiency, a marginal regression method is employed to speedup the tracking speed. Experimental results on real-world video sequences validate the high computational efficiency and demonstrate that the proposed tracking method can significantly enhance visual object tracking performance.

Second, a real-time part-based tracker is proposed based on the framework of correlation filter tracker and Bayesian inference. To effectively combine the individual response maps of each part, we adopt an adaptive weighting scheme, which can also be used to evaluate the part tracking performance. Moreover, to adapt to various appearance changes, a structural masking is exploited to improve the robustness of the proposed approach. Instead of using the predefined spatial layout, we discriminatively select the parts from multiple samples. Experimental results have prove the capability of our proposed method on OTB and VOT2014 datasets.

Third, we propose a framework for learning new color features for visual tracking tasks. The generic features are learned from a person re-identification dataset and applied to challenging visual object tracking. Moreover, The marginal regression based sparse coding method is adopted to speedup the training and feature extraction procedure. Experimental results demonstrate that the learned color features are robust to various situations and are able to improve the tracking performance.

Fourth, we develop a novel multi-model correlation filter based tracking method based on the high-level semantic category responses and middle convolutional layers, which are trained on ImageNet offline. Benefiting from the occlusion estimation method, our tracker can holistically combine multiple tracking models. A linear motion model is adopted to effectively re-detect the lost target. Extensive experiments have been done to verify the reliability of our proposed method.

For the real-world applications, real-time tracking is highly required. Although the sparse representation based tracking algorithms are robust to significant appearance changes, the computational complexity of sparse coding is usually high, which limits them to be adopted in practical systems. In the third and fifth parts of this thesis, sparse coding are used to represent the tracking target. To improve the tracking speed of proposed methods, marginal regression is applied to simplify the calculation of sparse coding with acceptable accuracy drop. In the fourth chapter of this thesis, correlation filter and discriminative part selection scheme play the important roles to speedup the proposed part-based tracker. Through the discriminative selection, this tracker can focus on the more robust parts of a target instead of considering all of them. And with the help of high speed correlation filter, this method can run in real-time with fairly well performance. In the sixth chapter of this thesis, cause the CNN features are used, it is difficult to run the tracker in real-time without GPU. However, we can still significantly improve the speed of the proposed tracker with the adaptive feature layer selection. Instead of extracting features from many intermediate convolutional layers, we can ignore many of these layers

when we use the classification scores to evaluation the current tracking performance. In summary, the trackers introduced in this thesis aim to track the target robustly as well as efficiently.

7.2 Future Works

There are many more challenges that need to be addressed to successfully solve the visual tracking task and to apply it in a real-world scenario. To conclude the thesis, I would like to discuss some research directions which can be explored in the future.

In the first piece of work, a sparse coding based temporally smooth tracker is developed. Although our sparse tracker can track the target well in most situations, it still cannot address the sequences with significant appearance changes due to the limitation of sparse representation. In the future, the dictionary learning used for sparse coding can be further improved to enhance the discriminative ability. In object recognition task, some effective discriminative dictionary learning algorithms [54, 136] have been developed. In future work, this kind of dictionary learning scheme could be modified to be used in visual tracking. For example, when we learn sparse coding representations, in addition to the temporal information, the structural and spatial information should also be considered to learn a more discriminative dictionary. Then, the structural information of the representation can be further exploited to handle the occlusion problem. Specifically speaking, when the tracker is used to track pedestrians, the head and upper trunk should be paid more attention due to less pose changing. Hence, the tracker should train the sparse representation with emphasis of these parts.

In the second piece of work, the proposed part-based tracker utilizes the traditional HOG feature for the correlation filter based classifier. However, by comparing the performance of conventional correlation filter based tracker [58] with [60], it can be found that the possibilities of drifting are significantly reduced with powerful features. Hence, in the

future work, more powerful features are worthy to be tried in our proposed tracking framework. In addition to the HOG features, we can concatenate color features or shallow convolutional features to represent the tracking objects. Furthermore, cause the proposed method is able to compare the similarity of tracking candidates in different frames with PSR and SCCM, we can take advantage of this scheme to adaptively determine the features. Consequently, the proposed method can track the target with the most robust features for different videos.

