An Exploration into Model-Free Online Visual Object Tracking

Gao Zhu

A thesis submitted for the degree of
Doctor of Philosophy
The Australian National University

January 2017

Except where otherwise indicated, this thesis is my own original work.

Gao Zhu
29 January 2017

To my parents and supportive wife, Tian Tian – I will never reach this PhD degree without you behind me.

Acknowledgments

Firstly, I would like to express my deep gratitude to the chair of my research panel: Associate Professor Hongdong Li, who accepted me as his PhD student about four years ago. He has closely mentored and taught me research skills since then and I am impressed of his innovative ideas and really enjoy discussing academic and life problems with him.

Secondly, a major credit would be given to another supervisor in my research panel: Professor Fatih Porikli. He joined the panel at my second year of PhD while I was changing the research topic from semantic object segmentation and localization to visual object tracking. I would not go through so many successful and interesting projects without his significant contribution, not only in technical advice but also in concrete writing support.

Yansheng Ming, who was a senior PhD student working on object contour and semantic segmentation when I started the PhD study, provided great helps in my first two years mentally and technically. We would have talked for hours about a new idea and his insightful and rigorous thinking had impacted me for the rest time of my PhD study.

I would like to acknowledge many fellow students: Ami Drory, Buyu Liu, Eric Hou, Gucan Long, Hao Zhu, Jiaolong Yang, Juan David Adarve, Jamie Wei, Moses Bangura, Pan Ji, Qingchen Liu, Yi Zhou, Yiran Zhong, Yon Hon Ng, Yun Hou, Zhixun Li, Zhiyong Sun, etc. and fellow senior researchers: Dr Chuong Nguyen, Dr Dingfu Zhou, Dr Laurent Kneip, Dr Mathieu Salzmann, Dr Mehrtash Harandi, Professor Richard Hartley, Associate Professor Stephen Gould, Dr Viorela Ila, Dr Xuming He, Dr Yuchao Dai, etc., for creating a great computer vision community, benefiting me in various ways.

Finally, family and friends are always my mental pillars. I am grateful for the trust and love they put on me. Especially to my wife, Tian Tian and my parents, thank you all for consistently backing me up and keeping me spiritually charged.

Abstract

This thesis presents a thorough investigation of model-free visual object tracking, a fundamental computer vision task that is essential for practical video analytics applications. Given the states of the object in the first frame, e.g., the position and size of the target, the computational methods developed and advanced in this thesis aim at determining target states in consecutive video frames automatically. In contrast to the tracking schemes that depend strictly on specific object detectors, model-free tracking provides conveniently flexible and competently general solutions where object representations are initiated in the first frame and adapted in an online manner at each frame.

We first articulate our motivations and intuitions in Chapter 1, formulate model-free online visual tracking, illustrate outcomes on two representative object tracking applications; drone control and sports video broadcasting analysis, and elaborate other relevant problems.

In Chapter 2, we review various tracking methodologies employed by state-of-the-art trackers and further review related background knowledge, including several important dataset benchmarks and workshop challenges, which are widely used for evaluating the performance of trackers, as well as commonly applied evaluation protocols in this chapter.

In Chapter 3 through Chapter 6, we then explore the model-free online visual tracking problem in four different dimensions: 1) learning a more discriminative classifier with a two-layer classification hierarchy and background contextual clusters; 2) overcoming the limit of conventionally used local-search scheme with a global object tracking framework based on instance-specific object proposals; 3) tracking object affine motion with a Structured Support Vector Machine (SSVM) framework incorporated with motion manifold structure; 4) an efficient multiple object model-free online tracking approach based on a shared pool of object proposals.

Lastly, as a conclusion and future work outlook, we highlight and summarize the contribution of this thesis and discuss several promising research directions in Chapter 7, based on latest work and their drawbacks of current state-of-the-art trackers.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Humans use their eyes and brains to see and visually sense the world around them, while computer vision is the science that aims to give a similar, if not better, capability to a machine or computer [Ballard and Brown, 1982]. It is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images and involves the development of a theoretical and algorithmic basis to achieve automatic visual understanding.

Recently, computer vision research has been receiving world-wide growing attention thanks to its crucial role in major applications such as self-driving cars and augmented/virtual reality. Computer vision solutions evolved significantly and their performance improved to a level that is now comparable with human perception in specific tasks as face recognition [Taigman et al., 2014] and object classification [Russakovsky et al., 2015].

Following the trend with a large number of papers published in top-tier computer vision conferences and journals every year, object tracking research keeps its momentum as a rapidly emerging field that continually attracts significant attention [Kristan et al., 2015; Milan et al., 2016]. Object tracking enables many higher-level objectives such as motion analysis, event detection, and activity understanding [Kristan et al., 2013; Patino et al., 2016]. Nevertheless, visual object tracking remains a challenging task [Smeulders et al., 2014; Wu et al., 2013, 2015].

Furthermore, the fast development of hardware/software technology in terms of computational power, form factor and price, opens potentially vast applications for tracking algorithms [TeuliÃÍre et al., 2011; Gomez-Balderas et al., 2013; Xing et al., 2011]. Typical applications could be found in camera surveillance systems, transport industry, sports video automatic analysis, medical imaging, mobile robotics, film post-production and human-computer interfaces.

1.2 Applications

We introduce two popular applications as a taste that how the visual object tracking algorithms can be applied to help solve real-life problems. Firstly, we check the latest advance of using visual tracking methods to assist controlling the Unmanned Areal Vehicles (UAVs) or drones. Then we look at the domain of sports in which the tracking algorithms are deployed to automatically extract and analyze players and other contextual information for improving their performances, without laborious and subjective human annotations.

1.2.1 Visual Tracking on Drones

Along with widely discussed self-driving cars, Unmanned Aerial Vehicles (UAVs) or drones are expected to have extensive applications in the future. One example is the Amazon Prime Air, a future delivery system designed to safely get packages to customers in 30 minutes or less using small UAVs. It is thought to have great potential to increase the overall safety and efficiency of the transportation system.

Meanwhile, the vision-based control of UAVs has become an active field of research in the last few years [Hérissé et al., 2010; TeuliÃÍre et al., 2011; Gomez-Balderas et al., 2013], majorly because vision provides a cheap, passive and rich source of information, while low-weight cameras can be embedded even on small-size flying UAVs. Most of the efforts have been concentrated on developing tracking-based control methods for autonomous take off, landing, stabilization and navigation, as shown in Figure 1.1 (a). Those methods apply either a known model (e.g., color histogram) of a target [TeuliÃÍre et al., 2011; Gomez-Balderas et al., 2013] or optical flow computation [Hérissé et al., 2010].

A successful product that can be found in the market is the DJI Phantom 4, which is promoted as the smartest flying camera drone, allowing to capture superb aerial images on phones or tablets. Not only does it fly intelligently with a tap and automatically create seamless tracking shots, it can autonomously avoid obstacles as shown in Figure 1.1 (b). A critical function to achieve this is called ActiveTrack,

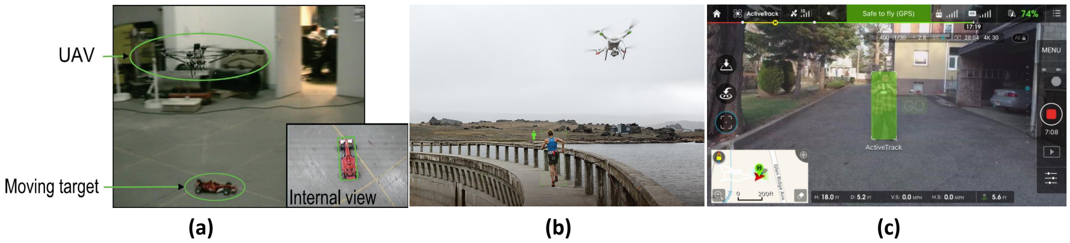


Figure 1.1: Tracking algorithms can be used to send visual information for controlling drones. (a) A color histogram based drone tracking system [TeuliÃÍre et al., 2011]; (b) In a real-life scenario, the DJI Phantom 4 uses ActiveTrack technology to track moving subjects and avoid obstacles automatically. (c) A screen capture of the Graphical User Interface (GUI) of the Phantom 4. The person over-shaded by green is the target.

which uses advanced visual tracking capabilities to fully automate tracking, so the camera can be pointed steadily at a moving subject and get a perfect shot while flying. An example screen capture of the Phantom 4 Graphical User Interface (GUI) is shown in Figure 1.1 (c). In this case, a person over-shaded with green color is to be tracked. The target could be initialized either manually by drawing a bounding box over the target or automatically by an object category detector searching around a click point of the user.

1.2.2 Visual Tracking for Automatic Sports Video Analysis

According to a report from Forbes, the sports market in North America alone was worth $60.5 billion in 2014. It is expected to reach $73.5 billion by 2019 and the biggest reason for such a growth is the increases in revenue derived from media rights deals. As a result, sports broadcast videos can be easily found on major television channels or various video websites such as YouTube. To take advantage of the incredibly huge amount of video data, vision technique such as visual object tracking is an important tool to provide crucial inputs for extensive higher-level applications.

One of such applications is for Canoe/Kayak Slalom (CK Slalom) competition Drory et al. [2017], in which negotiation of obstacles through gates is the fundamental skill and key determinant of overall performance. In race context where the winner is commonly decided by fractions of a second, developing an optimal strategy and techniques for negotiation of gates that minimizes overall course time-to-completion is critical for performance. Previous literature [Hunter, 2009] analyzed upstream gate negotiation strategies of 17 elite Slalom paddlers using manual extraction of spatial kinematic data of the boat and athlete's head from video footage obtained by over-

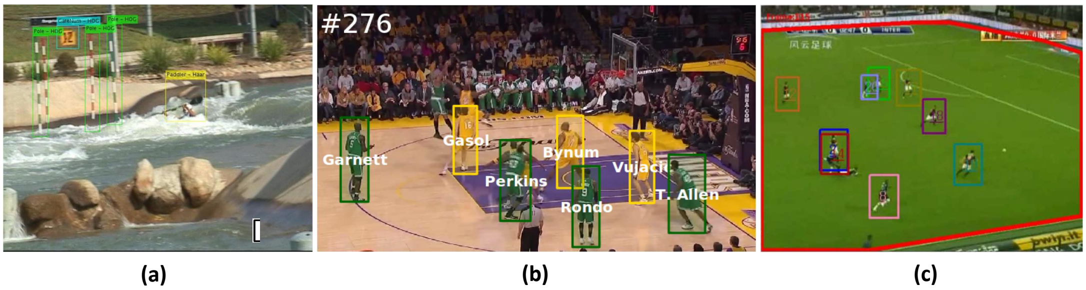


Figure 1.2: Tracking algorithms can be used to automatically analyze sports broadcast videos. (a) For Canoe/Kayak Slalom (CK Slalom) competition, automatic detection and tracking of Slalom paddlers as well as the ordered course obstacles provide the evidence base pre-requisite to derived race kinematics for analysis of performance; (b)(c) For basketball and soccer broadcast videos, intelligent video analysis systems deploy automatic player tracking and identification to gather game statistics for understanding the competitors' strength and weakness. Note that in (c), color information is used to segment the dominant playfield region from an image and estimate the players' locations.

head camera. The utility of the methodology used by [Hunter, 2009] is limited by the use of a custom calibration rig when there is no water on the course, obtrusive attachment of markers to the boat and athlete, and laborious object labeling for extraction of trajectory kinematic information. In contrast, Drory et al. [2017] deploy visual detection and tracking of Slalom paddlers as well as the ordered course obstacles to provide the evidence base pre-requisite to derived race kinematics for analysis of performance, as shown in Figure 1.2 (a).

