Advances in RGB and RGBD Generic Object Trackers

Thesis by

Adel A. Bibi

In Partial Fulfillment of the Requirements

For the Degree of

Master of Science

in Electrical Engineering

King Abdullah University of Science and Technology, Thuwal,

Kingdom of Saudi Arabia

The thesis of Adel A. Bibi is approved by the examination committee

Committee Chairperson: Dr. Bernard Ghanem

Committee Member: Dr. Wolfgang Heidrich

Committee Member: Dr. Tareq Al-Naffouri

ABSTRACT

Generic Object Tracking

Adel A. Bibi

Visual object tracking is a classical and very popular problem in computer vision with a plethora of applications such as vehicle navigation, human computer interface, human motion analysis, surveillance, auto-control systems and many more. Given the initial state of a target in the first frame, the goal of tracking is to predict states of the target over time where the states describe a bounding box covering the target. Despite numerous object tracking methods that have been proposed in recent years [1–4], most of these trackers suffer a degradation in performance mainly because of several challenges that include illumination changes, motion blur, complex motion, out of plane rotation, and partial or full occlusion, while occlusion is usually the most contributing factor in degrading the majority of trackers, if not all of them. This thesis is devoted to the advancement of generic object trackers tackling different challenges through different proposed methods. The work presented propose four new state-of-the-art trackers. One of which is 3D based tracker in a particle filter framework where both synchronization and registration of RGB and depth streams are adjusted automatically, and three works in correlation filters that achieve state-of-the-art performance in terms of accuracy while maintaining reasonable speeds.

Keywords: Trackers, Correlation Filters, Convolution Filters, Sparse Representation, RGBD Trackers, Synchronization, Registration.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation and gratitude to my supervisor, Prof. Bernard S. Ghanem for his unconditional support and help during my Masters. He has been a great source of knowledge and has inspired me with precious ideas. Without his guidance, encouragement and patience this work would not have been by any chance possible.

Additionally, I would like to thank all the members of the IVUL group from post-docs Dr. Baoyouan, Dr. Tianzhu Zhang, and Dr. Ganzhuan Yuan to students Fabian, Victor, Matthias, Jean, Lama, and Sara. Their comments and advises shaded the lights on to many aspects of this work. I would like in specific to thank Matthais Mueller and Jean Lahoud for their enriching side discussions that made this work possible. I can't miss my dear friends from high school and Kuwait University, Omar, Alaa, Ashraf, Essam, Haddad, and Ossama as they have always encouraged me in pursing graduate school and for being whom they are.

I am also very grateful to Prof. Wolfgang Heidrich and Prof. Tareq Al-Naffouri for being part of the Master Thesis committee.

Finally, my deepest love and gratitude is devoted to my parents Amer Bibi and Yasmin Al Najjar. They have and will always be an endless source of love, support and encouragement. In my darkest times, I found my way to light through their continuous prayers. To them, I dedicate this work.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

RGB	Red, Green, and Blue
RGBD	Red, Green, Blue, and Depth
HOG	Histogram of Oriented Gradients
CF	Correlation Filters
VOT	Visual Object Tracking
OTB	Online Benchmark Tracking
KCF	Kernelized Correlation Filters
CST	Circulant Sparse Tracker

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Visual object tracking is a classical problem in computer vision. It plays an important role in a plethora of applications, such as robotics, surveillance, and human-computer interaction to name a few. Object tracking can be defined as the task of localizing an object of interest (e.g. by an upright bounding box) in every frame starting from a given patch containing the object in the first frame [1–4]. The problem is very challenging because the object could undergo a variety of transformations making it harder to localize. Typical nuisances that have to be overcome by a successful object tracker include occlusion, in- and out-of-plane rotation, fast motion, illumination changes, etc. This thesis is devoted to the advancement of generic object trackers tackling different challenges through different proposed methods. The work includes RGBD and RGB based trackers and spans both the discriminative and generative based trackers which will be discussed indetails coming chapters.

1.1 Related Work

Trackers can be divided into two main categories depending on the use of depth information into RGB (Red, Green, and Blue) and RGBD (Red, Green, Blue, and Depth) based trackers where D in RGBD stands for depth.

RGB Trackers. RGB Trackers in general can be divided into two main categories: generative and discriminative trackers. Generative trackers adopt an appearance model to describe the target observations. The main objective of generative based trackers is to search for the target that is the most similar in appearance to an online learnt generative model. As such, the major contribution of this type of trackers is in developing suitable and versatile representative models that can reliably describe the object even when it undergoes different appearance changes. Examples of generative models include the mean shift [7], incremental (IVT) [8], fragment-based (Frag) [9], L1-min [10], multi-task (MTT) [11, 12], low-rank sparse [13], exclusive context modelling based tracker [14], occlusion detection based structural sparse learning based tracker [15] and structural sparse tracker [16].

On the other hand, discriminative trackers formulate visual object tracking as a binary classification problem that searches for the target location that is the most distinctive from the background. Examples of discriminative trackers include multiple instance learning tracking (MIL) [17], ensemble tracking [18], support vector tracking [19], all the correlation filter based trackers [20–25] and other multi-object based trackers like the tracklet association with identity constraints [26]. Despite the fact that correlation filters are considered to be part of the discrimative trackers, they have been considered lately as an independent group all together especially after their recent popularity. Correlation filters in fact have been used for a long time in classification problems, where the objective is to learn filter taps that minimize the energy response of the filter [27] or minimize the variance of the response over a set of given training examples [28]. The filter taps are usually computed in the time domain in a simple least squares formulation. Later, Bolme *et al.* proposed a method for learning the taps in the frequency domain [22], thus, achieving impressively high frame rates since all the necessary computations degenerate to elementary additions and multiplications. The correlation filter was extended even further to

handle multi-dimensional features (beyond gray scale), when its kernelized version (KCF) was proposed in [20, 21]. KCF solves for the filter taps very efficiently by utilizing kernel functions and the circulant structure of the underlying kernel matrix. Since then, using correlation filters for tracking has attracted more attention in the vision community. For example, some trackers use multiple KCF trackers to represent different parts of the object and track them jointly [24]. In [29], they learn an online random fern classifier over a larger region to identify failures and re-detect the object in case of long term occlusion or when the target exits out of the field-of-view. Similarly, Galoogahi *et al.* [25] propose a method to deal with the boundary effects of circularly shifted patches by pre-multiplying them with a masking matrix; however, the resulting optimization is unable to exploit circular structure. Despite KCF's popularity and versatility, some drawbacks of the method remain which will be discussed in detail in coming chapters.

RGBD Trackers. As for the RGBD domain, only a limited number of generic methods exist in the literature, owing to the novelty of the problem and the only recent availability of RGBD tracking data. In [5], a computationally expensive tracker is proposed that combines the SVM scores of two tracking-by-detection instances: one based on RGB and depth HOG features and the other is based on point cloud features. This scoring function is used to evaluate a dense sampling of 3D non-overlapping cells, thus, incurring a large computational cost. Also, the authors of [5] propose an expensive depth based histogram segmentation method (based on a strict assumption that the object lies in the nearest plane) to detect occlusion. Moreover, the conventional and high-speed KCF tracker [20, 21] was adapted to the RGBD domain in [30]. Similar to [5], occlusion is claimed to be handled by a segmentation of the depth histogram followed by a connected component analysis to incorporate spatial features of the object. As mentioned in the paper, the results of [30] on the Princeton benchmark relied on an apriori synchronization and re-aligning of the

provided RGB and depth sequences. To the best of our knowledge, this process was most probably done manually, since no elaboration was given on the method used. Other 3D trackers usually rely on a target-specific 3D object model, which is known apriori [31]. For example, the trackers of [32, 33] assume a 3D model of a hand is provided before tracking begins. Other methods like [34, 35] are category-specific RGBD trackers, which are mainly used for human detection and tracking.

1.2 Outline of the Thesis

In Chapter 2, we present a part-based sparse tracker in a particle filter framework where both the motion and appearance model are formulated in 3D. The motion model is adaptive and directed according to a simple yet powerful occlusion handling paradigm, which is intrinsically fused in the motion model. Also, since 3D trackers are sensitive to synchronization and registration noise in the RGB and depth streams, we propose automated methods to solve these two issues. Extensive experiments are conducted on a popular RGBD tracking benchmark, which demonstrate that our tracker can achieve superior results, outperforming many other recent and state-of-the-art RGBD trackers.

As for chapter 3, the work identifies the major drawbacks of a very computationally efficient and state-of- the-art-tracker known as the Kernelized Correlation Filter (KCF) tracker. These drawbacks include an assumed fixed scale of the target in every frame, as well as, a heuristic update strategy of the filter taps to incorporate historical tracking information (i.e. simple linear combination of taps from the previous frame). In our approach, we update the scale of the tracker by maximizing over the posterior distribution of a grid of scales. As for the filter update, we prove and show that it is possible to use all previous training examples to update the filter taps very efficiently using fixed-point optimization. We validate the efficacy of our approach on

two tracking datasets, VOT2014 and VOT2015.

Chapter 4 is dedicated to investigate another drawback of correlation filters (CF) based trackers that utilize the circulant structure of the training data to learn a linear filter that best regresses this data to a hand-crafted target response. These circularly shifted patches are only *approximations* to actual translations in the image, which become unreliable in many realistic tracking scenarios including fast motion, occlusion, etc. In these cases, the traditional use of a single centered Gaussian as the target response impedes tracker performance and can lead to unrecoverable drift. To circumvent this major drawback, in this chapter we propose a generic framework that can adaptively change the target response from frame to frame, so that the tracker is less sensitive to the cases where circular shifts do not reliably approximate translations. To do that, we reformulate the underlying optimization to solve for both the filter and target response *jointly*, where the latter is regularized by measurements made using actual translations. This joint problem has a closed form solution and thus allows for multiple templates, kernels, and multi-dimensional features. Extensive experiments on the popular OTB100 benchmark [6] show that our target adaptive framework can be combined with many CF trackers to realize significant overall performance improvement (ranging from 3%-13.5% in precision and 3.2%-13% in accuracy), especially in categories where this adaptation is necessary (e.g. fast motion, motion blur, etc.).

Lastly, chapter 5 proposes a method to incoorpertate correlation filters in a sparse representation framework. Sparse representation has been introduced to visual tracking by finding the best target candidate with minimal reconstruction error within the particle filter framework. However, most sparse representation based trackers have high computational cost, less than promising tracking performance, and limited feature representation. To deal with the above issues, we propose a novel circulant sparse tracker (CST), which exploits circular target templates. Because of the cir-

culant structure property, CST has the following advantages: (1) It can refine and reduce particles using circular shifts of target templates. (2) The optimization can be efficiently solved entirely in the Fourier domain. (3) High dimensional features can be embedded into CST to significantly improve tracking performance without sacrificing much computation time. Both qualitative and quantitative evaluations on challenging benchmark sequences demonstrate that CST performs better than all other sparse trackers and favorably against state-of-the-art methods.

Chapter 2

3D Part-Based Sparse Tracker with Automatic Synchronization and Registration

2.1 Introduction

Fortunately, there is a recent surge in the availability of affordable and increasingly reliable RGBD sensors that provide image and depth data such as the Microsoft Kinect, Asus Xtion, and PrimeSense. When depth data is available, there exist more visual cues that can help resolve the nuisances of object tracking, especially occlusion. Recently, a relatively large RGBD dataset for visual tracking (100 videos) was compiled and released to the public [5] in the form of an online competition, where the ground truth object tracks are mostly suppressed. Since then, much deserved attention has been brought towards developing robust RGBD visual trackers. Although only a few of these trackers exist, they easily and with a big margin outperform state-of-art RGB trackers on the same videos. This is a clear evidence of how depth information can be useful to visual tracking, especially regarding early and robust detection of occlusion and proper model update, both of which can significantly boost the performance of

any tracker.

Along with providing an RGBD benchmark [5], Song *et al.* proposed an RGBD tracker that performs quite well on the benchmark. They adopt an exhaustive search paradigm coupled with an SVM classifier trained on image/depth HOG and point cloud features, which reduce the tracker's runtime to only 0.26 FPS. An occlusion handling method is also adopted, where the tracked object is assumed to be contained in the plane closest to the camera. This method of handling occlusions makes it difficult to properly update the target's depth throughout the frames.

Recently, the work in [30] incorporated depth information into the popular Kernelized Correlation Filter tracker [20, 21]. This method demonstrates very promising performance with real-time speeds of up to 40 FPS. However, tracking is still done in the 2D image plane, which might not make use fully of the depth information. Also, an occlusion handling method was proposed that is very similar in spirit to [5], thus, suffering from similar difficulties.

Part-based RGB trackers have been prevalent for a while now. They demonstrate very desirable performance as compared to RGB trackers that use only one part, owing to the fact that an object can still be tracked as long as some (not necessarily) all its parts are visible in the next frame. This is most helpful in handling partial occlusions and recovering from complete occlusions. Inspired by this line of work [16, 31, 36], we propose a 3D part based-tracker, whereby the parts not only help in representing the target but also support the detection of partial or full occlusion.

When testing our tracker and other 3D methods on the RGBD Princeton benchmark [5], we realized that the provided RGB and depth sequences contain many synchronization and registration errors. A synchronization error occurs when an RGB frame is assigned to an incorrect depth frame in the sequence. This happens because the RGB and depth video streams are usually generated independently and although the frame rate of both cameras is similar, it might fluctuate slightly over time and

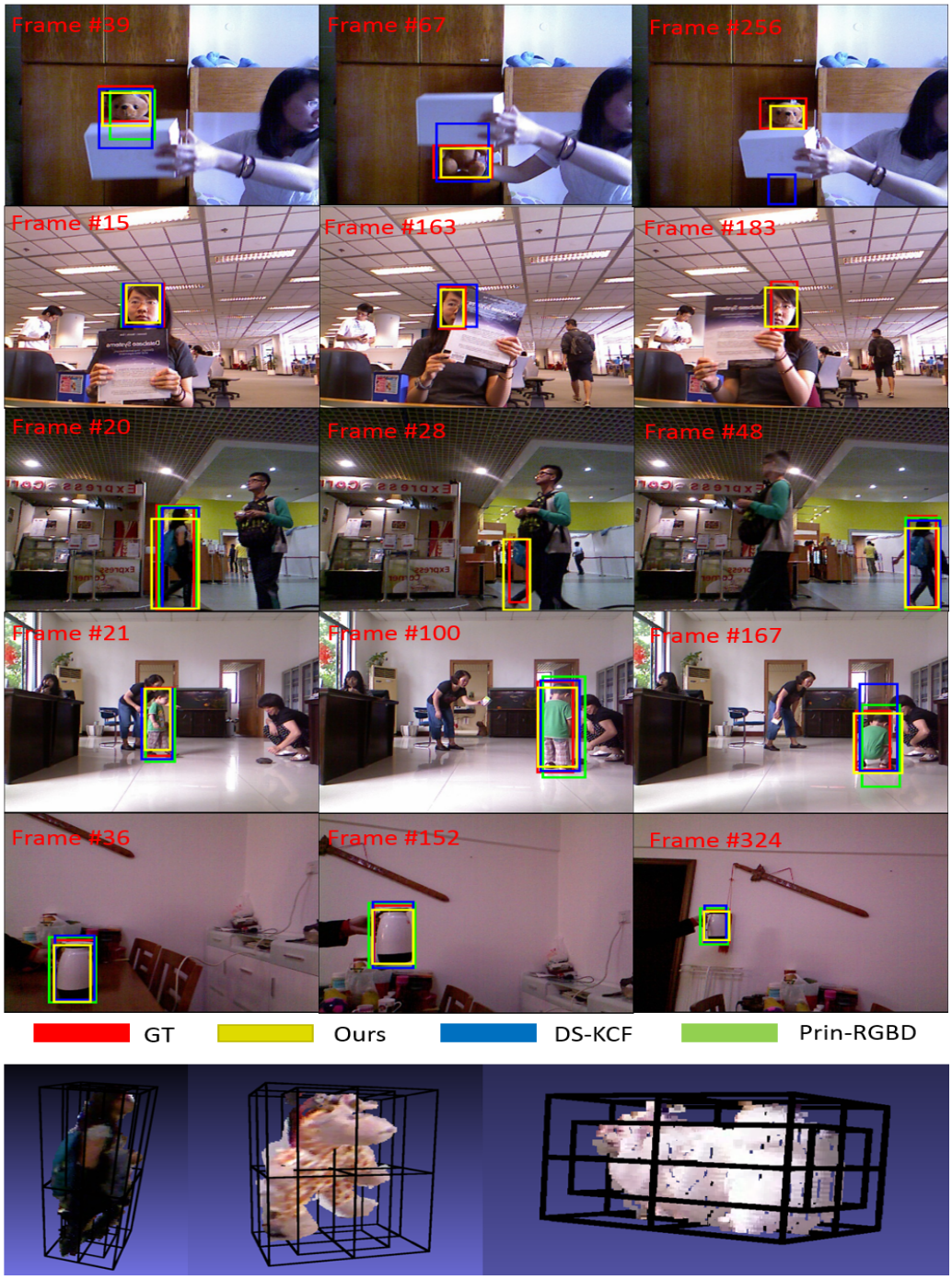

Figure 2.1: **Top**: Shows the tracking results on the videos "bear_front", "face_occ_5", and "new_ex_occ_4" respectively comparing our method, DS-KCF, and Princeton RGBD tracker. **Bottom**: Shows different 3D parts as proposed in this paper.

Figure 2.2: This figure shows the overall pipeline for the proposed method including the three major modules synchronization, registration, and 3D tracking.

there might also be dropped frames. This can cause substantial errors when tracking is done in 3D, since the projection to the image plane is affected. For example, even if the 3D tracking results are accurate, their resulting 2D bounding boxes, which are in turn used to evaluate the RGBD tracker, will not be since they are with respect to the depth image plane while the ground truth is in the RGB image plane. On the other hand, a registration error usually occurs due to imprecise calibration between the RGB and depth cameras. This imprecision manifests itself in erroneous assignment of depth values to RGB pixels. For example, background RGB pixels are assigned the target's depth values and vice versa. Due to both types of errors, the tracking performance of methods, which perform representation, sampling, and tracking in 3D, can be significantly affected. In fact, we suspect that this is the reason why the authors of a very recent RGBD tracker [30] mention that they mandate these errors be rectified (mostly probably manually) before tracking can be performed. In this paper, we suggest a simple yet effective method to automatically alleviate many of these errors that plague 3D tracking and possibly other applications that make use of RGBD stream data.

Contributions This work makes two main contributions. **(i)** To the best of our knowledge, we propose the first generic 3D part-based tracker with a very effective part-based occlusion handling method with desirable performance. At the time of submission, outranks all other trackers in the online Princeton RGBD benchmark [5]. **(ii)** We propose a simple method to synchronize and register RGBD videos for the purpose of single object tracking. We also provide to the vision community the

manual synchronization and registration of the videos provided in the benchmark.

2.2 Related Work

RGB Trackers. In general, RGB trackers can be divided into two main categories: discriminative and generative methods. Discriminative trackers formulate visual object tracking as a binary classification problem that searches for the target location that is the most distinctive from the background. Examples of discriminative trackers include multiple instance learning tracking (MIL) [17], ensemble tracking [18], support vector tracking [19], correlation filter based trackers [20–22], and the recent multiple experts entropy minimization (MEEM) tracker [37]. On the other hand, a generative tracker searches for a candidate target that is best represented by its current appearance model. As such, the major contribution of this type of tracker is in developing suitable and versatile representative models that can reliably describe the object even when it undergoes different appearance changes. Examples of generative models include the mean shift [7], incremental (IVT) [8], fragment-based (Frag) [9], L1-min [10], multi-task (MTT) [11, 12], low-rank sparse [13], and structural sparse tracker [16].

RGBD Trackers. As for the RGBD domain, only a limited number of generic methods exist in the literature, owing to the novelty of the problem and the only recent availability of RGBD tracking data. In [5], a computationally expensive tracker is proposed that combines the SVM scores of two tracking-by-detection instances: one based on RGB and depth HOG features and the other is based on point cloud features. This scoring function is used to evaluate a dense sampling of 3D non-overlapping cells, thus, incurring a large computational cost. In our proposed tracker, we avoid this unnecessary computation induced by naive 3D sampling by exploiting the object's part-based structure as well as its previous motion. In doing so, only a very small

number of 3D samples need to be evaluated at any given time. Also, the authors of [5] propose an expensive depth based histogram segmentation method (based on a strict assumption that the object lies in the nearest plane) to detect occlusion, unlike our tracker that infuses the occlusion handling scheme directly in the tracking process without any additional complexity.

Moreover, the conventional and high-speed KCF tracker [20, 21] was adapted to the RGBD domain in [30]. Similar to [5], occlusion is claimed to be handled by a segmentation of the depth histogram followed by a connected component analysis to incorporate spatial features of the object. As mentioned in the paper, the results of [30] on the Princeton benchmark relied on an apriori synchronization and re-aligning of the provided RGB and depth sequences. To the best of our knowledge, this process was most probably done manually, since no elaboration was given on the method used. In this paper, we show how this issue can be alleviated automatically. In [38], a 2D particle filter was adopted in which the sampling variance of each individual particle changes according to its occlusion state. Despite its good performance on the benchmark, its representation and motion models are both constructed in 2D, unlike our method that natively treats the problem in 3D, from both representation and motion perspectives. In [39], authors build an adaptive boosting classifier from a pool of features (color, depth, and grayscale) that is incrementally re-trained from frame-to-frame. Similar to the previous method, this tracker operates solely in representation in 2D with no clear method of handling occlusion.

Other 3D trackers usually rely on a target-specific 3D object model, which is known apriori [31]. For example, the trackers of [32,33] assume a 3D model of a hand is provided before tracking begins. Other methods like [34, 35] are category-specific RGBD trackers, which are mainly used for human detection and tracking.

In this chapter, we propose a 3D particle filter tracker that exploits sparse representation, object structure (parts), as well as, adaptive particle sampling and pruning,

all in a unified framework. We also present a simple yet effective method to automatically re-synchronize and re-register RGB and depth pairs for better tracking performance. Through extensive experiments, we show that our tracker outperforms all the state-of-the-art methods in Princeton RGBD benchmark by ranking first and third on the manually and automatically synchronized and registered data respectively in the online evaluation.

2.3 Part-Based Sparse 3D tracker

In this paper, we propose a 3D part-based particle-filter tracker, where both representation and motion models are constructed entirely in 3D. We also propose a simple yet effective method for handling structural information provided by the object parts and how it is used for both representation and particle pruning. We represent target candidates (particles) and their parts using a linear sparse representation. Moveover, an occlusion handling scheme supported by the part representation is integrated with the motion model so as to adaptively guide/direct how particles are sampled in the next frame. As compared to tracking-by-detection methods that densely sample the 3D space, our strategy produces only a small number of particles needed for 3D tracking. As mentioned earlier, synchronized and registered data is exceptionally important for 3D trackers; therefore, we propose an automatic method to synchronize and register RGBD sequences and apply it to those in the Princeton RGBD benchmark [5]. The overall pipeline of our method is illustrated in Figure 2.2.

2.3.1 Representation Model

Generative RGB trackers [10, 16, 40] and most RGBD trackers [30, 38] define their target candidates (e.g. particles) to be 2D bounding boxes in the RGB or depth image plane, from which features for representation are extracted. In our formulation, we

use a particle filter to sample 3D cuboid candidates. To limit the number of particles to a practical number, we propose an adaptive data-driven sampling approach. Each particle cuboid is divided into a pre-defined number of overlapping parts, each of which is defined also as a 3D cuboid. For convenience, all particles have the same part layout. One of these parts covers 65% volume of the entire particle (cuboid) and is located at its center as seen in Figure 2.1. This part captures holistic object information, while the other smaller parts capture information from different sides of the object (refer to Figure 2.1). Clearly, other (possibly hierarchical) part layouts can be used here too. Each part is represented using a $m = 13$ dimensional feature: ten color names and three 3D shape features, as proposed in [5]. We model each particle part as a sparse linear combination of a set of dictionary elements. To do this, we adopt a similar approach as in [10, 16, 40]. At the first frame, we collect several observations of the object (and its parts) by sampling multiple cuboids around its ground truth location. We build $K = 2$ sparsifying dictionaries (using KSVD [41], one for each type of feature and for each part. Therefore, the representation of each particle can be described mathematically as follows:

$$\min_{\mathbf{X}_k} \sum_{k}^{K} \|\mathbf{D}_k\mathbf{X}_k - \mathbf{Y}_k\|_F^2 + \lambda\|\mathbf{X}_k\|_{1,1}, \tag{2.1}$$

where $\mathbf{D}_k$ denotes the dictionary corresponding to the k^{th} feature type, such that $\mathbf{D}_k = [\hat{\mathbf{D}}_{1k}|\hat{\mathbf{D}}_{2k}|...|\hat{\mathbf{D}}_{Nk}|\mathbf{I}_{m_k}]$, where $\hat{\mathbf{D}}_{ik} \in \mathbb{R}^{m_k \times n_k}$, where m_k and n_k are the dimensionality of the k^{th} feature space and the number of atoms in dictionary k of the part i respectively, and $\mathbf{I}_{m_k}$ is an identity matrix that encodes sparse error (e.g. partial occlusion); therefore, $\mathbf{D}_k \in \mathbb{R}^{m_k \times (Nn_k + m_k)}$. For computational reasons, these dictionaries are not updated with time. We concatenate the k^{th} feature type for all the parts of a particle in the observation matrix $\mathbf{Y}_k \in \mathbb{R}^{m_k \times N}$, where N is the total number of parts ($N = 9$ in our experiments). The resulting sparse code matrix is denoted as $\mathbf{X}_k \in \mathbb{R}^{(Nn_k + m_k) \times N}$. The optimization in Eq (2.1) is non-smooth but convex. It

is the matrix form of the popular Lasso problem. It can be efficiently solved using a number of methods, including Alternating Direction Method of Multipliers (ADMM).

Temporal Coherence on Part Structure

Unlike sampling in 2D where the image plane is dense, most of the 3D point cloud scene is in fact empty. This may result in empty particle parts or completely empty particles. Scoring particles that are partially or completely empty using the objective in Eq (2.1) is not appropriate because the objective is not representative of these instances. That is, if one of the parts has disappeared, one of the two very different scenarios have occurred. Either the part is occluded, or the particle having that part is not representative enough. The first has to be associated with a low cost, while the latter with a high one. In Eq (2.1), an empty part will result in a low cost for both scenarios. That means the objective will favor parts that are empty; thus, leading the tracker to drift into completely empty regions in 3D space. Similarly, setting the cost too high for empty parts within a cuboid would favor dense regions and will lead to favoring any nearby object, even if it were the occluder.

To address this issue, we make use of the temporal coherence of the object's part structure, which is modeled using the distribution of 3D points within each part of the current target. For simplicity, each part is described with a single binary value (1 or 0), which depicts whether that part is empty or not. As such, each particle is described using an N-dimensional binary vector, denoted as the part-based structure feature. We expect that this feature changes gradually with time, i.e. many parts tend not to abruptly disappear or re-appear in the next frame. To preserve structural information in between consecutive frames and to determine occluded parts, we use the hamming distance between the binary structure feature of the current target and that of each particle sampled in the next frame. Only those particles with the minimum hamming distance are selected and represented using Eq (2.1). This strategy also helps prune

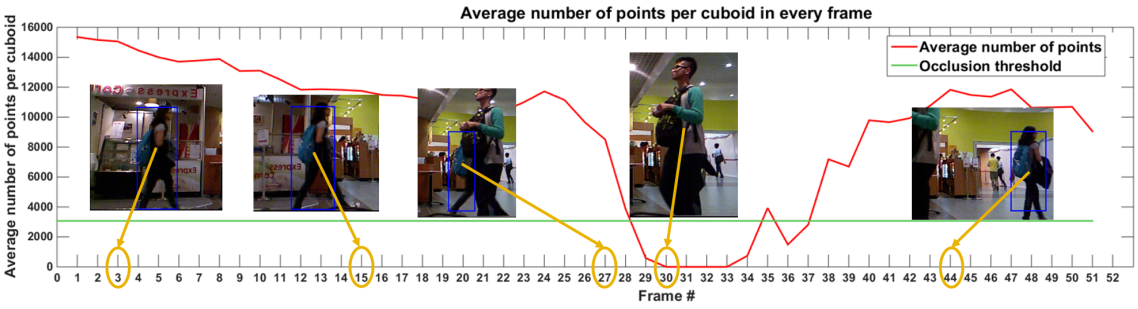

Figure 2.3: Shows the proposed occlusion handling scheme on video "new_ex_occ_4 ", where the number of points in all the particles decreases significantly in 3D space when the tracked object is occluded.

unlikely particles, thus, substantially reducing the overall computation time needed for representation.

2.3.2 Motion Model

In this part, we give a detailed explanation of how we use particle filters in our tracker. But first, we give a brief of particle filtering.

Particle Filter

The particle filter is a Bayesian sequential important sampling technique for estimating a posterior distribution of state variables $\mathbf{x}_t$ that characterize a dynamic mode. It consists of two major steps, the prediction and the update for re-sampling. At the new instance t, a new observation $\mathbf{z}_t$ is available and the new probability distribution given all observations and the previous state is given by:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \tag{2.2}$$

As for the update step, it is based on $p(\mathbf{z}_t|\mathbf{x}_t^i)$. As such, more particles will be sampled from a state that has been observed with a higher probability. It is computed as follows:

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

Adaptive Directed Sampling

Owing to the way objects move in 3D and since our particles are modeled as cuboids, we take the state vector $\mathbf{x}_t$ to represent a 3D rigid body transformation, i.e. $\mathbf{x}_t \in \mathbb{R}^6$, which is characterized by a 3D rotation (3 DoF) and translation (3 DoF). Note that scale could be incorporated in this setup; however, in most cases, the 3D scale of a target does not change dramatically from its initial value. Depending on the state of the object (whether it is occluded or not), the motion model changes accordingly.

No occlusion. To not sample particles unnecessarily, we do *not* use a zero-order motion model. Instead, we use 2D optical flow on the current target to the next RGB frame to compute a crude estimate of its new 3D location. Since only a crude estimate is needed, most optical flow methods can be used here. So, for mainly computational reasons, the basic Horn-Schunck optical flow [42] is sufficient. In fact, we experimented with more sophisticated large-displacement methods (e.g. the work in [43]), only to find that the tracking performance is only subtly affected. Given the pixel correspondences from optical flow, we can get a crude estimate of the rigid body transformation (rotation $\mathbf{R}$ and translation $\mathbf{t}$) of the current target. Then, we sample the particle states using a Gaussian distribution centered at the previous estimate $(\mathbf{R}, \mathbf{t})$ and with a diagonal covariance. Since the orientation of a target cuboid does not change much over time (e.g. a human walking on a planar surface), we set the variances of the translation parameters to be much larger than those of the rotation.

Occlusion. When the target is determined to be in an occlusion state (as we will describe in the next section), optical flow is no longer a valid measure. Therefore, we resort back to a zero-mean motion model with large translational variance, so as to recapture the target when it re-appears.

Occlusion Handling

We integrate the occlusion handling scheme with the particle filter formulation. As discussed earlier, each sampled particle represents a cuboid, which contains a certain number of 3D points. In case of occlusion, we observe that this number tends to decrease significantly in all particles all at once. By monitoring this change, we can determine if an object is being occluded or not. Since image resolution is inversely proportional to the distance from the camera, the number of points in a cuboid is also inversely proportional with depth. Therefore, we need to compensate for the number of points in depth by computing a depth-normalized measure: $t_i^j = \left(\frac{z_i^j}{z^1}\right)^2 \bar{t}^1$. Here, z_i^j and z^1 are the average depth values of particle i in frame j and the average depth of all particles in the first frame, respectively. $\bar{t}$ is the average number of points in all particles in the first frame. In case of occlusion at frame j, the depth-normalized number of points among all particles will be very low. If the average falls below some threshold, the object is identified to be in an occlusion state as illustrated in Figure 2.3.

2.3.3 Synchronizing and Registering RGBD Images

There are several videos in Princeton RGBD benchmark [5] with registration and synchronization issues that can substantially affect 3D tracking performance. We consider this as a fundamental problem that needs to be addressed because it not only affects 3D trackers but many other computer vision applications that involve operations on 3D data. In fact, it is clearly stated in previous work [30] that the RGB and depth sequences need to be synchronized and realigned properly before tracking can be applied. In the following, we propose a method to solve both problems.