In the third piece of work, a color feature learning framework is developed for visual tracking. The proposed color feature learning relies on approximate correction algorithms to correct the pixel values initially. However, we should eliminate the dependency on hand-crafted approaches, which are based on several assumptions about the statistics of color distribution in the images. Moreover, the color features are pre-trained from a human re-identification dataset in this project. Hence, learning color features from video data can be considered as a possible direction for future research. Specifically, the tracker can start with the pre-trained features. During tracking, it can preserve the tracking results and extract the color pixels in the same position of consecutive frames. This position can be chosen randomly inside the tracking results, while the pixel pairs should be extracted in neighboring frames to avoid significant pose changing. Then these pixel pairs will be used to fine tune the new color feature model.

In the last piece of work, the high-level semantic classification scores are exploited to detect occurrence of occlusion. However, when the tracking target is occluded by the object from the same category, the probability of drifting will increase. In future work, more types of features can be incorporated into the existing classification scores to improve the accuracy of occlusion detection at the expense of modest time consuming. Additionally, the shallower features (with more spatial information) can be considered together with the final classification scores to identify the target. With the help of both sophisticated low-level and semantic high-level features, we can well recognize, identify

and track the targets. Then this proposed tracker can even be extended to the filed of multiple object tracking, which means we can not only track several objects at the same time, but also give correct indexes to corresponding instances.

Author's Publications

- (i) **Ting Liu**, Gang Wang and Qingxiong Yang, “Real-time part-based visual tracking via adaptive correlation filters”, In *Conference on Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, 2015.
- (ii) **Ting Liu** , Gang Wang, Wang, Li and Chan, Kap Luk, “Visual tracking via temporally smooth sparse coding”, *IEEE Signal Processing Letters (SPL)*, 2015.
- (iii) **Ting Liu** Rahul Rama Varior and Gang Wang, “Visual tracking using learned color features”, In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on* , 2015.
- (iv) **Ting Liu** Gang Wang, Qingxiong Yang and Li Wang, “Part-based Tracking via Discriminative Correlation Filters”, In Proceedings of the *Circuits and Systems for Video Technology (CSVT), IEEE Transactions on*, 2016.
- (v) Li Wang, **Ting Liu**, Gang Wang, Kap Luk Chan and Qingxiong Yang. “Video tacking using learned hierarchical features”, *Image Processing (T-IP), IEEE Transactions on* , 2016.
- (vi) Rahul Rama Varior, Gang Wang, Jiwen Lu and **Ting Liu**, “Learning invariant color features for person re-identification”, *Image Processing (T-IP)*, IEEE Transactions on, 2016.
- (vii) Bing Wang, Li Wang, Bing Shuai, Zhen Zuo, **Ting Liu**, Kap Luk Chan, Gang Wang “Joint Learning of Convolutional Neural Networks and Temporally Constrained

Metrics for Tracklet Association”, In *Computer Vision and Pattern Recognition workshops (CVPRW)*, 2016 IEEE Conference on , 2016.

References

- [1] Y. Wu, J. Lim, and M.-H. Yang, “Online object tracking: A benchmark,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 2411–2418, 2013.
- [2] Q. Liu, X. Zhao, and Z. Hou, “Survey of single-target visual tracking methods based on online learning,” *IET Computer Vision*, vol. 8, no. 5, pp. 419–428, 2014.
- [3] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi, “Robust online appearance models for visual tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 10, pp. 1296–1311, 2003.
- [4] S. K. Zhou, R. Chellappa, and B. Moghaddam, “Visual tracking and recognition using appearance-adaptive models in particle filters,” *Image Processing, IEEE Transactions on*, vol. 13, no. 11, pp. 1491–1506, 2004.
- [5] M. Fussenegger, P. Roth, H. Bischof, R. Deriche, and A. Pinz, “A level set framework using a new incremental, robust active shape model for object segmentation and tracking,” *Image and Vision Computing*, vol. 27, no. 8, pp. 1157–1168, 2009.
- [6] H. Yang, Z. Song, and R. Chen, “An incremental pca-hog descriptor for robust visual hand tracking,” in *International Symposium on Visual Computing*, pp. 687–695, 2010.