With regard to other sports, tennis ball tracking techniques are widely deployed for judging if a ball hits inside the field [Yan et al., 2005]. There are also intelligent sports video analysis systems for basketball and soccer games, as demonstrated in Figure 1.2 (b)(c). Most of them focus on player tracking and identification [Xing et al., 2011; Lu et al., 2013], which enable various commercial applications. From the coaching staff's point of view, these technologies can be used to gather game statistics for analyzing their competitors' strength and weakness. TV broadcasting companies also benefit by using such systems to create star-camera-views video streams that highlight star players. Most of these tasks are currently performed by human annotators and automating these processes would significantly increase the production speed and reduce cost.

1.3 Problem Definition

As aforementioned in Section 1.1, visual object tracking is a substantial research area itself and it should be considered and addressed from various perspectives. In this thesis, the focus is on a more specific yet important component, namely model-free online visual object tracking. It aims at tracking generic objects that are initialized manually, or by any other means, at the first frame of the input video. Below, we define this objective in detail and explain its stance in relation to other tracking approaches.

- **Object-Level Tracking**

Our task is to track common objects, such as a pedestrian, a mug, a ball or a bottle, as shown in Figure 1.4. More typical tracking objects can be found as those commonly used in object detection/classification research domain [Everingham et al., 2015; Russakovsky et al., 2015].

In contrast, there is a large group of researchers working on two different levels of visual tracking problems:

(a) Pixel-level tracking or optical flow [Yang and Li, 2015] - to recover image motion at each pixel from spatio-temporal image brightness variations as shown in Figure 1.3 (a);

(b) Feature-level tracking [Shi and Tomasi, 1994; Lucas and Kanade, 1981] - to extract visual features (corners, textured areas) and "track" them over multiple frames.

For the second case, a famous example is the Kanade-Lucas-Tomasi (KLT) [Shi and Tomasi, 1994; Lucas and Kanade, 1981] feature tracker, which is an approach proposed mainly for dealing with the problem that traditional image registration techniques are generally costly. KLT makes use of spatial intensity information to direct the search for the position that yields the best match. It is faster than traditional techniques for examining far fewer potential matches between the images.

Note that an important tracking problem, also called motion segmentation (tracking) [Brox and Malik, 2010], usually takes feature/pixel tracking results as inputs. The motivation is that motion cue is extremely important as pure bottom-up segmentation results from static images are well known to be ambiguous at the object level. However as soon as objects move, the missing link can be established by analyzing the long-term motion traces, as shown in Figure 1.3 (b).

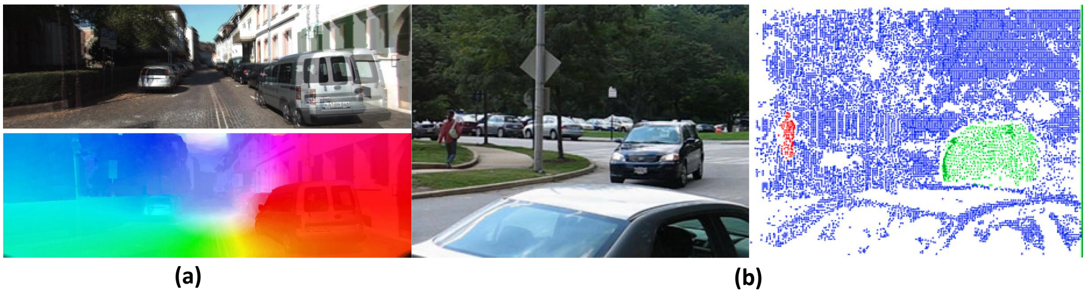


Figure 1.3: (a) A sample result of optical flow from [Yang and Li, 2015]. Top: overlay of two input frames. Bottom: estimated flow (overlayed onto the original image); (b) An example of motion segmentation [Brox and Malik, 2010]. Only motion information was used to successfully separate the car (green) and even the pedestrian (red) from the background.

- **Single-Camera 2D Tracking**

We address single-camera 2D tracking in this thesis. Typical videos from standard benchmarks [Wu et al., 2015; Kristan et al., 2015] are generated by commercial RGB cameras. It could also be thermal cameras as many infrared tracking benchmarks are proposed recently [Felsberg et al., 2015; Patino et al., 2016]. Furthermore, the camera is not always fixed in our problem, thus the motion of the target is harder to be predicted directly.

On the other side, three different dimensions of tracking topics can be found and they are discussed as below:

(a) Fixed-camera tracking [Liao et al., 2016; Denman et al., 2009] - they are widely used for surveillance applications. In these scenarios, static frames allow learning a background model for facilitating object tracking. It is also relatively easier for applying motion detection and optical flow algorithms [Denman et al., 2009] to assist object tracking.

(b) RGB-D camera or 3D tracking [Prisacariu and Reid, 2012; Ren et al., 2014] - RGB-D (in which “D” refers to a “depth” or “distance” channel) data has become conveniently available recently, as various commercial 3D devices released, e.g., Kinect, which is a line of motion sensing input devices by Microsoft for Xbox 360 and Xbox One video game consoles and Windows PCs. It promotes novel research such as: 2D to 3D pose tracking using a known 3D model [Prisacariu and Reid, 2012] and 3D object tracking from RGB-D data [Ren et al., 2014]. Differently, some work focus on automatic recovery of 3D human pose from monocular image sequences [Andriluka et al., 2010], using only 2D data.

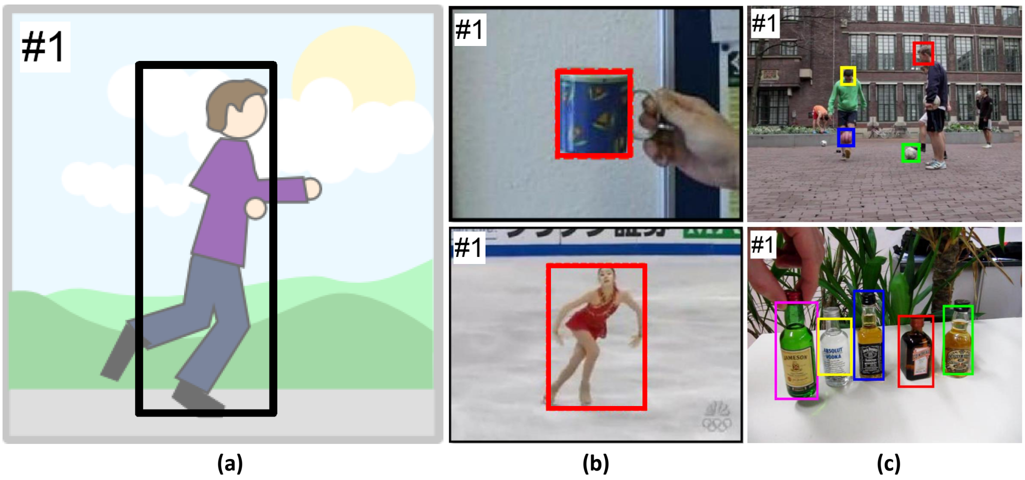


Figure 1.4: (a) A bounding box is initialized on the target at the first frame of an input video either manually by the user or automatically by a category-level object detector, such as DPM [Felzenszwalb et al., 2010]; (b) Initialization examples for single visual object tracking, which are represented using red bounding boxes. Note that no prior knowledge about the category of a target is given before tracking, i.e., the target could be any generic object, such as a mug and a skater as shown in the figure; (c) Initialization examples for multiple visual object tracking, which are represented using bounding boxes of various colors. Note that those targets can be of the same category (bottom) or different categories (top).

(c) Targets association across multiple cameras - multiple camera setting is common for many broadcast systems, such as in sports games like basketball [Shitrit et al., 2014], as more viewing angles can be covered. This naturally raises the computer vision problem of constantly tracking while retaining the identities of multiple targets across those cameras. Typical work can be found: (1) human re-identification Li et al. [2012], which is to match persons observed in non-overlapping camera views; (2) tracking multiple sports players whose paths may intersect repeatedly over long periods of time while retaining their individual identities [Shitrit et al., 2014].

• **Model-Free Online Tracking**

In tracking context, "model-free" [Wu et al., 2015; Kristan et al., 2013] is a widely used term that means no prior information of the target, particularly its class, is available in advance, except for the (manually or automatically) initialized bounding box at the first frame of an input video as shown in Figure 1.4. Since object class is not known, an object detector cannot be applied. In contrast to model-based tracking techniques that are designed and applicable specifically for known object types (such as vehicle, pedestrian, etc.) [Milan et al., 2016] for known applications (such as vehicle traffic monitoring), model-free tracker allows any generic objects and regions.

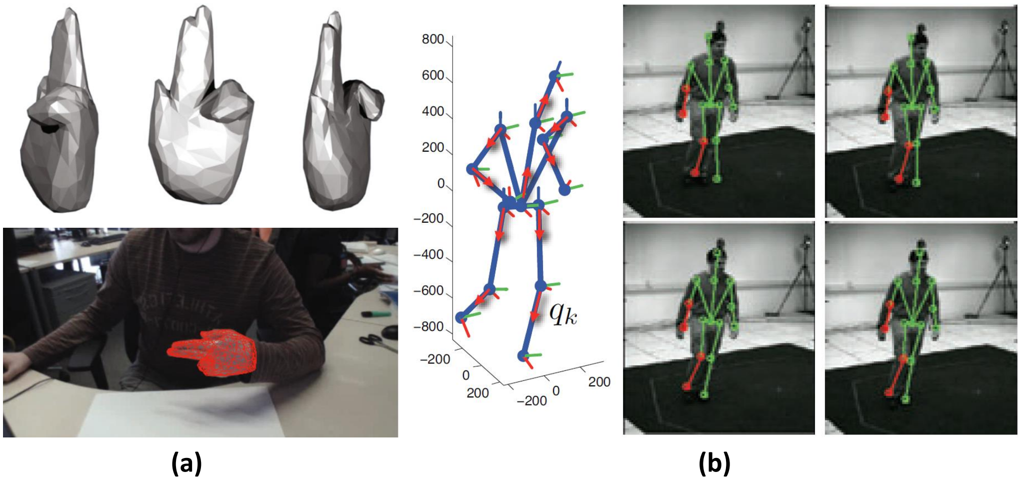


Figure 1.5: Demonstration of two 3D trackers. The first one (a) [Prisacariu and Reid, 2012] is for simultaneous region-based 2D segmentation and 2D to 3D pose tracking, using a known 3D model. The hand pose model on top is trained with the image on the bottom. The second one (b) [Shitrit et al., 2014] tracks 3D human pose only using monocular image evidence. The 3D pose (parametrized joints are marked with arrows) is shown on the left side of (b), with initial pose sequence after 2D-to-3D lifting (right top) and pose sequence after optimization of the 3D pose posterior (right bottom).