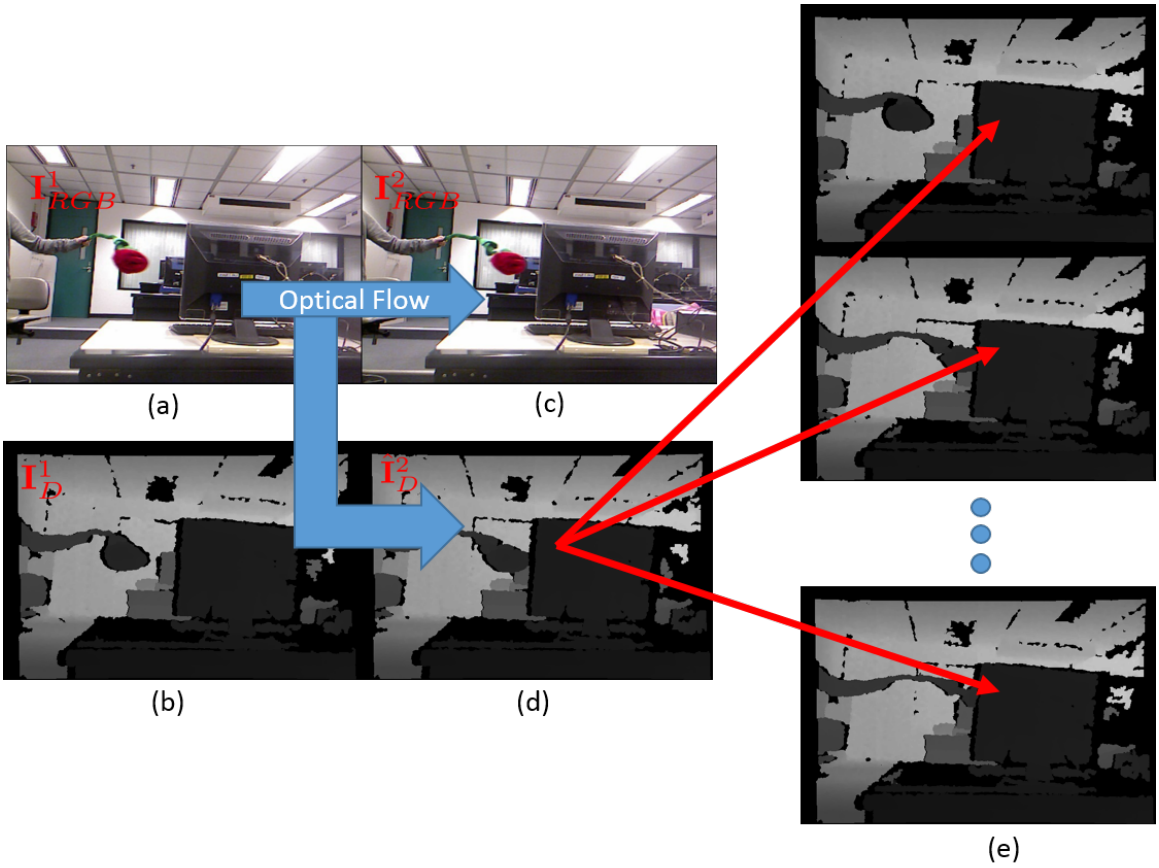

Figure 2.4: (a)&(b) is a previously synchronized RGBD image pair. (c)&(d) is an RGBD pair formed of the next RGB image in the sequence and its corresponding synthetic depth image generated using optical flow. (e) shows a set of possible matches to (c) most similar to (d).

Synchronization

As explained earlier, synchronization problems arise in RGBD videos because the RGB and depth streams tend to be acquired independently, with different frame rates, and sometimes frames are dropped. A commonly used method to resolve this issue is to assign a depth image to every RGB image that has the closest time stamp. This is what is done in the benchmark [5]. Obviously, this issue can also be resolved from a hardware perspective by simply increasing the frame rate of both cameras, thus, reducing the effect of time stamp offset. In this section, we propose an automatic method to alleviate the synchronization problem and formulate it as a matching task.

We take the RGB sequence as reference. We seek to match a depth image to every image in the RGB sequence. An RGB image is allowed to match to any depth image, whose time stamp is close enough to that of the RGB image. We start with a manually synchronized pair of RGB and depth images in the first frame, denoted as $\mathbf{I}^1_{RGB}$ and $\mathbf{I}^1_D$ respectively. Under the assumption that the depth values of a scene change smoothly between consecutive frames, we can compute a large-displacement optical flow [43] between the current synchronized RGB image $\mathbf{I}^1_{RGB}$ and the next one in the sequence $\mathbf{I}^2_{RGB}$. Only point correspondences $\{(\mathbf{x}^1_i, \mathbf{y}^2_i)\}_{i=1}^p$ with a large enough flow magnitude are maintained, since moving points give clear indication of when synchronization errors occur. Then, we generate a synthetic depth image $\hat{\mathbf{I}}^2_D$ by transferring over depth values, i.e. $\hat{\mathbf{I}}^2_D(\mathbf{y}^2_i) = \mathbf{I}^1_D(\mathbf{x}^1_i)$. If $\mathcal{C}$ is the set of frame indices identifying the depth images that are close in time stamp to $\mathbf{I}^2_{RGB}$, then we can synchronize $\mathbf{I}^2_{RGB}$ to the depth image $\mathbf{I}^{j^*}_D$, whose depth values at $\{\mathbf{y}^2_i\}_{i=1}^p$ are closest to those in $\hat{\mathbf{I}}^2_D$. We formulate this mathematically as follows:

$$j^* = \underset{j \in \mathcal{C}}{\operatorname{argmin}} \sum_{i=1}^{p} \left(\hat{\mathbf{I}}^2_D(\mathbf{y}^2_i) - \mathbf{I}^j_D(\mathbf{y}^2_i) \right)^2 \tag{2.4}$$

This strategy can be applied iteratively until all the images in the RGB sequence are synchronized. Refer to Figure 2.4 for an example.

Registration

Several videos in the Princeton RGBD benchmark [5] also suffer from incorrect registration between synchronized RGB and depth image pairs. Being able to correctly register these pairs is very important, since this registration defines the correspondences between pixels in the RGB image and those in the depth image, which in turn enable the generation of the 3D point cloud of the scene.

The usual strategy for registering these image pairs is to stereo calibrate both the depth and RGB cameras together. This calibration can be used to map pixels

from one of them to the other. This strategy is used in the RGBD benchmark [5]; however, in many cases, registration errors do occur and they could be as large as 30-40 pixels. Such an offset can negatively affect any RGBD based tracker, especially one that tracks the object in 3D. This offset is not uniformly random at each pixel, since the source of the error arises from perturbations in the extrinsic calibration parameters, i.e. the rotation and translation that transforms the coordinate system of the RGB camera to that of the depth. By assuming that the perturbation in the rotation is negligible w.r.t. the perturbation in the translation, it can be shown that the per-pixel offset in registration varies with the depth and image location of the pixel (refer to the **Appendix** for a mathematical treatment). However, this variation is structured. For example, neighboring pixels with similar depth values tend to have very similar offsets. This is why some of the registration errors in the benchmark (corresponding to videos showing a predominant foreground object infront of a far away background) can be easily fixed by simply translating the whole depth image in a single direction. To estimate the registration offsets from one frame to the next, we formulate a structured selection problem as shown in Eq (2.5). Note that we require the first RGBD image pair to be correctly registered.

$$\begin{aligned} \min_{\mathbf{z}_i \forall i} \quad & \frac{1}{p}\sum_{i}^{p}(d_1^i - \mathbf{d}_2^{i\,T}\mathbf{z}_i)^2 \\ s.t \quad & \mathbf{z}_i \in \{0,1\}^N \ \forall i, \quad \mathbf{1}^T\mathbf{z}_i = 1 \ \forall i, \ \text{rank}(\mathbf{Z}) \leq r \end{aligned} \tag{2.5}$$

Here, d_1^i is the depth of pixel i in the previous RGBD image pair, which is assumed to be correctly registered. We assume that the offset of this pixel in the next frame can be one of N possible offsets. The vector $\mathbf{d}_2^i \in \mathbb{R}^N$ contains the depth values in the next depth image corresponding to the N possible offsets. Moreover, the binary vector $\mathbf{z}_i$ can be viewed as a selection vector that chooses only one of the offset depth values in the next frame for pixel i. The selection vectors $\mathbf{Z} = [\mathbf{z}_1|...|\mathbf{z}_p]$ for the p pixels (i.e. matched pixels using optical flow) must be constrained to follow the

perturbation model described above. Since neighboring pixels with similar depths tend to have the same offset, we only expect a small number of distinct offsets to be selected among the p pixels. We formulate this as a low-rank constraint on the binary matrix $\mathbf{Z}$.

Computing the global solution of this binary problem cannot be done in polynomial time, so we seek a tradeoff between solution quality and computational efficiency. We observe that when $r = 1$, any feasible matrix $\mathbf{Z}$ is all zeros except for an entire row. We can exhaustively evaluate all N feasible matrices and return the one leading to the smallest objective value. This is equivalent to selecting a single offset for all p pixels. We can exploit this observation to efficiently compute an approximate solution when $r > 1$. This can be done in two ways. **(i)** We can pre-cluster the p pixels (e.g. according to their depth values d_1^i and their image locations) and then find the best rank-1 solution for each cluster independently. **(ii)** We first find the best rank-1 solution and remove all pixels whose offset has been found (i.e. their contribution to the overall objective is zero). Then, we reiterate this process on the remaining pixels at most $(r - 1)$ times or until no pixels are left.

2.4 Experimental Results

To evaluate the performance of our proposed tracker, we conduct extensive experiments on the well-known Princeton RGBD benchmark [5], comprising a total of 100 video sequences only five of which have publicly available ground truth tracks. The videos contain many challenges including partial and full occlusion, fast motion, out of plane rotation, background clutter, moving camera, and shape deformation and distortion. For comparison, the authors of [5] provide an online evaluation system that compares a tracker's performance with that of 20 others, 12 of which use depth information while the rest are popular state-of-the-art RGB trackers. The evaluation

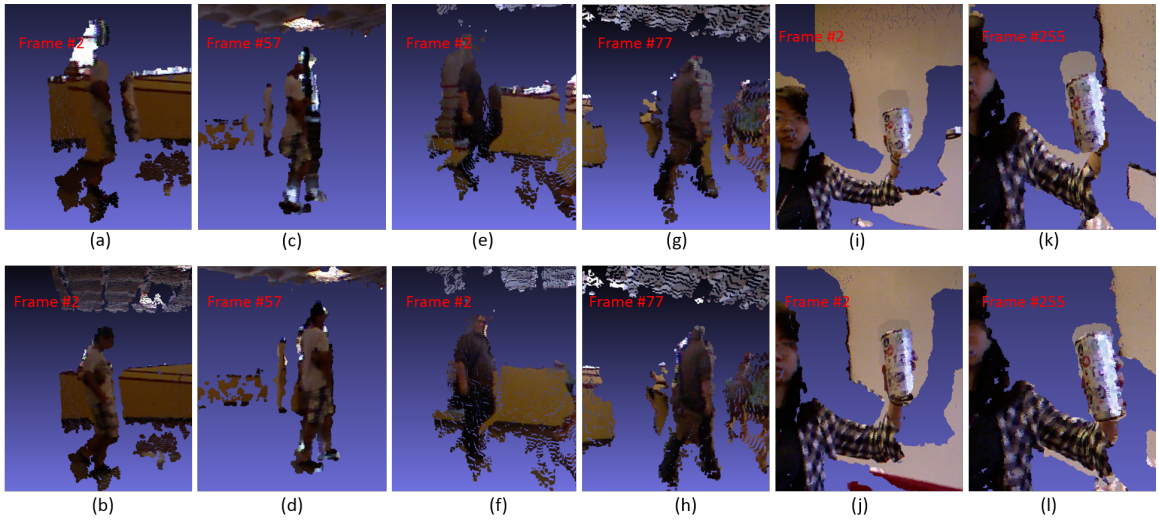

Figure 2.5: Pairs (a)&(b) and (c)&(d) denote the registered RGB and depth pairs as provided on the benchmark [5] compared with our method of registration on the video "new_student_center_no_occ"at frames 2 and 57 respectively. (e),(f) and (g),(h) for the video "new_student_center_3 "at frames 2 and 77 respectively. Similarly pairs (i),(j) and (k),(l) for the video "cup_book "at frames 2 and 255 respectively.

for the comparisons among trackers was based on the intersection over union criterion (IOU). For the details of the tracking criterion used for evaluation reader is referred to [5]. To evaluate the merits of our synchronization and registration method, we compare our tracker on both the RGBD data provided in the benchmark and the same data after both methods are applied. Moreover, we empirically validate the benefits of using multiple parts instead of a holistic representation for tracking.

2.4.1 Implementation Details

All our experiments are done using MATLAB R2014b on a 3.07GHz Intel(R) Xeon(R) with 48GB RAM. The number of parts $N = 9$, with $K = 2$ dictionaries learnt for each part. Only a total of 20 particles were used in our method As for particle sampling, we set the standard deviations of the translation component to $\{0.05, 0.05, 0.03\}$ for when the object is not in a state of occlusion and $\{0.35, 0.10, 0.2\}$ when it is. To reduce computational cost, we do not sample the rotation obtained from point cor-

Algorithm	Target type			Target Size		Movement		Occlusion		Motion Type	
	Human	Animal	Rigid	Large	Small	Slow	Fast	Yes	No	Passive	Active
Ours_Manual	0.81	0.64	0.73	0.80	0.71	0.75	0.75	0.73	0.78	0.79	0.73
Ours_Sync_Reg	0.74	0.66	0.70	0.77	0.65	0.76	0.68	0.67	0.76	0.75	0.69
Ours_Sync_Raw_Reg	0.68	0.58	0.71	0.76	0.61	0.74	0.64	0.62	0.74	0.75	0.64
Ours_Raw_Sync_Raw_Reg	0.64	0.57	0.67	0.71	0.58	0.73	0.60	0.57	0.73	0.72	0.61
Ours_One_Part_Manual	0.60	0.56	0.46	0.58	0.50	0.58	0.52	0.48	0.62	0.54	0.54

Table 2.1: Comparison between our proposed tracker in different design variations.

Algorithm	Avg. Rank	Target Type			Target Size		Movement		Occlusion		Motion Type	
		Human	Animal	Rigid	Large	Small	Slow	Fast	Yes	No	Passive	Active
Ours_Manual	2.27	0.81	0.64	0.73	0.80	0.71	0.75	0.75	0.73	0.78	0.79	0.73
OAPF[38]	2.63	0.64	0.85	0.77	0.73	0.73	0.85	0.68	0.64	0.85	0.78	0.71
RGBDOcc+OF[5]	2.81	0.74	0.63	0.78	0.78	0.70	0.76	0.72	0.72	0.75	0.82	0.70
Ours_Sync_Reg	3.72	0.74	0.66	0.70	0.77	0.65	0.76	0.68	0.67	0.76	0.75	0.69
DS-KCF[30]	4.54	0.67	0.61	0.76	0.69	0.70	0.75	0.67	0.63	0.78	0.79	0.66
RGBD+OF[5]	5.27	0.64	0.65	0.75	0.72	0.65	0.73	0.66	0.60	0.79	0.74	0.66
PCdet_flow[5]	7.27	0.51	0.52	0.73	0.63	0.56	0.74	0.53	0.55	0.64	0.75	0.53

Table 2.2: Tracking results from the online evaluation for our tracker on both the manually and automatically synchronized and registered data compared with the top 5 trackers.

respondences. Moreover, we set the Lasso parameter $\lambda = 0.05$ as a decent tradeoff between sparsity and meaningful reconstruction. For occlusion handling, when the depth-normalized average number of points in a frame for all particles falls below $0.2\bar{\mathbf{t}}^1$, the target is in an occlusion state.

As for synchronization, any RGB image is allowed to match to one depth image that is within 5 frames from it. For the registration problem, we use the offset in the previous frame as an initialization for the next one.

2.4.2 Evaluation of Design Choices

In the following paragraphs we show the impact of the tracking performance and some qualitative results on the automatically synchronized and registered data compared

with the provided one on the benchmark [5]. In defense of parts, we show the huge impact in performance when considering multiple parts as opposed to using only one.

Synchronization results. As discussed earlier, around 14% of Princeton RGBD benchmark [5] videos are in synchronization error, while other 8% need registration on top of synchronization. Since this is very important for 3D based trackers, Table 2.1 shows the proposed tracker results on the RGB and depth data as provided by [5] compared with the tracking results on our proposed method of synchronization. A notable improvement in performance in both the fast motion and occlusion category is expected. This is because for un-synchronized data during a fast motion, the RGB and depth image pairs will be significantly different. There is around 5% increase in performance in the occlusion category due to the fact that the occlusion detector fires at an in-correct occlusion instances.

Registration results. Around 12% of the testing videos provided by [5] suffer from severe incorrect registration. Table 2.1 also shows tracking results on the videos as given by [5] and on the data registered using the proposed method. Significant improvement in performance is noted in almost all categories, again this is mainly because 3D based trackers project the points back into the depth frame's reference, while the evaluation groundtruth is constructed on the RGB image. Also, the improvement is due to the fact that unregistered data will change the features representing the object completely causing color attributes of the background appearing on the object and vice versa. Figure 2.5 illustrates the problem of registration with a qualitative comparison for the registration method.

In the defense of parts. Table 2.1 summarizes the performance of our method when only one holistic part is used. Clearly, adding multiple parts substantially improves performance across *all* tracking categories. This is because the parts help provide a more robust representation of the target, as well as, structural information that is important to prune unnecessary and possibly confusing particles.

2.4.3 Results on the RGBD Tracking Benchmark

In Table 2.2, we summarize the performance of our tracker on both the manually, and automatically synchronized and registered data compared to the top-5 closest trackers on a total of 95 videos. We use the same category breakdown as the online benchmark. The list of competing trackers was shortened to show only the best 5 methods (refer to the **Appendix** for the entire table). Our part-based sparse tracker ranks first among all other methods on the manually synchronized and registered data while ranking third on the automatically synchronized and registered data. In some categories (e.g. Human), it registers an improvement of 7% when compared to the second best tracker. This is attributed to the use of parts and the temporal coherence in their structure, which is a reasonable assumption for humans. While on the the other hand, categories (e.g. Animal, Fast Motion) have their tracked target interacting in very close distance with other objects that have some similar cues in complex motion which makes it hard for the 3D tracker to identify the target uniquely.

Qualitative results. Figure (2.6) shows the tracking results in 3D of sample frames taken from two benchmark videos, namely "new_ex_occ_4 "and "three_people". Both these videos include examples of full occlusion. For visualization purposes, we only show two of the nine constituent parts, denoted as red and blue cuboids. We also backproject these cuboids into the image plane as upright bounding boxes. The yellow cuboids are instances when the object is determined to be in an occlusion state. Notice how our tracker is able to easily recover from this full occlusion.

2.5 Conclusion

In this chapter, we proposed a 3D part-based sparse tracker, which exploits parts to preserve temporal structural information and help in particle pruning. A fast yet

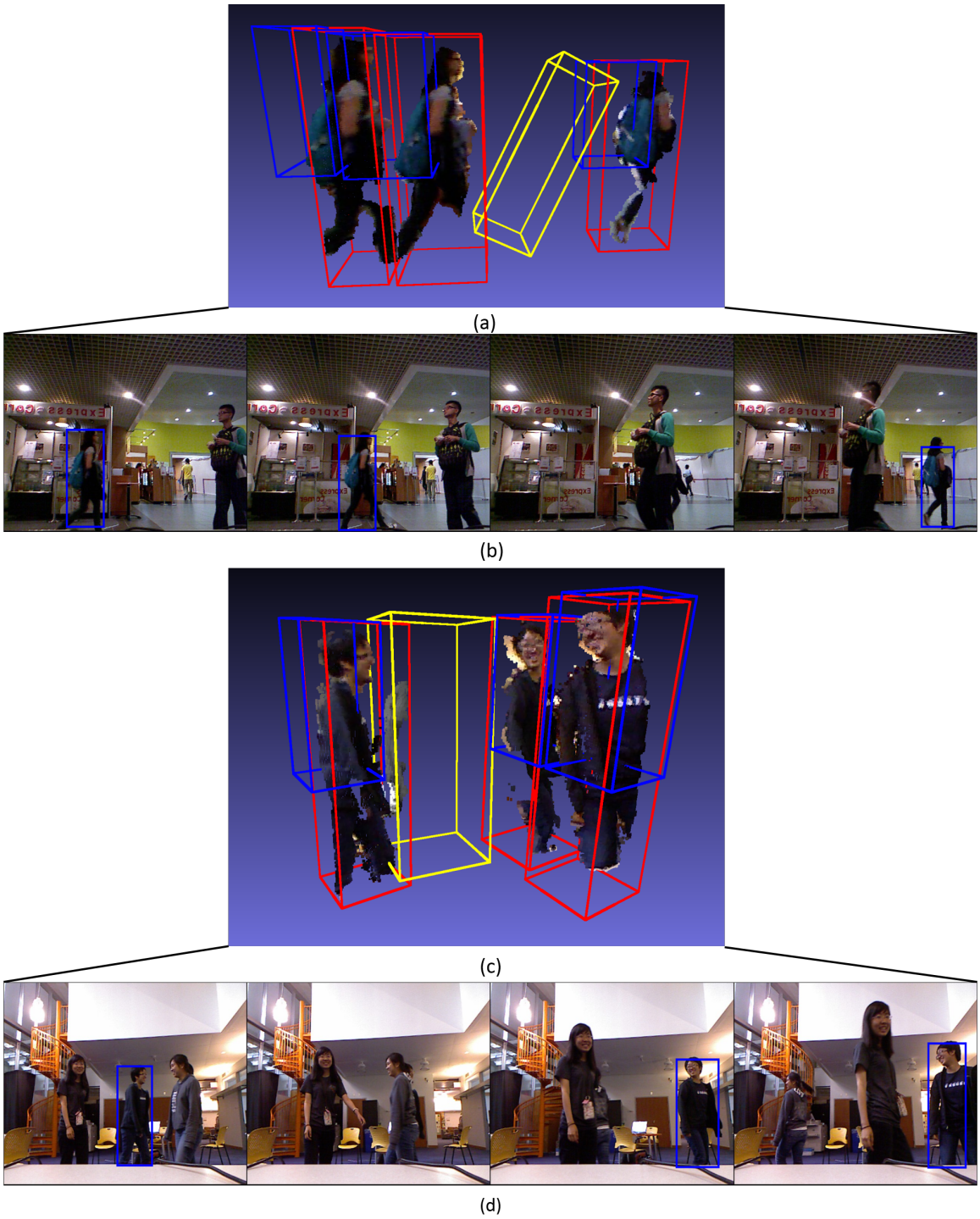

Figure 2.6: Images (a)&(c) show the tracking results in 3D respectively where the red and blue cuboids show two different parts tracked, while the yellow cuboid denotes the occlusion cuboid. Images (b)&(d) show the corresponding 2D tracking results of videos "new_ex_occ _4 "and "three_people "respectively.

powerful method was proposed to embed occlusion detection in the motion model framework. Since 3D trackers are sensitive to synchronization and registration noise, we proposed methods to correct for both, especially since no less than 30% of the videos on the popular RGBD tracking benchmark [5] suffer from these issues. Extensive experiments demonstrate the positive impact of each module of the proposed tracker. In fact, our tracker currently ranks first on the benchmark, as compared to many state-of-the-art trackers. For future work, we aim to exploit part-to-part spatial and appearance relationships to encode particles and to detect occlusion. Moreover, we plan to develop a strategy to incrementally build and maintain a prototypical 3D model of the target by registering its tracking results with time.

Chapter 3

Multi-Template Scale-Adaptive Kernelized Correlation Filters

3.1 Introduction

In this chapter, we build upon a correlation based tracker popularly known as the Kernelized Correlation Filter (KCF) tracker [20, 21]. KCF has achieved impressive results on the visual tracking benchmark [3] and ranked third in the VOT2014 challenge [44] achieving real-time performance. KCF, similar to other correlation filter based trackers, tries to find the best filter taps that maximize the response when correlated with a target template that looks similar in appearance to the training data. KCF solves the problem of tracking by solving a simple rigid regression problem over training data in the dual form, which allows the use of both multi-dimensional features and nonlinear kernels (e.g. Gaussian).

The combination of multi-dimensional features and kernels when learning the filter taps pushed the tracker to be among the best performers. This is made possible because of the over-sampling strategy used in KCF. Instead of taking random samples of the target to train over, an over-sampling method is used to consider all possible translations of the target in a given window. This over-sampling was considered previously to be a drawback because of the large number of redundant samples that

are required. However, when these samples are collected and organized properly, they form a circulant matrix that has very desirable properties, the most interesting of which is that its diagonalization can be efficiently computed using the DFT matrix. Using this over-sampling (in both training and detection), this DFT diagonalization property is used in formulating and solving the dual rigid regression problem entirely in the frequency domain. In fact, the only necessary operations are elements-wise addition, element-wise multiplication, and the FFT.

Due to the attractive properties highlighted above, KCF can accurately predict the target's location in the video frame in real-time. However, two performance-impeding drawbacks with KCF are its use of a fixed target scale in detection [23, 45] and a heuristic filter update rule that makes use of only one template at a time. We solve the issue of scaling through a voting scheme, which selects a scale that maximizes the posterior probability when the prior is Gaussian centered around the scale selected in the previous frame.

Our second contribution lies in how the filter taps are updated from frame-to-frame. In the landmark KCF paper [20, 21], it is argued that the filter taps can be computed in either the primal or the dual form each with its own limitations. Solving the problem in the primal form allows multiple base template in training, but only with one-dimensional features (i.e. gray scale images). On the other hand, solving it in the dual form allows the use of multi-dimensional features (e.g. HoG) and non-linear kernels, but only with a single training template. It was stated in [20] that using both multiple templates and multi-dimensional features is not possible. As can be seen in Table 3.1, using multiple templates in training with a linear kernel in the primal formulation gives very interesting results even outperforming KCF with non-linear kernels. In fact, using the primal formulation with multiple templates ranks first among all the considered trackers that use the same features. This observation motivates us to develop a kernelized correlation filter scheme that incorporates

multiple multi-dimensional templates in training within the dual formulation, while re-using properties of the resulting circulant matrix structure. In this paper, we show that this can be done through an iterative scheme where each iteration solves for multiple filters each corresponding to a training example with a set of constraints that encourage these filters to be the same. By incorporating multiple templates (e.g. tracking results from previous frames), our approach allows for a more systematic update scheme for the filter taps, as compared to the heuristic update rule used in KCF, KCF based trackers, and many other correlation filter based trackers [20, 22, 24, 29]. This iterative update scheme does not tradeoff performance and accuracy for computational efficiency, owing to the underlying circulant matrix structures that arise. By integrating both updates, we demonstrate that our tracker is superior to the original KCF formulation and other state-of-the-art trackers.

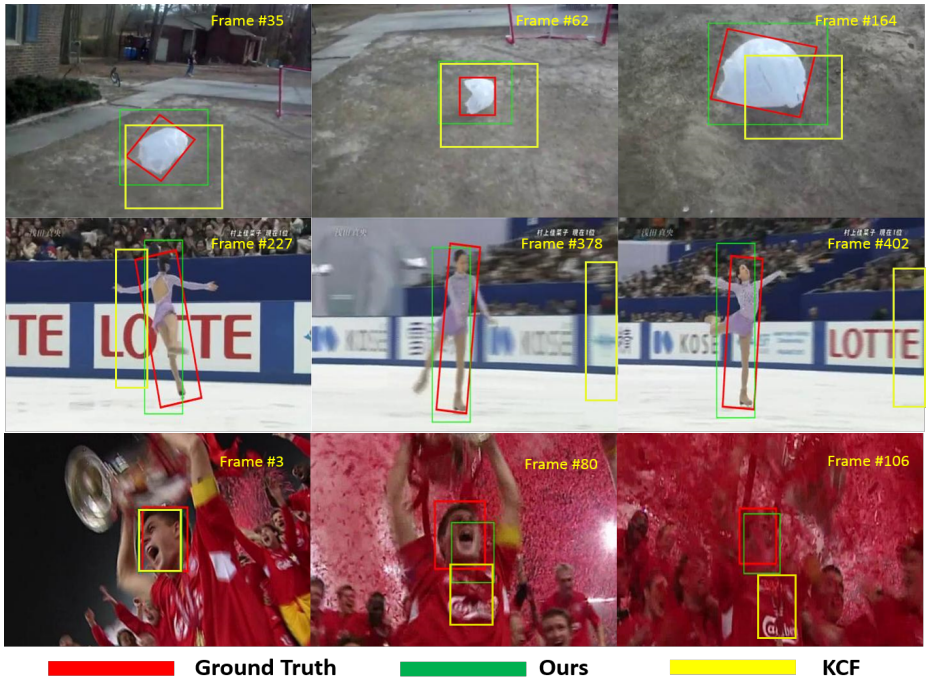

Figure 3.1: Comparison between our method and KCF on 3 sequences from the VOT2015 dataset.

	Acc. Rank	Rob. Rank	Rank
KCF_Gauss_HoG	3.23	2.83	3.03
KCF_Lin_HoG	3.35	2.79	3.07
KCF_Lin_Gray_Multi	**3.41**	**3.20**	**3.30**
KCF_Gauss_Gray	3.79	3.00	3.39
KCF_Lin_Gray	3.99	4.04	4.02

Table 3.1: Multi-templates vs single template updates

3.2 Related Work

Trackers in general can be divided into two main categories: generative and discriminative trackers. Generative trackers adopt an appearance model to describe the target observations. The main aim of the tracker is to search for the target that is the most similar in appearance to the generative model. Therefore, the major contribution of this type of tracker is in developing complex generative representative models that can reliably describe the object even when it undergoes different appearance changes. Examples on generative models include mean shift tracker [7], incremental tracker (IVT) [8], fragment-based tracker (Frag) [9], L1-min tracker [10], multi-task tracker (MTT) [12], low-rank sparse tracker [13], and structural sparse tracking [16] to name a few.

On the other hand, discriminative trackers formulate visual object tracking as a binary classification problem that searches for the target location that is the most distinctive from the background. Examples of discriminative trackers include multiple instance learning tracking (MIL) [17], ensemble tracking [18], support vector tracking [19], and all the correlation filter based trackers. Correlation filters have been used for a long time in classification problems, where the objective is to learn filter taps that minimize the energy response of the filter [27] or minimize the variance of the response over a set of given training examples [28]. The filter taps are usually computed in the time domain in a simple least squares formulation. Later, Bolme *et al.* proposed a method for learning the taps in the frequency domain [22], thus,

achieving impressively high frame rates since all the necessary computations degenerate to elementary additions and multiplications. The correlation filter was extended even further to handle multi-dimensional features (beyond gray scale), when its kernelized version (KCF) was proposed in [20, 21]. KCF solves for the filter taps very efficiently by utilizing kernel functions and the circulant structure of the underlying kernel matrix. Since then, using correlation filters for tracking has attracted more attention in the vision community. For example, some trackers use multiple KCF trackers to represent different parts of the object and track them jointly [24]. In [29], they learn an online random fern classifier over a larger region to identify failures and re-detect the object in case of long term occlusion or when the target exits out of the field-of-view. Despite KCF's popularity and versatility, some drawbacks of the method remain. We investigate and address two of these drawbacks in this paper, namely multiple template training and scale adaptation.

We organize the rest of the paper as follows. Section 3.3 is a brief description of the KCF tracker, while Section 3.4 describes our method. Section 3.5 provides empirical evidence for our method as compared to other trackers, while Section 2.5 concludes the paper.

3.3 Review of KCF Tracker

KCF tracker has gained attention recently for achieving very impressive results on the visual tracking benchmark [3], as well as, a high rank in the 2014 visual object tracking (VOT) competition [44]. Moreover, KCF has very attractive computational properties, since it can easily reach real-time frame rates. Much like other detection based trackers, KCF can be trained using a set of training templates. At the first frame, only one template is available. Some methods generate more templates by sampling patches around the first patch to collect more training data. On the oth-

er hand, KCF exhaustively samples the whole region around the target by taking advantage of cyclic shifts, which simulate translations of the target object.

Assuming for simplicity all examples are 1D and $\mathbf{x}$ represents the base template at the first frame, then $\mathbf{Px} = [x_n, x_1, ..., x_{n-1}]^T$ represents one circular shift of that base patch, where $\mathbf{P}$ is a permutation matrix. Then, the set representing all possible circular shifts is given by $\{\mathbf{P}^i\mathbf{x}|i = 0, ..., n-1\}$. Obviously in case of 2D signals, there will be two possible shifts one in either direction. The matrix containing all possible circular shifts is a circulant matrix $\mathbf{X}$, which is also known as the data matrix.