REFERENCES

- [7] J. Chiverton, X. Xie, and M. Mirmehdi, “Automatic bootstrapping and tracking of object contours,” *Image Processing, IEEE Transactions on*, vol. 21, no. 3, pp. 1231–1245, 2012.
- [8] J. Chiverton, M. Mirmehdi, and X. Xie, “On-line learning of shape information for object segmentation and tracking,” in *British Machine Vision Conference (BMVC) 2009*, pp. 21.1–21.11, 2009.
- [9] X. Liu, L. Lin, S. Yan, H. Jin, and W. Jiang, “Adaptive object tracking by learning hybrid template online,” *Circuits and Systems for Video Technology, IEEE transactions on*, vol. 21, no. 11, pp. 1588–1599, 2011.
- [10] Y. Xu, H. Zhou, Q. Wang, and L. Lin, “Realtime object-of-interest tracking by learning composite patch-based templates,” in *Image Processing (ICIP), 2012 IEEE International Conference on*, pp. 389–392, 2012.
- [11] J. Kwon and K. M. Lee, “Visual tracking decomposition,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 1269–1276, 2010.
- [12] J. Kwon and K. M. Lee, “Tracking by sampling trackers,” in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 1195–1202, 2011.
- [13] D. Ross, J. Lim, and M.-H. Yang, “Adaptive probabilistic visual tracking with incremental subspace update,” in *European Conference on Computer Vision (ECCV) 2004*, pp. 470–482, 2004.
- [14] J. Lim, D. A. Ross, R.-S. Lin, and M.-H. Yang, “Incremental learning for visual tracking,” in *Advances in Neural Information Processing Systems*, pp. 793–800, 2004.
- [15] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, “Incremental learning for robust visual tracking,” *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008.

- [16] K.-C. Lee and D. Kriegman, “Online learning of probabilistic appearance manifolds for video-based recognition and tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2008 IEEE Conference on*, vol. 1, pp. 852–859, 2005.
- [17] J. Wen, X. Li, X. Gao, and D. Tao, “Incremental learning of weighted tensor subspace for visual tracking,” in *Systems, Man and Cybernetics (SMC), 2009 IEEE International Conference on*, pp. 3688–3693, 2009.
- [18] X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, and J. Cheng, “Visual tracking via incremental log-euclidean riemannian subspace learning,” in *Computer Vision and Pattern Recognition (CVPR), 2008 IEEE Conference on*, pp. 1–8, 2008.
- [19] Y. Wu, J. Cheng, J. Wang, and H. Lu, “Real-time visual tracking via incremental covariance tensor learning,” in *Computer Vision (ICCV), 2009 IEEE International Conference on*, pp. 1631–1638, 2009.
- [20] X. Li, A. Dick, C. Shen, A. Van Den Hengel, and H. Wang, “Incremental learning of 3d-dct compact representations for robust visual tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 4, pp. 863–881, 2013.
- [21] X. Mei and H. Ling, “Robust visual tracking using ℓ_1 minimization,” in *Computer Vision (ICCV), 2009 IEEE International Conference on*, pp. 1436–1443, 2009.
- [22] H. Li, C. Shen, and Q. Shi, “Real-time visual tracking using compressive sensing,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1305–1312, 2011.
- [23] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski, “Robust and fast collaborative tracking with two stage sparse optimization,” in *European Conference on Computer Vision (ECCV) 2010*, pp. 624–637, 2010.