“Online tracking" means the tracker is executed in an online manner, ideally following the causal arrow of time, without using any future frame. It is different from offline object tracking where the whole video is available thus tracking can start at any frame toward any direction. Typically, a model capturing the appearance feature of the target would be learned firstly using the initialization bounding box. At the following frames, this model needs to be online updated to adapt to target appearance change, so “online” indicates both the fact that the tracking is carried out in a real-time way and also that the object model is online updated.

Along this line, two relevant tracking formulations can be found in the literature as discussed as below:

(a) Category-specific tracking - track a predefined and type-specific target. In this particular problem, object detector might be used to initialize the target and facilitate the tracking afterwards. The challenge lies in how to efficiently fuse the category-level prior knowledge into the specific object instance to be tracked. Various strategies could be applied such as a periodic regularization imposed by a prior classifier that was learned offline [Drory et al., 2017] or individual-specific detectors obtained through elementary manipulations of the thresholds of a category detector [Hall and Perona, 2014].

(b) Offline video tracking - different from online or real-time object tracking, offline visual tracking problem treats a single input video (or multiple videos) as available data. Typical algorithms generate a set of object bounding boxes in each frame (using category detectors like DPM [Felzenszwalb et al., 2010] or object proposal approaches [Zitnick and Dollár, 2014; Cheng et al., 2014]), then try to supervisely or unsupervisely associate (assign) those bounding boxes based on short-term or long-term consistency cues. They are also popularly named as video object co-localization [Joulin et al., 2014] or video object discovery [Kwak et al., 2015]. Instead, segmentation based object proposal methods [Carreira and Sminchisescu, 2012; van de Sande et al., 2011] can also be utilized. Such work are often known as video object/foreground co-segmentation [Lee et al., 2011].

1.4 Thesis Outline

As mentioned in Section 1.3, this thesis emphasizes on a specific tracking problem, i.e., model-free online visual object tracking. Firstly, we review various tracking strategies and frameworks employed by state-of-the-art trackers [Wu et al., 2015; Kristan et al., 2015] in Chapter 2. Then we introduce background knowledge, including several widely used benchmarks and popular workshop challenges, as well as commonly adopted tracker evaluation methods.

From Chapter 3 to Chapter 5, we present three work for addressing single object model-free tracking in perspectives of background cluster, global tracking with proposals and affine motion tracking, as elaborated in Section 1.4.1. In Chapter 6, we further propose a work for addressing multiple object model-free tracking by taking advantage of a shared pool of proposals, as explained in Section 1.4.2.

Lastly, we highlight and summarize our contribution of this thesis and also propose promising research directions as future work, based on the latest advance and also shortcomings existed in current state-of-the-art trackers, in Chapter 7.

1.4.1 Single Object Tracking

With regard to single object model-free tracking, in Chapter 3, we firstly observe that conventional tracking approaches for visual object tracking often assume that the task at hand is a binary foreground-versus-background classification problem where the background is a single, generic, and all-inclusive class [Zhu et al., 2017]. In contrast, we argue that the background appearance, for the most part, possesses a more complicated structure that would benefit from further partitioning into multiple contextual clusters. We build multiple fine-grained foreground-versus-contextual-cluster models that provide more discriminative classifications, and consequently more robust and accurate foreground object tracking. For each cluster, we employ a Structured output SVM (SSVM), and in an online manner, we combine the responses of multiple classifiers with a top level SSVM that models the tracked foreground object.

Then in Chapter 4, we address another common drawback, i.e., most existed tracking-by-detection methods employ a local search window around the predicted object location in the current frame. They assume the previous location is accurate, the trajectory is smooth, and the computational capacity permits a search radius that can accommodate the maximum speed yet small enough to reduce mismatches. These, however, may not be valid always, in particular for fast and irregularly moving objects. We thus present an object tracker [Zhu et al., 2016a,c] that is not limited to a local search window and has ability to probe efficiently the entire frame. Our method generates a small number of "high-quality" proposals by a novel instance-specific objectness measure and evaluates them against the object model that can be adopted from an existing tracking-by-detection approach as a core tracker. During the tracking process, we update the object model concentrating on hard false-positives supplied by the proposals, which help suppressing distractors caused by difficult background clutters, and learn how to re-rank proposals according to the object model.

Recognizing the importance of the object motion to visual tracking [Smeulders et al., 2014], we further present a novel and reliable object tracking method [Zhu et al., 2015] for image regions that undergo affine transformations such as translation, rotation, scale, dilatation and shear deformations, which span the six degrees of freedom of motion in Chapter 5. Our method takes advantage of the intrinsic Lie group structure of the 2D affine motion matrices and imposes this motion structure on a kernelized Structured output SVM (SSVM) classifier that provides an appearance based prediction function to directly estimate the object transformation between frames using geodesic distances on manifolds, unlike the existing methods proceeding by linearizing the motion.

1.4.2 Multiple Object Tracking

With regard to multiple object model-free tracking, as most previous methods for tracking of multiple objects follow the conventional "tracking by detection" scheme and focus on improving the performance of category-specific object detectors as well as the between-frame tracklet association, these methods are therefore heavily sensitive to the performance of the object detectors, leading to limited application scenarios. In Chapter 6, we overcome this issue by a novel model-free framework [Zhu et al., 2016b] that incorporates generic category-independent object proposals without the need to pre-train any object detectors. In each frame, our method generates a small number of target object proposals that are shared by multiple objects regardless of their category. This significantly improves the search efficiency in comparison to the traditional dense sampling approach. To further increase the discriminative power of our tracker among targets, we treat all other object proposals as the negative samples, i.e. as "distractors", and update them in an online fashion.

1.5 Publication

Conference

Zhu, G.; Porikli, F.; and Li, H., 2016b. Model-free multiple object tracking with shared proposals. In *Asian Conference on Computer Vision (ACCV)*

Zhu, G.; Porikli, F.; and Li, H., 2016a. Beyond local search: Tracking objects everywhere with instance-specific proposals. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Spotlight*

Zhu, G.; Porikli, F.; and Li, H., 2016c. Robust visual tracking with deep convolutional neural network based object proposals on PETS. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*

Kristan, M.; Matas, J.; Leonardis, A.; et al., 2015. The visual object tracking VOT2015 challenge results. In *International Conference on Computer Vision Workshops (ICCVW)*

Felsberg, M.; Berg, A.; Hager, G.; et al., 2015. The thermal infrared visual object tracking VOT-TIR2015 challenge results. In *International Conference on Computer Vision Workshops (ICCVW)*

Zhu, G.; Porikli, F.; Ming, Y.; and Li, H., 2015. Lie-Struck: Affine tracking on Lie groups using structured SVM. In *IEEE Winter conference on Applications of Computer Vision (WACV)*

Zhu, G.; Ming, Y.; and Li, H., 2014. Object category detection by incorporating mid-level grouping cues. In *International Conference on Image Processing (ICIP)*

Zhu, G.; Ming, Y.; and Li, H., 2013. Object cut as minimum ratio cycle in a superpixel boundary graph. In *International Conference on Digital Image Computing: Techniques and Applications (DICTA)*

Journal

Drory, A.; Zhu, G.; Li, H.; and Hartley, R., 2017. Rapid automated detection and tracking of slalom paddlers using cascade classifiers and discriminative correlation filters. *Computer Vision and Image Understanding (CVIU)*, (2017)

Zhu, G.; Porikli, F.; and Li, H., 2017. Not all negatives are equal: Learning to track with multiple background clusters. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, (2017)

Chapter 2

Background and Related Work

In this chapter, we provide an overview of model-free online object (single-camera, 2D) visual tracking problem. Focusing on tracking-by-detection framework, in Section 2.1, we review various tracking methods. Then in Section 2.2, we mention popular benchmark datasets and challenges, as well as the widely used evaluation metrics and protocols. We also briefly introduce an online Structured Support Vector Machine (SSVM) tracking framework in Section 2.3, considering most of our works are built based on it.

2.1 Algorithms

Extensive survey or review papers for visual object tracking [Yilmaz et al., 2006; Li et al., 2013c; Wang et al., 2015b; Wu et al., 2015; Smeulders et al., 2014] can be found along with the fast developing of visual tracking research itself. Especially for the recent ten years, numerous new trackers are proposed each year with the state-of-the-art performance and no single tracker can stay at the top of the benchmarks [Wu et al., 2015] or workshop challenges [Kristan et al., 2015; VOT2016] for long.

In this thesis, we focus the tracker literature review on the modern "tracking-by-detection" framework [Yilmaz et al., 2006; Li et al., 2013c; Wang et al., 2015b]. As shown in Figure 2.1, it is typically composed of motion model, observation model and model updater. Motion model generates a set of candidates which might contain the target in the current frame based on the estimation from the previous frame. Observation model judges whether a candidate is the target based on the features extracted from it. Model updater online updates the observation model to adapt the change of the object appearance. We note that certain tracking systems additionally incorporate an ensemble post-processor component to combine the outputs of their constituent trackers [Zhu et al., 2017; Wang et al., 2015b] .

In the following sections, we firstly review two important learning strategies for training the object appearance model in Section 2.1.1. Then we discuss various work which attempt to address the online model-free tracking problem from alternative perspectives, in Section 2.1.2. Lastly, we go through several deep convolutional neural network based tracking approaches that are recently proposed.

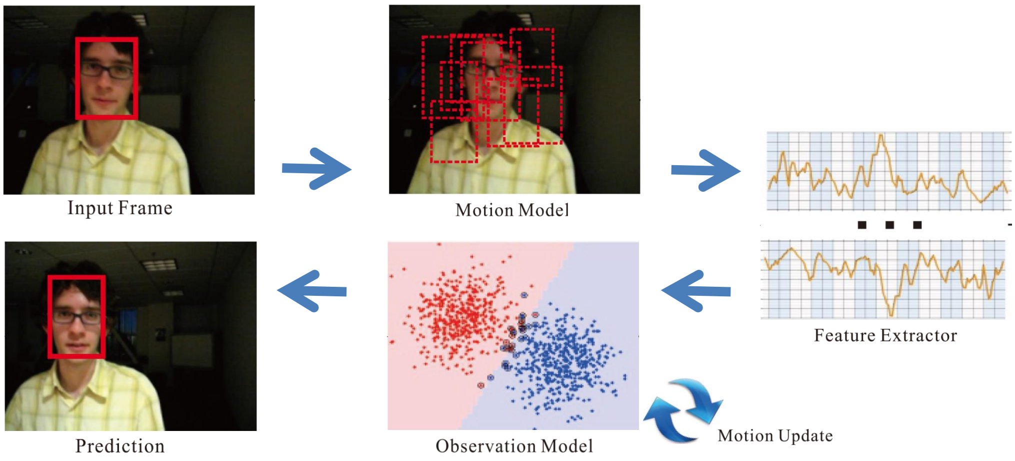


Figure 2.1: Pipeline of a typical modern visual tracking system [Wang et al., 2015b], whose success is widely corroborated by benchmark evaluations [Kristan et al., 2016; Wu et al., 2015; Smeulders et al., 2014].