The goal of KCF training is to learn the filter taps $\mathbf{w}$ that minimize the error between the filtered circular shifts of the training template $\mathbf{x}$ and the regression targets $\mathbf{y}$, which take on Gaussian values the highest of which is at the middle of the patch (i.e. when no circular shift exists).

$$\min_{\mathbf{w}} \sum_{i}^{n} (f(\mathbf{w}; \mathbf{P}^i\mathbf{x}) - y_i)^2 + \lambda ||\mathbf{w}||_2^2 \tag{3.1}$$

The representation function is given as $f(\mathbf{w}) = \mathbf{w}^T\phi(\mathbf{x})$, where $\phi(\mathbf{x})$ denotes a mapping of the template $\mathbf{x}$ into another space that can possibly be of a higher dimension. Therefore, Eq (3.1) can be re-written as,

$$\min_{\mathbf{w}} ||\Phi\mathbf{w} - \mathbf{y}||_2^2 + \lambda ||\mathbf{w}||_2^2 \tag{3.2}$$

where Φ contains the mapping of all the circular shifts of template $\mathbf{x}$. The most efficient solution to Eq (3.2) highly depends on the nature of the mapping function $\phi(.)$.

ϕ is linear. In this case, $\phi(\mathbf{x}) = \mathbf{x}$, $\Phi = \mathbf{X}$, and the solution is known to be: $\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$. In fact, Eq (3.2) can be solved directly and efficiently in terms of the filter taps $\mathbf{w}$ (primal formulation) by exploiting the circulant matrix

structure of $\mathbf{X}$ (i.e. it can be diagonalized using the DFT matrix), as follows:

$$\hat{\mathbf{w}}^* = \frac{\hat{\mathbf{x}}^* \odot \hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda} \tag{3.3}$$

where $\hat{\mathbf{z}}$ and $\hat{\mathbf{z}}^*$ denote the FFT of signal $\mathbf{z}$ and its complex conjugate, respectively. Interestingly, when *multiple* templates are available in training, it can easily be shown that the optimal filter taps can be computed in a similar way [46]:

$$\hat{\mathbf{w}}^* = \frac{\sum_{j=1}^{m} \hat{\mathbf{x}}_j^* \odot \hat{\mathbf{y}}}{\sum_{j=1}^{m} \hat{\mathbf{x}}_j^* \odot \hat{\mathbf{x}}_j + \lambda} \tag{3.4}$$

ϕ is non-linear. In this case, the single template $\mathbf{x}$ can be chosen to have multiple features (e.g. HoG), but the solution to Eq (3.2) cannot be computed efficiently in the primal form. However, using a kernel decomposition of the filter taps to transform the problem into its *dual* form, which is also a ridge regression problem. The dual variable solution α for this kernelized version, $\mathbf{w} = \Phi^T \alpha$, in the dual form can be shown to be: $\alpha = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}$, where $\mathbf{K} = \Phi_{\mathbf{x}} \Phi_{\mathbf{x}}^T$ which is known as the kernel matrix. This kernel matrix evaluates the kernel (e.g. Gaussian) on all pairs of the circular shifts of the template $\mathbf{x}$, thus, these shifts can contain multiple features, as opposed to the case when ϕ is linear. As shown in [20], if the kernel used is permutation invariant, matrix $\mathbf{K}$ is also circulant and thus we can exploit its DFT diagonalization property to compute the dual solution α (in the frequency domain) efficiently, as follows:

$$\hat{\alpha}^* = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{xx} + \lambda} \tag{3.5}$$

where $\hat{\mathbf{k}}^{xx}$ is defined to be the FFT of the kernel correlation, and $\hat{\mathbf{w}}$, $\hat{\mathbf{y}}$ denotes the

FFT of the filter taps and the target patches respectively.

Target Detection. After the optimal taps are learned for the template $\mathbf{x}$, some variants of KCF trackers update these taps as a convex combination of these taps and those from a previous frame [20, 22, 45]. In this way, historical information can be propagated, despite the fact that it is done heuristically and non-adaptively. Detecting the target in the next frame can be simply done by applying the filter $\mathbf{w}$ to some search regions in the frame. In other words, we apply $f(\mathbf{z}; \mathbf{w}) = \mathbf{w}^T \phi(\mathbf{z})$ where $\mathbf{z}$ is the sample to be evaluated over, and $\Phi_{\tilde{\mathbf{x}}}$ is the latest model that has been updated. If the dual form of the solution is computed, then the detection formula is: $f(\mathbf{z}; \alpha) = \alpha^T \Phi_{\tilde{\mathbf{x}}} \phi(\mathbf{z})$. To measure the response over all circular shifts (i.e. translations) of the signal $\mathbf{z}$, the mapping of these circular shifts is computed in the same way it is done in the training, namely using a circulant matrix Φ_z. In this case, all the filter responses (in the frequency domain) can be computed efficiently: $\hat{\mathbf{f}}(\mathbf{z}; \alpha) = (\hat{k}^{\tilde{\mathbf{x}}\mathbf{z}} \odot \hat{\alpha})$, where $\hat{k}^{\tilde{\mathbf{x}}\mathbf{z}}$ is the FFT of $\Phi_{\tilde{\mathbf{x}}} \Phi_z^T$. It is worthwhile to note that it was stated in [20] that solving the ridge regression problem in Eq (3.2) for multi-dimensional feature vectors with non-linear kernels, as well as, multiple templates is not possible. We will address this issue next.

3.4 Multiple Template Scale Adaptive KCF

The following subsections will provide a detailed derivation of our proposed tracker.

3.4.1 Multiple Templates

Although the primal formulation of the rigid regression problem allows for training with multiple templates (i.e. more than one circulant matrix $\mathbf{X}$) [46], it does not directly allow the use of multi-dimensional features and non-linear kernels. Experimental results in Table 3.1 demonstrate a significant performance improvement when

using multiple templates in training as compared to using only one, while utilizing the same features. In the seminal KCF work [20], the authors argue that multiple templates are feasible only for linear kernels in the primal formulation. On the other hand, the dual form allows for non-linear, multidimensional features, but using only one template. They argue that using both is not possible. In this paper, we show how filter taps can be computed using multiple templates with multi-dimensional features and non-linear kernels in the dual formulation.

We formulate the problem as follows. Ideally, at frame $n+1$, there are n samples, on which the filter $\mathbf{w}$ should be trained. In this case, the data matrix containing the templates and all their shifted versions is no longer circulant, but rather blockwise circulant. In fact, the augmented data matrix $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_n]^T$, where each $\mathbf{X}_i$ is a circulant matrix generated from the i^{th} template (i.e. the detected patch at frame i). Therefore, the multiple template kernelized correlation problem can be formulated as:

$$\min_{\mathbf{w}} \left\| \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_n \end{pmatrix} \mathbf{w} - \begin{pmatrix} \mathbf{y} \\ \mathbf{y} \\ \vdots \\ \mathbf{y} \end{pmatrix} \right\|_2^2 + \lambda ||\mathbf{w}||_2^2 \tag{3.6}$$

Note that Φ_i is the mapping matrix evaluated for template $\mathbf{x}_i$ and its circulant shifts. Since $\mathbf{X}_i$ is circulant, Φ_i is also circulant. In what follows, we will provide details on how this optimization problem can be solved iteratively using the inherent properties of block circulant matrices. Without loss of generality, our derivation will be highlighted for 1-D signals with two training examples (i.e. $n = 2$), but the results can be easily extended to 2-D signals and to multiple training examples. The resulting problem can then be written as:

$$\min_{\mathbf{w}} \quad \left\| \begin{pmatrix} \Phi_1 \\ \Phi_2 \end{pmatrix} \mathbf{w} - \begin{pmatrix} \mathbf{y} \\ \mathbf{y} \end{pmatrix} \right\|_2^2 + \lambda ||\mathbf{w}||_2^2 \tag{3.7}$$

By adding auxiliary variables, Eq (3.7) can be re-written by considering two independent filters under the constraint that these filters are the same, as follows:

$$\begin{aligned} \min_{\mathbf{w}_1, \mathbf{w}_2} \quad & \left\| \begin{pmatrix} \Phi_1 \mathbf{w}_1 - \mathbf{y} \\ \Phi_2 \mathbf{w}_2 - \mathbf{y} \end{pmatrix} \right\|_2^2 + \lambda \left\| \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{pmatrix} \right\|_2^2 \\ \text{subject to:} \quad & \mathbf{w}_1 = \mathbf{w}_2. \end{aligned} \tag{3.8}$$

This problem can be solved by replacing the hard constraint with a soft one in the objective: $\frac{\mu}{2}||\mathbf{w}_1 - \mathbf{w}_2||_2^2$. This additional regularizer encourages that both filters be the same. The value of μ is increased in each iteration. In each iteration, we solve for both $\mathbf{w}_1$ and $\mathbf{w}_2$ via alternating fixed-point optimization. The method starts by initializing a solution for $\mathbf{w}_2$ and uses that to update $\mathbf{w}_1$. Then, we use the updated $\mathbf{w}_1$ to solve for $\mathbf{w}_2$, so on and so forth, until a stopping criterion has been met. Since the original problem is convex, this strategy is guaranteed to converge to a global minimum. We initialize the filters with the solution to the single-template KCF. Therefore, in the j^{th} iteration, we solve the following two coupled problems:

$$\begin{aligned} &\min_{\mathbf{w}_1} \quad ||\Phi_1 \mathbf{w}_1 - \mathbf{y}||_2^2 + \lambda ||\mathbf{w}_1||_2^2 + \mu ||\mathbf{w}_1 - \mathbf{w}_2^j||_2^2 \\ &\min_{\mathbf{w}_2} ||\Phi_2 \mathbf{w}_2 - \mathbf{y}||_2^2 + \lambda ||\mathbf{w}_2||_2^2 + \mu ||\mathbf{w}_2 - \mathbf{w}_1^{j+1}||_2^2 \end{aligned} \tag{3.9}$$

Solving Eq (3.9) Note that the problems above have the exact same form, so we will only derive the solution to one of them. To solve Eq (3.9), we formulate its dual in Eq (3.10) to allow for non-linear kernel functions and multidimensional features.

$$\min_{\mathbf{w}_1} \sum_i ({\mathbf{w}_1}^T \phi(\mathbf{x_i}) - y_i)^2 + \lambda ||\mathbf{w}_1||_2^2 + \mu ||\mathbf{w}_1 - \mathbf{b}||_2^2 \tag{3.10}$$

Here, we take $\mathbf{b} = \mathbf{w}_2^j$ for readability. Following the same strategy as in [20], we set the gradient of Eq (3.10) to zero and identify the left hand side to the right. By doing so, we obtain: $\mathbf{w}_1 = \Phi_1^T \mathbf{a}_1 + k\mathbf{b}$, where $a_1^i = -\frac{1}{\lambda+\mu}({\mathbf{w}_1}^T \phi(\mathbf{x_i}) - y_i)$ and $k = \frac{\mu}{\lambda+\mu}$. By substituting the dual formulation of $\mathbf{w}_1$ into Eq (3.10) and setting its gradient to zero, we obtain the following linear system:

$$(\Phi_1 \Phi_1^T + (\lambda+\mu)\mathbf{I})\mathbf{a}_1 = \mathbf{y} - \left[k\mathbf{I} + \Big(\lambda k + \mu(k-1)\Big)\left(\Phi_1 \Phi_1^T\right)^{-1}\right]\tilde{\mathbf{b}}$$

where $\tilde{\mathbf{b}} = \Phi_1 \mathbf{b} = \Phi_1 \Phi_2^T \mathbf{a}_2$. Now Eq (3.11) looks very similar to the solution derived using a single training sample in the original KCF formulation but with an extra additive term. A similar approach can be used to solve this linear system efficiently. Using the FFT diagonalization of both $\Phi_1 \Phi_1^T$ and $\Phi_1 \Phi_2^T$ (both are circulant matrices), we can invert the left hand side and compute the right hand side efficiently, as follows:

$$\tilde{\mathbf{b}} = \mathbf{F} diag(\hat{k}^{x_2 x_1}) \mathbf{F}^H \mathbf{a}_2 = \mathbf{F}(\hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \tag{3.11}$$

where $\hat{\mathbf{k}}^{x_2 x_1}$ is the FFT of the correlation kernel vector between the two signals $\mathbf{x}_1$ and $\mathbf{x}_2$. Similarly,

$$(\Phi_1 \Phi_1^T)^{-1} \tilde{\mathbf{b}} = \mathbf{F}((\hat{k}^{x_1 x_1})^{-1} \odot \hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \tag{3.12}$$

This result extends easily to 2-D signals and to multiple features, as in [20]. Therefore, the final solution would be very similar to the solution given in [20] with a modified target vector. In this way, we avoid using heuristics to combine the effect of

the current template with historical information as done in previous work [20, 22, 45].

$$\hat{\mathbf{a}}_1 = \frac{\hat{\psi}}{\hat{\mathbf{k}}^{x_1 x_1} + (\lambda + \mu)} \tag{3.13}$$

where $\hat{\psi} = \hat{\mathbf{y}} - \left[k + \Big(\lambda k + \mu(k-1)\Big)\Big(\frac{1}{\hat{k}^{x_1 x_1}}\Big)\right] \odot \tilde{\mathbf{b}}$.

As stated earlier, we update $\mathbf{w}_1$ and $\mathbf{w}_2$ (equivalently $\hat{\mathbf{a}}_1$ and $\hat{\mathbf{a}}_2$) by increasing the tradeoff coefficient μ in each iteration. Upon convergence, we obtain the solution to the original problem in Eq (3.7). The stopping criterion we use in our experiments depends on the variation in the overall objective cost in previous iterations. Once that variation is small enough, the optimization process is terminated. Here, we note that evaluating the cost function efficiently is non-trivial; however, we can use diagonalization ideas similar to those seen earlier to compute it very quickly.

3.4.2 Scale Integration

One of the main drawbacks of KCF is that it does not address the target scale issue. We use a simple yet effective method (similar in spirit to [23, 45]) to counteract this issue. Specifically, we apply max-pooling over multiple scales in the detection phase of the tracker. The main difference between our approach and previous work is that we maximize over the posterior probability instead of the likelihood, as illustrated in the following equation.

$$\max_i \quad P(s_i|\mathbf{y}) = P(\mathbf{y}|s_i)P(s_i) \tag{3.14}$$

where s_i represents the i^{th} scale and $P(\mathbf{y}|s_i)$ is the likelihood that is defined by the maximum detection response at the i^{th} scale: $\hat{\mathbf{f}}(\mathbf{z}) = (\hat{k}^{\hat{\mathbf{x}}\mathbf{z}} \odot \hat{\alpha})$. The prior term $P(s_i)$ is assumed to follow a Gaussian distribution, which is centered around the previous scale and has a standard deviation σ which is set experimentally. This will allow for smooth transition between scales, since the target's scale is assumed to not change

much between consecutive frames. This strategy produces more stable detections than previous methods [23, 45] that only use the likelihood.

3.5 Experimental Results

We conduct two experiments to evaluate the efficiency and accuracy of our proposed tracker. First, we compare our tracker against state-of-art trackers that participated in the VOT2014 challenge [44], which comprises 25 challenging sequences. Secondly, we evaluate our tracker using the VOT2015 toolkit [47] on a set of 60 challenging videos. We show how each of the proposed modifications to the original KCF tracker (i.e. the use of multiple templates in training and scale adaptation in detection) leads to improvement in overall performance.

3.5.1 Features and Parameters

In the implementation, we used HoG features with a Gaussian kernel function. The σ used in the Gaussian kernel is set to 0.5, as in [20]. The HoG cell size is 4x4 and the number of orientation bins is 9. The extracted feature vector is multiplied by a Hanning window, as described in [22]. The regularization parameter over the energy term is set experimentally to $\lambda = 10^{-4}$. Also, we set $\mu = 10^{-5}$ as an initial value and it is doubled every iteration. The search grid of scales is set to be $[0.76, 0.80...1.20, 1.24]$ of the original size of the target. The stopping criterion used in the optimization of Section 3.4.1 is based on the standard deviation of the overall cost in the past 5 iterations. The stopping threshold for this standard deviation is set to $\eta = 10^{-5}$.

3.5.2 Experimental Setup

In all our experiments, we run our proposed approach in MATLAB on an Intel(R) Xeon(R) 2.67GHz CPU with 32GB RAM.

Datasets: We conduct the experiments on 2 different datasets, namely VOT2014[44] and VOT2015[47]. VOT2014 has 25 annotated videos in total, while VOT2015 comprises 60 videos. Both datasets pose challenging problems to object tracking, including partial occlusion, illumination change, motion blur, and background clutter. Both datasets have annotations to account for non-standard rectangles that can be rotated or scaled.

Evaluation Methodology: We use the VOT2015 toolkit to evaluate the results for both VOT2014 and VOT2015. The toolkit reports three measures of evaluation: accuracy, robustness, and speed, along with the overall tracking score. Our proposed tracker is deterministic, so it is evaluated by the toolkit three consecutive times and the average score over the three turns is recorded. As for accuracy, we use the same criterion used in the Pascal VOC, namely the overlap ration (VOR) [48], which is defined as:

$$\text{VOR} = \frac{area(ROI_{T_i} \cap ROI_{G_i})}{area(ROI_{T_i} \cup ROI_{G_i})} \tag{3.15}$$

where T and G denote the target and ground truth bounding boxes respectively. As for robustness, it measures the number of times the tracker loses the object. Therefore, a robustness score of 0 means the tracker does not lose the object at all, during the video throughout the 3 runs. A failure is defined when the overlap measure is below some threshold for a certain number of consecutive frames. The toolkit automatically re-initializes the tracker when a failure happens allowing the tracker to resume from the re-initialized frame. Obviously, a smaller robustness score is desired. The toolkit also reports the speed of the tracker in processing each frame.

3.5.3 Results

VOT2014 Experiments: We compare our method with some of the available results of the state-of-art trackers on the VOT2014 dataset. The trackers we used

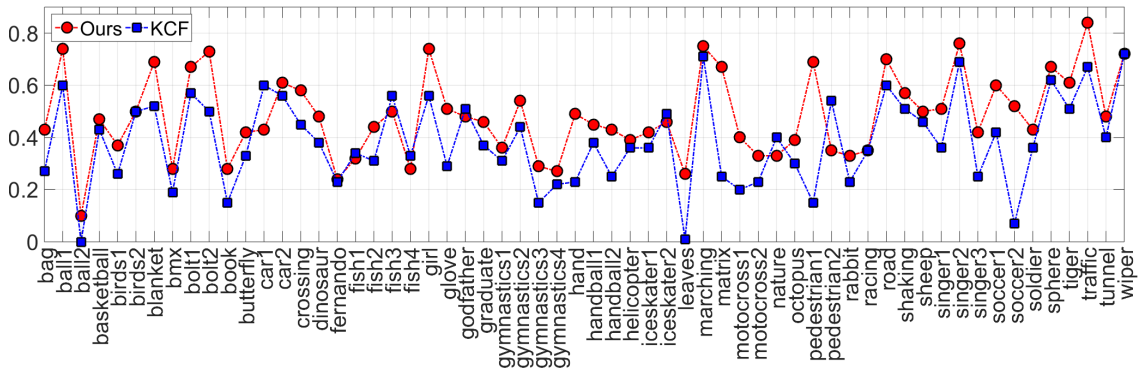

Figure 3.2: Accuracy results on VOT2015 dataset for 60 videos, comparing our proposed method and KCF.

for comparisons were NCC (baseline), Struck [49], MIL[17], IVT[8], KCF [20], and KCF_Scale trackers. In the VOT2014 challenge, a scaled version of KCF was introduced which we denote as KCF_Scale. Since the parameter setup for this tracker is not available and to ensure a fair comparison with our method, we run KCF_Scale with the same methodology used in VOT2014 but with the same parameters we use in our tracker. It is clear from Table 3.2 that our method (incorporating KCF with multiple templates and scale adaptation) substantially surpasses all the other trackers in both accuracy and robustness.

	Acc. Rank	Rob. Rank	Rank	FPS
Ours	2.68	2.96	2.82	25.1
KCF_Scale	3.00	3.48	3.24	47.42
KCF	3.46	3.13	3.29	66.5
Struck	3.68	3.85	3.75	19.77
MIL	5.02	3.88	4.45	1.94
IVT	5.00	4.65	4.81	27.51
NCC	5.21	6.08	5.65	27.9

Table 3.2: Comparison on the VOT2014 dataset.

VOT2015 Experiments: Since the tracking results of other trackers has not been released on VOT2015 just yet, we use this dataset to demonstrate how each of our proposed components (multiple templates and scale adaptation) contributes to the

overall improvement of our approach over the original KCF tracker. To do this, we compare the KCF tracker with: **(KCF+MT)** which is our multiple template version of the KCF (refer to Section 3.4.1) and **(KCF+MT+Sc)** which is the latter tracker but with scale adaptation (refer to Section 3.4.2). In Table 3.3, we compare the performance and runtime of these three trackers. Clearly, adding any of the two components to the original KCF tracker improves its performance. By adding both, the improvement in performance is even more significant. It may be noted that the improvement by adding the multiple template update may not be as significant as for adding the scale for this dataset. This is primarily due to the fact that this dataset in particular includes examples of a target object undergoing substantial variations in scale. The effect of adding the MT component is more evident in the VOT2014 dataset, where less scale variations are encountered. For a detailed per-video comparison, we show the accuracy of our method and KCF on all 60 VOT2015 videos in Figure 3.2. It is clear from the plot that our tracker outperforms KCF in the majority of videos.

	Acc. Rank	Rob. Rank	Rank	FPS
KCF+MT+Sc	1.80	1.95	1.88	24.3
KCF+MT	2.04	2.04	2.04	31.54
KCF	2.16	2.01	2.08	44.76

Table 3.3: Comparisons on the VOT2015 dataset.

3.6 Conclusion

We have identified the main drawbacks of the KCF tracker that cause failure and we propose two components to address these drawbacks. First, we show that it is possible to incorporate multiple multi-dimensional templates in computing the optimal filter taps. By reformulating the kernel correlation problem and by using fixed-point optimization, we demonstrate that using multiple templates improves the performance of

the original KCF tracker that uses only one template and a heuristic update scheme. Second, we address the problem of fixed scale tracking. We use a similar grid search approach as in previous methods, but we week the scale with maximum posterior instead of likelihood. This subtle difference makes the tracker more robust to gradual scale changes. Our experimental results on VOT2014 and VOT2015 show that our tracker substantially outperforms many state-of-the-art trackers.

Chapter 4

Target Response Adaptation for Correlation Filter Tracking

4.1 Introduction

CF based trackers [20–22,25,50] have gained much attention lately for their attractive performance both in speed and accuracy. The key idea behind CF trackers is that a learned filter is used to localize the object in the next frame by identifying the location of maximal correlation/convlution response (detection step). Then, it is updated by computing a filter, whose correlation with training templates (most often the current tracking result) closely resembles a hand-crafted target response, usually taken to be a Gaussian centered at the current tracking result (training step) [20, 21, 25, 50–52]. A recent development in this tracking paradigm and the main reason behind its computational efficiency is the use of circulant structure in the training step. In many cases (e.g. when the background is homogenous and no occlusion occurs), the circular shifts of the training templates actually represent translations in the image domain. This means that the *motion* of a template is inherently accounted for by these circular shifts.

Despite its merits, there are two main drawbacks in the traditional CF tracking paradigm. **(i)** Since the detection step of the tracker might be inaccurate (e.g. due

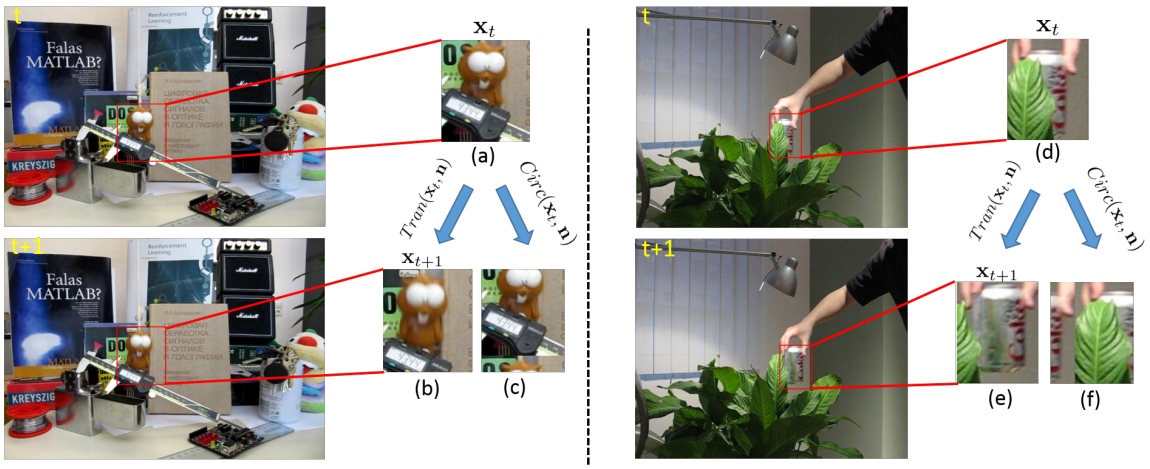

Figure 4.1: Shows examples where circular shifts do not represent actual translations. Patches (a) and (b) of video *Lemming* show the object in two consecutive frames, where the target was partially occluded and the occluder is within the filter window. The circular shift corresponding to the actual translation of the object in the next frame is given in patch (c). Note that both the occluder and target are shifted. $\text{Circ}(\mathbf{x}, \mathbf{n})$, and $\text{Tran}(\mathbf{x}, \mathbf{n})$ denote an $\mathbf{n}$ circular shift and actual translation applied to the patch $\mathbf{x}$, respectively. Similarly, we show patches (d) and (e) of video *Coke*, where fast motion and partial occlusion occur. In both examples, translations and their approximations (circular shifts) are quite different. This discrepancy will severely affect the detection step (and in turn the training step) of any CF based tracker at that frame.

to fast motion, motion blur, etc.), the localization of the object in the next frame is erroneous. Moreover, since the target response is independent of the frame, error will be propagated to the newly computed filter and the tracker becomes at risk of unrecoverable drift. **(ii)** The target response used in the training step is independent of the observed frame and assumes that circular shifts correspond to actual translations, which is not the case in some scenarios (refer to Figure 4.1). Obviously, this approximation is not reliable for many tracking nuisances including fast motion, occlusion, and motion blur. Since the target response is not adaptive to the observed frame, the tracker cannot easily recover from errors in the detection step. In this paper, we propose to tackle both drawbacks by jointly solving for the best filter and target response in each frame, where the latter is regularized using actual translation measurements of correlation and not approximated using circular shifts.

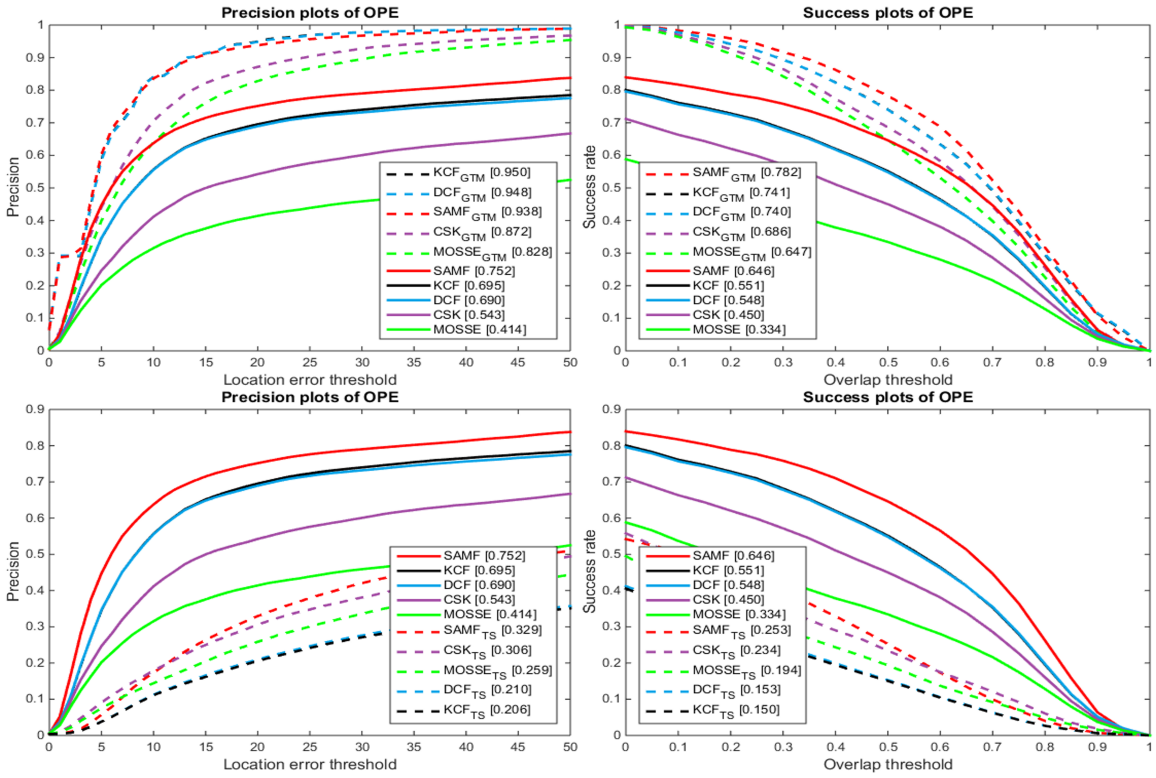

Figure 4.2: The first row demonstrates the impressive performance of five CF based trackers (MOSSE_{GTT}, DCF_{GTT}, CSK_{GTT}, KCF_{GTT}, and SAMF_{GTT}) when the target response is obtained from the ground truth of OTB100 [6]. The second row shows the sensitivity of the tracking results to the target response. By only perturbing the target response by at most 2 pixels, the performance drops significantly. This motivates the importance of designing more robust and effective target response.

As mentioned earlier, the selection of the target response in CF tracking is intimately related to the assumed motion model. Traditionally, this model is simplistic, strict, and prone to drift. Therefore, using/designing a more effective motion model (or equivalently, designing a better target response) is crucial. We justify this observation in Figure 4.2 by changing the traditional target response in two different ways and reporting the precision and accuracy results of several CF based trackers in both cases. Note that the traditional detection step is not altered in either case.

In the first experiment (top row of Figure 4.2), the target response (motion model) is *optimal* because it is generated by centering the traditional Gaussian target at the ground truth object location in each frame, irrespective of the detection location.

As compared to using the traditional response, all CF trackers perform significantly better (in both metrics), especially those that use simple grayscale features, which usually make it difficult to reliably detect the object in the next frame. For example, the precision of the basic MOSSE tracker [22] increases from 14% to 82%. Note that perfect performance is not achieved here (even for trackers with scale adaptation) because the detection step still introduces errors. This experiment suggests that it is useful to design more realistic target responses, instead of only focusing on incorporating more complex features that tend only to impact the detection step in CF tracking. In the second experiment (bottom row of Figure 4.2), perturbations in the detection step are simulated by randomly shifting the traditional Gaussian target response around the detected location by at most 2 pixels in each frame. Clearly, CF tracking performance drops significantly for all methods because it could not recover from the drift. We also conclude from this experiment that designing a better target response is important for more robust CF tracking. This design should be able to handle errors/perturbations in the detection step, thus, allowing the tracker to recover from drift.

4.2 Related Work

There have been many advances in the field of object tracking, so we only provide a brief overview of the two main categories of trackers (generative and discriminative) in the literature and focus on those that are most relevant to our proposed framework (CF trackers).

Generative Trackers. They adopt an appearance model to describe a set of target observations. The aim of these trackers is to search for the target that is best represented by the updated generative model. Therefore, learning a representative appearance model that can identify the target, even when it undergoes appearance

changes is the main emphasis of these trackers. Examples of generative trackers include incremental tracker (IVT) [8], mean shift tracker [7], L1-min tracker [10], multi-task tracker (MTT) [12], low-rank sparse tracker [13], and structural sparse tracker [16] to name a few.

Discriminative Trackers. They formulate object tracking as a classification problem, where the regions around the previous location of the target are given scores to regress to (e.g. using a classifier to predict background or foreground). Examples of discriminative trackers include multiple instance learning (MIL) [17], ensemble tracking [18], support vector tracking [19], and correlation filter (CF) based trackers like [20–22, 45].