REFERENCES

- [24] F. Chen, Q. Wang, S. Wang, W. Zhang, and W. Xu, “Object tracking via appearance modeling and sparse representation,” *Image and Vision Computing*, vol. 29, no. 11, pp. 787–796, 2011.
- [25] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, “Robust visual tracking via structured multi-task sparse learning,” *International Journal of Computer Vision*, vol. 101, no. 2, pp. 367–383, 2013.
- [26] T. Bai and Y. F. Li, “Robust visual tracking with structured sparse representation appearance model,” *Pattern Recognition*, vol. 45, no. 6, pp. 2390–2404, 2012.
- [27] X. Jia, H. Lu, and M.-H. Yang, “Visual tracking via adaptive structural local sparse appearance model,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1822–1829, 2012.
- [28] X. Lu, Y. Yuan, and P. Yan, “Robust visual tracking with discriminative sparse learning,” *Pattern Recognition*, vol. 46, no. 7, pp. 1762–1771, 2013.
- [29] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.
- [30] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.
- [31] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, *et al.*, “Least angle regression,” *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.
- [32] K. Balasubramanian, K. Yu, and G. Lebanon, “Smooth sparse coding via marginal regression for learning sparse representations,” in *International Conference on Machine learning (ICML) 2013*, pp. 289–297, 2013.

- [33] D. L. Donoho, “For most large underdetermined systems of linear equations the minimal ℓ_1 -norm solution is also the sparsest solution,” *Communications on Pure and Applied Mathematics*, vol. 59, no. 6, pp. 797–829, 2006.
- [34] J. Fan and J. Lv, “Sure independence screening for ultrahigh dimensional feature space,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 5, pp. 849–911, 2008.
- [35] C. R. Genovese, J. Jin, L. Wasserman, and Z. Yao, “A comparison of the lasso and marginal regression,” *The Journal of Machine Learning Research*, vol. 98888, no. 1, pp. 2107–2143, 2012.
- [36] S. Avidan, “Ensemble tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 2, pp. 261–271, 2007.
- [37] H. Grabner, M. Grabner, and H. Bischof, “Real-time tracking via on-line boosting,” in *British Machine Vision Conference (BMVC) 2006*, vol. 1, p. 6, 2006.
- [38] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in *European Conference on Computer Vision (ECCV) 2008*, pp. 234–247, 2008.
- [39] B. Babenko, M.-H. Yang, and S. Belongie, “Visual tracking with online multiple instance learning,” in *Computer Vision and Pattern Recognition (CVPR), 2009 IEEE Conference on*, pp. 983–990, 2009.
- [40] B. Zeisl, C. Leistner, A. Saffari, and H. Bischof, “On-line semi-supervised multiple-instance boosting,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 1879–1879, 2010.
- [41] W. Li, L. Duan, I. W.-H. Tsang, and D. Xu, “Batch mode adaptive multiple instance learning for computer vision tasks,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 2368–2375, 2012.

- [42] T. Parag, F. Porikli, and A. Elgammal, “Boosting adaptive linear weak classifiers for online learning and tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2008 IEEE Conference on*, pp. 1–8, 2008.
- [43] K. Sankaranarayanan and J. W. Davis, “One-class multiple instance learning and applications to target tracking,” in *Asian Conference on Computer Vision (ACCV) 2012*, pp. 126–139, 2012.
- [44] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, “On-line random forests,” in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE International Conference on*, pp. 1393–1400, 2009.
- [45] K. Zhang, L. Zhang, and M.-H. Yang, “Real-time compressive tracking,” in *European Conference on Computer Vision (ECCV) 2012*, pp. 864–877, 2012.
- [46] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel, “Robust tracking with weighted online structured learning,” in *European Conference on Computer Vision (ECCV) 2012*, pp. 158–172, 2012.
- [47] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, “Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 10, pp. 1728–1740, 2008.
- [48] M. Yang, J. Yuan, and Y. Wu, “Spatial selection for attentional visual tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2007 IEEE Conference on*, pp. 1–8, 2007.
- [49] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 7, pp. 1409–1422, 2012.