2.1.1 Generative and Discriminative based Object Model Learning

Typically speaking, two object appearance model learning strategies can be found in the tracking literature [Li et al., 2013c; Wang et al., 2015b; Wu et al., 2015; Smeulders et al., 2014], i.e., generative and discriminative learning approaches, respectively. Generative learning based models mainly concentrate on how to construct an object representation in specific feature spaces, while discriminative learning based appearance models aim to maximize the inter-class separability between the object and background regions using discriminative learning techniques.

• Generative Learning based Methods

To be specific, for generative learning based methods, a vast scope of visual tracker can be found and we summarize them as below:

(1) Holistic templates (e.g., based on raw intensity values) are used for tracking [Matthews et al., 2004] since the early work of Lucas and Kanade (LK) [Lucas and Kanade, 1981]. They do not take large appearance variability into account and hence result in poor performance when the visual properties of a target object change significantly.

(2) Subspace-based tracking approaches [Hager and Belhumeur, 1998; Ross et al., 2008] are proposed to better account for appearance changes, applying low-dimensional representations for tracking, which is found to be more robust to illumination [Hager and Belhumeur, 1998] and appearance variation [Ross et al., 2008].

(3) Sparse representation is another popular object model for generative model based trackers. As a well known work, [Mei and Ling, 2011] used a dictionary of holistic intensity templates composed of target and trivial templates, and determined the target location by solving multiple $\ell 1$ minimization problems. To better handle occlusion and improve run-time performance, [Zhong et al., 2012] proposed a collaborative tracking algorithm that combined a sparsity-based discriminative classifier and a sparsity-based generative model.

(4) Color histograms are widely deployed as appearance descriptions [Comaniciu et al., 2003; TeuliÃÍre et al., 2011; Gomez-Balderas et al., 2013] among earlier tracking approaches. [Comaniciu et al., 2003] applied the mean shift algorithm to object tracking on the basis of a color histogram and Collins [Collins, 2003] extended the mean shift tracking algorithm to deal with the scale variation of target objects. It however yielded inferior performance on recent benchmarks [Wu et al., 2015], as color measurements can vary significantly over an image sequence [Danelljan et al., 2014b].

• **Discriminative Learning based Methods**

A known drawback of the generative model based methods is that they often ignore the influence of the background, and consequently suffer from distractions caused by the background regions with similar appearance to the foreground object. In contrast, discriminative learning based appearance models are trained using both the object and background regions. Various classifiers can be employed as shown in the literature, such as Support Vector Machine (SVM) [Avidan, 2004], Structured output SVM (SSVM) [Hare et al., 2011], boosting [Grabner et al., 2006] and online multi-instance boosting [Babenko et al., 2009].

An inspiring example is proposed by [Possegger et al., 2015], which applied an efficient color-histograms based discriminative object model to identify potentially distracting regions in advance. This knowledge is then exploited to adapt the object representation beforehand so that distractors are suppressed and the risk of drifting is significantly reduced. It achieved state-of-the-art performance on standard large benchmarks [Kristan et al., 2014; Kristan et al., 2013]. Comparatively, earlier tracking approaches [Comaniciu et al., 2003; PÃl'rez et al., 2002] employing color histograms tend to drift towards nearby regions with similar appearance [Possegger et al., 2015] as we mentioned before. This demonstrates the advance when the background information is considered when training the object appearance model.

Another recent breakthrough comes from the Discriminative Correlation Filter (DCF) based approaches [Danelljan et al., 2014b; Henriques et al., 2012; Danelljan et al., 2014a; Hong et al., 2015b]. These methods learn a correlation filter by performing a circular sliding window operation on the training samples, which then

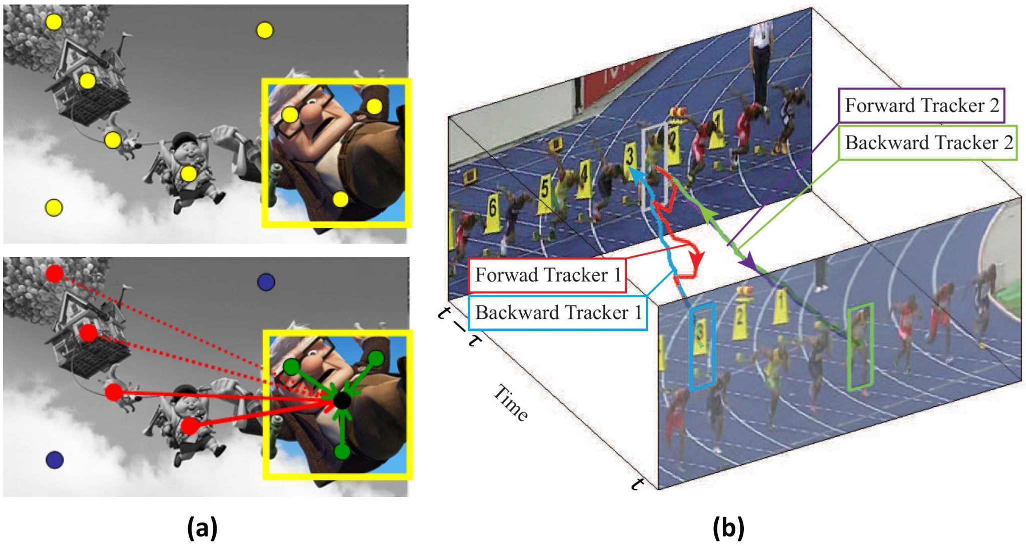


Figure 2.2: Two different ways of exploiting the contextually supportive information: spatial and temporal. (a) Spatial support: (top) a frame with the target object marked; (Bottom) spatial supporters which are features that vote for the position of the object, since their motion appears correlated. They can belong to the object itself (green) or not (red). Uncorrelated features (blue) are discarded. (b) Temporal support: the forward-backward trajectory analysis, where the purple forward tracker is successful and the red one is not.

facilitates efficient training and detection with the Fast Fourier Transform (FFT). Extensive trackers that deploy this scheme can be found in recent literature [Danelljan et al., 2015; Henriques et al., 2015; Danelljan et al., 2014a; Hong et al., 2015b] and they are among the top performed trackers on standard benchmarks [ALOV300; Kristan et al., 2015]. For example, [Danelljan et al., 2015] introduced a spatial regularization component to penalize correlation filter coefficients, depending on their spatial locations. This allows the correlation filters to be learned on a significantly larger set of negative training samples, without corrupting the positive samples. Notably, it is the best tracker that did not exploit additional tracking data for offline learning in the VOT2015 challenge [Kristan et al., 2015].

2.1.2 Alternative Approaches

Since model-free online visual object tracking is a challenging high-level problem, similar to object detection and classification [Russakovsky et al., 2015; Everingham et al., 2015], there are numerous tracking work which attempted to address this task from different perspectives as summarized below:

• **Tracking with Spatially Contextual Support**

Spatial context is widely and successfully used in detection and segmentation approaches [Divvala et al., 2009; Mottaghi et al., 2014]. For object tracking, there are also work trying to take advantage of it. Towards incorporating larger receptive fields, [Yang et al., 2009] proposed a context-aware tracking algorithm that considers a set of auxiliary objects as the spatial context of the foreground. These auxiliary objects need to satisfy conditions such as persistent co-occurrence with the foreground and consistent motion correlation. These conditions may not be easily satisfied in practice.

[Grabner et al., 2010; Dinh et al., 2011; Possegger et al., 2015] used similar concepts termed as "distracters" and "supporters". Distracters Dinh et al. [2011]; Possegger et al. [2015] are regions that have similar appearance as the target, and supporters Grabner et al. [2010]; Dinh et al. [2011] are regions or features around the target with consistent co-occurrence and motion correlation in a short time span as shown in Figure 2.2 (a). These methods require careful maintaining models for distracters and supporters. [Li et al., 2011] showed that the high-order contextual information from samples can increase the robustness of the classifier to noise. The high-order context is defined as a group of samples having some common properties. Each sample in the high-order context is influenced by other samples in the same high-order context. For their tracker, the similarity measure depends on not only two individual samples but also their corresponding contexts.

• **Tracking with Temporally Contextual Support**

The major differences between object detection and object tracking can be summarized in two aspects: (1) tracking is carried on a specific object instance, while detection is always for category-level objects; (2) trackers run on ordered image sequences, while a detector can be applied on any single image. Respectively, temporal context could potentially be an important cue for object tracking, if could be exploited.

In the literature, [Lee et al., 2015] proposed a framework to trace a target forwardly and backwardly over a time interval as shown in Figure 2.2 (b), with multiple component trackers. Then by analyzing the pair of the forward and backward trajectories and measuring the robustness with the geometry similarity, the cyclic weight, and the appearance similarity from the forward and backward trajectories, the optimal component tracker which yielded the maximum robustness score was selected and its forward trajectory was used as the final tracking result.

Alternatively,[Zhang et al., 2014a] used a tracker and its historical snapshots to constitute an expert ensemble, where the best expert was selected to restore the current tracker when needed based on a minimum entropy criterion, so as to correct undesirable model update. The base tracker in their formulation exploited an online SVM on a budget algorithm and an explicit feature mapping method was used for efficient model update and inference.

- **Parts and Segmentation based Tracking**

Parts based object representation such as DPM [Felzenszwalb et al., 2010] is widely found in object detection/localization research domain [Bourdev et al., 2010] as conventional bounding box representation inevitably incorporates background noise into the model, especially for non-rectangular shapes. This issue is however significantly alleviated due to the high flexibility of the parts based model.

In the visual object tracking literature, parts based model is mainly deployed to handle the occlusion and appearance change of the target [Jia et al., 2012; Cai et al., 2013; Li et al., 2015; Zhang and van der Maaten, 2014]. Among them, [Li et al., 2015] attempted to identify and exploit the reliable patches that could be tracked effectively through the whole tracking process, using a probability model under a sequential Monte Carlo framework. [Zhang and van der Maaten, 2014] proposed a novel multi-object model-free tracker by incorporating spatial constraints between the parts (or objects). The spatial constraints were learned along with the part (object) detectors using an online structured SVM algorithm.

Segmentation-based representation is another popular way to address the non-rectangular object shape and articulated deformation of the target in visual tracking [Ren and Malik, 2007; Godec et al., 2011; Duffner and Garcia, 2013; Wang et al., 2011], e.g., to track sports players in a broadcast video as shown in Figure 1.2. Although it has shown promising progress for video object segmentation by generating object region proposals and link them across frames, in recent literature [Grundmann et al., 2010; Lee et al., 2011], segmentation based approaches are typically computationally intensive. Moreover, it is also challenging for those methods to deal with cluttered background and occlusions during the tracking, which leads to unstable results.

2.1.3 Deep Convolutional Neural Networks based Tracking

Although significant achievements have been obtained by deep Convolutional Neural Networks (CNNs) for object detection and classification tasks Everingham et al. [2015]; Russakovsky et al. [2015], there are comparably limited adaptations of CNNs for tracking task and most CNNs based trackers use such networks to learn better features [Zhu et al., 2016c; Hong et al., 2015a]. In their pioneering work, Li et al. [2014] employed a candidate pool of multiple CNNs as a data-driven model of different instances of the target object. Inspired by this, [Ma et al., 2015a] interpreted the hierarchies of convolutional layers as a nonlinear counterpart of an image pyramid representation and adaptively learned correlation filters on each convolutional layer to encode the target appearance.