CF Trackers. Using correlation filters for tracking started with Bolme *et al.* [22], where the formulation was constructed in the frequency domain for efficiency, thus, reaching a runtime of 600-700 FPS. Seminal followup work by Henriques *et al.* [20, 21] formulate the problem in the spatial domain, but solved it efficiently in the frequency domain. This is possible by exploiting circulant structure in the optimization. This method (denoted as the kernel correlation filter tracker or KCF [20]) can incorporate both non-linear kernels (e.g. Gaussian) and multi dimensional features (e.g. HOG). Many improvements have recently been made to this popular tracker to address several limiting issues. For instance, the work in [23, 45] proposes an adaptive scale version of KCF and also made use of the colornames feature. Another approach proposes a multi-template version of KCF [53] by solving a constrained ridge regression problem. Recent work extends KCF to enable part-based tracking [24], where multiple KCF trackers (one for each part) are run independently and the response map of all trackers is computed.

Another variant of CF based trackers changes the objective to spatially focus the filter energy at the center, thus, reducing undesirable boundary effects [50]. Unlike other CF based trackers, their final formulation cannot fully exploit circulant matrix

structure into having a closed form element wise solution. Similarly, Galoogahi *et al.* [25] propose a method to deal with the boundary effects of circularly shifted patches by pre-multiplying them with a masking matrix; however, the resulting optimization is also unable to exploit circular structure.

Contributions. To the best of our knowledge, we are the first to investigate the effect of designing an adaptive target response in the context of CF tracking.**(2)** Unlike previous work that fixes the target response for all frames, our proposed method adaptively changes it to account for motion and boundary effects caused by circular shifts. The resulting joint optimization can be solved efficiently by exploiting the underlying circulant structure. **(3)** Extensive experiments on the popular online tracking benchmark OTB100 [6] show that our adaptive target framework can be applied to many CF trackers to improve their performance, while remaining computationally efficient.

4.3 Correlation Trackers

Similar to other discriminative methods, correlation filters need a set of training examples to learn a filter. In tracking, the first image patch is the only available training example. Other discriminative trackers usually collect positive examples from image patches close to the object in the first frame and negative ones from patches that are farther away. Computational complexity can increase significantly as the number of training patches increases. However, CF based trackers collect dense samples by circularly shifting the patch in the first frame (thus approximating translations) to construct a circulant matrix, which has very desirable properties.

Assuming for simplicity that all the training examples are 1D where $\mathbf{x} \in \mathbb{R}^n$ represents the template in the first frame. Then, $\mathbf{P}\mathbf{x} = [x_n, x_1, \ldots, x_{n-1}]$ represents 1 circular shift of $\mathbf{x}$, where $\mathbf{P}$ is a permutation matrix. Concatenating the set of all possible circularly shifted templates forms a circulant matrix that is also referred to

as the data matrix. Correlation filtering seeks a filter $\mathbf{w}$ that minimizes the following ridge regression problem:

$$\underset{\mathbf{w}}{\text{minimize}} \;\; ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \lambda||\mathbf{w}||_2^2 \tag{4.1}$$

The data matrix $\mathbf{X} \in \mathbb{R}^{n \times n}$ can either contain the template $\mathbf{x}$ and all its shifts or a kernelized version of them when using circulant structure preserving kernels [20]. Here, $\mathbf{y}$ is the target response, which is usually assumed to be Gaussian centered around the base patch. Eq 4.1 can be solved in either the primal or the dual domain each with its own pros and cons.

Primal Domain. Here, the filter $\mathbf{w}$ (the primal variable) is computed to solve Eq 4.1, which admits a closed form solution in the primal domain given by $\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$. Due to the circulant structure of $\mathbf{X}$, it can be diagonalized and the matrix inversion can be done efficiently. The filter solution (in FFT form) is given by $\hat{\mathbf{w}} = \frac{\hat{\mathbf{x}}^* \odot \hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda}$, where $\hat{\mathbf{w}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{x}}$ are the FFTs of $\mathbf{w}$, $\mathbf{y}$, and $\mathbf{x}$, respectively. The $*$ denotes the complex conjugate and all operations are element wise [20]. The primal formulation also enables the use of multiple templates without changing the solution mechanism much. In this case, Eq 4.1 is solved with $\mathbf{X}$ replaced with $\tilde{\mathbf{X}}$, which is the blockwise concatenation of the circulant matrices of all the templates. It can be easily shown [20] that the optimal filter for k templates is given by $\hat{\mathbf{w}} = \frac{\sum_{j=1}^{k} \hat{\mathbf{x}}_j^* \odot \hat{\mathbf{y}}}{\sum_{j=1}^{k} \hat{\mathbf{x}}_j^* \odot \hat{\mathbf{x}}_j + \lambda}$. However, this formulation does not facilitate the use of kernels because the solution is not written as a function of bi-products of the circulant matrices.

Dual Domain. Conversely, Eq 4.1 can also be solved in the dual domain, where the solution is $\alpha = (\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I})^{-1}\mathbf{y}$. Here, α is the dual variable and it is related to the primal variable through $\mathbf{w} = \mathbf{X}^T\alpha$. The dual formulation admits the solution as a function of bi-products of the circulant data matrix allowing the use of the

kernel trick. The dual solution (in FFT form) is $\hat{\alpha} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda}$, where $\hat{\alpha}$ is the FFT of α. Unfortunately, using k templates in the dual domain can no longer be done efficiently because $\mathbf{X}\mathbf{X}^T$ is now a matrix with circulant blocks, which has an inversion computational complexity of $n^2O(k^3)$ compared to $kO(nlog(n))$ in the primal domain.

4.4 Learning Adaptive Target Responses for CF Tracking

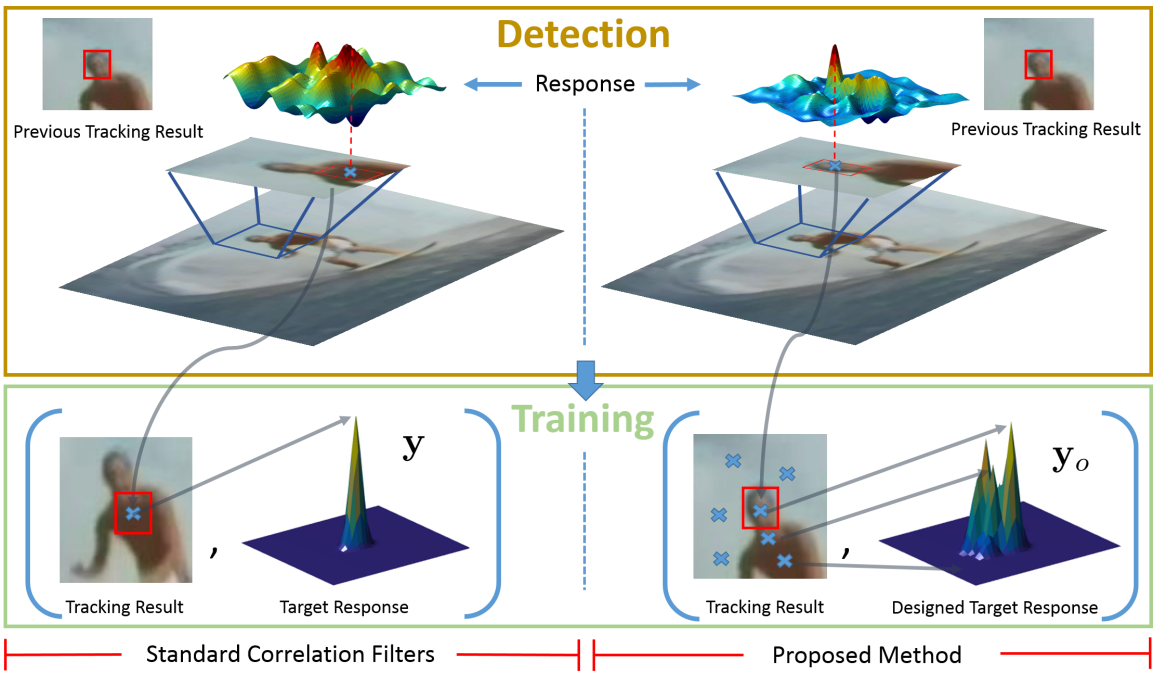

Figure 4.3: Shows the pipeline of both standard CF based trackers and our approach. Both involve a detection and training step with a key difference that our approach uses the current detection frame to sample actual translations, which will be used to construct a prior for the target response. In fact, standard CF tracking is a special case of our formulation. When only one translation is sampled around the current tracking result, our approach reduces to the standard CF model.

As seen in Figure 4.2, exploiting a reliable motion model in CF based tracking (i.e. designing a better target response $\mathbf{y}$) can significantly boost performance. In this paper, we do this by adaptively changing $\mathbf{y}$ in every frame. The peak values of

$\mathbf{y}$ not only favor a training template over another based on appearance information, but also based on prior motion information as well. To do this, we solve the following joint optimization:

$$\underset{\mathbf{w},\mathbf{y}}{\text{minimize}} \;\; ||\tilde{\mathbf{X}}\mathbf{w} - \mathbf{y}||_2^2 + \lambda_1||\mathbf{w}||_2^2 + \lambda_2||\mathbf{y} - \mathbf{y}_o||_2^2 \tag{4.2}$$

As compared to the classical CF formulation in Eq 4.1, ours does not assume that the target response $\mathbf{y}$ is known apriori, but instead its constructed prior $\mathbf{y}_o$ is known. In what follows, we discuss how $\mathbf{y}_o$ is computed, how Eq 4.2 is solved for k templates with $\tilde{\mathbf{X}} \in \mathbb{R}^{kn \times n}$ being a block circulant matrix, and how the single template solution follows directly. We will also provide the new detection equation for the objective in Eq 4.2, as well as, an exposition of how our method differs from all other CF based trackers (refer to Figure 4.3). All the derivations are done for a 1D example, but they can be easily extended to the 2D. More details can be found in the **Appendix**.

Construction of $\mathbf{y}_o$. In Eq 4.2, the target $\mathbf{y}$ is assumed to follow the noise model: $\mathbf{y} = \mathbf{y}_o + \mathbf{n}$, where $\mathbf{y} \in \mathbb{R}^n$ and $\mathbf{y} \sim \mathcal{N}(\mathbf{y}_o, \text{diag}^{-1}(\frac{1}{2\lambda_2}))$. At the first frame, a standard KCF [20] filter is learnt by solving Eq 4.1. In the next frame, a fixed number p translations are sampled, at which the previous filter is correlated with the image. These correlations are used to fill the corresponding p entries in $\mathbf{y}_o$. As motivated earlier, this process generates correlation scores for actual translations, which can offset the limiting effects of using their approximations (i.e. circular shifts). We choose $p \ll n$, so the computational burden remains reasonable for our tracking scenario. The rest of the entries in $\mathbf{y}_o$ are computed from the p computed values by Gaussian interpolation. We did experiment with more sophisticated types of interpolation (e.g. bilateral filtering using the image patch as a guide). This resulted in unnoticeable change in performance and an increase in computation cost.

In subsequent frames and to encode motion information, the aforementioned translations are sampled using a standard Kalman filter with a constant velocity motion

model. Other motion models can be used here, such as particle filters.

Multiple Template Solution to Eq 4.2. Here, $\tilde{\mathbf{X}}^\top = \begin{bmatrix}\mathbf{X}_1^\top & \mathbf{X}_2^\top & \dots & \mathbf{X}_k^\top\end{bmatrix} \in \mathbb{R}^{kn\times n}$, which is the concatenation of all the circulant matrices generated from all the templates. By introducing the variable $\mathbf{z}^\top = \begin{bmatrix}\mathbf{w}^\top & \mathbf{y}^\top\end{bmatrix}$, Eq 4.2 (in its primal form) can be written as:

$$\underset{\mathbf{z}}{\text{minimize}} \;\; \|\tilde{\mathbf{G}}\mathbf{z}\|_2^2 + \lambda_1\|\mathbf{E}\mathbf{z}\|_2^2 + \lambda_2\|\mathbf{D}\mathbf{z} - \mathbf{y}_o\|_2^2, \tag{4.3}$$

where $\tilde{\mathbf{G}} = \begin{bmatrix}\tilde{\mathbf{X}} & -\tilde{\mathbf{I}}\end{bmatrix} \in \mathbb{R}^{kn\times 2n}$, $\tilde{\mathbf{I}}^\top = [\mathbf{I} \;\cdots\; \mathbf{I}]$, $\mathbf{E} = \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \in \mathbb{R}^{n\times 2n}$, and $\mathbf{D} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \in \mathbb{R}^{n\times 2n}$. The problem is convex quadratic, so a global solution can be easily derived (refer to the **Appendix** for details of this derivation). In its dual form, Eq 4.3 becomes:

$$\underset{\alpha}{\text{minimize}} \; \|\mathbf{D}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha - \mathbf{y}_o\|_2^2 + \lambda_1\|\mathbf{E}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha\|_2^2 + \|\mathbf{G}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha\|_2^2, \tag{4.4}$$

where α is the dual variable and is related to $\mathbf{z}$ through $\mathbf{z} = \tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha$ and $\tilde{\mathbf{K}} = \Big(\lambda_1\mathbf{E}^T\mathbf{E} + \mathbf{G}^T\mathbf{G}\Big)$. Solving Eq 4.4 is straightforward, as it is equivalent to solving the following linear system:

$$\mathbf{D}\tilde{\mathbf{K}}^{-1}\Big(\lambda_2\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{E}^T\mathbf{E} + \mathbf{G}^T\mathbf{G}\Big)\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha = \lambda_2\mathbf{D}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\mathbf{y}_o \tag{4.5}$$

By using the inverse lemma, the closed form solution to Eq 4.5 is:

$$\hat{\alpha}^* = \lambda_2 diag^{-1}(\Upsilon)\left(\frac{\frac{1}{k}\Big(\sum_i^k \hat{\mathbf{x}}_{1i}^*\Big) \odot \Big(\sum_i^k \hat{\mathbf{x}}_{1i}\Big) \odot \hat{\mathbf{y}}_0^*}{\sum_i^k(\hat{\mathbf{x}}_{1i}^* \odot \hat{\mathbf{x}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{x}}_{1i}^* \odot \sum_i^k \hat{\mathbf{x}}_{1i})} + \frac{\hat{\mathbf{y}}_o^*}{k}\right), \tag{4.6}$$

where

$$\Upsilon = \left(\frac{\frac{-1}{k}\sum_i^k (\hat{\mathbf{x}}_{1i}^* \odot \hat{\mathbf{x}}_{1i}) + \frac{k+\lambda_2}{k}(\sum_i^k \hat{\mathbf{x}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{x}}_{1i}) + \frac{\lambda_1(k+\lambda_2)}{k}}{\sum_i^k (\hat{\mathbf{x}}_{1i}^* \odot \hat{\mathbf{x}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{x}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{x}}_{1i})} \right) \odot \left(\frac{\frac{1}{k^2}\sum_i^k \hat{\mathbf{x}}_{1i}^* \odot \sum_i^k \hat{\mathbf{x}}_{1i}}{\sum_i^k (\hat{\mathbf{x}}_{1i}^* \odot \hat{\mathbf{x}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{x}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{x}}_{1i})} + \frac{1}{k} \right) \tag{4.7}$$

Here, $\hat{\mathbf{x}}_{1i}$ denotes the FFT of the first row of $\mathbf{X}_i$ where all the operations are element wise and thus computationally attractive. Moreover, the solution for one single template can be easily found by setting $k = 1$ in Eq 4.6 to obtain:

$$\hat{\alpha} = \frac{\left(\frac{\lambda_2}{\lambda_1}(\hat{\mathbf{x}}_1 \odot \hat{\mathbf{x}}_1^*) + \lambda_2\right) \odot \hat{\mathbf{y}}_o}{\frac{\lambda_2}{\lambda_1^2}(\hat{\mathbf{x}}_1 \odot \hat{\mathbf{x}}_1^* \odot \hat{\mathbf{x}}_1 \odot \hat{\mathbf{x}}_1^*) + \frac{1+2\lambda_2}{\lambda_1}(\hat{\mathbf{x}}_1 \odot \hat{\mathbf{x}}_1^*) + (1+\lambda_2)}, \quad \text{for} \;\; k = 1 \tag{4.8}$$

Detection Formula. The previous solution is used to train the filter $\mathbf{w}$ or the dual variables α. As for the detection, a similar approach is used as in [20], where a circulant data matrix of the test sample $\mathbf{u}$ is considered for detection. The following is the detection formula for a single template case:

$$\mathbf{T}(\mathbf{u}) = \mathbf{U}\mathbf{w} = \mathbf{X}^T\alpha = \frac{1}{\lambda_1}\mathbf{F}diag(\hat{\mathbf{u}} \odot \hat{\mathbf{x}}_1^*)\hat{\alpha}^* \;\Rightarrow\; \hat{\mathbf{T}}(\mathbf{u}) \;\; = \frac{1}{\lambda_1}\hat{\mathbf{u}}^* \odot \hat{\mathbf{x}}_1 \odot \hat{\alpha}, \tag{4.9}$$

where $\hat{\mathbf{T}}$ is the FFT of the detection over all circular shifts of a sample $\mathbf{u}$. It is important to note that when $\lambda_2 \to \infty$ the soft constraint becomes a hard one, where our formulation reduces back to the original CF tracking formulation with a target response $\mathbf{y}_o$. Therefore, the standard CF tracking framework can be viewed as a special case of our adaptive formulation.

Comparison to CF Based Trackers. As discussed earlier, CF based trackers [20–22, 45, 50], as shown in Figure 4.3, exploit two main steps: detection and training while the target response used during the training is assumed to be independent of the frame and taken to be Gaussian centered at the window center. This inherently

assumes that the detected location of the window is correct.

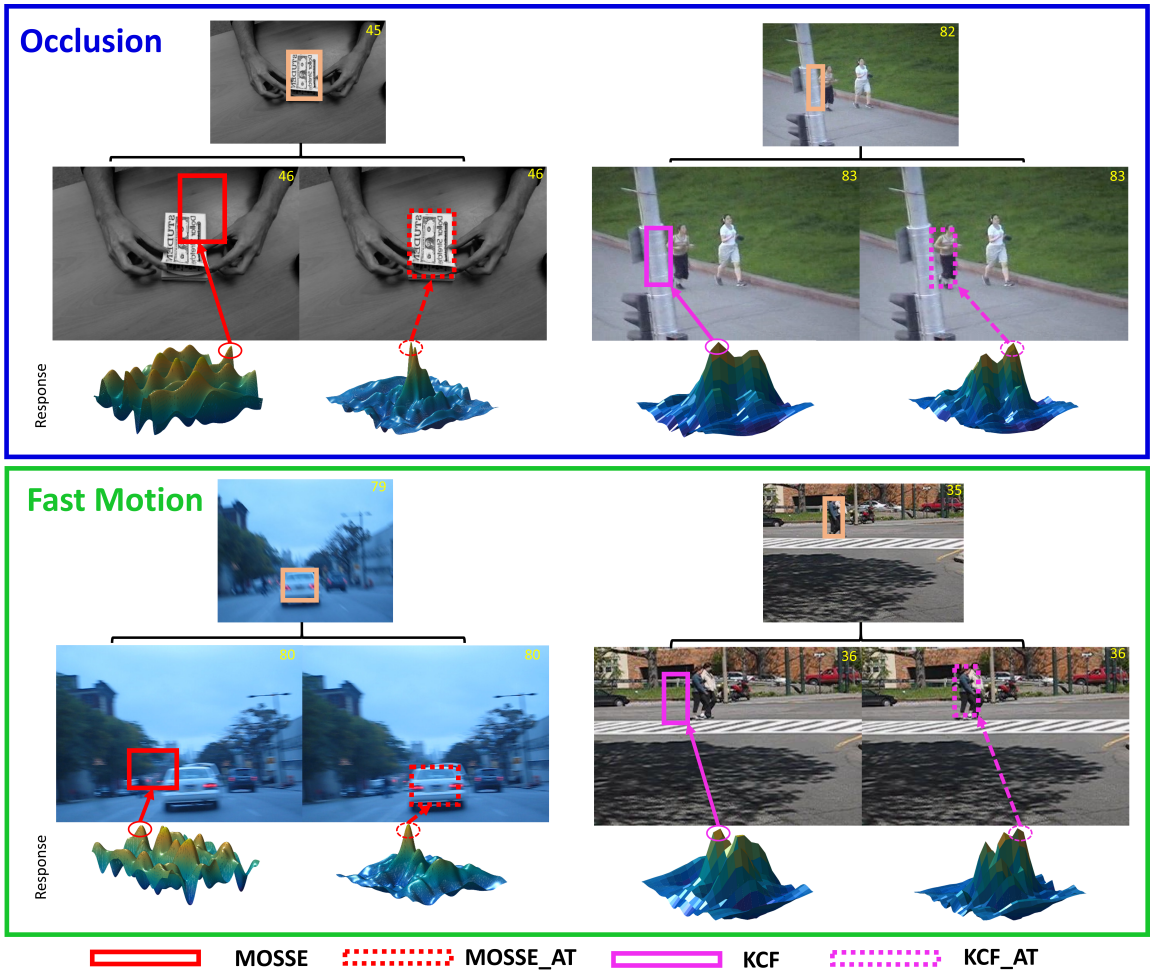

Figure 4.4: The first row shows tracking results on occlusion sequences (from left to right: *Coupon* and *Jogging1*) for MOSSE and KCF, along with their adaptive target versions MOSSE_{AT} and KCF_{AT}. In the second row, we show similar results for two fast motion sequences (from left to right: *BlurCar2* and *Couple*) for the same trackers.

When errors (even as small as a few pixels) arise in the detection, the target response $\mathbf{y}$ is not centered properly and these errors propagate into the filter estimation. This error propagation usually leads to tracker drift if multiple subsequent detection errors are encountered. This is illustrated in Figure 4.2. Obviously, this detection/training process is not fault tolerant and has difficulties recovering from errors. In comparison, our approach assumes that $\mathbf{y}$ is unknown and estimates it at every frame by making use of a target response prior $\mathbf{y}_o$, which exploits correlation

values at actual translations in the next frame to help the filter update regress to more realistic target values. As such, our proposed strategy is less prone to error propagation than the classical CF procedure. We illustrate this conclusion with a qualitative example in Figure 4.4, where two trackers (MOSSSE and KCF) are compared against their target adpative versions, when they encounter occlusion and fast motion. When our adaptive target method is used, the corresponding response maps are less noisy with the simpler tracker (MOSSE) and better localized with KCF (that uses more sophisticated features). The response map is biased towards the correct target location, since actual translations are used in the correlation measurements of the training step.

4.5 Experiments

We validate our adaptive target response framework by integrating it into five popular CF-based trackers. The experiments are run on the OTB100 dataset [6], which comprises 100 challenging video sequences including all 50 videos from its previous version OTB50 [3]. As compared to other tracking datasets, OTB100 contains a higher percentage of sequences that experience fast motion, motion blur, and occlusion.

Baseline Trackers. They differ in terms of the features used, kernels applied, and their ability to adapt to object scale variations. Particularly, MOSSE [22] uses grayscale features and a linear kernel, while CSK [21] uses the same features but with a gaussian kernel. DCF [20] uses HOG features along with a linear kernel, while KCF [20] uses the same features but with a gaussian kernel. The four aforementioned trackers do not adapt to scale changes, so we choose SAMF [45] to represent CF-based trackers that are scale variant. Note that DSST [23] is another option of this type, but we only include SAMF in our evaluation because it outperforms DSST on OTB100 [6] and their methodology is very similar. Applying our framework to the five baseline trackers gives rise to their adaptive target variants: $\text{MOSSE}_{AT}, \text{CSK}_{AT}, \text{DCF}_{AT}$,

KCF$_{AT}$, and SAMF$_{AT}$.

In fact, our framework can be applied to any CF-based tracker, but the aforementioned ones (and most trackers of this type in general) use a formulation that allows for the direct and efficient implementation of our target adaptation. Other trackers, such as SRDCF [50], are included in the evaluation but not modified for target response adaptation because the closed form solutions in Eqs 4.6&4.8 do not apply directly to the underlying optimization in these trackers. For example, SRDCF adds spatial regularization to $\mathbf{w}$, which impedes the effective exploitation of circulant structure. Nevertheless, we provide details in the **Appendix** on how our framework can be extended to include SRDCF and trackers with similar formulations.

Implementation Details and Parameters. In all our experiments, we use MATLAB on an Intel(R) Xeon(R) 2.67GHz CPU with 32GB RAM. For all the baseline trackers, we use the original parameters provided by the authors. The best regularization parameters λ_1 and λ_2 are selected for each baseline tracker. They are $\{(10^{-1}, 10^{-2}), (10^{-1}, 10^{-4}), (10^{-6}, 10^{-2}), ((10^{-3}, 10^{-5}), ((10^{-3}, 10^{-2})\}$ for MOSSE$_{AT}$, CSK$_{AT}$, DCF$_{AT}$, KCF$_{AT}$, and SAMF$_{AT}$ respectively. For simplicity and fair comparison, we consider $k = 1$ templates in our experiments for all trackers. The standard update rule for the newly computed filter [20–22] is used, where the learning rate is set to $(0.02, 0.01, 0.01, 0.01, 0.015)$ for MOSSE$_{AT}$, CSK$_{AT}$, DCF$_{AT}$, KCF$_{AT}$, and SAMF$_{AT}$ respectively.

As for the number of translations used to form the prior target response $\mathbf{y}_o$, we set $p = 13$ for trackers with grayscale features (MOSSE$_{AT}$ and CSK$_{AT}$) and $p = 7$ for those with HOG features (DCF$_{AT}$, KCF$_{AT}$, and SAMF$_{AT}$). This discrepancy is due to cell size used in HOG (or any other patch based feature). The granularity of each translation is dependent on the cell size of the feature, which is taken to be 4 pixels for HOG. In this case, the minimum translation possible would be 4 pixels, thus, allowing for larger translations with a smaller number of translation samples.

For the case of $p = 13$, translations are initialized from the set of $\{0, 3, -3, 5, -5\}$ pixels in a grid fashion, while this set is $\{0, 4\}$ when $p = 7$. Several expirements have been conducted on several choices of p, but it turns out that increasing p beyond 13 have marginal impact on performance. The padding region was set to 2 for all trackers. Moreover, the scaling function of SAMF$_{AT}$ is the same as that of SAMF, where 9 scales are considered with the same step size as the original implementation [45].

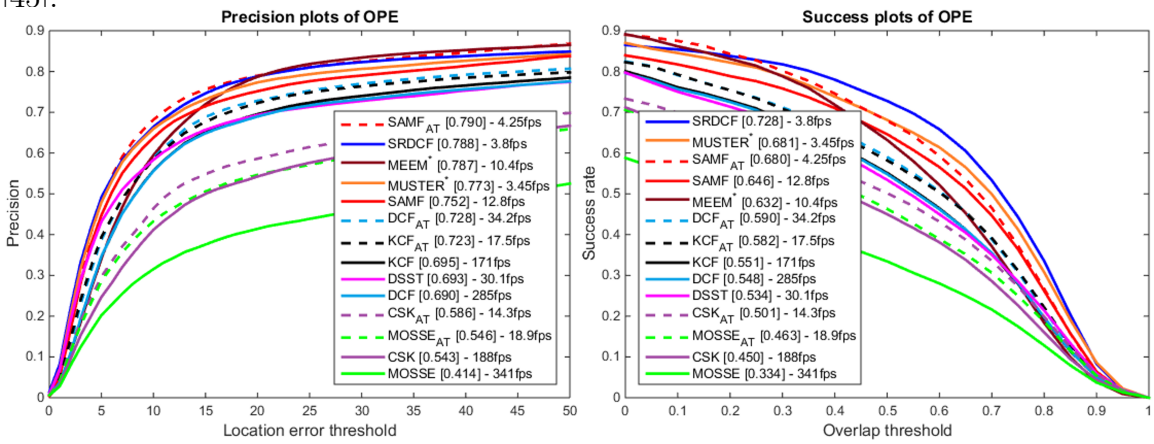

Figure 4.5: Precision and accuracy results for five baseline CF trackers, their adaptive target variants, as well as, other state-of-the-art methods. Trackers denoted by * are either not CF trackers or only use a CF tracker as a baseline for a generic framework.

Quantitative Results. We run all baseline and adaptive target trackers on OTB100 [6]. Following its standard evaluation strategy, we show the overall precision and accuracy plots of all trackers in Figure 4.5. The precision is defined as the average number of frames per video that are at most 20 pixels away from the ground truth, while the accuracy is defined as the average number of frames per video where the intersection over union with ground truth is at least 0.5. For a complete comparison, we also show the results of other state-of-the-art trackers including, SRDCF [50], MUSTER [52], MEEM [37], and DSST [23]. All trackers with target adaptation improve in performance, where the improvement ranges from 4% for sophisticated trackers like SAMF to 15% for basic ones like MOSSE. It is worthwhile to note that SAMF$_{AT}$ achieves state-of-art performance in precision and is tied with MUSTER

for second place right after SRDCF in accuracy. The reason behind this ranking discrepancy between the two metrics is primarily due to the scaling modality used in SAMF. Evidence of this phenomenon also arises in Figure 4.2, where SAMF accuracy is worse than its precision, even when the target response is optimal.

In Figure 4.6, we show an extensive comparison of the baseline trackers and their adaptive target variants on sequences with attributes (fast motion, motion blur, occlusion, and low resolution) that are expected to benefit the most from our adaptive framework. In fact, the performance of all the baseline trackers improves with target adaptation, some more than others in certain attributes. In general, trackers that use multi-dimensional sophisticated features experience less improvement than those that use grayscale features. For example, since there are more severe object translations in the subset of videos in the motion blur category that do not belong to the fast motion category, the range of improvement in the former category (6%-24.1%) is higher than the latter (2.6%-25.8%). In the occlusion category, the trackers with grayscale features (MOSSE and CSK) are the only ones with significant improvement (13% and 7.7% respectively). On the other hand, trackers that use sophisticated features and/or non-linear kernels have less improvement (i.e. DCF_{AT}, KCF_{AT}, and SAMF_{AT}). The reason behind this non-uniform improvement among trackers is two-fold. First, the occlusion category comprises about 50% of the whole OTB dataset, so occlusion videos contain many other attributes (some that do not benefit from target adaptation), thus, making the improvement less obvious. Secondly, more sophisticated features (e.g. HOG) play an important role in making the detection step of the CF tracker more robust to occlusion. The low resolution category witnesses the largest improvements overall. Interestingly, for videos with this attribute, basic trackers like MOSSE and CSK can outperform more established trackers like SAMF, when they exploit target adaptation. In fact, even SAMF improves by 15.7% here. Since our method makes use of correlation scores from actual translations to bias the

target response, the learned filter is better at localizing a smaller object (i.e. whose dimensions are more comparable to its frame-to-frame translation), when compared to traditional CF trackers that only use circular shifts. This is because a standard cosine window is applied to the patch [20,21] which is proportional to object's size. This limits the motion search for the object in standard CF trackers unlike our method that allows for the detection of larger translations.

Qualitative Results. Figure 4.7 shows qualitative results comparing the five baseline trackers to their target adaptive variants. For the first row (the *BlurOwl* sequence), the target undergoes fast motion along with motion blur. Unlike SAMF, SAMF$_{AT}$ is able to track the object throughout the complete sequence. In the second row (the *Human4* sequence), KCF is unable to keep tracking the target as it undergoes partial occlusion. When the occluder appears inside the filter window, the circular shifts are no longer good approximations of actual translations and the tracker drifts. Similar behaviour arises when the DCF, KCF, and MOSSE trackers are applied to the *Freeman4*, *Coke*, and *Woman* sequences, respectively. In all these sequences, the target adaptive version of each CF tracker is able to consistently maintain the track.

4.6 Conclusion

In this chapter, we propose a generic framework for correlation filter (CF) based trackers to counter the problem of fast motion, motion blur, and occlusion in videos. Our approach efficiently solves for both the filter and target response jointly, whereby the target response is regularized using correlation scores evaluated at sampled translations. Experiments demonstrate significant improvement in performance when our adaptive target framework is applied to many CF trackers. The proposed method is generic and can be incorporated into any CF based tracker. For future work, we aim to investigate more systematic and effective strategies for sampling the translations from frame to frame.