- [50] Z. Kalal, J. Matas, and K. Mikolajczyk, “Online learning of robust object detectors during unstable tracking,” in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE International Conference on*, pp. 1417–1424, 2009.
- [51] Z. Kalal, J. Matas, and K. Mikolajczyk, “Pn learning: Bootstrapping binary classifiers by structural constraints,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 49–56, 2010.
- [52] S. Hare, A. Saffari, and P. H. Torr, “Struck: Structured output tracking with kernels,” in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 263–270, 2011.
- [53] X. Wang, G. Hua, and T. X. Han, “Discriminative tracking by metric learning,” in *European Conference on Computer Vision (ECCV) 2010*, pp. 200–214, 2010.
- [54] N. Jiang, W. Liu, and Y. Wu, “Learning adaptive metric for robust visual tracking,” *Image Processing, IEEE Transactions on*, vol. 20, no. 8, pp. 2288–2300, 2011.
- [55] N. Jiang, W. Liu, and Y. Wu, “Order determination and sparsity-regularized metric learning adaptive visual tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1956–1963, 2012.
- [56] Y. Cong, J. Yuan, and Y. Tang, “Object tracking via online metric learning,” in *Image Processing (ICIP), 2012 IEEE International Conference on*, pp. 417–420, 2012.
- [57] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, “Visual object tracking using adaptive correlation filters,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2544–2550, 2010.
- [58] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “Exploiting the circulant structure of tracking-by-detection with kernels,” in *European Conference on Computer Vision (ECCV) 2012*, pp. 702–715, 2012.

REFERENCES

- [59] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 3, pp. 583–596, 2015.
- [60] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer, “Adaptive color attributes for real-time visual tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 2014.
- [61] Y. Li and J. Zhu, “A scale adaptive kernel correlation filter tracker with feature integration,” in *European Conference on Computer Vision (ECCV) 2014*, pp. 254–265, 2014.
- [62] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, “Accurate scale estimation for robust visual tracking,” in *British Machine Vision Conference (BMVC) 2014*, 2014.
- [63] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.
- [64] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 580–587, 2014.
- [65] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, “Deep networks for saliency detection via local estimation and global search,” in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pp. 3183–3192, 2015.
- [66] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *Computer Vision and Pattern Recognition (CVPR), 2009 IEEE Conference on*, pp. 248–255, 2009.

- [67] H. Li, Y. Li, F. Porikli, *et al.*, “Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking,” in *British Machine Vision Conference (BMVC) 2014*, vol. 1, p. 3, 2014.
- [68] H. Li, Y. Li, and F. Porikli, “Robust online visual tracking with a single convolutional neural network,” in *Asian Conference on Computer Vision (ACCV) 2014*, pp. 194–209, 2014.
- [69] J. Fan, W. Xu, Y. Wu, and Y. Gong, “Human tracking using convolutional neural networks,” *Neural Networks, IEEE Transactions on*, vol. 21, no. 10, pp. 1610–1623, 2010.
- [70] N. Wang and D.-Y. Yeung, “Learning a deep compact image representation for visual tracking,” in *Advances in Neural Information Processing Systems*, pp. 809–817, 2013.
- [71] S. Hong, T. You, S. Kwak, and B. Han, “Online tracking by learning discriminative saliency map with convolutional neural network,” *International Conference on Machine learning (ICML) 2015*, 2015.
- [72] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, “Hierarchical convolutional features for visual tracking,” in *Computer Vision (ICCV), 2015 IEEE International Conference on*, 2015.
- [73] L. Wang, W. Ouyang, X. Wang, and H. Lu, “Visual tracking with fully convolutional networks,” in *Computer Vision (ICCV), 2015 IEEE International Conference on*, pp. 3119–3127, 2015.
- [74] T. Woodley, B. Stenger, and R. Cipolla, “Tracking using online feature selection and a local generative model,” in *British Machine Vision Conference (BMVC) 2007*, pp. 1–10, 2007.