Wang et al. [2015a] made a similar observation. They found that the top layer of the convolutional neural networks encoded more semantic category-level information while the lower layer carried more instance-level discriminative information used to separate the target and distractors from the background. In their work, a General Network (GNet) that captured the category information of the target was built on top of the selected feature maps of the conv5-3 layer. and a Specific Network (SNet) that discriminated the target from background with similar appearance was built on top of the selected feature maps of the conv4-3 layer. Both GNet and SNet were initialized in the first frame to perform foreground heat map regression for the target and adopted different online update strategies.

[Hong et al., 2015a] maneuvered similarly on the feature map generated from the deep CNN layers. It drew samples near the target location in the previous frame and extracted feature descriptors using a pre-trained CNNs model. To deal with the spatial information loss due the pooling operation, they further adopted the target specific feature map generated by back-projecting the information corresponding to the identified label to visualize the region of interest. An online SVM was employed to classify each sample then those positive samples were used to construct the salience map.

The most noticeable work is [Nam and Han, 2016], which pre-trained a CNNs network using videos found on the tracking benchmarks [Kristan et al., 2015; Wu et al., 2015] with ground truth trajectories, instead of using object detection dataset [Russakovsky et al., 2015] or an offline learned network [Krizhevsky et al., 2012; Szegedy et al., 2013; Girshick, 2015] like other CNNs tracker mentioned above. Their network was composed of shared layers and multiple branches of domain-specific layers. They trained the network with respect to each domain iteratively to obtain generic target representations in the shared layers. Note that this tracker achieved the best result among a large number of submitted trackers on the latest VOT2015 challenge [Kristan et al., 2015].

2.2 Datasets

Inspired by object classification [Everingham et al., 2015; Li et al., 2006; Martin et al., 2001; Russakovsky et al., 2015] where standard evaluation benchmarks have been established from earlier ages, visual tracking community, albeit only recently, started to adopt large-scale benchmarks for performance evaluation [Wu et al., 2013, 2015; Smeulders et al., 2014] .This delay is partially due to the fact that the tracking objective can be defined in different ways as elaborated in Section 1.3. Besides, preparing labeled videos takes much more effort, thus some benchmarks [Kristan et al., 2014; Kristan et al., 2015] resorted augmentation such as rotated bounding boxes to provide highly accurate ground-truth values for comparing results.

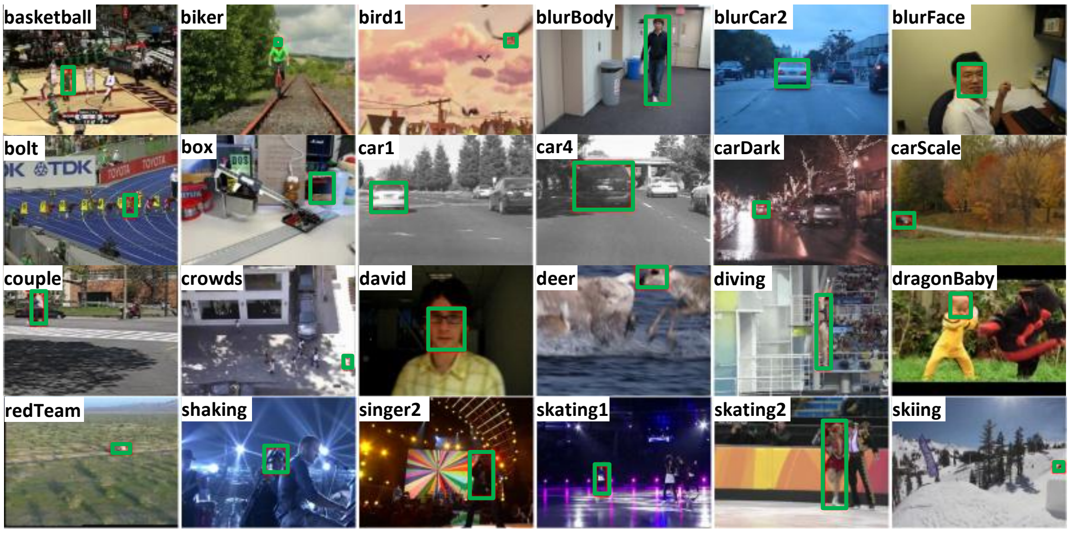


Figure 2.3: Sample sequences with ground truths illustrated as green bounding boxes from the TB50 benchmark dataset [Wu et al., 2015], which contains 50 difficult and representative testing instances (selected from TB100 [Wu et al., 2015]) for experimental evaluation with detailed attributes annotated.

Earlier tracking dataset either focused on high-level event interpretation algorithms, such as PETS (Performance Evaluation of Tracking and Surveillance) [Ferryman and Shahrokni, 2009], or emphasized on evaluation of surveillance systems and event detection, e.g., CAVIAR, i-LIDS, ETISEO, or specialized on tracking specific objects like faces Kasturi et al. [2009] and sports analytics (CVBASE).

In comparison, two notably large benchmarks are recently proposed: the visual Tracker Benchmark (TB50 and TB100) by [Wu et al., 2013, 2015] and the experimental survey – Amsterdam Library of Ordinary Videos (ALOV) by [Smeulders et al., 2014]. Both benchmarks compare a significant number of recent trackers using the source code obtained from the original authors. We review the details and differences between them in the following parts.

- **Visual Tracker Benchmark**

Arguably, this benchmark [Wu et al., 2013, 2015] enabled the first large-scale evaluation of model-free visual object trackers. It builds a toolkit with standard protocols for comparing recently published algorithms. Earlier tracking video sequences were often not supported by ground-truth annotations. The reported quantitative results in the literature were inconsistent since the trackers are not initialized with a consistent protocol and evaluated on the same platform. To facilitate a fair performance evaluation, the visual tracker benchmark collected and annotated most of the commonly used tracking sequences.

An earlier version of the visual tracker benchmark, named as Online Tracking Benchmark (OTB) [Wu et al., 2013], contains 50 video sequences with full bounding box annotations. The total number of frames is more than 29000, and for a particular sequence, this number varies from tens to thousands, e.g. *deer* (71 frames), *skiing* (81 frames), *dog1* (1350 frames), *doll* (3872 frames), etc. A later work [Wu et al., 2015] expanded the number of sequences to 100: visual Tracker Benchmark (TB100). Since some of the targets are similar or less challenging, they also selected 50 difficult and representative ones: TB50 dataset, for an in-depth analysis. Note that as humans are the most important target in practice, the TB100 dataset contains more sequences of this category (36 body and 26 face/head videos). Sample sequences with ground truth annotations are shown in Figure 2.3.

To further analyze the strength and weakness of a tracker, the visual tracker benchmark additionally annotate each sequence globally with various visual attributes [Wu et al., 2015]. Some common attributes are:

(1) Deformation - non-rigid object deformation.

(2) Background Clutters - the background near the target has similar color or texture as the target.

(3) Illumination Variation - the illumination in the target region is significantly changed.

(4) Fast Motion - the motion of the ground truth is larger than t_m pixels ($t_m = 20$).

(5) Low Resolution - the number of pixels inside the ground-truth bounding box is less than t_r ($t_r = 400$).

(6) Motion Blur - the target region is blurred due to the motion of target or camera.

(7) Occlusion - the target is partially or fully occluded.

Note that in this benchmark, individual sequence is not per-frame annotated. For example, a sequence has the *occlusion* attribute if the target is occluded at any frame in the sequence. Although many factors could contribute to a tracker's performance, these attributes help us to diagnose it in a more detailed way.

- **ALOV300 Benchmark**

As we mentioned, the tracking videos in TB100 are mostly collected from existed literature and targets are largely human-related such as a body, a face and a head. Furthermore, many videos are obtained under well-controlled conditions, e.g., several of them are particularly recorded by the researchers in their offices. Differently, the Amsterdam Library of Ordinary Videos, ALOV300 [Smeulders et al., 2014], aims to cover as diverse circumstances as possible: illuminations, transparency, specularity, confusion with similar objects, clutter, occlusion, zoom, severe shape changes, motion patterns, low contrast, and so on (thirteen aspects [Chu and Smeulders, 2010]).

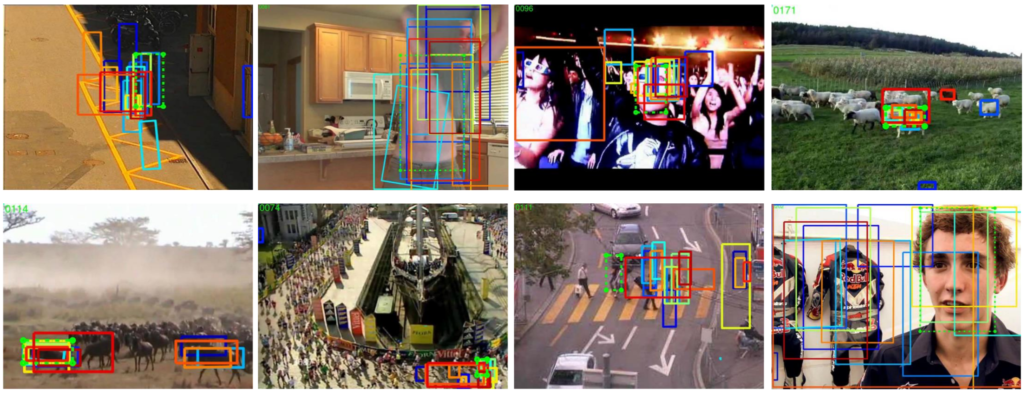


Figure 2.4: Sample sequences from the ALOV300 dataset [Smeulders et al., 2014]. We overlap them with results of trackers to illustrate the difficulties of this benchmark. 315 testing instances are employed for experimental evaluation with 13 attributes [Chu and Smeulders, 2010] annotated.

Preference is given to assorted short videos over longer ones to maximize the diversity, while composing the ALOV300 dataset. The dataset consists of 315 video sequences as shown in Figure 2.4, whose main source is real-life videos from YouTube with 64 different types of targets ranging from a human face, a person, a ball to a can. The average length of normal videos is 9.2 seconds with a maximum of 35 seconds. One additional category contains ten long videos with a duration between one and two minutes, which includes three videos (car and motorbike) from [Kalal et al., 2012] and three videos from the 3DPeS dataset [Baltieri et al., 2011] with varying illumination conditions and complex-motion objects.

The total number of frames in ALOV300 is 89364 and the data are annotated by a rectangular bounding box every fifth frame. In rare cases, when motion is rapid, the annotation is more frequent. The ground truth of the intermediate frames are then acquired by linear interpolation to reduce human manual labeling effort. ALOV300 is publicly available and new results can also be uploaded to their website for directly comparing to other participated trackers .

2.2.1 Workshop Challenges

Despite the success of standard evaluation benchmarks, such as TB100 [Wu et al., 2015] and ALOV300 [Smeulders et al., 2014], they however present certain limits. For example, on both benchmarks, the attributes are annotated globally although they may occupy only a short sub-sequence of frames in a video. To be specific, a sequence is annotated as “occlusion” if the target is occluded anywhere in the sequence, while it does not usually last throughout the entire sequence. This leads to inaccurate evaluation of a specific attribute, e.g., an occlusion might occur at the end of the sequence, while the poor performance is in fact due to some other effects occurring at the beginning of the sequence.