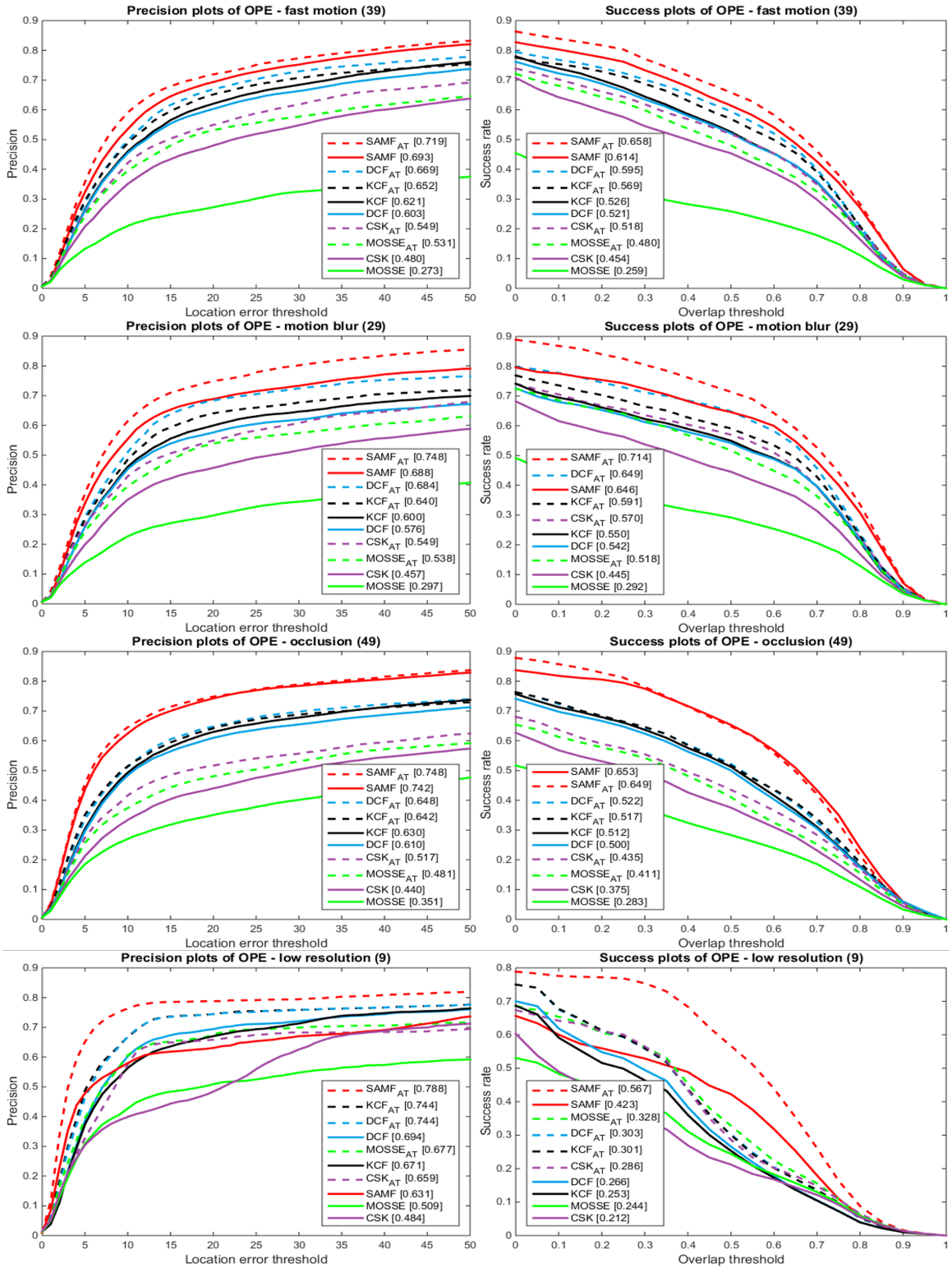

Figure 4.6: Precision and accuracy results for the fast motion, motion blur, occlusion, and low resolution categories in OTB100 [6].

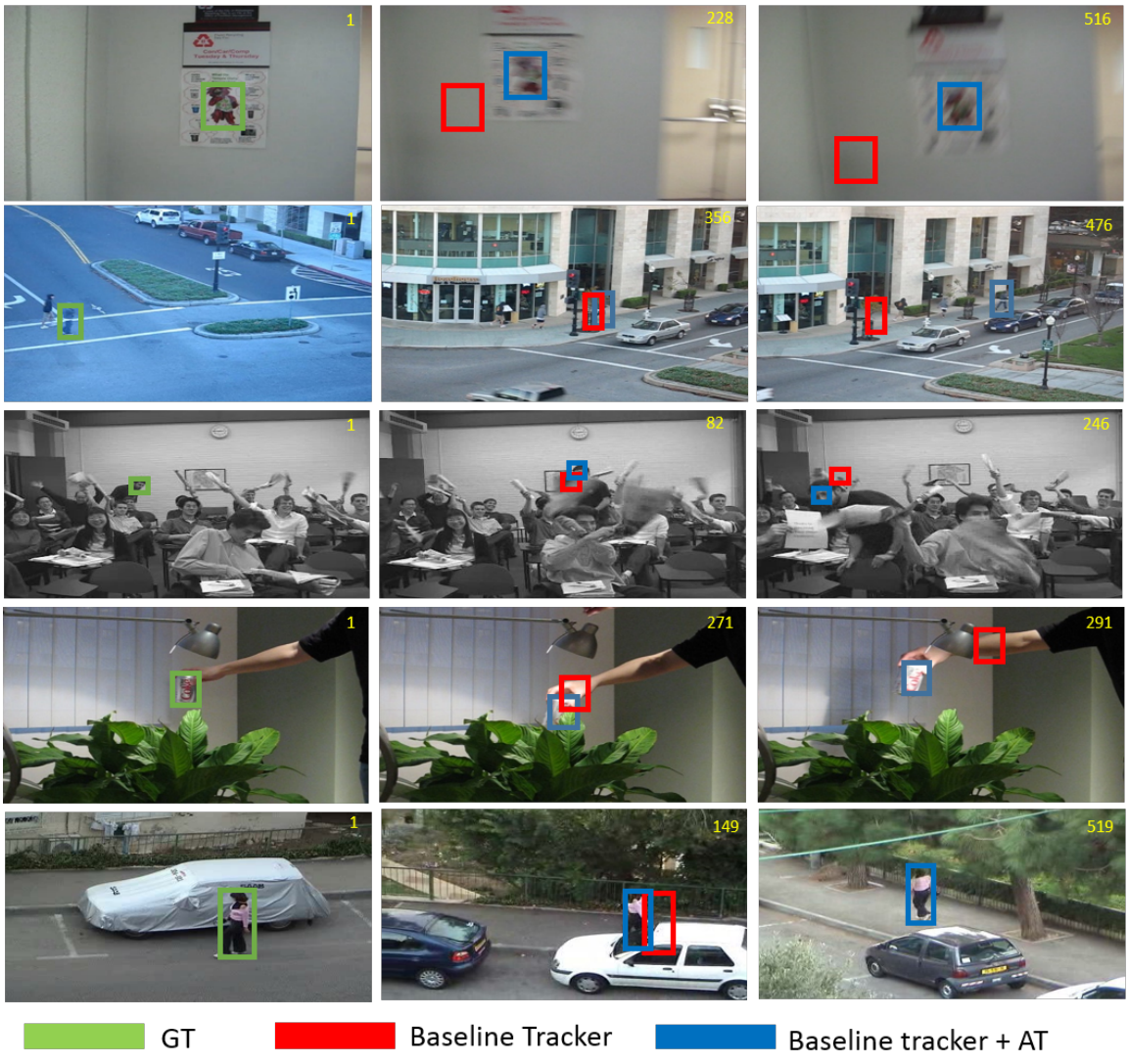

Figure 4.7: Shows tracking results for five videos (from top to bottom: *BlurOwl*, *Human4*, *Freeman4*,*Coke*, and *Woman*). In each row, a different baseline tracker is applied (from top to bottom: SAMF, KCF, DCF, CSK, and MOSSE) along with its adaptive target variant.

Chapter 5

In Defense of Sparse Tracking: Circulant Sparse Tracker

5.1 Introduction

Recently, sparse representation has been developed for object tracking [54–62]. The seminal work in [58] was the first to successfully apply sparse representation to visual tracking using particle filtering. Here, the tracker represents each target candidate as a sparse linear combination of dictionary templates that can be dynamically updated to maintain an up-to-date target appearance model. This representation has been shown to be robust against partial occlusions, thus, leading to improved tracking performance. However, sparse trackers generally suffer from the following drawbacks: **(1)** They remain computationally expensive, despite recent speedup attempts [63]. Sparse trackers perform computationally expensive ℓ_1 minimization at each frame. In a particle filter framework, computational cost grows linearly with the number of particles sampled. It is this computational bottleneck that precludes the use of these trackers in real-time scenarios. **(2)** They have limited overall performance. Based on results in the VOT2014 challenge [44] and the visual tracking benchmark [3], sparse trackers do not achieve a better accuracy than other types of trackers, such as KCF [20] and Struck [49]. **(3)** They are limited in the features they can use

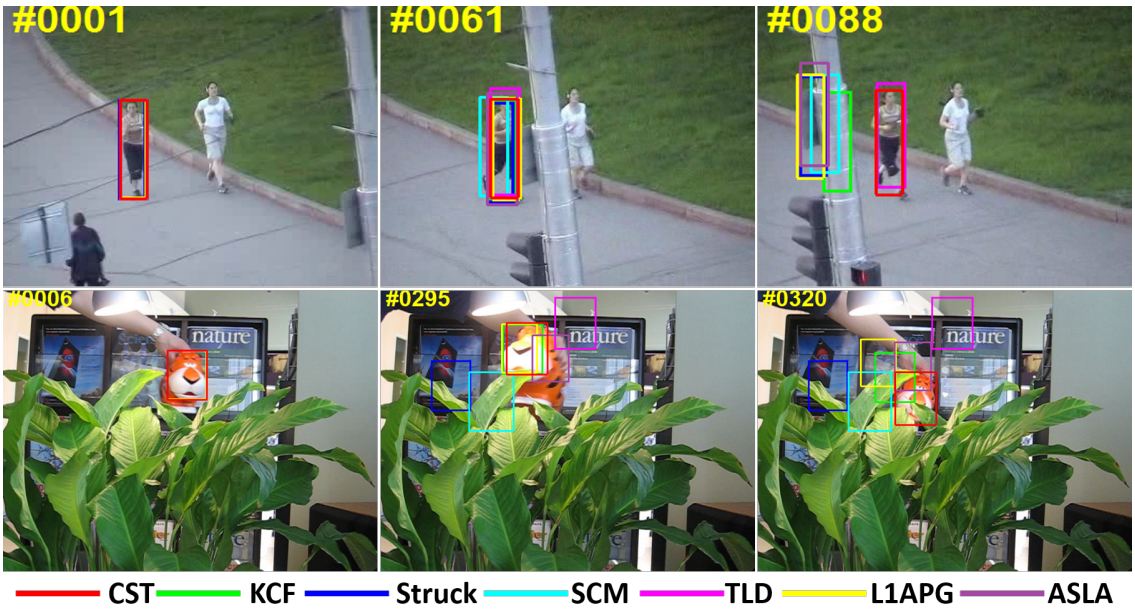

Figure 5.1: Comparison of our CST tracker with state-of-the-art methods (KCF, Struck, SCM, ASLA, L1APG, and TLD) on two videos from a visual tracking benchmark [3]. On the *Jogging* sequence, only TLD and CST can track well when partial occlusion happens. On the *Tiger* sequence, our CST can track the target throughout the whole sequence, while other trackers suffer from drift. The sparse trackers (ASLA, L1APG, SCM) fail to track these sequences. Overall, the proposed CST method performs favorably against the state-of-the-art trackers.

for representation. Due to the high computational cost, most sparse trackers make use of gray-scale pixel appearance and cannot adopt more representative features, such as, HOG and SIFT. As a result, sparse trackers have a bottleneck in tracking performance due to their limited feature representation. Due to these issues, sparse trackers cannot achieve promising results as shown in Figure 5.1 and are viewed as inadequate solutions to robust visual tracking.

To deal with the above issues, we propose a novel circulant sparse tracker (CST) with robustness and computational efficiency for visual tracking using particle filters. Here, learning the representation of each particle is again viewed as a sparse encoding problem. Similar to the above work, the next target state is decided by particles that have high similarity with a dictionary of target templates. Unlike previous methods that need to sample more and more candidates to refine the target's location, we

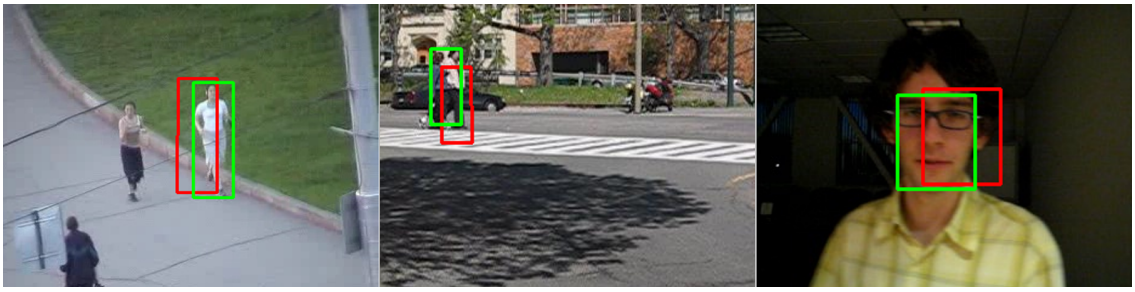

Figure 5.2: Examples on three video sequences to show particle refinement via the proposed CST method. The bounding boxes with red color are the sampled particles. Due to random sampling, the sampled particles are far away from the target object. With the proposed CST method, these particles can be refined and translated to better state denoted with bounding boxes with green color.

embed *translated* versions of the templates in the dictionary using circulant matrices. By doing this, an accurate sparse representation of each particle can be efficiently estimated and the target object can be *collectively* localized using a small number of particles. Given an object image patch $\mathbf{a}$ of $M \times N$ pixels, where all the circular shifts of $\mathbf{a}_{\bar{m},\bar{n}}$, $(\bar{m},\bar{n}) \in \{0,1,\ldots,M-1\} \times \{0,1,\ldots,N-1\}$, are generated as target templates $\hat{\mathbf{A}} = [\mathbf{a}_{0,0},\ldots,\mathbf{a}_{M-1,N-1}]$. These circulant shifts approximate the translated versions of $\mathbf{a}$. Then, each particle can be represented as a sparse linear combination of dictionary templates $\hat{\mathbf{A}}$. Based on the learned sparse coefficient, we know which element of $\hat{\mathbf{A}}$ has the highest similarity with the target candidate. The circular shift of these elements can be adopted to translate and refine each particle to make it more similar to the target.

In Figure 5.2, we show three examples of this particle refinement. Since particles are sampled randomly from a posterior distribution, they can be far away from the target object as shown in red bounding boxes. If we estimate the target's state based on only a few of these particles, the tracker can easily drift. By embedding circulant versions of the target templates in the dictionary, particles can be refined as shown (in green). Because of this refinement, the number of particles needed for tracking can be reduced dramatically, thus, making the tracker much more efficient. Moreover, by exploiting the blockwise circulant structure of the dictionary, all computation

can be efficiently done in the Fourier domain. Also, this allows us to seamlessly exploit more sophisticated feature representations (e.g. HOG or SIFT) which existing sparse trackers cannot because of the prohibitive computational cost. As a result, the proposed CST tracker can achieve much better tracking performance than state-of-the-art trackers as shown in Figure 5.1.

Contributions: The contributions of this work are three-fold. **(1)** We propose a novel circulant sparse tracker, which is a robust and effective sparse coding method that exploits circulant structure in target templates to obtain better tracking results than traditional sparse trackers. To the best of our knowledge, this is the first work to exploit the circulant structure of target templates for sparse representation. **(2)** Due to the circular shifts of target templates, we can refine particles and reduce their number. Moreover, due to the circulant property, we can apply CST efficiently in the Fourier domain. This makes our tracking method computationally attractive in general and faster than traditional sparse trackers in particular. **(3)** Because all the computation can be performed efficiently in the Fourier domain, we can use more sophisticated feature representations, which significantly improve tracking performance.

5.2 Related Work

Visual tracking has been studied extensively in the literature [1,64,65]. In this section, we briefly overview trackers that are most relevant to our work.

Generative vs. Discriminative Tracking: Visual tracking algorithms can be generally categorized as either generative or discriminative. Generative trackers typically search for the best image regions, which are similar to the tracked targets [7, 9, 66–68]. Black et al. [67] learn an off-line subspace model to represent the target object for tracking. In [7], the mean shift tracking algorithm represents a target with nonparametric distributions of color features and locates the object with

mode shifts. Kwon et al. [66] decompose the observation model into multiple basic observation models to cover a wide range of pose and illumination variation. In [68], the IVT tracker learns an incremental subspace model to adapt appearance changes. As compared to generative trackers, discriminative approaches cast tracking as a classification problem that distinguishes the tracked targets from the background [17,18,20,23,37,49,69–71]. Avidan [18] combines a set of weak classifiers into a strong one to do ensemble tracking. Grabner et al. [69] introduce an online boosting tracking method with discriminative feature selection. The Struck tracker [49] adopts an online structured output support vector machine for adaptive visual tracking. In [20], the KCF tracker exploits the circulant structure of adjacent image patches in a kernel space with HOG features. Zhang et al. [37] utilize multiple experts using entropy minimization to address the model drift problem in visual tracking. Hong et al. [71] cooperate short-term processing and long-term processing in visual tracking.

Sparse Tracking: Sparse linear representation has recently been introduced to object tracking with demonstrated success [54–61, 72]. In the seminal ℓ_1 tracking work [58], a candidate region is represented by a sparse linear combination of target and trivial templates where the coefficients are computed by solving a constrained ℓ_1 minimization problem with non-negativity constraints. As this method entails solving one ℓ_1 minimization problem for each particle, the computational complexity is significant. An efficient ℓ_1 tracker with minimum error bound and occlusion detection was subsequently developed [57]. In addition, methods based on dimensionality reduction, as well as, orthogonal matching pursuit [56] and accelerated proximal gradient descent [60] have been proposed to make ℓ_1 tracking more efficient. Recently, several tracking algorithms have been proposed to learn the sparse representations of all particles jointly [12, 62]. In [12], learning the representation of each particle is viewed as an individual task and a multi-task learning formulation for all particles is proposed based on joint sparsity. In [62], a multi-task multi-view joint sparse representation

for visual tracking is introduced. Based on the performance results in the VOT2014 challenge [44] and online tracking benchmark [3], sparse trackers clearly have drawbacks in both efficiency and accuracy. In defense of this type of tracker, we propose an effective and efficient circulant sparse tracker.

5.3 Circulant Sparse Tracking

In this section, we give a detailed description of our particle filter based circulant sparse tracking method that makes use of the circulant structure of target templates to represent particles. Similar to [58], we assume an affine motion model between consecutive frames. Therefore, the state of a particle $\mathbf{s}_t$ consists of six affine transformation parameters (2D linear transformation and translation). By applying an affine transformation based on $\mathbf{s}_t$, we crop the region of interest $\mathbf{y}_t$ from the image and normalize it to the same size as the target templates in our dictionary, i.e. the target size in the first frame). The state transition distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ is modeled using a zero-mean Gaussian with diagonal covariance. The observation model $p(\mathbf{y}_t|\mathbf{s}_t)$ reflects the similarity between an observed image region $\mathbf{y}_t$ corresponding to a particle $\mathbf{s}_t$ and target templates of the current dictionary. In this work, $p(\mathbf{y}_t|\mathbf{s}_t)$ is computed based on the distance between the refined particle and the template dictionary. At each frame, the target's state is estimated as a weighted average of the states of the refined particles.

5.3.1 Circulant Sparse Model

In the t^{th} frame, we sample n particles. Consider one of these particles as an example. Its feature representation is denoted as $\mathbf{x} \in \mathbb{R}^d$. Here, for simplicity, we assume $\mathbf{x}$ is a 1D signal with pixel color values; however, this can be easily extended to the 2D case with HOG features, as we will discuss in Section 5.3.2. Each $\mathbf{x}$ is represented

as a linear combination $\mathbf{z}$ of dictionary templates $\mathbf{D}$, such that $\mathbf{x} = \mathbf{D}\mathbf{z}$. In many visual tracking scenarios, target objects are often corrupted by noise or partially occluded. To address this issue, an error term $\mathbf{e}$ can be added to indicate the pixels in $\mathbf{x}$ that are corrupted or occluded: $\mathbf{x} = \mathbf{D}\mathbf{z} + \mathbf{e}$. Then, we can rewrite: $\mathbf{x} = \mathbf{A}\mathbf{c}$, where $\mathbf{A} = [\mathbf{D}\ \mathbf{I}]$, $\mathbf{c} = [\mathbf{z}; \mathbf{e}]$, and $\mathbf{I}$ is a $d \times d$ identity matrix. Dictionary $\mathbf{D}$ can be constructed from an overcomplete sampling of the target. Given K of these *base* samples $\mathbf{a}_k$ with $k = 1, \ldots, K$ each having $M \times N$ pixels, $\mathbf{A}$ can be constructed as $\mathbf{A} = [\mathbf{A}_1, \ldots, \mathbf{A}_k, \ldots, \mathbf{A}_K]$. Here, $\mathbf{A}_k$ contains all the circular shifts of the k-th base sample $\mathbf{a}_k^{\bar{m},\bar{n}}$, where $(\bar{m}, \bar{n}) \in \{0, 1, \ldots, M-1\} \times \{0, 1, \ldots, N-1\}$, $\mathbf{A}_k = [\mathbf{a}_k^{0,0}, \ldots, \mathbf{a}_k^{M-1,N-1}]$, and $\mathbf{A}_{K+1} = \mathbf{I}$ to represent the trivial templates. Therefore, each $\mathbf{A}_k \in \mathbb{R}^{d \times d}$ is circulant, and $\mathbf{A} \in \mathbb{R}^{d \times Kd}$ is blockwise circulant. As mentioned earlier, the circulant shifts of the base templates approximate their 2D translations. The particle representation can be obtained by solving the ℓ_1 minimization problem (5.1).

$$\min_{\mathbf{c}} \frac{1}{2} \|\mathbf{x} - \mathbf{A}\mathbf{c}\|_2^2 + \lambda \|\mathbf{c}\|_1 \tag{5.1}$$

Solving (5.1) with large-scale $\mathbf{A}$ is infeasible in the visual tracking task. To deal with this issue, we solve the dual problem of (5.1) in the Fourier domain. We introduce a dummy variable $\mathbf{r}$ along with a set of equality constraints:

$$\min_{\mathbf{c},\mathbf{r}} \frac{1}{2} \|\mathbf{r}\|_2^2 + \lambda \|\mathbf{c}\|_1 \quad s.t. \quad \mathbf{r} = \mathbf{A}\mathbf{c} - \mathbf{x} \tag{5.2}$$

Using $\mathbf{z}$ to denote the Lagrange multipliers, we can write the Lagrangian of this problem as

$$\mathcal{L}(\mathbf{c}, \mathbf{r}, \mathbf{z}) = \frac{1}{2} \|\mathbf{r}\|_2^2 + \lambda \|\mathbf{c}\|_1 + \mathbf{z}^\top (\mathbf{A}\mathbf{c} - \mathbf{x} - \mathbf{r}) \tag{5.3}$$

Distributing $\mathbf{z}$ across the subtraction and grouping terms involving $\mathbf{c}$ and $\mathbf{r}$, the resulting dual function is:

$$\max_{\mathbf{z}} \min_{\mathbf{c},\mathbf{r}} \mathbf{z}^\top \mathbf{A}\mathbf{c} + \lambda \|\mathbf{c}\|_1 + \frac{1}{2}\|\mathbf{r}\|_2^2 - \mathbf{z}^\top \mathbf{r} - \mathbf{z}^\top \mathbf{x} \tag{5.4}$$

Using the definition of the conjugate function to the ℓ_1-norm and ℓ_2-norm squared [73], we get the dual problem of (5.1) as shown in (5.5).

$$\min_{\mathbf{z}} \quad \frac{1}{2}\mathbf{z}^\top \mathbf{z} + \mathbf{z}^\top \mathbf{x} \quad s.t. \quad \left\|\mathbf{A}^\top \mathbf{z}\right\|_\infty \leq \lambda \tag{5.5}$$

5.3.2 Optimization in the Fourier Domain

In this section, we present algorithmic details on how to efficiently solve the optimization problem (5.5) in the Fourier domain using the fast first-order Alternating Direction Method of Multipliers (ADMM) [74]. We introduce a variable θ to make $\theta = \mathbf{A}^\top \mathbf{z}$ and $\|\theta\|_\infty \leq \lambda$. By introducing augmented Lagrange multipliers to incorporate the equality constraints into the objective function, we obtain a Lagrangian function that can be optimized through a sequence of simple closed form update operations in (5.6), where $\mathbf{c}$ and $u > 0$ are the Lagrange multiplier and penalty parameter, respectively. Since the dual solution to the dual problem of a convex optimization is the primal solution in general, then the Lagrange multiplier here is actually the sparse code vector $\mathbf{c}$ from (5.1) [75].

$$\begin{aligned} \mathcal{L}(\mathbf{c}, \mathbf{z}, \theta) &= \frac{\mathbf{z}^\top \mathbf{z}}{2} + \mathbf{z}^\top \mathbf{x} + \mathbf{c}^\top (\mathbf{A}^\top \mathbf{z} - \theta) + \frac{u}{2}\left\|\mathbf{A}^\top \mathbf{z} - \theta\right\|_2^2 \\ &\Rightarrow \max_{\mathbf{c}} \min_{\mathbf{z},\theta} \mathcal{L}(\mathbf{c}, \mathbf{z}, \theta) \end{aligned} \tag{5.6}$$

The ADMM method iteratively updates the variables θ and $\mathbf{z}$ (one at a time) by minimizing (5.6) and then performs gradient ascent on the dual to update $\mathbf{c}$. By up-

dating these variables iteratively, convergence can be guaranteed [74]. Consequently, we have three update steps corresponding to all three variables as follows.

Update θ: Given $(\mathbf{c}, \mathbf{z})$, θ is updated by solving the optimization problem (5.7) with the solution (5.8).

$$\theta = \arg\min_{\theta} \frac{u}{2} \left\| \mathbf{A}^\top \mathbf{z} - \theta \right\|_2^2 - \mathbf{c}^\top \theta \tag{5.7}$$

$$\Rightarrow \theta = \mathcal{P}_{\mathcal{B}_\lambda^\infty}(\mathbf{A}^\top \mathbf{z} + \frac{\mathbf{c}}{u}) \tag{5.8}$$

Here, $\mathcal{P}_{\mathcal{B}_\lambda^\infty}$ represents the projection operator onto $\mathcal{B}_\lambda^\infty$, and $\mathcal{B}_\lambda^\infty = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\|_\infty \leq \lambda\}$.

Note that, all circulant matrices are made diagonal by the Discrete Fourier Transform (DFT), regardless of the generating vector [76]. If $\mathbf{X}$ is a circulant matrix, it can be expressed with its base sample $\mathbf{x}$ as

$$\mathbf{X} = \mathcal{F} diag(\hat{\mathbf{x}}) \mathcal{F}^H, \tag{5.9}$$

where $\mathcal{F}$ is a constant matrix that does not depend on $\mathbf{x}$, and $\hat{\mathbf{x}}$ denotes the DFT of the generating vector: $\hat{\mathbf{x}} = \mathcal{F}\mathbf{x}$. The constant matrix $\mathcal{F}$ is known as the DFT matrix. $\mathbf{X}^H$ is the Hermitian transpose, i.e., $\mathbf{X}^H = (\mathbf{X}^*)^\top$, and $\mathbf{X}^*$ is the complex-conjugate of $\mathbf{X}$. For real numbers, $\mathbf{X}^H = \mathbf{X}^T$. In (5.8), $\mathbf{A}^\top \mathbf{z} = [\mathbf{A}_1^\top \mathbf{z}; \ldots; \mathbf{A}_{K+1}^\top \mathbf{z}]$, and it can be obtained via (5.10). Here, $\mathcal{F}^{-1}$ denotes the inverse DFT, while $\odot$ denotes the element-wise product. The $\mathbf{a}_k^\top$ is the base sample of circulant matrix $\mathbf{A}_k$.

$$\theta = \mathcal{P}_{\mathcal{B}_\lambda^\infty}(\mathcal{F}^{-1}[\hat{\mathbf{a}}_1^* \odot \hat{\mathbf{z}} + \frac{1}{u}\hat{\mathbf{c}}_1; \ldots; \hat{\mathbf{a}}_{K+1}^* \odot \hat{\mathbf{z}} + \frac{1}{u}\hat{\mathbf{c}}_{K+1}]) \tag{5.10}$$

Update $\mathbf{z}$: Given $(\mathbf{c}, \theta)$, updating $\mathbf{z}$ can be shown to be a least squares problem,

whose solution is given by (5.11).

$$\mathbf{z} = (\mathbf{A}\mathbf{A}^\top + \frac{1}{u}\mathbf{I})^{-1}(\mathbf{A}\theta - \frac{1}{u}\mathbf{x} - \frac{1}{u}\mathbf{A}\mathbf{c}) \tag{5.11}$$

Here, $\mathbf{A}\theta = \sum_{k=1}^{K} \mathbf{A}_k\theta_k = \mathcal{F}\sum_{k=1}^{K} \hat{\mathbf{a}}_k^* \odot \hat{\theta}_k^*$, $\mathbf{A}\mathbf{c} = \sum_{k=1}^{K} \mathbf{A}_k\mathbf{c}_k = \mathcal{F}\sum_{k=1}^{K} \hat{\mathbf{a}}_k^* \odot \hat{\mathbf{c}}_k^*$, and $\mathbf{A}\mathbf{A}^\top = \sum_{k=1}^{K} \mathbf{A}_k\mathbf{A}_k^\top = \mathcal{F}diag(\sum_{k=1}^{K} \hat{\mathbf{a}}_k \odot \hat{\mathbf{a}}_k^*)\mathcal{F}^H$. Therefore, the update $\mathbf{z}$ in (5.11) can be computed as $\mathbf{z} = \mathcal{F}^{-1}(\hat{\mathbf{z}})$, where $\hat{\mathbf{z}}$ is defined in (5.12). The fraction denotes element-wise division. Note that no expensive matrix inversion is required here. This is the defining difference between sparse representation on a traditional dictionary and that on a blockwise circulant one.

$$\hat{\mathbf{z}} = \frac{\sum_{k=1}^{K} (\hat{\mathbf{a}}_k \odot \hat{\theta}_k - \frac{1}{u}\hat{\mathbf{a}}_k \odot \hat{\mathbf{c}}_k) - \frac{1}{u}\hat{\mathbf{x}}}{\sum_{k=1}^{K} \hat{\mathbf{a}}_k \odot \hat{\mathbf{a}}_k^* + \frac{1}{u}} \tag{5.12}$$

Update c: Given $(\mathbf{z}, \theta)$, the sparse code $\mathbf{c}$ is updated in (5.13). The penalty parameter is increased from one iteration to the next: $u = \rho u$, where $\rho > 1$.

$$\mathbf{c} = \mathbf{c} + u(\mathbf{A}^\top\mathbf{z} - \theta) \tag{5.13}$$

In the Fourier domain, $\mathbf{c}$ can be updated as (5.14).

$$\mathbf{c} = \mathbf{c} + u(\mathcal{F}^{-1}[\hat{\mathbf{a}}_1^* \odot \hat{\mathbf{z}} - \hat{\theta}_1; \ldots; \hat{\mathbf{a}}_{K+1}^* \odot \hat{\mathbf{z}} - \hat{\theta}_{K+1}]) \tag{5.14}$$

The ADMM algorithm that solves (5.6) is shown in Algorithm 1, where convergence is reached when the change in the objective function or solution $\mathbf{z}$ is below a pre-defined threshold (e.g., $\tau = 10^{-3}$ in this work). Note that, in Algorithm 1, for a 1D signal $\mathbf{x}$, the $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the 1D DFT and its inverse. When $\mathbf{x}$ is 2D, $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the 2D DFT and its inverse. As such, the optimization in the Fourier domain is efficient for 2D image patches with multiple channels, which is a useful property

Algorithm 1: The optimization for (5.6) via ADMM.

Input : Dictionary $\mathbf{A}$ and Particle $\mathbf{x}$. Initialization of λ, $\theta = 0$, $\mathbf{z} = 0$, $\mathbf{c} = 0$, and $u > 0$.

Output: Particle Representation $\mathbf{c}$.

while *not converged* **do**
  Update θ via (5.10);
  Update $\mathbf{z}$ via (5.12);
  Update $\mathbf{c}$ as in (5.14);
end

for visual tracking.

5.3.3 The Proposed CST Tracker

In this section, we give a detailed description of the proposed CST tracker, including the template model update, target state estimation, and feature representation.