REFERENCES

- [75] X. Zhang, W. Hu, S. Maybank, and X. Li, “Graph based discriminative learning for robust and efficient object tracking,” in *Computer Vision (ICCV), 2007 IEEE International Conference on*, pp. 1–8, 2007.
- [76] Q. Yu, T. B. Dinh, and G. Medioni, “Online tracking and reacquisition using co-trained generative and discriminative trackers,” in *European Conference on Computer Vision (ECCV) 2008*, pp. 678–691, 2008.
- [77] M. Yang, Y. Wu, and S. Lao, “Intelligent collaborative tracking by mining auxiliary objects,” in *Computer Vision and Pattern Recognition (CVPR), 2006 IEEE Conference on*, vol. 1, pp. 697–704, 2006.
- [78] Y. Lei, X. Ding, and S. Wang, “Visual tracker using sequential bayesian learning: discriminative, generative, and hybrid,” *Systems, Man, and Cybernetics, Part B (Cybernetics), IEEE Transactions on*, vol. 38, no. 6, pp. 1578–1591, 2008.
- [79] C. Shen, J. Kim, and H. Wang, “Generalized kernel-based visual tracking,” *Circuits and Systems for Video Technology, IEEE transactions on*, vol. 20, no. 1, pp. 119–130, 2010.
- [80] W. Zhong, H. Lu, and M.-H. Yang, “Robust object tracking via sparsity-based collaborative model,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1838–1845, 2012.
- [81] T. Liu, G. Wang, L. Wang, and K. L. Chan, “Visual tracking via temporally smooth sparse coding,” *IEEE Signal Processing Letters*, vol. 22, no. 9, pp. 1452–1456, 2015.
- [82] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.
- [83] B. Wang, G. Wang, K. L. Chan, and L. Wang, “Tracklet association with on-line target-specific metric learning,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1234–1241, 2014.

- [84] D. L. Donoho, “Compressed sensing,” *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [85] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, 2006.
- [86] B. A. Olshausen and D. J. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, vol. 381, no. 6583, p. 607, 1996.
- [87] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, 2009.
- [88] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, “Minimum error bounded efficient l1 tracker with occlusion detection,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1257–1264, 2011.
- [89] C. Bao, Y. Wu, H. Ling, and H. Ji, “Real time robust l1 tracker using accelerated proximal gradient approach,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1830–1837, 2012.
- [90] A. Adam, E. Rivlin, and I. Shimshoni, “Robust fragments-based tracking using the integral histogram,” in *Computer Vision and Pattern Recognition (CVPR), 2006 IEEE Conference on*, vol. 1, pp. 798–805, 2006.
- [91] S. Stalder, H. Grabner, and L. Van Gool, “Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition,” in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE International Conference on*, pp. 1409–1416, 2009.

REFERENCES

- [92] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, “Locally orderless tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1940–1947, 2012.
- [93] R. T. Collins, “Mean-shift blob tracking through scale space,” in *Computer Vision and Pattern Recognition (CVPR), 2003 IEEE Conference on*, vol. 2, pp. II–234, 2003.
- [94] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 5, pp. 564–577, 2003.
- [95] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, “Color-based probabilistic tracking,” in *European Conference on Computer Vision (ECCV) 2002*, pp. 661–675, 2002.
- [96] L. Sevilla-Lara and E. Learned-Miller, “Distribution fields for tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1910–1917, 2012.
- [97] Y. Wu, B. Shen, and H. Ling, “Online robust image alignment via iterative convex optimization,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1808–1814, 2012.
- [98] T. B. Dinh, N. Vo, and G. Medioni, “Context tracker: Exploring supporters and distracters in unconstrained environments,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1177–1184, 2011.
- [99] D. Wang, H. Lu, and M.-H. Yang, “Least soft-threshold squares tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 2371–2378, 2013.