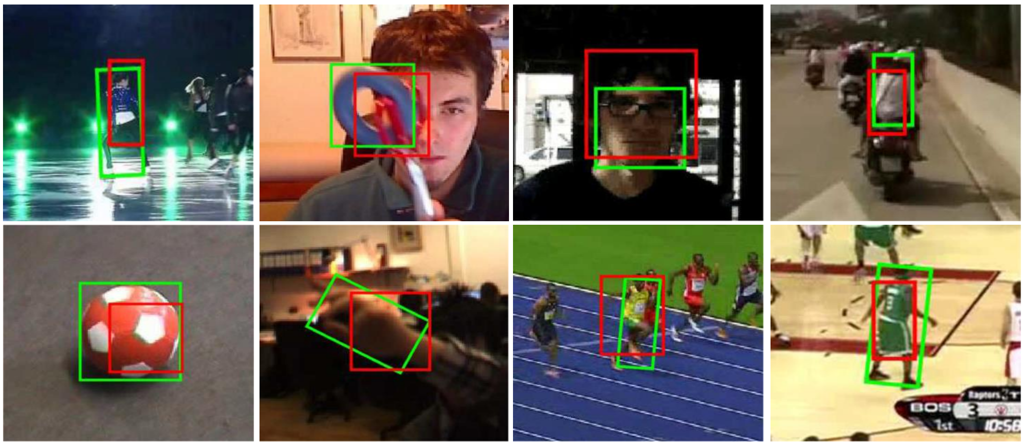


Figure 2.5: Sample video frames from the VOT challenge [Kristan et al., 2016]. The red bounding boxes are examples that overlap (Intersection-over-union, IoU) the ground truth (green) at 0.5, which is a threshold used to indicate the failure of a tracker [Wu et al., 2015; Smeulders et al., 2014].

More importantly, those standard benchmarks lack an efficient way for dynamically managing the trackers participated for comparison, as they were picked manually by the authors and there are no easy ways to add and update them. In contrast, as mentioned in Chapter 1, visual object tracking is a highly attractive and active research domain with a consistently high number of work published in high prole conferences every year (∼40 papers [Kristan et al., 2013]). It is thus extremely important for a benchmark to provide a mechanism for maintaining the results, like those object detection challenges [Everingham et al., 2015; Russakovsky et al., 2015].

Recognizing those issues, the Visual Object Tracking (VOT) workshop challenge [Kristan et al., 2016, 2015; Kristan et al., 2014; Kristan et al., 2013] were organized to provide an evaluation platform that goes beyond the current state-of-the-art. In particular, they have compiled a labeled dataset collected from widely used sequences showing a balanced set of various objects and scenes. Several features in benchmarking short-term trackers were introduced through these challenges and we summarize them below:

(1) The most active single-object model-free visual tracking benchmark, continually organized from VOT2013 [Kristan et al., 2013] to VOT2016, jointly with top-tier conferences (ICCV and ECCV) and 70 trackers were evaluated in the latest challenge.

(2) The dataset is fully annotated with rotated bounding boxes to more faithfully denote the target position as shown in Figure 2.5 and all the sequences are labeled per-frame with visual attributes. This is crucial to facilitate in-depth analysis, as the performance measures computed from global attribute annotations [Wu et al., 2015;

Smeulders et al., 2014] are significantly biased toward the dominant attributes in the sequence, while the bias is reduced with per-frame annotation, even in presence of miss annotations [Kristan et al., 2015].

(3) The latest challenge, VOT2016, includes 60 sequences through an automatic sequence selection protocol from an original pool of 356 sequences. They are then automatically clustered (using k-means clustering) according to their similarity in terms of various globally calculated sequence visual attributes [Kristan et al., 2015].

• Thermal Infrared Visual Tracking Challenge

In comparison, the Thermal Infra-Red Visual Object Tracking (VOT-TIR) challenge [Felsberg et al., 2015; VOT-TIR2016] aims at comparing short-term single-object visual trackers that work on thermal infrared sequences. The main advantages of thermal cameras are their ability to see in total darkness, their robustness to illumination changes and shadow effects, and reduced privacy intrusion. As these cameras improve in image quality and resolution while decrease in both price and size, they have been commonly used in various applications [Gade and Moeslund, 2014], e.g., cars, surveillance systems and for military purposes. The VOT-TIR challenge has been featured as a sub-challenge to the VOT challenges [Kristan et al., 2015].

2.2.2 Evaluation Protocol

In this section, we discuss the evaluation methodologies employed by TB100 [Wu et al., 2015], ALOV300 [Smeulders et al., 2014] and VOT challenges [Kristan et al., 2015, 2016], respectively. To be specific, they are organized in two categories: evaluation strategies and evaluation metrics.

• Evaluation Methods

The most straightforward way to evaluate a tracker is to initialize the tracker in the first frame using the ground truth annotation as shown in Figure 1.4, then let it run until the end of a sequence. This strategy is employed by benchmarks TB100 [Wu et al., 2015] and ALOV300 [Smeulders et al., 2014]. Some evaluation metrics (such as Intersection-over-Union (IoU) [Everingham et al., 2015] that is widely used in object detection benchmarks) can then be applied to evaluate how well the results match with the ground truths, reflecting the performance of the tracker.

This evaluation approach is referred as One-Pass Evaluation (OPE) in TB100 [Wu et al., 2015]. Although it is simple to be applied, this method has two major drawbacks. Firstly, a tracking algorithm may be sensitive to initialization in the first frame, and its performance with different initial states or frames may vary significantly. Secondly, most algorithms do not have re-detection or target recovery mechanisms and the tracking results after tracking failures do not provide meaningful information.

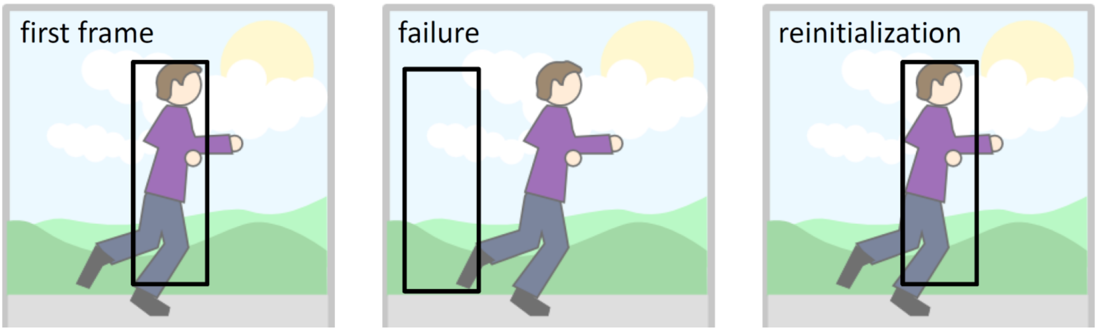


Figure 2.6: VOT challenges [Kristan et al., 2016] employ a re-initialization evaluation scheme. After the tracker loses the target during tracking, which is the case when the overlap measure (IOU) with the ground truth becomes zero, the tracker is re-initialized five frames after the failure. This scheme measures the robustness of trackers by counting how many times they fail in a sequence.

To understand the importance of an initial bounding box for tracking, [Smeulders et al., 2014] investigated the stability of the trackers by shifting the initial target bounding box by 20% of the width to the right so that the target was partially covered. This experiment demonstrates a tracker's robustness to initialization misalignment. Furthermore, [Wu et al., 2015] proposed two metrics to analyze whether a tracking algorithm is robust to different initialization states by perturbing them temporally (i.e., starting at different frames) or spatially (i.e., starting with different bounding boxes), referred as Temporal Robustness Evaluation (TRE) and Spatial Robustness Evaluation (SRE), respectively.

To solve these issues, VOT challenges [Kristan et al., 2016] employ a novel re-initialization evaluation scheme instead, as shown in Figure 2.6. After the tracker loses the target during tracking, the tracker is re-initialized five frames after the failure. The failure is typically determined by the IoU metric [Everingham et al., 2015] when it becomes zero with the ground truth. This scheme allows measuring the robustness of a tracker by counting how many times it failed in a sequence. Note that it also provides a more efficient way to utilize the video data, as most trackers are not expected to perform re-detection [Kristan et al., 2016] while the values of performance measures become irrelevant after the point of tracking failure and including them in the computation of a global performance measure introduces significant distortions as we discussed before.

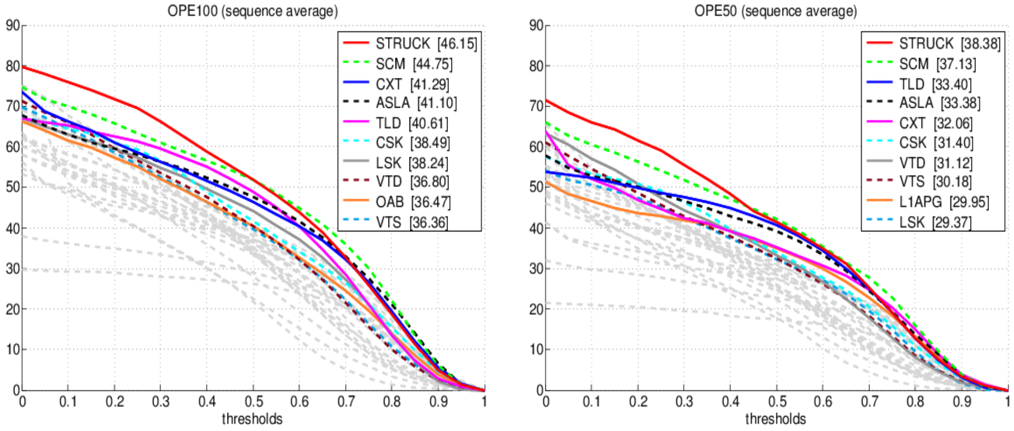


Figure 2.7: *Success plots* of various trackers on TB100 and TB50 [Wu et al., 2015]. The y and x axis are success rates and overlap (IoU) thresholds respectively. The trackers on the legend are ranked using the Area Under the Curve (AUC). Note that TB50 is significantly harder than TB100 as we mentioned in Section 2.2, with roughly 8 percentages down for the top-performed tracker, Struck [Hare et al., 2011].

- **Evaluation Metrics**

To evaluate the performance of a tracker against others, a large group of work including TB100 [Wu et al., 2015] and ALOV300 [Smeulders et al., 2014], choose several basic measures. Typical metrics include center error, region overlap (IoU), failure rate [Wu et al., 2015] and F-score [Smeulders et al., 2014], or more sophisticated measures, such as CoTPS [Carvalho et al., 2012; Nawaz and Cavallaro, 2013], which combines several measures. A nice property of the combined measures is that they provide a single score to rank the trackers. A downside is that they offer little insight into the tracker performance which limits their interpretability. Furthermore, all measures strongly depend on the experimental setup within which they are computed.