Model Update: In the proposed tracker, the model consists of target templates $\mathbf{A}$, which are updated dynamically to handle frame-to-frame changes in target appearance. For simplicity, we adopt the adaptive strategy in (5.15) by considering the current target appearance $\mathbf{x}$. Here, we only update the target template $\mathbf{a}_k$ whose corresponding sparse representation has a maximum absolute value among all other templates.

$$\mathcal{F}(\mathbf{a}_k)^t = (1-\eta)\mathcal{F}(\mathbf{a}_k)^{t-1} + \eta\mathcal{F}(\mathbf{x}) \tag{5.15}$$

Here, η is a learning rate parameter, and the $\mathbf{a}_k$ is the base sample of circulant matrix $\mathbf{A}_k$.

Target State Estimation: The target state is decided based on the n sampled particles with their states $\mathbf{s}_i$ and representations $\mathbf{c}_i, i = 1, \ldots, n$. For the i-th particle, its state $\mathbf{s}_i$ can be refined to $\bar{\mathbf{s}}_i$ by applying the translation corresponding to the circular shift of the target template whose corresponding coefficient in $\mathbf{c}_i$ has the

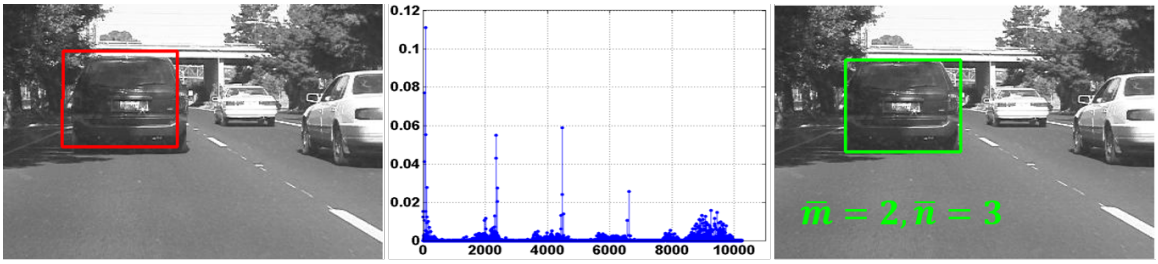

Figure 5.3: Illustration for particle refinement: (a) the original sampled particle in red, (b) the learned coefficient $\mathbf{c}$, and (c) the refined particle in green with a circular shift ($\bar{m} = 2, \bar{n} = 3$). Here, $K = 5$ and $\mathbf{x}$ is $41 \times 50 \times 31$ using HOG.

maximum absolute value. In Figure 5.2, we show three examples of this particle state refinement. Then, the target state $\mathbf{s}$ can be estimated via a weighted average of all the refined particle states: $\mathbf{s} = \sum_i \pi_i \bar{\mathbf{s}}_i$. Here, π_i denotes the confidence score of the i-th particle, and it is defined as $\pi_i = \max(|\mathbf{c}_i|)$. Once π_i of each particle is computed, they are normalized to predict the final target state.

Image Representation: Most existing sparse trackers make use of vectorized grayscale values to represent the image patch $\mathbf{x}$. It tends to be infeasible to adopt more sophisticated and higher-dimensional features, such as HOG and SIFT, since the added computational cost would be significant. On the other hand, our formulation (5.1) seamlessly enables the use of richer feature representations, without sacrificing much computationally. Algorithm (1) is used as is but with each target template $\mathbf{a}_k$ represented using the new feature. Similar to the grayscale case, the optimization can be efficiently solved in the Fourier domain, since only element-wise operations are needed. As such, we make HOG features feasible for sparse trackers.

For a more intuitive view of the proposed method, we visualize an empirical example to show how the proposed CST tracker works in Figure 5.3. Clearly, embedding circulant shifts of base templates into the sparse representation is helpful in refining each particle's state and localizing the target accurately.

5.4 Experiments

In this section, we present experimental results. (1) We introduce the experimental setup. (2) We show computational cost analysis in detail. (3) We perform a comprehensive evaluation of different features in sparse trackers. (4) We provide quantitative, qualitative, and attribute-specific comparisons with state-of-the-art trackers.

5.4.1 Experimental Setup

All experiments are implemented in MATLA on an Intel(R) Xenon(R) 2.70 GHz CPU with 64 GB RAM.

Parameters: The λ in (5.1) is set to 1. The learning rate η in (5.15) is set to 0.03. We use the same parameter values and initialization for all the sequences. All the parameter settings are available in the source code to be released for accessible reproducible research and comparative analysis.

Datasets and Evaluation Metrics: We evaluate the proposed method on a large benchmark dataset [3] that contains 50 videos with comparisons to state-of-the-art methods. The performance of our approach is quantitatively validated by three metrics used in [3,23] including distance precision (DP), centre location error (CLE), and overlap success (OS). The DP is computed as the relative number of frames in the sequence where the centre location error is smaller than a certain threshold. As in [3], the DP values at a threshold of 20 pixels are reported. The CLE is computed as the average Euclidean distance between the ground-truth and the estimated centre location of the target. The OS is defined as the percentage of frames where the bounding box overlap surpasses a threshold, which follows from the widely used PASCAL evaluation criterion. We report the results at a threshold of 0.5, which correspond to the PASCAL evaluation criteria. We provide results using the average DP, CLE, and OS over all 50 sequences. In addition, we plot the precision and success plots as in [3].

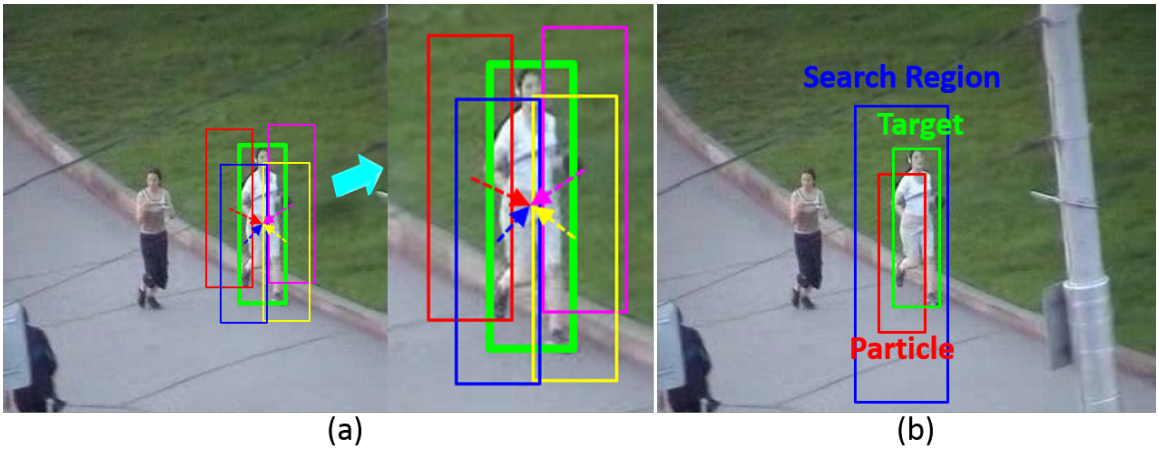

Figure 5.4: Particle reducing strategy via (a) particle refinement and (b) search region padding. See text for more details.

The average distance precision is plotted over a range of thresholds in the precision plot. In the legend, we report the average distance precision score at 20 pixels for each method. The average overlap precision is plotted in the success plot. The area under the curve (AUC) is included in the legend. We also report the speed of the trackers in average frames per second (FPS) over all image sequences.

5.4.2 Particle Refinement Strategy

In this section, we discuss how the proposed CST can refine particles, thus, effectively reducing the number of particles needed for accurate tracking. This involves two strategies: particle refinement and search region padding. As shown in Figure 5.4 (a), particles (denoted in different colors) are translated towards the target object (denoted in green color). The translation of each particle is set to be the circular shift corresponding to the highest absolute valued sparse coefficient in that particle's sparse code. This allows the sampled particles in the next frame to be closer to the target object. Moreover, the tracked region is 2 times the size of the target, to provide for context and additional search samples. This region determines the number of circulant shifts. As shown in Figure 5.4 (b), one particle (denoted in

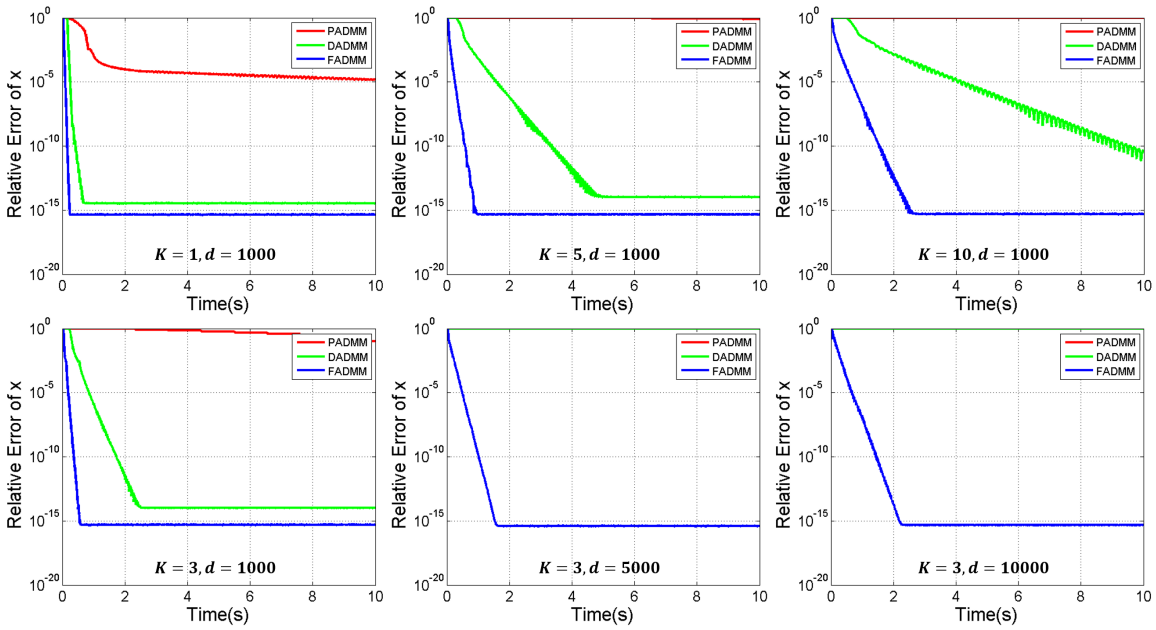

Figure 5.5: Comparisons on computational cost with state-of-the-art methods for different choices of K and d. The proposed FADMM achieves the best.

red) is quite far away from the target object (denoted in green). However, its search region (denoted in blue) can still cover the target object well. As a result, the particle can still be shifted towards the target object using via our proposed model. In our implementation, both particles and the base samples of target templates have the same search region. In addition, they are also weighted by a cosine window for robustness. Although enlarging the search region increases the computational cost, it adds robustness to the tracker against fast motion, partial occlusion, etc. To balance efficiency and accuracy in visual tracking, we adopt this search strategy using 2 times the size of the target. In the experiments, we test the tacking results with different number of particles, $n = 10$, $n = 20$, and $n = 50$, and they have very similar results. Therefore, we set $n = 20$ in our experiments. However, in traditional sparse trackers [12, 60], more than hundreds of particles are used.

	CST-HOG	**CST-Color**	L1APG [60]	SCM [72]	ASLA [61]	MTT [12]
OS	**68.2**	48.9	44.0	**61.6**	51.1	44.5
DP	**77.7**	54.3	48.5	**64.9**	53.2	47.5
CLE	**40.4**	86.2	77.4	**54.1**	73.1	94.5
FPS	2.2	**3.0**	2.4	0.4	**7.5**	1.0

Table 5.1: Comparison with state-of-the-art trackers on the 50 benchmark sequences. Our approach performs favorably against existing methods in overlap success (OS) (%), distance precision (DP) (%) and centre location error (CLE) (in pixels). The first and second highest values are highlighted by red and blue color. Additionally our method is faster compared to the best performing existing sparse tracker (SCM).

5.4.3 Computational Cost Evaluation

To evaluate the computational speedup of using Algorithm 1, we follow the experimental setting in [75]. A primary measure of performance is the relative error $r_n(\mathbf{x})$ as a function of CPU time t_n after n iterations: $r_n(\mathbf{x}) = \|\mathbf{x}_n - \mathbf{x}_0\|_2 / \|\mathbf{x}_0\|_2$, where $\mathbf{x}_n$ is the estimate after n iterations. We compare the two best state-of-the-art methods [75] to solve (5.1). The first is PADMM, which solves the primal problem (5.1) with ADMM, and the second is DADMM, which solves the dual problem (5.5) with ADMM. We denote Algorithm 1 as FADMM, which solves the dual problem (5.5) with ADMM in Fourier domain. We generate the target template matrix $\mathbf{A}$ of size $d \times Kd$ with K base samples $\mathbf{a}_k, k = 1, \ldots, K$. The observation $\mathbf{x}$ is computed by $\mathbf{A}\mathbf{x}_0$, where $\mathbf{x}_0$ is a sparse vector. The support of $\mathbf{x}_0$ is chosen at random, and the nonzero entries of $\mathbf{x}_0$ are i.i.d. according to a uniform distribution in the interval $[-10, 10]$. In this experiment, the $\mathbf{x}$ and $\mathbf{a}_k$ are 1D signals. For different choices of K and d, we compute the relative errors in estimating $\mathbf{x}_0$ as a function of CPU time using all the algorithms. Figure 5.5 shows the average relative errors over 20 trials. We observe in the three plots that FADMM is the fastest algorithm in all the cases, followed by DADMM. Moreover, the proposed FADMM is the only method that achieves near-machine precision ($r_n(\mathbf{x}) \leq 10^{-10}$) in solving (5.5). Clearly, it is much more advantageous computationally to solve the circulant sparse representation problem using our method than generic ADMM solvers.

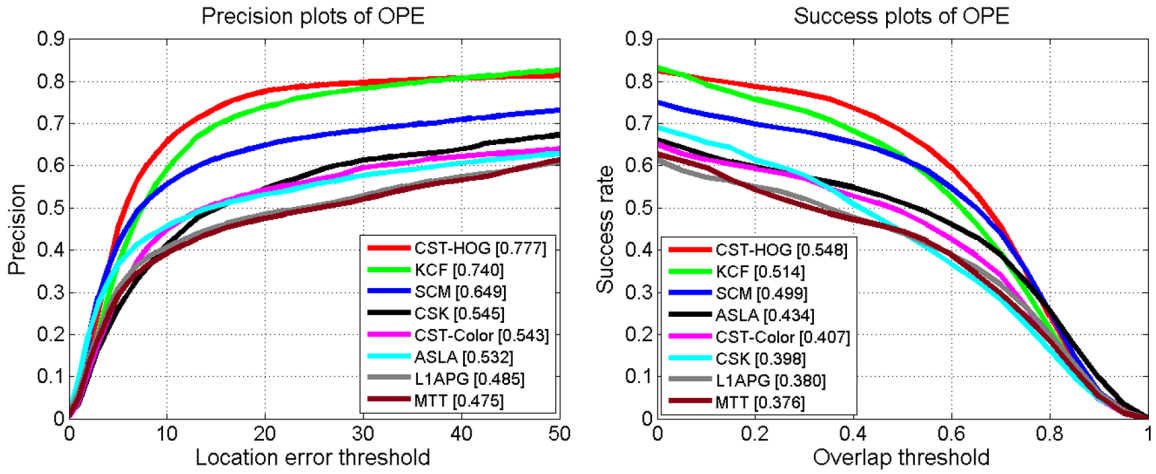

Figure 5.6: Comparisons of different sparse trackers by using precision and success plots over all the 50 sequences. The legend contains the area-under-the-curve score for each tracker. Our CST method performs favorably against the state-of-the-art trackers.

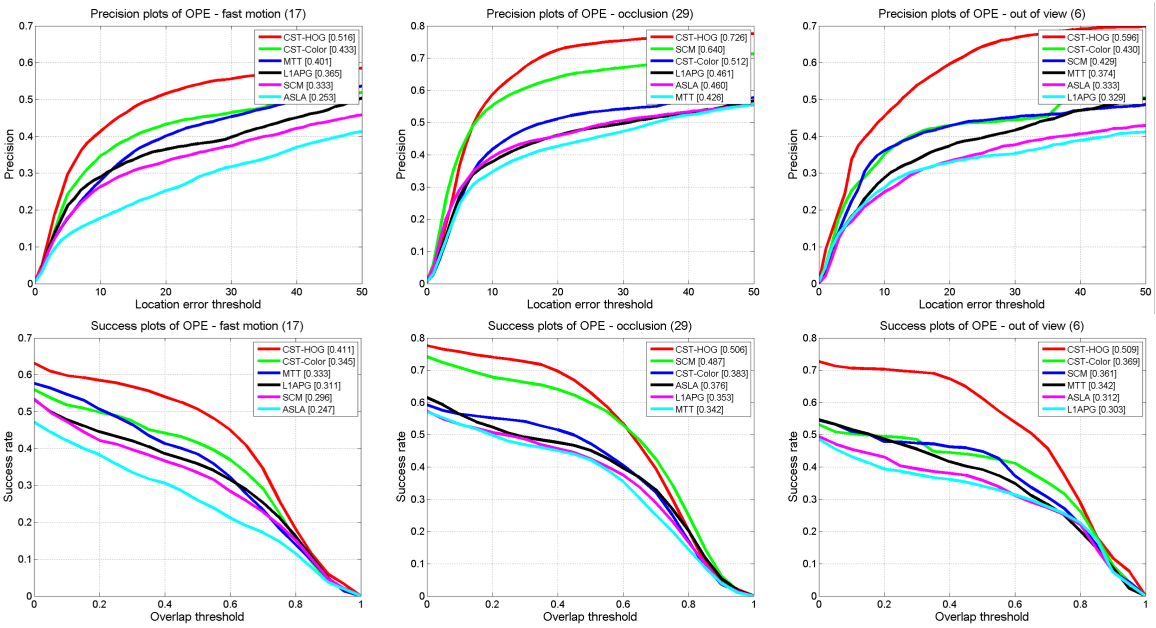

Figure 5.7: Overlap success plots over three tracking challenges of fast motion, occlusion, and out-of-view. The legend contains the AUC score for each tracker. The proposed CST method performs favorably against the state-of-the-art trackers.

5.4.4 Image Feature Evaluation

Here, we implement the proposed CST method with two different features: HOG (CST-HOG) and gray color (CST-Color). We report the results in one-pass evaluation

(OPE) using the distance precision and overlap success rate in Figure 5.6, which shows that replacing the conventional intensity values with HOG features significantly improves the tracking performance by 23.4% and 14.1% in terms of precision and success rate, respectively. Similarly, the HOG based tracker reduces the average CLE from 86.2 to 40.4 pixels as shown in Table 5.1. In summary, our results clearly suggest that the HOG based image representation improves the tracking performance, which is also demonstrated by comparing KCF to CSK [20]. Their results are also shown in Figure 5.6 for comparison. In what follows, we employ HOG features to represent particles.

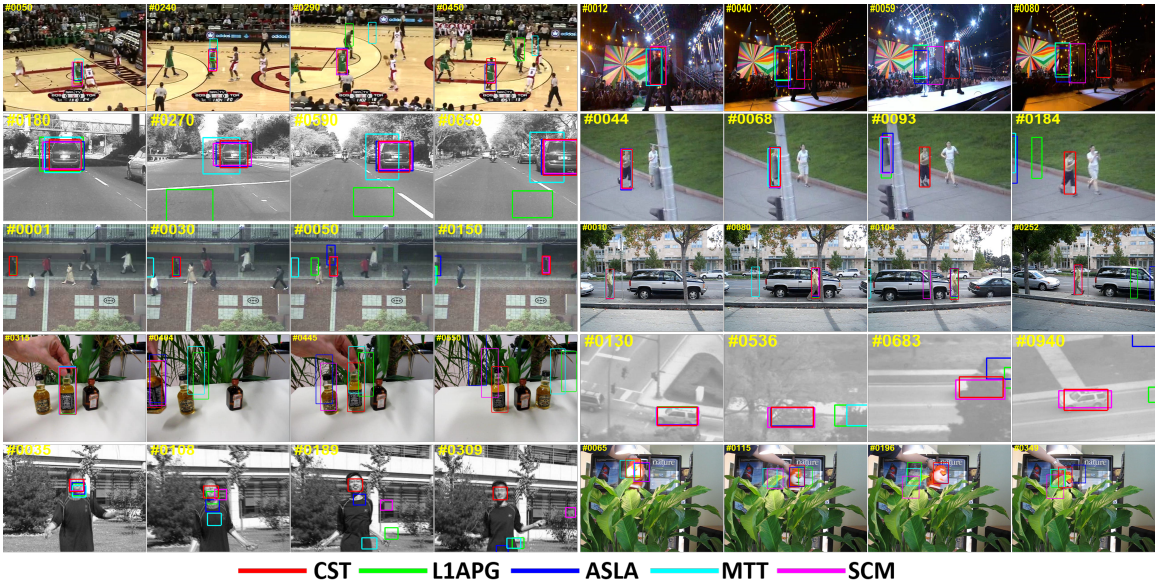

Figure 5.8: Tracking results of the top 5 sparse trackers (denoted in different colors and lines) in our evaluation on 10 challenging sequences (from left to right and top to down are basketball, singer2, car4, jogging-1, subway, david3, liquor, suv, jumping, and tiger1 respectively).

5.4.5 Sparse Tracking Evaluation

We evaluate the proposed algorithm on the benchmark with comparisons to the top 4 sparse trackers in [3], namely SCM [72], ASLA [61], L1APG [60], and MTT [12]. The details of the 4 trackers in the benchmark evaluation can be found in [3]. We report

the results using average OS, DP, CLE, and FPS over all sequences in Table 5.1, and present the results in OPE using the distance precision and overlap success rate in Figure 5.6, attribute-based evaluation in Figure 5.7, and qualitative comparison in Figure 5.8. Table 5.1 shows that our algorithm outperforms state-of-the-art sparse trackers. Among the sparse trackers in the literature, the proposed CST method achieves the best results with an average OS of 68.2%, DP of 77.7%, and CLE of 40.4 pixels. Compared with the second-best method (SCM), CST registers a performance improvement of 6.6% in average OS and 12.8% in term of average DP. In term of average CLE, CST has about a 13.7 pixel improvement over SCM. Moreover, CST achieves much higher frame rate than the second-best performing sparse tracker. Note that our tracker can be made even faster with some code optimization.

Figure 5.6 contains the precision and success plots illustrating the mean distance and overlap precision over all the 50 sequences. In both precision and success plots, our approach shows the best results and significantly outperforms the best existing sparse method (SCM). In Figure 5.7, we analyze tracking performance based on some tracking attributes of the video sequences [3]. Note that the benchmark annotates 11 such attributes to describe the different challenges in the tracking problem, e.g., occlusions or out-of-view. These attributes are useful for analyzing the performance of trackers in different aspects. Due to space constraints, we present the success and precision plots of OPE for 3 attributes in Figure 5.7 and more results can be found in the **Appendix**. We note that the proposed tracking method performs well in dealing with challenging factors including fast motion, occlusion, and out of view. In Figure 5.8, we show a qualitative comparison among the sparse trackers on 10 challenging sequences. The SCM tracker performs well in handling scale change (*basketball* and *car4*). However, it drifts when target undergo heavy occlusion (*jogging-1*) and fast motion (*jumping* and *tiger1*). L1APG is the most similar sparse method to CST, as they both solve an ℓ_1 minimization problem but with different optimization tech-

niques (APG vs. ADMM). It fails to handle fast motion (*jumping* and *tiger1*), and background clutter (*singer2*), where using only grayscale intensity is less effective in discriminating targets from the cluttered background. MTT and ASLA do not perform well with partial occlusion (*suv* and *subway*). Overall, the proposed CST tracker performs very well in tracking objects on these challenging sequences. In addition, we compare the center location error frame-by-frame on the 10 sequences, which shows that our method performs well against existing trackers. Due to the space limitation, the results can be found in the **Appendix**.

Discussion: The above results clearly demonstrate the effectiveness and efficiency of our proposed CST tracker. Here, we highlight the following conclusions among the existing sparse trackers. **(1)** The circulant structure of target temples can improve tracking performance. L1APG and CST-Color have similar objective functions with grayscale intensity features. The differences are that the L1APG solves the ℓ_1 minimization problem in the spatial domain, but we construct the circulant matrix as target templates and solve the problem in the Fourier domain. Compared with the L1APG tracker, CST registers a 4.8% and 2.7% improvement in average DP and OS, respectively, while maintaining very comparable runtimes. **(2)** The circulant structure of target temples can make the HOG feature feasible in sparse trackers, and improve tracking performance significantly. In traditional sparse trackers, it is computationally infeasible to adopt HOG features due to the high computational cost. For example, given a target object with 50×50 pixels, if we adopt HOG feature with 31 bins and vectorize the image patch, the resulting representation has $77,500$ dimensions. As a result, the trivial templates have $77,500$ elements, which significantly increases the complexity. However, owing to the circulant structure used in CST, the optimization can be solved efficiently by using 2D image patches with multiple channels in the Fourier domain. Moreover, comparing CST-HOG and CST-Color shows that HOG can significantly improve tracking performance. **(3)** Compared with the

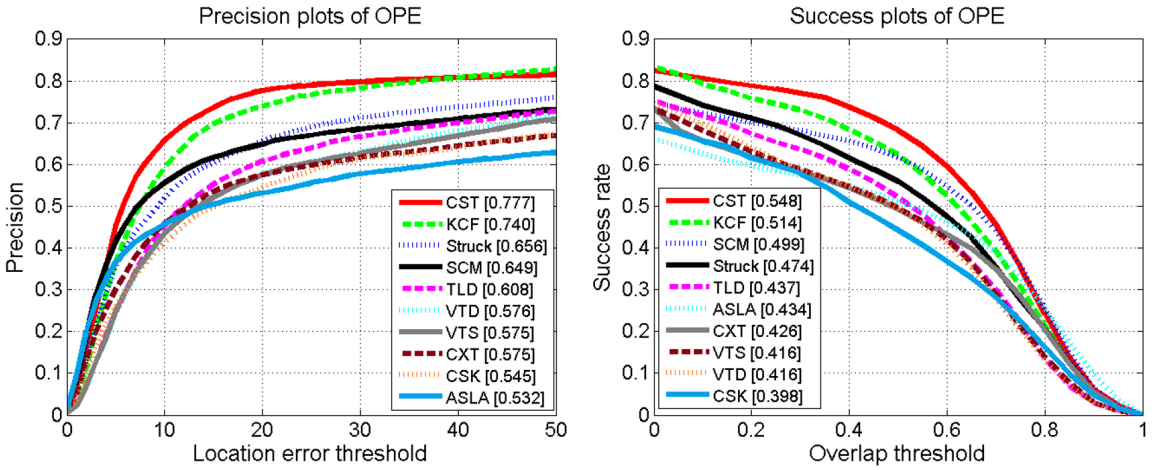

Figure 5.9: Precision and success plots over all the 50 sequences using OPE among 29 trackers in [3]. The proposed CST method performs favorably against the state-of-the-art trackers.

KCF tracker, our CST trackers register a 3.7% and 3.4% improvement in terms of average DP and OS, respectively. This is probably due to the proposed sparse model with particle filtering, which can better handle partial occlusion and fast motion.

5.4.6 Comparison with State-of-the-Art

We evaluate our CST tracker on the tracking benchmark with comparisons to 29 trackers in [3], whose details can be found in [3]. We report the precision and success plots in Figure 5.9, thus, illustrating the mean distance and overlap precision over all the 50 sequences. In both precision and success plots, CST registers the best performance among all trackers and significantly outperforms the best existing sparse tracking method (SCM).

For a more thorough evaluation, we also include in the comparison the following very recent trackers with their corresponding (DP, OS, FPS) results: MEEM (83.5% , 57.6%, 6.2) [37], TGPR (71.4% , 51.5%, 0.36) [77], RPT (81.9% , 57.9%, 3.1) [78], MUSTer (86.5%, 64.1%, 0.34) [71], and DSST (73.7%, 55.4%, 32.7) [23]. Among these trackers, the proposed CST is better than TGPR, and achieves comparable

performance as DSST. The MUSTer, MEEM, and RPT methods achieve better performance than our CST method. Compared with the best existing MUSTer tracker, sparse trackers still have room for improvement; however, our proposed tracker is much faster. Moreover, the short-term and long-term strategy used in MUSTer [71], as well as, the multiple expert framework proposed in MEEM [37] are generic schemes that can be also be adopted in sparse trackers to further improve their precision. For example, CST can exploit a combination of short-term and a long-term dictionaries to obtain much better performance.

5.5 Conclsuion

In this chapter, we propose a novel circulant sparse appearance model for object tracking under the particle filter framework. Due to the circulant structure property of target templates, the proposed tracker can make use of HOG feature, and effectively refine sampled particles to dramatically reduce the number of particles needed for tracking. Moreover, the optimization can be efficiently solved in Fourier domain. Experimental results compared with several state-of-the-art methods on challenging sequences demonstrate the effectiveness and robustness of the proposed algorithm.

Chapter 6

Conclusion

This thesis is devoted to the study of possible ways in improving existing generic object trackers either by building upon previous trackers to enhance upon or by proposing new tracker framework all together. All trackers submitted and accepted to the conferences achieved the state-of-art-performance in the time of submission. Trackers discussed in this thesis include one RGBD based tracker and three RGB based trackers. As part of the proposed RGBD tracker tracker, a major drawback was found in the current available dataset which is incorrect synchronization and registration. A powerful effective method for synchronization and registration was proposed to handle these nuisances. All the RGB trackers proposed are correlation based. Each tracker proposed was mainly to handle different drawbacks that include multi-template update, fast motion, motion blur, and occlusion. Also a new tracker framework is proposed in which the properties of correlation trackers were leveraged by coupling it in the sparse representation-particle-filter framework.

REFERENCES

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

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

[3] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Computer vision and pattern recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2411–2418.

[4] A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, July 2014.

[5] S. Song and J. Xiao, "Tracking revisited using rgbd camera: Unified benchmark and baselines," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 233–240.

[6] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 9, pp. 1834–1848, 2015.

[7] 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.

[8] T. Poggio and G. Cauwenberghs, "Incremental and decremental support vector machine learning," *Advances in neural information processing systems*, vol. 13, p. 409, 2001.

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

[10] X. Mei and H. Ling, "Robust visual tracking using &# x2113; 1 minimization," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 1436–1443.

[11] 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.

[12] ——, "Robust visual tracking via multi-task sparse learning," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2042–2049.

[13] ——, "Low-rank sparse learning for robust visual tracking," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 470–484.

[14] T. Zhang, B. Ghanem, S. Liu, C. Xu, and N. Ahuja, "Robust visual tracking via exclusive context modeling," *Cybernetics, IEEE Transactions on*, vol. 46, no. 1, pp. 51–63, 2016.

[15] T. Zhang, B. Ghanem, C. Xu, and N. Ahuja, "Object tracking by occlusion detection via structured sparse learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, pp. 1033–1040.

[16] T. Zhang, S. Liu, C. Xu, S. Yan, B. Ghanem, N. Ahuja, and M.-H. Yang, "Structural sparse tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 150–158.

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

[18] S. Avidan, "Ensemble tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 2, pp. 261–271, 2007.

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

[20] 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.

[21] ——, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 702–715.

[22] D. S. Bolme, J. R. Beveridge, B. Draper, Y. M. Lui *et al.*, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2544–2550.

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

[24] T. Liu, G. Wang, and Q. Yang, "Real-time part-based visual tracking via adaptive correlation filters," *Intelligence*, p. 2345390, 2015.

[25] H. Kiani Galoogahi, T. Sim, and S. Lucey, "Correlation filters with limited boundaries," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4630–4638.

[26] B. Wu, S. Lyu, B.-G. Hu, and Q. Ji, "Simultaneous clustering and tracklet linking for multi-face tracking in videos," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2856–2863.

[27] A. Mahalanobis, B. Vijaya Kumar, and D. Casasent, "Minimum average correlation energy filters," *Applied Optics*, vol. 26, no. 17, pp. 3633–3640, 1987.

[28] B. Vijaya Kumar, "Minimum-variance synthetic discriminant functions," *JOSA A*, vol. 3, no. 10, pp. 1579–1584, 1986.

[29] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5388–5396.

[30] M. Camplani, S. Hannuna, M. Mirmehdi, D. Damen, A. Paiement, L. Tao, T. Burghardt, and U. Bristol, "Real-time rgb-d tracking with depth scaling kernelised correlation filters and occlusion handling," in *British Machine Vision conference (BMVC)*, 2015.