REFERENCES

- [100] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.
- [101] T. Liu, G. Wang, and Q. Yang, “Real-time part-based visual tracking via adaptive correlation filters,” in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, June 2015.
- [102] A. Rodriguez, V. N. Boddeti, B. V. Kumar, and A. Mahalanobis, “Maximum margin correlation filter: A new approach for localization and classification,” *Image Processing, IEEE Transactions on*, vol. 22, no. 2, pp. 631–643, 2013.
- [103] L. Cehovin, M. Kristan, and A. Leonardis, “An adaptive coupled-layer visual model for robust visual tracking,” in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 1363–1370, 2011.
- [104] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, “Part-based multiple-person tracking with partial occlusion handling,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1815–1821, 2012.
- [105] B. Yang and R. Nevatia, “Online learned discriminative part-based appearance models for multi-human tracking,” in *European Conference on Computer Vision (ECCV) 2012*, pp. 484–498, 2012.
- [106] L. Cehovin, M. Kristan, and A. Leonardis, “Robust visual tracking using an adaptive coupled-layer visual model,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 4, pp. 941–953, 2013.
- [107] P. Chockalingam, N. Pradeep, and S. Birchfield, “Adaptive fragments-based tracking of non-rigid objects using level sets,” in *Computer Vision (ICCV), 2009 IEEE International Conference on*, pp. 1530–1537, 2009.

REFERENCES

- [108] S. Shahed Nejhum, J. Ho, and M.-H. Yang, “Visual tracking with histograms and articulating blocks,” in *Computer Vision and Pattern Recognition (CVPR), 2008 IEEE Conference on*, pp. 1–8, 2008.
- [109] J. Kwon and K. M. Lee, “Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling,” in *Computer Vision and Pattern Recognition (CVPR), 2009 IEEE Conference on*, pp. 1208–1215, 2009.
- [110] T. Zhang, K. Jia, C. Xu, Y. Ma, and N. Ahuja, “Partial occlusion handling for visual tracking via robust part matching,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1258–1265, 2014.
- [111] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel, “Part-based visual tracking with online latent structural learning,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 2363–2370, 2013.
- [112] D. Wang, H. Lu, and M.-H. Yang, “Online object tracking with sparse prototypes,” *Image Processing, IEEE Transactions on*, vol. 22, no. 1, pp. 314–325, 2013.
- [113] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, “Robust tracking-by-detection using a detector confidence particle filter,” in *Computer Vision (ICCV), 2009 IEEE International Conference on*, pp. 1515–1522, 2009.
- [114] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojř, G. Fernandez, A. Lukežič, A. Dimitriev, *et al.*, “The visual object tracking vot2014 challenge results,” in *European Conference on Computer Vision (ECCV Workshops) 2014*, pp. 191–217, 2014.
- [115] J. Kwon and K. M. Lee, “Visual tracking decomposition,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 1269–1276, 2010.