Specifically, [Wu et al., 2015] uses graphic plots to visually show the percentage of frames for which the estimated object location is within a certain metric threshold of the ground truth. One such an metric is *precision*, which measures the object location accuracy in terms of center error. Alternatively, *success plots* use the region overlap (IoU [Everingham et al., 2015]) instead, as shown in Figure 2.7. A drawback of performance plots is that they typically become cluttered when comparing several trackers on several sequences in the same plot. To address this, Smeulders et al. [Smeulders et al., 2014] calculate a performance measure per sequence for a tracker and order these values from highest to lowest, thus obtaining a so-called survival curve. The performance of several trackers is then compared on the entire dataset by visualizing their survival curves.

Another group of work use ranking-based methodologies [Pang and Ling, 2013; Everingham et al., 2015; Kristan et al., 2016]. Especially in [Pang and Ling, 2013], as previous evaluation work may have subjective biases towards the new tracker which typically performs the best as well as the difficulty to optimally tune all its competitors and sometimes the selected testing sequences, they proposed a novel approach towards inhibiting subjective bias in evaluating trackers by analyzing the results between the "second bests", which were widely collected from existed tracking papers published in major computer vision venues in recent years. In VOT challenges, [Kristan et al., 2016] further introduced the concept of equally-ranked trackers. For each tracker, a group of so-called equivalent trackers containing trackers performing indistinguishably was determined and a corrected rank was then calculated.

2.3 Online Structured SVM Tracking

Our works do not particularly prefer a certain type of classifiers as the core tracking component, although we choose a recently successful and efficient structured outputted SVM framework [Hare et al., 2011; Babenko et al., 2009], since it integrates the learning and tracking, avoiding the need for ad-hoc update strategies widely used in conventional trackers [Wu et al., 2015]. Note that other object models could be easily incorporated, e.g., we deployed a normalized cross correlation (NCC) template matching method to investigate the efficiency of the instance-specific object proposal approach while working with simple object models in Chapter 4.

Support vector machine technique [Cortes and Vapnik, 1995] has been one of the most commonly-used classification tools in machine learning and computer vision. It usually takes a set of training examples and learns a hyperplane defining the decision boundary, which is then used to make predictions. Recently, the SVM has been extended beyond classification so that it can also be used for structured prediction problems [Tsochantaridis et al., 2005; Babenko et al., 2009]. In tracking scenario, given an estimated 2D bounding box position $\mathbf{p}_{n-1}$ containing the target object at frame $n-1$, $\mathbf{x}_n^{\mathbf{p}_{n-1}}$ denotes the region of image within the bounding box at frame n. Object tracking is then formulated as learning a prediction function $f : \mathcal{X} \rightarrow \mathcal{Y}$, which directly estimates the object transformation $\mathbf{y}_n$ between frame $n-1$ and n [Hare et al., 2011]. A discriminant function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ is incorporated to achieve this:

$$\mathbf{y}_n = f(\mathbf{x}_n^{\mathbf{p}_{n-1}}) = \arg\max_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{x}_n^{\mathbf{p}_{n-1}}, \mathbf{y}). \tag{2.1}$$

F should give a large value to pairs $(\mathbf{x}, \mathbf{y})$ that are well matched. Then the estimated bounding box position at frame n is obtained as

$$\mathbf{p}_n = \mathbf{p}_{n-1} \circ \mathbf{y}_n. \tag{2.2}$$

Following the structured output SVM framework of [Tsochantaridis et al., 2005; Babenko et al., 2009], we represent discriminant function as the form of $F(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) = \langle \mathbf{w}, \phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) \rangle$, where $\phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y})$ is a joint kernel mapping implicitly defined by the kernel identity $k_{xy}((\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}), (\mathbf{x}_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}})) = \langle \phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}), \phi(\mathbf{x}_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}}) \rangle$. Given a set of example pairs $\{(\mathbf{x}_1^{\mathbf{p}_0}, \mathbf{y}_1), \dots, (\mathbf{x}_n^{\mathbf{p}_{n-1}}, \mathbf{y}_n)\}$, the following minimising problem can be formulated in order to learn the discriminant function:

$$
\begin{aligned}
\min_{\mathbf{w}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i, \\
\text{s.t.} \quad & \xi_i \geq 0, \quad \forall i \\
& \langle \mathbf{w}, \Delta\phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) \rangle \geq \mathcal{L}(\mathbf{y}_i, \mathbf{y}) - \xi_i, \quad \forall i, \forall \mathbf{y} \neq \mathbf{y}_i,
\end{aligned}
\tag{2.3}
$$

where $\Delta\phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) = \phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}_i) - \phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y})$ and C is a constant blending weight for the soft-margin errors. $\mathcal{L}(\mathbf{y}_i, \mathbf{y})$ is a loss function which decreases as a possible output $\mathbf{y}$ approaches the true output $\mathbf{y}_i$.

Note that the optimization of convex function (2.3) does not differ from usual SVM primal formulation except that there can be an infeasible large number of constraints (the number of training samples multiplies the size of the output space), which can even become infinite. However, not all constraints will be active at any time as demonstrated as follows:

$$
\xi_i \geq \max_{\mathbf{y} \neq \mathbf{y}_i} \mathcal{L}(\mathbf{y}_i, \mathbf{y}) - \langle \mathbf{w}, \Delta\phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) \rangle, \quad \forall i. \tag{2.4}
$$

Thus the minimization of (2.3) can be optimized by *constraint generation*, i.e., estimate $\mathbf{w}$ using fixed subsets of constraints and then add new constraints by finding the $\mathbf{y}$ that maximizes the right-hand side of (2.4). This alternation is repeated until convergence, generally with a smaller set of constraints Tsochantaridis et al. [2005]; Babenko et al. [2009] compared to the number of constraints in (2.3).

Dual Problem

Using standard Lagrangian duality techniques, (2.3) can be converted into its equivalent dual form Tsochantaridis et al. [2005]; Bordes et al. [2007, 2008]; Hare et al. [2011]:

$$
\begin{aligned}
\min_{\beta} \quad & \sum_{i,\mathbf{y}} \mathcal{L}(\mathbf{y}_i, \mathbf{y})\beta_i^{\mathbf{y}} + \frac{1}{2}\sum_{i,\mathbf{y},j,\bar{\mathbf{y}}} \beta_i^{\mathbf{y}}\beta_j^{\bar{\mathbf{y}}} \langle \phi(x_i^{\mathbf{p}_{i-1}}, \mathbf{y}), \phi(x_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}}) \rangle, \\
\text{s.t.} \quad & \beta_i^{\mathbf{y}} \leq C\delta(\mathbf{y}_i, \mathbf{y}), \quad \forall i, \forall \mathbf{y} \\
& \sum_{\mathbf{y}} \beta_i^{\mathbf{y}} = 0, \quad \forall i
\end{aligned}
\tag{2.5}
$$

where $\delta(\mathbf{y}_i, \mathbf{y}) = 1$ if $\mathbf{y}_i = \mathbf{y}$ and 0 otherwise, C is the same constant as in (2.3). Following Bordes et al. [2007, 2008], we refer to those pairs $(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y})$ for which $\beta_i^{\mathbf{y}} \neq 0$ as support vectors. Only the support vector $(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}_i)$ will have $\beta_i^{\mathbf{y}_i} > 0$ due to the constraints in (2.5), while any other support vector will have $\beta_i^{\mathbf{y}} < 0$, $\mathbf{y} \neq \mathbf{y}_i$. They are referred as positive and negative support vectors respectively.

The discriminant function can also be represented in the dual form as

$$\mathbf{w} = \sum_{j,\bar{\mathbf{y}}} \beta_j^{\bar{\mathbf{y}}} \phi(x_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}}), \tag{2.6}$$

$$F(x_i^{\mathbf{p}_{i-1}}, \mathbf{y}) = \sum_{j,\bar{\mathbf{y}}} \beta_j^{\bar{\mathbf{y}}} \langle \phi(x_i^{\mathbf{p}_{i-1}}, \mathbf{y}), \phi(x_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}}) \rangle. \tag{2.7}$$

Then the maximizing problem in the right-hand side of (2.4) is expressed as

$$\begin{aligned} &\max_{\mathbf{y}\neq\mathbf{y}_i} \mathcal{L}(\mathbf{y}_i, \mathbf{y}) + \langle \mathbf{w}, \phi(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) \rangle \\ &= \max_{\mathbf{y}\neq\mathbf{y}_i} \mathcal{L}(\mathbf{y}_i, \mathbf{y}) + \sum_{j,\bar{\mathbf{y}}} \beta_j^{\bar{\mathbf{y}}} \langle \phi(x_i^{\mathbf{p}_{i-1}}, \mathbf{y}), \phi(x_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}}) \rangle \\ &= \max_{\mathbf{y}\neq\mathbf{y}_i} \mathcal{L}(\mathbf{y}_i, \mathbf{y}) + F(x_i^{\mathbf{p}_{i-1}}, \mathbf{y}) \end{aligned} \tag{2.8}$$

2.3.1 Online Structured SVM Optimization

As for model-free tracking problem, there is no offline labelled data for training except the first frame which is assumed to be annotated. Thus Hare et al. [2011] introduce an online structured output SVM framework Bordes et al. [2007, 2008] for optimizing (2.5), which can be summarized mainly to two basic operations: select a triplet $\{i, \mathbf{y}_+, \mathbf{y}_-\}$ and optimize their corresponding coefficients $\beta_i^{\mathbf{y}_+}$ and $\beta_i^{\mathbf{y}_-}$ using an SMO-style step Platt [1999]. The parameter i in the triplet is randomly selected from 1 to n, while for a given i, $\mathbf{y}_+$ and $\mathbf{y}_-$ are chosen with respect to the gradient of (2.5):

$$g_i(\mathbf{y}) = \mathcal{L}(\mathbf{y}_i, \mathbf{y}) + F(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}). \tag{2.9}$$

$\mathbf{y}_-$ can then be chosen as $\mathbf{y}_- = \max_{\mathbf{y}\in\mathcal{Y}} g_i(\mathbf{y})$, which is consistent with (2.8) for generating active constraints (2.4) as described before. This corresponds to finding the most important sample to be a negative support vector, i.e., the one that has a high discriminant value while possessing a high output loss.

The details of the online structured output SVM optimization can be outlined as below (Algorithm 2.1).

Algorithm 2.1 Online Structured SVM Optimization

Given support vectors $\mathcal{S}_{n-1} = \{(\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}) \mid \beta_i^{\mathbf{y}} \neq 0, i = 1, \ldots, n-1\}$ and a new example pair $(\mathbf{x}_n^{\mathbf{p}_{n-1}}, \mathbf{y}_n)$ obtained at the testing frame n.
Set $\mathcal{S}_n = \mathcal{S}_{n-1}$.

1. ProcessNew: select the triplet $\{n, \mathbf{y}_+, \mathbf{y}_-\}$, where $\mathbf{y}_+ = \mathbf{y}_n$ and $\mathbf{y}_- = \max_{\mathbf{y} \in \mathcal{Y}} g_n(\mathbf{y})$. Optimize the triplet using SMO Bordes et al. [2007, 2008]. If the resulted coefficients $\beta_n^{\mathbf{y}_+}$ and $\beta_n^{\mathbf{y}_-}$ are not zero, add them into $\mathcal{S}_n$.