[31] Y. Xiang, C. Song, R. Mottaghi, and S. Savarese, "Monocular multiview object tracking with 3d aspect parts," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 220–235.

[32] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3d tracking of hand articulations using kinect." in *BMVC*, vol. 1, no. 2, 2011, p. 3.

[33] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, "Realtime and robust hand tracking from depth," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 1106–1113.

[34] M. Luber, L. Spinello, and K. O. Arras, "People tracking in rgb-d data with on-line boosted target models," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 3844–3849.

[35] W. Choi, C. Pantofaru, and S. Savarese, "Detecting and tracking people using an rgb-d camera via multiple detector fusion," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1076–1083.

[36] 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*. IEEE, 2014, pp. 1258–1265.

[37] J. Zhang, S. Ma, and S. Sclaroff, "Meem: Robust tracking via multiple experts using entropy minimization," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 188–203.

[38] S. O. H. S. Y. L. Kourosh Meshgi, Shin-ichi Maeda and S. Ishii, "Occlusion aware particle filter tracker to handle complex and persistent occlusions," *Computer Vision and Image Understanding*, 2015 [Under Review].

[39] G. M. García, D. A. Klein, J. Stückler, S. Frintrop, and A. B. Cremers, *Adaptive multi-cue 3D tracking of arbitrary objects*. Springer, 2012.

[40] Z. Hong, X. Mei, D. Prokhorov, and D. Tao, "Tracking via robust multi-task multi-view joint sparse representation," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 649–656.

[41] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *Signal Processing, IEEE Transactions on*, vol. 54, no. 11, pp. 4311–4322, 2006.

[42] B. K. Horn and B. G. Schunck, "Determining optical flow," in *1981 Technical symposium east*. International Society for Optics and Photonics, 1981, pp. 319–331.

[43] T. Brox, C. Bregler, and J. Malik, "Large displacement optical flow," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 41–48.

[44] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojíř, G. Fernandez, A. Lukežič, A. Dimitriev *et al.*, "The visual object tracking vot2014 challenge results," in *Computer Vision-ECCV 2014 Workshops*. Springer, 2014, pp. 191–217.

[45] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Computer Vision-ECCV 2014 Workshops*. Springer, 2014, pp. 254–265.

[46] V. N. Boddeti, T. Kanade, and B. Kumar, "Correlation filters for object alignment," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2291–2298.

[47] "The vot 2015 evaluation kit. http://www.votchallenge.net."

[48] 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.

[49] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 263–270.

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

[51] M. Danelljan, F. Khan, M. Felsberg, and J. Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1090–1097.

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

[53] A. Bibi and B. Ghanem, "Multi-template scale-adaptive kernelized correlation filters," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 50–57.

[54] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski, "Robust and fast collaborative tracking with two stage sparse optimization," in *ECCV*, 2010.

[55] B. Liu, J. Huang, L. Yang, and C. Kulikowski, "Robust visual tracking with local sparse appearance model and k-selection," in *CVPR*, 2011.

[56] H. Li, C. Shen, and Q. Shi, "Real-time visual tracking with compressed sensing," in *CVPR*, 2011.

[57] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Minimum error bounded efficient l1 tracker with occlusion detection," in *CVPR*, 2011.

[58] X. Mei and H. Ling, "Robust Visual Tracking and Vehicle Classification via Sparse Representation," *TPAMI*, vol. 33, no. 11, pp. 2259–2272, 2011.

[59] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *ECCV*, 2012.

[60] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l_1 tracker using accelerated proximal gradient approach," in *CVPR*, 2012.

[61] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *CVPR*, 2012.

[62] Z. Hong, X. Mei, D. Prokhorov, and D. Tao, "Tracking via robust multi-task multi-view joint sparse representation," in *ICCV*, 2013.

[63] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l1 tracker using accelerated proximal gradient approach," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1–8.

[64] Y. Pang and H. Ling, "Finding the best from the second bests - inhibiting subjective bias in evaluation of visual tracking algorithms," in *ICCV*, 2013.

[65] A. Smeulder, D. Chu, R. Cucchiara, S. Calderara, A. Deghan, and M. Shah, "Visual tracking: an experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2013.

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

[67] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *IJCV*, pp. 63–84, 1998.

[68] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental Learning for Robust Visual Tracking," *IJCV*, vol. 77, no. 1, pp. 125–141, 2008.

[69] H. Grabner, M. Grabner, and H. Bischof, "Real-Time Tracking via On-line Boosting," in *BMVC*, 2006.

[70] 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.

[71] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *CVPR*, 2015, pp. 749–758.

[72] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model." in *CVPR*, 2012, pp. 1838–1845.

[73] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press, New York, NY, USA*, 2004.

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

[75] A. Y. Yang, Z. Zhou, A. Ganesh, S. Sastry, and Y. Ma, "Fast ℓ_1-minimization algorithms for robust face recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 8, pp. 3234–3246, 2013.

[76] R. M. Gray, "Toeplitz and circulant matrices: A review," in *Now Publishers*, 2006.

[77] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with gaussian process regression," 2014.

[78] Y. Li, J. Zhu, and S. C. H. Hoi, "Reliable patch trackers: Robust visual tracking by exploiting reliable patches." in *CVPR*, 2015, pp. 353–361.

[79] A. Y. Yang, Z. Zhou, A. G. Balasubramanian, S. S. Sastry, and Y. Ma, "Fast-minimization algorithms for robust face recognition," *Image Processing, IEEE Transactions on*, vol. 22, no. 8, pp. 3234–3246, 2013.

APPENDICES

A Treatment of the Formulation Proposed in Chapter 2

A.1 Effects of Incorrect Calibration to 2D RGB/Depth Image Registration

In this part of the appendix, we show that 3D perturbation of a point cloud due to an in-correct registration causes a 2D translation in the image plane after projection that is a function of depth and 3D location. $\mathbf{p}_d$, and $\mathbf{P}_d$ denote the 2D image plane locations and the 3D locations in the depth camera reference respectively where $\mathbf{P}_d = [X_1\ Y_1\ Z_1]^T$. Similarly, $\mathbf{p}_{rgb}$, and $\mathbf{P}_{rgb}$ which are with respect to the RGB camera reference. $\mathbf{R}$, and $\mathbf{T}$ are the extrinsic calibration parameters, while $\mathbf{K}_d$, and $\mathbf{K}_{rgb}$ are the depth and RGB intrinsic parameters respectively.

In our formulation to the problem in chapter 2, we assume for simplicity that the in-accuracy of registration is in the extrinsic parameters only, specifically the translational part. Therefore, our model is given by $\mathbf{P}_{rgb} = \mathbf{P_d} + \Delta T$ where the RGB image plane projection is given by $\mathbf{p}_{rgb} = \frac{1}{z_2}\mathbf{K_{rgb}}[\mathbf{P}_d + \Delta T]$, while if the registration is done correctly then $\tilde{\mathbf{p}}_{rgb} = \frac{1}{z_1}\mathbf{K_{rgb}}\mathbf{P}_d$. Then perturbation in the 2D translation is given by:

$$\delta \mathbf{t}_{rgb} = \mathbf{K_{rgb}}[\mathbf{P_d}\frac{Z_1 - Z_2}{Z_1 Z_2} - \frac{1}{Z_2}\Delta T] = \begin{pmatrix} \alpha\delta x \\ \alpha\delta y \\ \alpha \end{pmatrix}$$

$$\delta x = \frac{\mathbf{f}_x}{\alpha}[X_1\frac{Z_1 - Z_2}{Z_1 Z_2} - \frac{\Delta X}{z_2}] + \frac{\mathbf{o}_x}{\alpha}[\frac{Z_1 - Z_2}{Z_2} - \frac{\Delta Z}{Z_2}]$$

$$\delta y = \frac{\mathbf{f}_y}{\alpha}[Y_1\frac{Z_1 - Z_2}{Z_1 Z_2} - \frac{\Delta Y}{z_2}] + \frac{\mathbf{o}_y}{\alpha}[\frac{Z_1 - Z_2}{Z_2} - \frac{\Delta Z}{Z_2}]$$

B Treatment of the Formulation Proposed in Chapter 3

In this part of the appendix, we followup our discussion from chapter 3 on the solution to the following problem:

$$\begin{aligned}\min_{w} f(w) &= \min_{\mathbf{w}} \frac{1}{2}||\Phi_1\mathbf{w} - \mathbf{y}||_2^2 + \frac{\lambda}{2}||\mathbf{w}||_2^2 + \frac{\mu}{2}||\mathbf{w} - \mathbf{b}||_2^2 \\ &= \min_{\mathbf{w}} \frac{1}{2}\sum_i (\mathbf{w}^T\phi(\mathbf{x_i}) - y_i)^2 + \frac{\lambda}{2}||\mathbf{w}||_2^2 + \frac{\mu}{2}||\mathbf{w} - \mathbf{b}||_2^2\end{aligned} \tag{B.1}$$

B.1 Dual Formulation

Since B.1 is convex quadratic, a stationary point is necessary and sufficient for global optimality.

$$\begin{aligned}\nabla_w f(w) &= \sum_i (\mathbf{w}^T\phi(\mathbf{x_i}) - y_i)\phi(\mathbf{x_i}) + \lambda\mathbf{w} + \mu(\mathbf{w} - \mathbf{b}) = 0 \\ &\Rightarrow \sum_i (\mathbf{w}^T\phi(\mathbf{x_i}) - y_i)\phi(\mathbf{x_i}) - \mu\mathbf{b} = -(\lambda + \mu)\mathbf{w} \\ \mathbf{w} &= -\frac{1}{\lambda + \mu}\sum_i (\mathbf{w}^T\phi(\mathbf{x_i}) - y_i)\phi(x_i) + \frac{\mu}{\lambda + \mu}\mathbf{b}\end{aligned} \tag{B.2}$$

Let the dual variables a_i be $a_i = -\frac{1}{\lambda+\mu}(\mathbf{w}^T\phi(\mathbf{x_i}) - y_i)$ and $k = \frac{\mu}{\lambda+\mu}$ Then, $\mathbf{w} = \Phi_1^T\mathbf{a}_1 + k\mathbf{b}$

Therefore, the dual objective and its solution is given as follows:

$$
\begin{aligned}
f(a) &= \frac{1}{2}\sum_i((\Phi_1^T\mathbf{a}_1 + k\mathbf{b})^T\phi(\mathbf{x_i}) - y_i)^2 + \frac{\lambda}{2}||\Phi_1^T\mathbf{a}_1 + k\mathbf{b}||_2^2 + \frac{\mu}{2}||\Phi_1^T\mathbf{a}_1 + k\mathbf{b} - \mathbf{b}||_2^2 \\
&= \frac{1}{2}\sum_i(\mathbf{a}_1^T\Phi_1\phi(\mathbf{x_i}) + k\mathbf{b}^T\phi(\mathbf{x_i}) - y_i)^2 + \frac{\lambda}{2}||\Phi_1^T\mathbf{a}_1 + k\mathbf{b}||_2^2 + \frac{\mu}{2}||\Phi_1^T\mathbf{a}_1 + (k-1)\mathbf{b}||_2^2 \\
&= \frac{1}{2}||\Phi_1\Phi_1^T\mathbf{a}_1 + k\Phi_1\mathbf{b} - \mathbf{y}||_2^2 + \frac{\lambda}{2}||\Phi_1^T\mathbf{a}_1 + k\mathbf{b}||_2^2 + \frac{\mu}{2}||\Phi_1^T\mathbf{a}_1 + (k-1)\mathbf{b}||_2^2
\end{aligned} \tag{B.3}
$$

$$
\begin{aligned}
\nabla_a f(a) &= \Phi_1\Phi_1^T(\Phi_1\Phi_1^T\mathbf{a}_1 + k\Phi_1\mathbf{b} - \mathbf{y}) + \lambda\Phi_1(\Phi_1^T\mathbf{a}_1 + k\mathbf{b}) + \mu\Phi_1(\Phi_1^T\mathbf{a}_1 + (k-1)\mathbf{b}) = 0 \\
&= (\Phi_1\Phi^T\Phi_1\Phi_1^T + (\lambda+\mu)\Phi_1\Phi_1^T)\mathbf{a}_1 + (k\Phi_1\Phi_1^T\Phi_1 + (\lambda k + \mu(k-1))\Phi_1)\mathbf{b} - \Phi_1\Phi_1^T\mathbf{y} \\
&\Rightarrow (\Phi_1\Phi_1^T\Phi_1\Phi_1^T + (\lambda+\mu)\Phi_1\Phi_1^T)\mathbf{a}_1 = -(k\Phi_1\Phi_1^T\Phi_1 + (\lambda k + \mu(k-1))\Phi_1)\mathbf{b} + \Phi_1\Phi_1^T\mathbf{y} \\
&\Rightarrow (\Phi_1\Phi_1^T + (\lambda+\mu)\mathbf{I})\mathbf{a}_1 = -(k\Phi_1 + (\lambda k + \mu(k-1))(\Phi_1\Phi_1^T)^{-1}\Phi_1)\mathbf{b} + \mathbf{y}
\end{aligned} \tag{B.4}
$$

The assumption made here is that the FFT of the base tempalte used doesn't have an exact zero in it. This leads to having $\Phi_1\Phi_1^T$ non-singular. Since $\mathbf{b} = \Phi_2^T\mathbf{a}_2$ for the first frame (which is the primal dual transformation of the standard KCF) then, $\Phi_1 b = \Phi_1\Phi_2^T\mathbf{a}_2 = \hat{\mathbf{b}}$. Then equation B.4 becomes:

$$
(\Phi_1\Phi_1^T + (\lambda+\mu)\mathbf{I})\mathbf{a}_1 = -(k\mathbf{I} + (\lambda k + \mu(k-1))(\Phi_1\Phi_1^T)^{-1})\hat{\mathbf{b}} + \mathbf{y} \tag{B.5}
$$

Now all is left is to compute both $\hat{\mathbf{b}}$ and $(\Phi_1\Phi_1^T)^{-1}\hat{\mathbf{b}}$ very efficiently. First, evaluating $\hat{\mathbf{b}} = (\Phi_1\Phi_2^T)\mathbf{a}_2$, I'll denote by $\tilde{\mathbf{b}}$ to be the Fourier transform of $\mathbf{b}$. Now since it can be easily shown that $\Phi_1\Phi_2^T$ is circulant, then it can be diagonalized using FFT

as follows:

$$\hat{\mathbf{b}} = (\Phi_1\Phi_2^T)\mathbf{a}_2 = \mathbf{F}diag(\hat{k}^{x_2x_1})\mathbf{F}^H\mathbf{a}_2 \tag{B.6}$$

$$= \mathbf{F}diag(\hat{k}^{x_2x_1})\hat{\mathbf{a}}_2^* \tag{B.7}$$

$$= \mathbf{F}(\hat{k}^{x_2x_1} \odot \hat{\mathbf{a}}_2^*) \tag{B.8}$$

$$(\Phi_1\Phi_1^T)^{-1}\tilde{\mathbf{b}} = (\mathbf{F}diag(\hat{k}^{x_1x_1})\mathbf{F}^H)^{-1}\hat{\mathbf{b}} \tag{B.9}$$

$$= \mathbf{F}diag^{-1}(\hat{k}^{x_1x_1})\mathbf{F}^H\hat{\mathbf{b}} \tag{B.10}$$

$$= \mathbf{F}diag^{-1}(\hat{k}^{x_1x_1})(\hat{k}^{x_2x_1} \odot \hat{\mathbf{a}}_2^*) \tag{B.11}$$

$$= \mathbf{F}((\hat{k}^{x_1x_1})^{-1} \odot \hat{k}^{x_2x_1} \odot \hat{\mathbf{a}}_2^*) \tag{B.12}$$

B.2 Efficient Computation of Data Term in the Objective Function

Here, we show a more detailed derivation for how to efficiently compute the regression cost that will be used in the stopping criterion for the alternating fixed-point optimization of Eq B.1(The results can be extended to any dimensional signal).

$$\begin{aligned}||\Phi_2\mathbf{w}_2 - \mathbf{y}||_2^2 &= (\Phi_2\mathbf{w}_2 - \mathbf{y})^H(\Phi_2\mathbf{w}_2 - \mathbf{y}) = \mathbf{w}_2^H\Phi_2^H\Phi_2\mathbf{w}_2 - \mathbf{w}_2^H\Phi_2^H\mathbf{y} - \mathbf{y}^H\Phi_2\mathbf{w}_2 + \mathbf{y}^H\mathbf{y} \\ &= \mathbf{a}_2^H\Phi_2\Phi_2^H\Phi_2\Phi_2^H\mathbf{a}_2 - \mathbf{a}_2^H\Phi_2\Phi_2^H\mathbf{y} - \mathbf{y}^H\Phi_2\Phi_2^H\mathbf{a}_2 + \mathbf{y}^H\mathbf{y} \\ &= \mathbf{a}_2^H\mathbf{F}(\hat{\mathbf{k}}^{22} \odot \hat{\mathbf{k}}^{22} \odot \hat{\mathbf{a}}_2^*) - \mathbf{a}_2^H\mathbf{F}(\hat{\mathbf{k}}^{22} \odot \hat{\mathbf{y}}_2^*) - \mathbf{y}^H\mathbf{F}(\hat{\mathbf{k}}^{22} \odot \hat{\mathbf{a}}^*) + \mathbf{y}^H\mathbf{y}\end{aligned} \tag{B.13}$$

C Treatment of the Formulation Proposed in Chapter 4

In this part of the appendix, we followup our discussion from chapter 4 on the solution to the following problem:

$$\min_{\mathbf{w},\mathbf{y}} \quad ||\mathbf{A}\mathbf{w} - \mathbf{y}||_2^2 + \lambda_1||\mathbf{w}||_2^2 + \lambda_2||\mathbf{y} - \mathbf{y}_0||_2^2 \tag{C.1}$$

The problem is convex quadratic and a stationary point is necessary and sufficient for global optimality. The following sections will discuss how Problem C.1 is solved in the primal domain, dual domain, and for both single and multiple templates along with the formula used to generate the response map, whose maximum value determines the current detection. Lastly, we discuss a one way of incorporating SRDCF with our proposed target adaptive framework.

C.1 Solution to Problem C.1 in the Primal Domain

C.1.1 Using a Single Template

Problem C.1 can be rewritten in terms of $\mathbf{z}$ with $\mathbf{z}^T = \begin{bmatrix}\mathbf{w}^T & \mathbf{y}^T\end{bmatrix}$:

$$f(\mathbf{z}) = || \begin{bmatrix}\mathbf{A} & -\mathbf{I}\end{bmatrix} \mathbf{z}||_2^2 + \lambda_1|| \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix} \mathbf{z}||_2^2 + \lambda_2|| \begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix} \mathbf{z} - \mathbf{y}_0||_2^2 \quad , \tag{C.2}$$

where $\mathbf{w} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{y} \in \mathbb{R}^n$, $\mathbf{y}_0 \in \mathbb{R}^n$, $\mathbf{z} \in \mathbb{R}^{2n}$, then:

$$\nabla_{\mathbf{z}} f(\mathbf{z}) = \begin{bmatrix} \mathbf{A}^T\mathbf{A} & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{I} \end{bmatrix} \mathbf{z} + \lambda_1 \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{z} + \lambda_2 \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{z} - \lambda_2 \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \mathbf{y}_0 = 0$$

$$\nabla_{\mathbf{z}} f(\mathbf{z}) = \begin{bmatrix} \mathbf{A}^T\mathbf{A} + \lambda_1\mathbf{I} & -\mathbf{A}^T \\ -\mathbf{A} & (1+\lambda_2)\mathbf{I} \end{bmatrix} \mathbf{z} = \lambda_2 \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \mathbf{y}_0$$

$$\begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix} \begin{bmatrix} diag(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^* + \lambda_1) & -diag(\hat{\mathbf{a}}_1^*) \\ -diag(\hat{\mathbf{a}}_1) & diag(1+\lambda_2) \end{bmatrix} \begin{bmatrix} \mathbf{F}^H & \mathbf{0} \\ \mathbf{0} & \mathbf{F}^H \end{bmatrix} \mathbf{z} = \lambda_2 \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \mathbf{y}_0$$

$$\begin{bmatrix} diag(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^* + \lambda_1) & -diag(\hat{\mathbf{a}}_1^*) \\ -diag(\hat{\mathbf{a}}_1) & diag(1+\lambda_2) \end{bmatrix} \begin{bmatrix} \hat{\mathbf{w}}^* \\ \hat{\mathbf{y}}^* \end{bmatrix} = \lambda_2 \begin{bmatrix} \mathbf{0} \\ \mathbf{F}^H \end{bmatrix} \mathbf{y}_0$$

$$\begin{bmatrix} \hat{\mathbf{w}}^* \\ \hat{\mathbf{y}}^* \end{bmatrix} = \lambda_2 \begin{bmatrix} diag(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^* + \lambda_1) & -diag(\hat{\mathbf{a}}_1^*) \\ -diag(\hat{\mathbf{a}}_1) & diag(1+\lambda_2) \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{F}^H \end{bmatrix} \mathbf{y}_0$$

Note that the inverse lemma states:

$$\begin{bmatrix} \mathbf{B} & \mathbf{N} \\ \mathbf{V} & \mathbf{C} \end{bmatrix}^{-1} = \begin{bmatrix} (\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1} & -(\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1}\mathbf{N}\mathbf{C}^{-1} \\ -\mathbf{C}^{-1}\mathbf{V}(\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1} & \mathbf{C}^{-1}\mathbf{V}(\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1}\mathbf{N}\mathbf{C}^{-1} + \mathbf{C}^{-1} \end{bmatrix} \tag{C.3}$$

Then:

$$\begin{aligned} \mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V} &= diag(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1 + \lambda_1) - diag(\hat{\mathbf{a}_1^*})diag^{-1}(1+\lambda_2)diag(\hat{\mathbf{a}_1}) \\ (\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1} &= diag\Big(\frac{1+\lambda_2}{\lambda_2(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^*) + \lambda_1(1+\lambda_2)}\Big) \\ -(\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1}\mathbf{N}\mathbf{C}^{-1} &= diag\Big(\frac{\hat{\mathbf{a}}_1^*}{\lambda_2(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^*) + \lambda_1(1+\lambda_2)}\Big) \end{aligned} \tag{C.4}$$

Since:

$$\hat{\mathbf{w}}^* = -\lambda_2(\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1}\mathbf{N}\mathbf{C}^{-1}\mathbf{F}^H\mathbf{y}_o \tag{C.5}$$

Then:

$$\hat{\mathbf{w}}^* = \frac{\lambda_2(\hat{\mathbf{a}}_1^* \odot \hat{\mathbf{y}}_o^*)}{\lambda_2(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^*) + \lambda_1(1+\lambda_2)} \Rightarrow \quad \hat{\mathbf{w}} = \frac{\lambda_2(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{y}}_o)}{\lambda_2(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{a}}_1^*) + \lambda_1(1+\lambda_2)} \tag{C.6}$$

Detection Formula with a Single Template

As for the detection formula in the primal domain, we consider a new test sample $\mathbf{u}$. For detection, we construct all the circular shifts of $\mathbf{u}$ in matrix $\mathbf{U}$. Therefore, the response map on the test sample is:

$$\mathbf{T}(\mathbf{u}) = \mathbf{U}\mathbf{w} = \mathbf{F}diag(\hat{\mathbf{u}})\mathbf{F}^H\mathbf{w} \Rightarrow \hat{\mathbf{T}}^*(\mathbf{u}) = \hat{\mathbf{u}} \odot \hat{\mathbf{w}}^* \quad \Rightarrow \hat{\mathbf{T}}(\mathbf{u}) = \hat{\mathbf{u}}^* \odot \hat{\mathbf{w}} \tag{C.7}$$

C.1.2 Using Multiple Templates

Following is the derivation of the solution for the multiple template case in primal domain, i.e. when $\tilde{\mathbf{A}} \in \mathbb{R}^{kn \times n}$, k is the total number of templates, and $\tilde{\mathbf{A}}^T = \begin{bmatrix} \mathbf{A}_1^T & \mathbf{A}_2^T & \dots & \mathbf{A}_k^T \end{bmatrix}$ and $\tilde{\mathbf{I}}^T = \begin{bmatrix} \mathbf{I}_1 & \mathbf{I}_2 & \dots & \mathbf{I}_k \end{bmatrix}$. Then:

$$f(\mathbf{z}) = || \begin{bmatrix} \tilde{\mathbf{A}} & -\tilde{\mathbf{I}} \end{bmatrix} \mathbf{z}||_2^2 + \lambda_1 || \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{z}||_2^2 + \lambda_2 || \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{z} - \mathbf{y}_0||_2^2$$

$$\nabla f(\mathbf{z}) = \begin{bmatrix} \tilde{\mathbf{A}}^T\tilde{\mathbf{A}} & -\tilde{\mathbf{A}}^T\tilde{\mathbf{I}} \\ -\tilde{\mathbf{I}}^T\tilde{\mathbf{A}} & \tilde{\mathbf{I}}^T\tilde{\mathbf{I}} \end{bmatrix} \mathbf{z} + \lambda_1 \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{z} + \lambda_2 \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{z} - \lambda_2 \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \mathbf{y}_0 = 0$$

$$\nabla f(\mathbf{z}) = \underbrace{\begin{bmatrix} \tilde{\mathbf{A}}^T\tilde{\mathbf{A}} + \lambda_1\mathbf{I} & -\tilde{\mathbf{A}}^T\tilde{\mathbf{I}} \\ -\tilde{\mathbf{I}}^T\tilde{\mathbf{A}} & (k+\lambda_2)\mathbf{I} \end{bmatrix}}_{\Gamma} \mathbf{z} = \lambda_2 \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \mathbf{y}_0$$

Now, note the following: $\tilde{\mathbf{A}}^T\tilde{\mathbf{A}} = \sum_{i=1}^{k} \mathbf{A}_i^T\mathbf{A}_i$, $\tilde{\mathbf{I}}^T\tilde{\mathbf{I}} = \sum_{i=1}^{k} \mathbf{I}_i^T\mathbf{I}_i = k\mathbf{I}$, and $\tilde{\mathbf{A}}^T\tilde{\mathbf{I}} = \sum_{i=1}^{k} \mathbf{A}_i^T$, and that $\tilde{\mathbf{I}}^T\tilde{\mathbf{A}} = \sum_{i=1}^{k} \mathbf{A}_i$. It is clear that $\tilde{\mathbf{A}}^T\tilde{\mathbf{A}}$ and $\tilde{\mathbf{I}}^T\tilde{\mathbf{I}}$ are circulant which is a sum of circulant matrices. Therefore, the matrix Γ is block wise circulant such that $\Gamma \in \mathbb{R}^{2n\times 2n}$. Similar to the single template case, we have:

$$\begin{bmatrix} \tilde{\mathbf{w}}^* \\ \tilde{\mathbf{y}}^* \end{bmatrix} = \lambda_2 \begin{bmatrix} \sum_{i=1}^{k} diag(\hat{\mathbf{a}}_{1i} \odot \hat{\mathbf{a}}_{1i}^* + \frac{\lambda_1}{k}) & -\sum_{i=1}^{k} diag(\hat{\mathbf{a}}_{1i}^*) \\ -\sum_{i=1}^{k} diag(\hat{\mathbf{a}}_{1i}) & diag(k+\lambda_2) \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{F^H} \end{bmatrix} \mathbf{y}_0 \tag{C.8}$$

Using the inverse lemma in Eq C.3 on Γ, we get:

$$\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V} = diag(\sum_{i}^{k}(\hat{\mathbf{a}}_{1i} \odot \hat{\mathbf{a}}_{1i}) + \lambda_1) - \frac{diag(\sum_i^k \hat{\mathbf{a}}_{1i}^*)diag(\sum_i^k(\hat{\mathbf{a}}_{1i}))}{(k+\lambda_2)} \tag{C.9}$$

$$\begin{aligned} &-(\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V})^{-1}\mathbf{N}\mathbf{C}^{-1} = \\ &diag\left(\frac{\sum_i^k \hat{\mathbf{a}}_{1i}^*}{(k+\lambda_2)(\sum_i^k \hat{\mathbf{a}}_{1i} \odot \hat{\mathbf{a}_{1i}}) + (k+\lambda_2)\lambda_1 - (\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})}\right) \end{aligned} \tag{C.10}$$

Then:

$$\hat{\mathbf{w}}^* = \frac{\lambda_2(\sum_i^k \hat{\mathbf{a}_{1i}^*}) \odot \hat{\mathbf{y}}_o^*}{(k+\lambda_2)(\sum_i^k \hat{\mathbf{a}}_{1i} \odot \hat{\mathbf{a}_{1i}}) + (k+\lambda_2)\lambda_1 - (\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} \tag{C.11}$$

It is to be noted that when $k = 1$, then Eq C.11 reduces to the single template case in Eq C.6, since $\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) = (\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})$ for $k = 1$.

Detection Formula with Multiple Templates

Here, the detection formula is similar to Eq C.7.

C.2 Solution to Problem C.1 in the Dual Domain

C.2.1 Using a Single Template

The optimization problem becomes:

$$f(\mathbf{z}) = || \begin{bmatrix} \mathbf{A} & -\mathbf{I} \end{bmatrix} \mathbf{z}||_2^2 + \lambda_1 || \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{z}||_2^2 + \lambda_2 || \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{z} - \mathbf{y}_0||_2^2 \tag{C.12}$$

For simpler notation, let $\mathbf{G} = \begin{bmatrix} \mathbf{A} & -\mathbf{I} \end{bmatrix}$, also let $\mathbf{E} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}$, and $\mathbf{D} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix}$

$$\begin{aligned} f(\mathbf{z}) &= ||\mathbf{G}\mathbf{z}||_2^2 + \lambda_1 ||\mathbf{E}\mathbf{z}||_2^2 + \lambda_2 ||\mathbf{D}\mathbf{z} - \mathbf{y}_0||_2^2 \\ &= \lambda_2 \sum_i \Big(\mathbf{z}^T \mathbf{d}_i - y_{0i}\Big)^2 + \lambda_1 ||\mathbf{E}\mathbf{z}||_2^2 + ||\mathbf{G}\mathbf{z}||_2^2 \end{aligned} \tag{C.13}$$

Then:

$$\begin{aligned} &\nabla_{\mathbf{z}} f(\mathbf{z}) = 2\lambda_2 \sum_i \Big(\mathbf{z}^T \mathbf{d}_i - y_{0i}\Big)\mathbf{d}_i + 2\lambda_1 \mathbf{E}^T \mathbf{E}\mathbf{z} + 2\mathbf{G}^T \mathbf{G}\mathbf{z} = 0 \\ &\lambda_2 \sum_i \Big(\mathbf{z}^T \mathbf{d}_i - y_{0i}\Big)\mathbf{d}_i = -\Big(\lambda_1 \mathbf{E}^T \mathbf{E} + \mathbf{G}^T \mathbf{G}\Big)\mathbf{z} \\ &\mathbf{z} = -\lambda_2 \Big(\lambda_1 \mathbf{E}^T \mathbf{E} + \mathbf{G}^T \mathbf{G}\Big)^{-1} \sum_i \Big(\mathbf{z}^T \mathbf{d}_i - y_{0i}\Big)\mathbf{d}_i \end{aligned} \tag{C.14}$$

Let $\alpha_i = -\lambda_2 \Big(\mathbf{z}^T \mathbf{d}_i - y_{0i}\Big)$. Then $\mathbf{z} = \Big(\lambda_1 \mathbf{E}^T \mathbf{E} + \mathbf{G}^T \mathbf{G}\Big)^{-1} \mathbf{D}^T \alpha$. Let $\mathbf{K} = \Big(\lambda_1 \mathbf{E}^T \mathbf{E} + \mathbf{G}^T \mathbf{G}\Big)$.