REFERENCES

- [116] Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, and S. Z. Li, “Robust deformable and occluded object tracking with dynamic graph,” *Image Processing, IEEE Transactions on*, vol. 23, no. 12, pp. 5497–5509, 2014.
- [117] T. Liu, R. R. Varior, and G. Wang, “Visual tracking using learned color features,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 1976–1980, 2015.
- [118] D. Gray and H. Tao, “Viewpoint invariant pedestrian recognition with an ensemble of localized features,” *European Conference on Computer Vision (ECCV) 2008*, pp. 262–275, 2008.
- [119] K. Nummiaro, E. Koller-Meier, and L. Van Gool, “An adaptive color-based particle filter,” *Image and Vision Computing*, vol. 21, no. 1, pp. 99–110, 2003.
- [120] R. Rama Varior, G. Wang, and J. Lu, “Learning invariant color features for person re-identification,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*.
- [121] D. S. Cheng, M. Cristani, M. Stoppa, L. Bazzani, and V. Murino, “Custom pictorial structures for re-identification,” in *British Machine Vision Conference (BMVC) 2011*, vol. 2, p. 6, 2011.
- [122] J. Yang, K. Yu, Y. Gong, and T. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *Computer Vision and Pattern Recognition (CVPR), 2009 IEEE Conference on*, pp. 1794–1801, 2009.
- [123] K. Yu, T. Zhang, and Y. Gong, “Nonlinear learning using local coordinate coding,” in *Advances in Neural Information Processing Systems*, pp. 2223–2231, 2009.
- [124] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

REFERENCES

- [125] L. Torresani, M. Szummer, and A. Fitzgibbon, “Efficient object category recognition using classemes,” in *European Conference on Computer Vision (ECCV) 2010*, pp. 776–789, 2010.
- [126] L. Li-Jia, H. Su, L. Fei-Fei, and E. P. Xing, “Object bank: A high-level image representation for scene classification & semantic feature sparsification,” in *Advances in Neural Information Processing Systems*, pp. 1378–1386, 2010.
- [127] S. Sadanand and J. J. Corso, “Action bank: A high-level representation of activity in video,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1234–1241, 2012.
- [128] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [129] K. Shafique, M. W. Lee, and N. Haering, “A rank constrained continuous formulation of multi-frame multi-target tracking problem,” in *Computer Vision and Pattern Recognition (CVPR), 2008 IEEE Conference on*, pp. 1–8, 2008.
- [130] M. Yang, T. Yu, and Y. Wu, “Game-theoretic multiple target tracking,” in *Computer Vision (ICCV), 2007 IEEE International Conference on*, pp. 1–8, 2007.
- [131] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojir, G. Fernandez, A. Lukežič, A. Dimitriev, A. Petrosino, A. Saffari, B. Li, B. Han, C. Heng, C. Garcia, D. Pangršič, G. Häger, F. S. Khan, F. Oven, H. Possegger, H. Bischof, H. Nam, J. Zhu, J. Li, J. Y. Choi, J.-W. Choi, J. ao F. Henriques, J. van de Weijer, J. Batista, K. Lebeda, K. Öfjäll, K. M. Yi, L. Qin, L. Wen, M. E. Maresca, M. Danelljan, M. Felsberg, M.-M. Cheng, P. Torr, Q. Huang, R. Bowden, S. Hare, S. Y. Lim, S. Hong, S. Liao, S. Hadfield, S. Z. Li, S. Duffner, S. Golodetz, T. Mauthner, V. Vineet, W. Lin, Y. Li, Y. Qi, Z. Lei, and Z. Niu, “The visual object tracking vot2014 challenge results,” 2014.

REFERENCES

- [132] J. Gao, H. Ling, W. Hu, and J. Xing, “Transfer learning based visual tracking with gaussian processes regression,” in *European Conference on Computer Vision (ECCV) 2014*, pp. 188–203, 2014.
- [133] J. Zhang, S. Ma, and S. Sclaroff, “MEEM: robust tracking via multiple experts using entropy minimization,” in *European Conference on Computer Vision (ECCV) 2014*, pp. 188–203, 2014.
- [134] K. Zhang, Q. Liu, Y. Wu, and M.-H. Yang, “Robust tracking via convolutional networks without learning,” *arXiv preprint arXiv:1501.04505*, 2015.
- [135] N. Wang, S. Li, A. Gupta, and D.-Y. Yeung, “Transferring rich feature hierarchies for robust visual tracking,” *arXiv preprint arXiv:1501.04587*, 2015.
- [136] Q. Zhang and B. Li, “Discriminative k-svd for dictionary learning in face recognition,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2691–2698, 2010.