By substituting the dual variables α into Eq C.13, we obtain:

$$f(\alpha) = \lambda_2 \sum_i \left(\alpha^T \mathbf{D}\mathbf{K}^{-1}\mathbf{d}_i - y_{0i}\right)^2 + \lambda_1 ||\mathbf{E}\mathbf{K}^{-1}\mathbf{D}^T\alpha||_2^2 + ||\mathbf{G}\mathbf{K}^{-1}\mathbf{D}^T\alpha||_2^2$$

Then, the new dual objective is given by:

$$f(\alpha) = \lambda_2 ||\mathbf{D}\mathbf{K}^{-1}\mathbf{D}^T\alpha - \mathbf{y}_0||_2^2 + \lambda_1 ||\mathbf{E}\mathbf{K}^{-1}\mathbf{D}^T\alpha||_2^2 + ||\mathbf{G}\mathbf{K}^{-1}\mathbf{D}^T\alpha||_2^2 \tag{C.15}$$

By setting the gradient to zero, the solution to Problem C.15 is obtained by solving the following linear system:

$$\left(\lambda_2 \mathbf{D}\mathbf{K}^{-1}\mathbf{D}^T\mathbf{D}\mathbf{K}^{-1}\mathbf{D}^T + \lambda_1 \mathbf{D}\mathbf{K}^{-1}\mathbf{E}^T\mathbf{E}\mathbf{K}^{-1}\mathbf{D}^T + \mathbf{D}\mathbf{K}^{-1}\mathbf{G}^T\mathbf{G}\mathbf{K}^{-1}\mathbf{D}^T\right)\alpha = \lambda_2 \mathbf{D}\mathbf{K}^{-1}\mathbf{D}^T\mathbf{y}_0$$

$$\mathbf{D}\mathbf{K}^{-1}\underbrace{\left(\lambda_2 \mathbf{D}^T\mathbf{D} + \lambda_1 \mathbf{E}^T\mathbf{E} + \mathbf{G}^T\mathbf{G}\right)}_{\Psi}\mathbf{K}^{-1}\mathbf{D}^T\alpha = \lambda_2 \mathbf{D}\mathbf{K}^{-1}\mathbf{D}^T\mathbf{y}_0 \tag{C.16}$$

$$\mathbf{D}\begin{bmatrix}(\mathbf{A}^T\mathbf{A} + \lambda_1\mathbf{I}) & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{I}\end{bmatrix}^{-1}\underbrace{\begin{bmatrix}(\mathbf{A}^T\mathbf{A} + \lambda_1\mathbf{I}) & -\mathbf{A}^T \\ -\mathbf{A} & (1+\lambda_2)\mathbf{I}\end{bmatrix}}_{\Psi}\begin{bmatrix}(\mathbf{A}^T\mathbf{A} + \lambda_1\mathbf{I}) & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{I}\end{bmatrix}^{-1}$$
$$\mathbf{D}^T\alpha = \lambda_2 \mathbf{D}\begin{bmatrix}(\mathbf{A}^T\mathbf{A} + \lambda_1\mathbf{I}) & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{I}\end{bmatrix}^{-1}\mathbf{D}^T\mathbf{y}_0$$

(C.17)

By using the inverse lemma and the diagonalization properties of circulant matrices (similar to inverting the circulant diagonal matrix in the single template case),

one can show the following:

$$\begin{aligned}\mathbf{K} &= \begin{bmatrix}(\mathbf{A}^T\mathbf{A}+\lambda_1\mathbf{I}) & -\mathbf{A}^T\\ -\mathbf{A} & \mathbf{I}\end{bmatrix}\\ \mathbf{K}^{-1} &= \begin{bmatrix}\frac{1}{\lambda_1}\mathbf{I} & \frac{1}{\lambda_1}\mathbf{A}^T\\ \frac{1}{\lambda_1}\mathbf{A} & \frac{1}{\lambda_1}\mathbf{A}\mathbf{A}^T+\mathbf{I}\end{bmatrix}\end{aligned} \tag{C.18}$$

Then:

$$\begin{aligned}\mathbf{K}^{-1}\Psi\mathbf{K}^{-1} &= \begin{bmatrix}\mathbf{I} & \frac{\lambda_2}{\lambda_1}\mathbf{A}^T\\ \mathbf{0} & \frac{\lambda_2}{\lambda_1}\mathbf{A}\mathbf{A}^T+(1+\lambda_2)\mathbf{I}\end{bmatrix}\begin{bmatrix}\frac{1}{\lambda_1}\mathbf{I} & \frac{1}{\lambda_1}\mathbf{A}^T\\ \frac{1}{\lambda_1}\mathbf{A} & \frac{1}{\lambda_1}\mathbf{A}\mathbf{A}^T+\mathbf{I}\end{bmatrix}\\ &= \begin{bmatrix}\frac{1}{\lambda_1}\mathbf{I}+\frac{\lambda_2}{\lambda_1^2}\mathbf{A}^T\mathbf{A} & \frac{1+\lambda_2}{\lambda_1}\mathbf{A}^T+\frac{\lambda_2}{\lambda_1^2}\mathbf{A}^T\mathbf{A}\mathbf{A}^T\\ \frac{\lambda_2}{\lambda_1^2}\mathbf{A}\mathbf{A}^\mathbf{T}\mathbf{A}+\frac{1+\lambda_2}{\lambda_1}\mathbf{A} & \frac{\lambda_2}{\lambda_1^2}\mathbf{A}\mathbf{A}^T\mathbf{A}\mathbf{A}^T+\frac{1+2\lambda_2}{\lambda_1}\mathbf{A}\mathbf{A}^T+(1+\lambda_2)\mathbf{I}\end{bmatrix}\end{aligned} \tag{C.19}$$

Therefore, by substituting Eq C.19 into Eq C.17, we get the linear system:

$$\Big(\frac{\lambda_2}{\lambda_1^2}\mathbf{A}\mathbf{A}^T\mathbf{A}\mathbf{A}^T+\frac{1+2\lambda_2}{\lambda_1}\mathbf{A}\mathbf{A}^T+(1+\lambda_2)\mathbf{I}\Big)\alpha = \lambda_2\Big(\frac{1}{\lambda_1}\mathbf{A}\mathbf{A}^T+\mathbf{I}\Big)\mathbf{y}_0 \tag{C.20}$$

The previous problem can be efficiently diagonalized and solved as follows:

$$\begin{aligned}&\mathbf{F}\Big(\frac{\lambda_2}{\lambda_1^2}diag(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*\odot\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+\frac{1+2\lambda_2}{\lambda_1}diag(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+(1+\lambda_2)\mathbf{I}\Big)\hat{\alpha}^*\\ &=\mathbf{F}\Big(\frac{\lambda_2}{\lambda_1}diag(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+\lambda_2\mathbf{I}\Big)\hat{\mathbf{y}_o^*}\\ &\hat{\alpha}^* = \frac{\Big(\frac{\lambda_2}{\lambda_1}(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+\lambda_2\Big)\odot\hat{\mathbf{y}_o}^*}{\frac{\lambda_2}{\lambda_1^2}(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*\odot\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+\frac{1+2\lambda_2}{\lambda_1}(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+(1+\lambda_2)}\\ &\hat{\alpha} = \frac{\Big(\frac{\lambda_2}{\lambda_1}(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+\lambda_2\Big)\odot\hat{\mathbf{y}}_o}{\frac{\lambda_2}{\lambda_1^2}(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*\odot\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+\frac{1+2\lambda_2}{\lambda_1}(\hat{\mathbf{a}}_1\odot\hat{\mathbf{a}}_1^*)+(1+\lambda_2)}\end{aligned} \tag{C.21}$$

Detection Formula with a Single Template

$$\begin{aligned}\mathbf{T}(\mathbf{u}) = \mathbf{U}\mathbf{w} = \mathbf{U}\mathbf{E}\mathbf{K}^{-1}\mathbf{D}^T\alpha = \mathbf{U}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\frac{1}{\lambda_1}\mathbf{I} & \frac{1}{\lambda_1}\mathbf{A}^T \\ \frac{1}{\lambda_1}\mathbf{A} & \frac{1}{\lambda_1}\mathbf{A}\mathbf{A}^T + \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{0} \\ \alpha\end{bmatrix} = \frac{1}{\lambda_1}\mathbf{U}\mathbf{A}^T\alpha \\ = \frac{1}{\lambda_1}\mathbf{F}diag(\hat{\mathbf{u}} \odot \hat{\mathbf{a}}_1^*)\hat{\alpha}^* \\ \hat{\mathbf{T}}(\mathbf{u}) = \frac{1}{\lambda_1}\hat{\mathbf{u}}^* \odot \hat{\mathbf{a}}_1 \odot \hat{\alpha}\end{aligned} \tag{C.22}$$

C.2.2 Using Multiple Templates

For multiple templates, the only difference is that $\tilde{\mathbf{G}}^T = \begin{bmatrix}\mathbf{A}_1^T & \mathbf{A}_2^T & \dots & \mathbf{A}_k^T \\ -\mathbf{I} & -\mathbf{I} & \dots & -\mathbf{I}\end{bmatrix}$ Therefore, the new dual objective is given as follows:

$$\mathbf{D}\tilde{\mathbf{K}}^{-1}\Big(\lambda_2\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{E}^T\mathbf{E} + \tilde{\mathbf{G}^T}\tilde{\mathbf{G}}\Big)\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha = \lambda_2\mathbf{D}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\mathbf{y}_0 \tag{C.23}$$

where

$$\tilde{\mathbf{K}} = \begin{bmatrix}(\sum_i^k \mathbf{A}_i^T\mathbf{A}_i + \lambda_1\mathbf{I}) & -\sum_i^k \mathbf{A}_i{}^T \\ -\sum_i^k \mathbf{A}_i & k\mathbf{I}\end{bmatrix} \tag{C.24}$$

$$\tilde{\mathbf{K}} = \begin{bmatrix}\mathbf{F} & \mathbf{0} \\ \mathbf{0} & \mathbf{F}\end{bmatrix}\begin{bmatrix}(\sum_i^k diag(\hat{\mathbf{a}}_i^* \odot \hat{\mathbf{a}}_i) + \lambda_1\mathbf{I}) & -\sum_i^k diag(\hat{\mathbf{a}}_i^*) \\ -\sum_i^k diag(\hat{\mathbf{a}}_i) & k\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{F}^H & \mathbf{0} \\ \mathbf{0} & \mathbf{F}^H\end{bmatrix} \tag{C.25}$$

To find $\tilde{\mathbf{K}}$, we use the inverse lemma in Eq C.3 again. First, we need to find: $\mathbf{B} - \mathbf{N}\mathbf{C}^{-1}\mathbf{V}$:

$$\begin{aligned}
\mathbf{B}-\mathbf{N}\mathbf{C}^{-1}\mathbf{V} &= \sum_i^k diag(\hat{\mathbf{a}}_i^* \odot \hat{\mathbf{a}}_i) + \lambda_1\mathbf{I} - \frac{1}{k}\sum_i^k diag(\hat{\mathbf{a}}_i^*)\sum_i^k diag(\hat{\mathbf{a}}_i) \\
&= diag(\sum_i^k \hat{\mathbf{a}}_i^* \odot \hat{\mathbf{a}}_i) + \lambda_1\mathbf{I} - \frac{1}{k}diag(\sum_i^k \hat{\mathbf{a}}_i^*)diag(\sum_i^k \hat{\mathbf{a}}_i) \\
&= diag(\sum_i^k (\hat{\mathbf{a}}_i^* \odot \hat{\mathbf{a}}_i) + \lambda_1) - \frac{1}{k}diag(\sum_i^k \hat{\mathbf{a}}_i^* \odot \sum_i^k \hat{\mathbf{a}}_i) \\
&= diag\Big(\sum_i^k (\hat{\mathbf{a}}_i^* \odot \hat{\mathbf{a}}_i) + \lambda_1 - \frac{1}{k}\Big(\sum_i^k \hat{\mathbf{a}}_i^* \odot \sum_i^k \hat{\mathbf{a}}_i\Big)\Big)
\end{aligned} \tag{C.26}$$

Then, the inverse of $\tilde{\mathbf{K}}$ is given as follows $\tilde{\mathbf{K}}^{-1} = \begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix} \Lambda \begin{bmatrix} \mathbf{F}^H & \mathbf{0} \\ \mathbf{0} & \mathbf{F}^H \end{bmatrix}$

where

$$\Lambda = \begin{bmatrix} \frac{1}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} & \frac{\frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^*}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} \\ \frac{\frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} & \frac{\frac{1}{k^2}\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i}}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} + \frac{1}{k} \end{bmatrix}$$

And since:

$$\begin{aligned}
\tilde{\Psi} = (\lambda_2\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{E}^T\mathbf{E} + \tilde{\mathbf{G}}^T\tilde{\mathbf{G}}) &= \begin{bmatrix} (\sum_i^k \mathbf{A}_i^T\mathbf{A}_i + \lambda_1\mathbf{I}) & -\sum_i^k \mathbf{A}_i{}^T \\ -\sum_i^k \mathbf{A}_i & (k+\lambda_2)\mathbf{I} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix} \Omega \begin{bmatrix} \mathbf{F}^H & \mathbf{0} \\ \mathbf{0} & \mathbf{F}^H \end{bmatrix}
\end{aligned} \tag{C.27}$$

where $\Omega = \begin{bmatrix} \sum_i^k diag(\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1\mathbf{I} & -\sum_i^k diag(\hat{\mathbf{a}}_{1i}^*) \\ -\sum_i^k diag(\hat{\mathbf{a}}_{1i}) & (k+\lambda_2)\mathbf{I} \end{bmatrix}$

Then, $\tilde{\mathbf{K}}^{-1}\tilde{\Psi}\tilde{\mathbf{K}}^{-1} = \begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix} \Lambda\Omega\Lambda \begin{bmatrix} \mathbf{F}^H & \mathbf{0} \\ \mathbf{0} & \mathbf{F}^H \end{bmatrix}$

Then:

$$\Lambda\Omega = \begin{bmatrix} \mathbf{1} & \frac{\frac{\lambda_2}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^*}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} \\ \mathbf{0} & \frac{\frac{-1}{k}\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\frac{k+\lambda_2}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})+\frac{\lambda_1(k+\lambda_2)}{k}}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} \end{bmatrix} \tag{C.28}$$

Then:

$$\Lambda\Omega\Lambda = \begin{bmatrix} \mathbf{1} & \frac{\frac{\lambda_2}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^*}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} \\ \mathbf{0} & \frac{\frac{-1}{k}\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\frac{k+\lambda_2}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})+\frac{\lambda_1(k+\lambda_2)}{k}}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} & \frac{\frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^*}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} \\ \frac{\frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})} & \frac{\frac{1}{k^2}\sum_i^k \hat{\mathbf{a}}_{1i}^*\odot\sum_i^k \hat{\mathbf{a}}_{1i}}{\sum_i^k(\hat{\mathbf{a}}_{1i}^*\odot\hat{\mathbf{a}}_{1i})+\lambda_1-\frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*)\odot(\sum_i^k \hat{\mathbf{a}}_{1i})}+\frac{1}{k} \end{bmatrix} \tag{C.29}$$

$$= \begin{bmatrix} \cdots & \cdots \\ & \\ \cdots & \Upsilon \end{bmatrix}$$

Note that we need to compute $\mathbf{D}\tilde{\mathbf{K}}^{-1}\tilde{\Psi}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T$, so only the last block Υ is relevant. So, $\mathbf{D}\mathbf{K}^{-1}\Psi\mathbf{K}^{-1}\mathbf{D}^T = \mathbf{F}\Upsilon\mathbf{F}^H$, where

$$\Upsilon = \Bigg(\Big(\frac{\frac{\lambda_1}{k}\sum_i^k \hat{\mathbf{a}}_{1i} + \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}) \odot (\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}^*) - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{a}}_{1i})^2}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{a}}_{1i})} - \frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i} \Big) \\ \odot \Big(\frac{\frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^*}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{a}}_{1i})} \Big) \Bigg) \\ \Bigg(+ \Big(\frac{\frac{k+\lambda_2-1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{a}}_{1i})}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{a}}_{1i})} + \frac{k+\lambda_2}{k} \Big) \\ \odot \Big(\frac{\frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i}}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^*) \odot (\sum_i^k \hat{\mathbf{a}}_{1i})} + \frac{1}{k} \Big) \Bigg) \tag{C.30}$$

and since:

$$\mathbf{D}\tilde{K}^{-1}\mathbf{D}^T = \mathbf{F}\left(\frac{\frac{1}{k}\Big(\sum_i^k \hat{\mathbf{a}}_{1i}^*\Big) \odot \Big(\sum_i^k \hat{\mathbf{a}}_{1i}\Big)}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} + \frac{1}{k} \right) \mathbf{F}^H \tag{C.31}$$

Therefore, solution to Problem C.23 is computed as follows:

$$\mathbf{F}\Upsilon\mathbf{F}^H\alpha = \lambda_2 \mathbf{F}\left(\frac{\frac{1}{k}\Big(\sum_i^k \hat{\mathbf{a}}_{1i}^*\Big) \odot \Big(\sum_i^k \hat{\mathbf{a}}_{1i}\Big)}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} + \frac{1}{k} \right) \mathbf{F}^H \mathbf{y}_o \tag{C.32}$$

$$\hat{\alpha}^* = \lambda_2 \Upsilon^{-1}\left(\frac{\frac{1}{k}\Big(\sum_i^k \hat{\mathbf{a}}_{1i}^*\Big) \odot \Big(\sum_i^k \hat{\mathbf{a}}_{1i}\Big) \odot \hat{\mathbf{y}_0}^*}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} + \frac{\hat{\mathbf{y}_0}^*}{k} \right) \tag{C.33}$$

Detection Formula with Multiple Templates

$$\begin{aligned} \mathbf{T}(\mathbf{u}) = \mathbf{U}\mathbf{w} = \mathbf{U}\mathbf{E}\tilde{\mathbf{K}}^{-1}\mathbf{D}^T\alpha = \mathbf{F}\frac{\hat{\mathbf{u}} \odot \frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \hat{\alpha}^*}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} \\ \hat{\mathbf{T}}(\mathbf{u}) = \frac{\hat{\mathbf{u}}^* \odot \frac{1}{k}\sum_i^k \hat{\mathbf{a}}_{1i} \odot \hat{\alpha}}{\sum_i^k (\hat{\mathbf{a}}_{1i}^* \odot \hat{\mathbf{a}}_{1i}) + \lambda_1 - \frac{1}{k}(\sum_i^k \hat{\mathbf{a}}_{1i}^* \odot \sum_i^k \hat{\mathbf{a}}_{1i})} \end{aligned} \tag{C.34}$$

C.3 Integrating SRDCF

Instead of re-deriving a closed form solution to SRDCF's formulation, one can use alternating optimization, where the filter $\mathbf{w}$ and the target response $\mathbf{y}$ are updated in an iterative fashion keeping one of them fixed at any given time. The target adapted SRDCF objective becomes:

$$\min_{\mathbf{w},\mathbf{y}} \quad ||\mathbf{A}\mathbf{w} - \mathbf{y}||_2^2 + \lambda_1||\mathbf{X}\mathbf{w}||_2^2 + \lambda_2||\mathbf{y} - \mathbf{y}_0||_2^2, \tag{C.35}$$

where $\mathbf{X}$ is the weight mask over the filter. Using alternating optimization, it can be minimized by solving the next two optimization problems at iteration j.

$$\mathbf{w}^{j+1} = \arg\min_{\mathbf{w}} ||\mathbf{A}\mathbf{w} - \mathbf{y}^j||_2^2 + \lambda_1||\mathbf{X}\mathbf{w}||_2^2 \tag{C.36}$$

$$\mathbf{y}^{j+1} = \arg\min_{\mathbf{y}} ||\mathbf{A}\mathbf{w}^{j+1} - \mathbf{y}||_2^2 + \lambda_2||\mathbf{y} - \mathbf{y}_0||_2^2 \tag{C.37}$$

Problem C.36 can be solved using the standard SRDCF solution, while Problem C.37 has a closed form solution as follows:

$$\mathbf{y}^{j+1} = \frac{1}{1+\lambda_2}\left(\mathbf{A}\mathbf{w}^{j+1} + \lambda_2\mathbf{y}_o\right) \tag{C.38}$$

To compute $\mathbf{y}^{j+1}$ efficiently in the Fourier domain, we can use the following strat-

egy:

$$\mathbf{y}^{j+1} = \frac{1}{1+\lambda_2}(\mathbf{F}diag(\hat{\mathbf{a}}_1)\mathbf{F}^H\mathbf{w}^{j+1} + \lambda_2\mathbf{y}_o) \tag{C.39}$$

$$= \frac{1}{1+\lambda_2}(\mathbf{F}diag(\hat{\mathbf{a}}_1)\hat{\mathbf{w}}^{*j+1} + \lambda_2\mathbf{y}_o) \tag{C.40}$$

$$\Rightarrow \hat{\mathbf{y}}^* = \frac{1}{1+\lambda_2}(diag(\hat{\mathbf{a}}_1)\hat{\mathbf{w}}^{*j+1} + \lambda_2\hat{\mathbf{y}}_o^*) \tag{C.41}$$

$$= \frac{1}{1+\lambda_2}(\hat{\mathbf{a}}_1 \odot \hat{\mathbf{w}}^{*j+1} + \lambda_2\hat{\mathbf{y}}_o^*) \tag{C.42}$$

D Treatment of the Formulation Proposed in Chapter 5

Say $\mathbf{A}$ is block wise circulant, such that $\mathbf{A}_i$ is circulant $\forall i$. $\mathbf{A} = [\mathbf{A}_1|\mathbf{A}_2|\mathbf{A}_3|...|\mathbf{A}_N]$ where $\mathbf{A}_i \in \mathbb{R}^{d\times d}$. Then, $\mathbf{A} \in \mathbb{R}^{d\times Nd}$ and N denotes the number of samples (Training examples). $\mathbf{y} \in \mathbb{R}^{d\times 1}$, and $\mathbf{x} \in \mathbb{R}^{Nd\times 1}$ Then the primal formulation for the Lasso is given by:

$$P1 : \underset{\mathbf{x}}{\text{minimize}} \quad ||\mathbf{A}\mathbf{x} - \mathbf{y}||_2^2 + \lambda||\mathbf{x}||_1 \tag{D.1}$$

By setting $\mathbf{r} = A\mathbf{x} - \mathbf{y}$, then P1 can be re-written as:

$$\begin{aligned} P2 : \underset{\mathbf{x},\mathbf{r}}{\text{minimize}} \quad & ||\mathbf{r}||_2^2 + \lambda||\mathbf{x}||_1 \\ \text{subject to} \quad & \mathbf{r} = \mathbf{A}\mathbf{x} - \mathbf{y} \end{aligned} \tag{D.2}$$

D.1 Dual Formulation

Follows the dual formulation [79]:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{r}} \mathcal{L}(\mathbf{x},\mathbf{r},\Psi) &= \min_{\mathbf{x},\mathbf{r}} ||\mathbf{r}||_2^2 + \lambda||\mathbf{x}||_1 + \Psi^T(\mathbf{A}\mathbf{x} - \mathbf{y} - \mathbf{r}) \\ &= \min_{\mathbf{r}} (||\mathbf{r}||_2^2 - \Psi^T\mathbf{r}) + \min_{\mathbf{x}}(\lambda||\mathbf{x}||_1 + \Psi^T\mathbf{A}\mathbf{x}) - \Psi^T\mathbf{y} \\ &= -\Psi^T\mathbf{y} - \frac{1}{4}||\Psi||_2^2 - \lambda\max_{\mathbf{x}}(\frac{\mathbf{x}^T(-\mathbf{A}^T\Psi)}{\lambda} - ||\mathbf{x}||_1) \\ &= -\Psi^T\mathbf{y} - \frac{1}{4}||\Psi||_2^2 - \mathbb{1}_{(\frac{||-\mathbf{A}^T\Psi||_\infty}{\lambda} \leq 1)} \end{aligned} \tag{D.3}$$

Therefore, the dual problem of D.1 and D.2 is given by:

$$\begin{aligned} P3: \underset{\Psi}{\text{minimize}} \quad & \frac{1}{4}||\Psi||_2^2 + \Psi^T \mathbf{y} \\ \text{subject to} \quad & ||\mathbf{A}^T \Psi||_\infty \leq \lambda \end{aligned} \tag{D.4}$$

D.2 ADMM Formulation

To solve the problem using ADMM, problem D.4 will be set in the standard format by letting:

$$\begin{aligned} f(\Psi) &= \frac{1}{4}||\Psi||_2^2 + \Psi^T \mathbf{y} \\ g(\zeta) &= \mathbf{1}_{(||\zeta||_\infty \leq \lambda)} \end{aligned} \tag{D.5}$$

where $\zeta = A^T \Psi$. Then the dual problem can be re-witten as:

$$\begin{aligned} \underset{\Psi,\zeta}{\text{minimze}} \quad & f(\Psi) + g(\zeta) \\ \text{subject to} \quad & \zeta = A^T \Psi \end{aligned} \tag{D.6}$$

The augmented lagrangian is given as follows:

$$\mathcal{L}(\Psi, \zeta, \gamma) = f(\Psi) + g(\zeta) + \gamma^T (A^T \Psi - \zeta) + \frac{\rho}{2}||A^T \Psi - \zeta||_2^2 \tag{D.7}$$

Note that: $\gamma \in \mathbb{R}^{Nd \times 1}$ and $\zeta \in \mathbb{R}^{Nd \times 1}$

D.2.1 Updating Ψ

$$\Psi^{k+1} = \underset{\Psi}{\text{argmin}} \quad f(\Psi) + (\gamma^k)^T (\mathbf{A}^T \Psi - \zeta^k) + \frac{\rho}{2}||\mathbf{A}^T \Psi - \zeta^k||_2^2 \tag{D.8}$$

Then the solution is given by:

$$(\rho \mathbf{A}\mathbf{A}^T + \frac{1}{2}\mathbf{I})\Psi^{k+1} = \mathbf{A}(\rho \zeta^k - \gamma^k) - \mathbf{y} \tag{D.9}$$

$\mathbf{A}\mathbf{A}^H$ can actually be computed very efficiently as follows:

$$\begin{aligned}\mathbf{A}\mathbf{A}^H &= \sum_i^N A_i A_i^H = \sum_i^N (\mathbf{F} diag(\hat{\mathbf{a}}_i) diag(\hat{\mathbf{a}}_i^*) \mathbf{F}^H) \\ &= \mathbf{F} diag\Big(\sum_i^N \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) \mathbf{F}^H\end{aligned} \tag{D.10}$$

Then the update rule will collapse to be:

$$\begin{aligned}\Psi^{k+1} &= (\rho \mathbf{F} diag\Big(\sum_i^n \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) \mathbf{F}^H + \frac{1}{2}\mathbf{F}\mathbf{F}^H)^{-1}(\mathbf{A}(\rho\zeta^k - \gamma^k) - \mathbf{y}) \\ &= \mathbf{F}(\rho diag\Big(\sum_i^N \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2}\mathbf{I})^{-1}\mathbf{F}^H \mathbf{A}(\rho\zeta^k - \gamma^k) - \mathbf{F}(\rho diag\Big(\sum_i^N \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2}\mathbf{I})^{-1}\hat{\mathbf{y}}^* \\ &= \mathbf{F}\frac{\Gamma^k}{(\rho\Big(\sum_i^N \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2})} - \mathbf{F}\frac{\hat{\mathbf{y}}^*}{(\rho\Big(\sum_i^N \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2})}\end{aligned} \tag{D.11}$$

Where the division is element wise operation. Since:

$$\begin{aligned}\mathbf{A}(\rho\zeta^k - \gamma^k) &= \mathbf{F}[diag(\hat{\mathbf{a}}_1) \mid diag(\hat{\mathbf{a}}_2) \mid .... \mid diag(\hat{\mathbf{a}}_N)][\mathbf{F}^H \otimes \mathbf{I}](\rho\zeta^k - \gamma^k) \\ &= \mathbf{F}\sum_i^N diag(\hat{\mathbf{a}}_i)(\rho\hat{\zeta}^{*k} - \hat{\gamma}^{*k})_i = \mathbf{F}\sum_i^N \hat{\mathbf{a}}_i \odot (\rho\hat{\zeta}^{*k} - \hat{\gamma}^{*k})_i = \mathbf{F}\Gamma^k\end{aligned} \tag{D.12}$$

Then,

$$\begin{aligned}\Psi^{k+1} &= \mathbf{F}\frac{\Gamma^k}{(\rho\Big(\sum_i^n \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2})} - \mathbf{F}\frac{\hat{\mathbf{y}}^*}{(\rho\Big(\sum_i^n \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2})} \\ &= \mathbf{F}\frac{(\Gamma^k - \hat{\mathbf{y}}^*)}{(\rho\Big(\sum_i^N \hat{\mathbf{a}}_i \odot \hat{\mathbf{a}}_i^*\Big) + \frac{1}{2})}\end{aligned} \tag{D.13}$$

D.2.2 Updating ζ

$$\begin{aligned}
\zeta^{k+1} &= \underset{\zeta^k}{\text{argmin}} \quad g(\zeta^k) + \frac{\rho}{2}||A^T\Psi^{k+1} - \zeta^k||_2^2 + (\gamma^k)^T(A^T\Psi^{k+1} - \zeta^k) \\
&= \underset{\zeta^k}{\text{argmin}} \quad \frac{\rho}{2}||A^T\Psi^{k+1} - \zeta^k||_2^2 + (\gamma^k)^T(A^T\Psi^{k+1} - \zeta^k) \\
&\quad \text{subject to} \quad ||\zeta^k||_\infty \leq \lambda \\
&= \underset{\zeta^k}{\text{argmin}} \quad ||\zeta^k - A^T\Psi^{k+1}||_2^2 - \frac{2}{\rho}(\gamma^k)^T\zeta^k \\
&\quad \text{subject to} \quad ||\zeta^k||_\infty \leq \lambda
\end{aligned} \tag{D.14}$$

Note that:

$$\begin{aligned}
&\underset{\zeta^k}{\text{argmin}} \quad ||(\zeta^k - A^T\Psi^{k+1}) - \frac{1}{\rho}\gamma^k||_2^2 \\
&= \underset{\zeta^k}{\text{argmin}} \quad ||\zeta^k - A^T\Psi^{k+1}||_2^2 - \frac{2}{\rho}(\gamma^k)^T(\zeta^k - A^T\Psi^{k+1}) + \frac{1}{\rho^2}||\gamma^k||_2^2 \\
&= \underset{\zeta^k}{\text{argmin}} \quad ||\zeta^k - A^T\Psi^{k+1}||_2^2 - \frac{2}{\rho}(\gamma^k)^T\zeta^k
\end{aligned} \tag{D.15}$$

Then the dual problem can be re-written as:

$$\begin{aligned}
&\underset{\zeta^k}{\text{argmin}} \quad ||\zeta^k - (A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k)||_2^2 \\
&\text{subject to} \quad ||\zeta^k||_\infty \leq \lambda
\end{aligned} \tag{D.16}$$

Then the solution for problem D.16 is given by projecting on to the L_∞ ball:

$$\zeta^{k+1} = \begin{cases}
A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k & \text{if } ||A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k||_\infty \leq \lambda \\
\text{proj}(A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k)_i & \text{if } \forall\ i \text{ such that} \\
 & ||A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k||_\infty > \lambda
\end{cases} \tag{D.17}$$

where:

$\text{proj}(A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k)_i = \lambda sign(A^T\Psi^{k+1} + \frac{1}{\rho}\gamma^k)_i$

Therefore, the update for ζ^{k+1} is given by:

$$A^H\Psi^{k+1} + \frac{1}{\rho}\gamma^k = \begin{bmatrix} \mathbf{F}diag(\hat{\mathbf{a}}_1^* \odot \hat{\Psi}^{k+1^*}) + \frac{1}{\rho}\gamma_1^k \\ \mathbf{F}diag(\hat{\mathbf{a}}_2^* \odot \hat{\Psi}^{k+1^*}) + \frac{1}{\rho}\gamma_2^k \\ \vdots \\ \mathbf{F}diag(\hat{\mathbf{a}}_N^* \odot \hat{\Psi}^{k+1^*}) + \frac{1}{\rho}\gamma_N^k \end{bmatrix} \tag{D.18}$$

And the final update is given by D.17 with the vector condition defined for efficacy as in D.18.

D.2.3 Updating γ

$$\gamma^{k+1} = \gamma^k + \rho(A^T\Psi^{k+1} - \zeta^{k+1}) \tag{D.19}$$

E Accepted and Submitted Papers

E.1 Accepted and Submitted Conference Papers

• [**C**1] **A**. **Bibi** and B. Ghanem "Multi-Template Scale-Adaptive Kernelized Correlation Filters". *IEEE International Conference in Computer Vision Workshops (ICCVW)*, Santiago, Chile, NOV 2015.

• [**C**2] **A**. **Bibi**, T. Zhang and B. Ghanem "3D Part-Based Sparse Tracker with Automatic Synchronization and Registration". To appear in *IEEE Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, June 2016.

• [**C**3] T. Zhang, **A**. **Bibi** and B. Ghanem "In Defense of Sparse Tracking: Circulant Sparse Tracker". To appear in *IEEE Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, June 2016.

• [**C**4] **A**. **Bibi**, M. Muelluer and B. Ghanem "Target Response Adaptation for Correlation Filter Tracking". *IEEE European Conference in Computer Vision (ECCV) (Under Review)*, Amsterdam, March 2016.