2. ProcessOld: select the triplet $\{i, \mathbf{y}_+, \mathbf{y}_-\}$ for a random i, where $\mathbf{y}_+ = \min_{\mathbf{y} \in \mathcal{Y}} g_i(\mathbf{y})$ and $\mathbf{y}_- = \max_{\mathbf{y} \in \mathcal{Y}} g_i(\mathbf{y})$. Optimize the triplet using SMO, then add, remove or update them in $\mathcal{S}_n$, according to the resulted $\beta_i^{\mathbf{y}_+}$ and $\beta_i^{\mathbf{y}_-}$.

3. Optimize: select the triplet $\{i, \mathbf{y}_+, \mathbf{y}_-\}$ for a random i, where $\mathbf{y}_+ = \min_{\mathbf{y} \in \mathcal{Y}_i} g_i(\mathbf{y})$, $\mathbf{y}_- = \max_{\mathbf{y} \in \mathcal{Y}_i} g_i(\mathbf{y})$ and $\mathcal{Y}_i = \{\mathbf{y} \in \mathcal{Y} | \beta_i^{\mathbf{y}} \neq 0\}$. Optimize the triplet using SMO. If $\beta_i^{\mathbf{y}_+}$ or $\beta_i^{\mathbf{y}_-}$ is zero, remove it from $\mathcal{S}_n$, otherwise update the corresponding coefficients in $\mathcal{S}_n$. Repeat this step for N_o times.

4. Repeat step 2 to step 3 for N_a times.

Loss Function and Joint Kernel Mapping

Loss function $\mathcal{L}(\mathbf{y}_i, \mathbf{y})$ plays an important role at the training stage (2.3), as it quantifies the loss associated with a prediction $\mathbf{y}$, if the true output value is $\mathbf{y}_i$ Tsochantaridis et al. [2005]. It allows to address the issue raised in the previous works that all negative samples being treated equally Hare et al. [2011]. Following Blaschko and Lampert [2008]; Hare et al. [2011], a bounding box overlap rate based loss function can be defined as below:

$$\mathcal{L}(\mathbf{y}_i, \mathbf{y}) = 1 - \mathcal{O}(\mathbf{y}_i, \mathbf{y}), \tag{2.10}$$

$$\mathcal{O}(\mathbf{y}_i, \mathbf{y}) = \frac{(\mathbf{p}_{i-1} \circ \mathbf{y}_i) \cap (\mathbf{p}_{i-1} \circ \mathbf{y})}{(\mathbf{p}_{i-1} \circ \mathbf{y}_i) \cup (\mathbf{p}_{i-1} \circ \mathbf{y})}. \tag{2.11}$$

Equation (2.11) measures the degree of overlap between two bounding boxes ($\mathbf{p}_{i-1} \circ \mathbf{y}_i$ and $\mathbf{p}_{i-1} \circ \mathbf{y}$) at frame i.

To define the joint kernel mapping, Blaschko and Lampert [2008]; Hare et al. [2011] use a restriction kernel, which crops a region of an image and then applies a standard image kernel between pairs of such patches:

$$k_{xy}((\mathbf{x}_i^{\mathbf{p}_{i-1}}, \mathbf{y}), (\mathbf{x}_j^{\mathbf{p}_{j-1}}, \bar{\mathbf{y}})) = k(\mathbf{x}_i^{\mathbf{p}_{i-1} \circ \mathbf{y}}, \mathbf{x}_j^{\mathbf{p}_{j-1} \circ \bar{\mathbf{y}}}). \tag{2.12}$$

The applied image kernel generally computes statistics or features (such as bag of visual words representation) of the two image patches and then compares them.

Budget Maintainment

Similar to Hare et al. [2011], we set the number of support vectors within an upper limit, because the computational and memory costs increase with the number of support vectors, while the number of training examples can be large in the tracking procedure. Employing the method proposed in Wang et al. [2010], we remove support vector $(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y})$ that results in the smallest change to the coefficient vector w, as measured by $\|\Delta\mathbf{w}\|^2$, given below:

$$\begin{aligned}\|\Delta\mathbf{w}\|^2 = (\beta_r^{\mathbf{y}})^2\{&\langle\phi(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y}), \phi(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y})\rangle + \\ &\langle\phi(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y}_r), \phi(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y}_r)\rangle - 2\langle\phi(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y}), \phi(\mathbf{x}_r^{\mathbf{p}_{r-1}}, \mathbf{y}_r)\rangle\}.\end{aligned} \tag{2.13}$$

At each time the budget is exceeded, we remove the support vector which produces the minimum $\|\Delta\mathbf{w}\|^2$.

Chapter 3

Learning to Track with Multiple Background Clusters

From Chapter 3 to Chapter 5, we present three pioneering works for single object model-free online visual tracking challenge.

In this chapter, we address it from the perspective of building multiple fine-grained foreground-versus-contextual-cluster models that provide more discriminative classifications, and consequently more robust and accurate foreground object tracking.

3.1 Introduction

Visual object tracking confronts with major challenges due to object appearance and scene illumination variation, partial and full occlusion, background clutter, and noise. To build a tracker that is robust to such issues, tracking-by-detection techniques learn adaptive object models, e.g. classifiers, in an online fashion and then search for the best match in the consecutive frames.

Depending on the basic learning strategy, tracking-by-detection approaches can be grouped into generative and discriminative learning categories. Generative learning based models mainly concentrate on how to construct an object representation in specific feature spaces, including the subspace learning Ross et al. [2008], sparse representation Mei and Ling [2011]; Li et al.; Jia et al. [2012] and so on Li et al. [2013a]. A known drawback of these methods is that they often ignore the influence of the background, and consequently suffer from distractions caused by the background regions with similar appearance to the foreground object.

In contrast, discriminative learning based appearance models aim to maximize the inter-class separability between the object and background regions using discriminative learning techniques, including SVMs Avidan [2004]; Li et al. [2011]; Yang et al. [2014], random forest Santner et al. [2010], and multiple instance learning Babenko et al. [2009], to name a few. Among the main challenges of discriminative methods one can consider how to maintain positive and negative training samples, and how to build a powerful classifier out of them. As the number of the processed frames increases, the number of positive, and in particular negative, samples could inflate. Thus, the design of an adequate model update strategy for discriminative learning is not trivial.

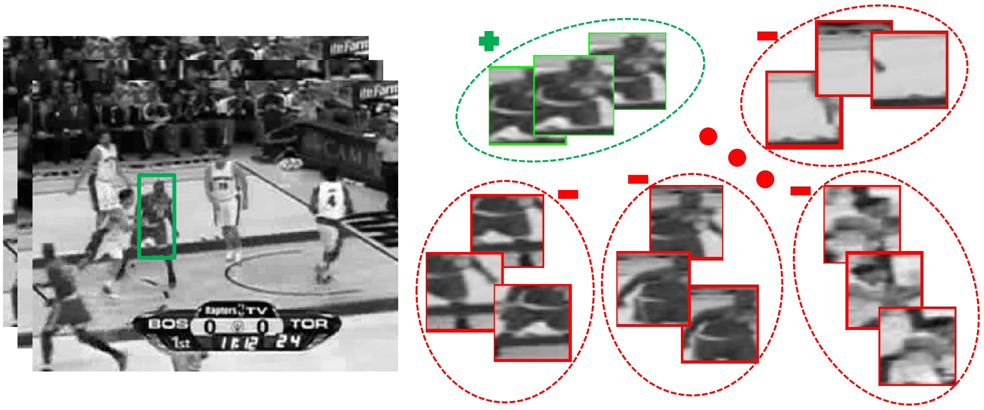


Figure 3.1: Instead of training only one classifier to separate the set of positive samples and the set of negative samples, this paper explores the implicit data structure underneath the negative samples by fine-grain partitioning the negatives into multiple clusters. Note that, each background cluster is meaningful, either corresponding to the shifted-versions of the object or further away yet visually similar negative samples of *pure* background samples.

In this paper, we propose to exploit the underlying distribution structure of the training samples to reduce the burden of the classification task, hence increase the discriminative power of the object tracker. To this end, we utilize the weak visual structure of background, i.e., the negative sample space, by explicitly grouping background samples into multiple contextual clusters. Here, a contextual cluster means a group of samples that exhibit similar visual properties and possibly spatial proximity as dissected more in the experimental section.

We observe that these contextual clusters emerge mainly as two distinct groups: the shifted-versions of the object window, which help better localization albeit cause confusion or drift if not modeled properly, and ordinary non-object like background samples, which encourage better detection yet can cause sudden jumps in the subsequent frames if neglected. We show that explicitly building multiple foreground-versus-cluster classifiers increases the discriminative power by preventing object window drifts and avoiding inaccurate assignments. In other words, using multiple background models is preferable to employing only one.

We exploit structured output support vector machines (SSVM) to obtain an individual tracker for each contextual object-cluster pair. First, we train independently each classifier with their respective contextual clusters using low-level feature descriptors such as histogram of intensity. Then, a unifying SSVM is constructed with all negative samples to learn the importance weights corresponding to each contextual SSVM, by concatenating their responses of the training sample into a feature vector. This lends itself to naturally and optimally combining the outcomes from multiple trackers. More details can be found in Section 3.3.2.

On the TB50 dataset Wu et al. [2015], our method improves the *precision score* around 11.3% overall. For specific attributes, the performance improvement is up to 23.5% for the deformation case, 17.0% for the fast motion case, 23.7% for the motion blur case, 12.2% for the occlusion, and 7.6% for the background clutter case in comparison to the baseline tracker that uses a single background model. Similarly, our results on the popular VOT2014 Kristan et al. [2014] and OTB datasets Wu et al. [2013] are superior to the baseline tracker by a significant margin.

3.2 Related Work

For completeness, we provide a brief overview of the most relevant works and refer readers to the object tracking surveys Wu et al. [2015]; Li et al. [2013c]; Smeulders et al. [2014].

Among notable approaches, Avidan [2004] proposed an SVM-based tracking-by-detection algorithm for distinguishing the object from its close neighborhood. Tian et al. [2007] utilized the ensemble version of the linear SVM classifiers that can be weighted according to their discriminative abilities at each frame. Henriques et al. [2015] addressed the high redundancy of the negative samples due to overlapping pixels with circulant matrix and diagonalized it with the Discrete Fourier Transform, reducing both storage and computation by several orders of magnitude. Li et al. [2013b] partitioned the entire image sequence into spatially and temporally adjacent sub-sequences. They then trained an SVM classifier for object/non-object classification on each of these sub-sequences. A spatiotemporal weighted Dempster-Shafer scheme was presented to combine the discriminative information from these classifiers. Nevertheless, none of these algorithms consider the available contextual information as we do.

Towards incorporating larger receptive fields, Yang et al. [2009] proposed a context-aware tracking algorithm that considers a set of auxiliary objects as the context of the foreground. These auxiliary objects need to satisfy conditions such as persistent co-occurrence with the foreground and consistent motion correlation. These conditions may not be easily satisfied in practice. Grabner et al. [2010]; Dinh et al. [2011]; Possegger et al. [2015] used similar concepts termed as ‘distracters’ and ‘supporters’. Distracters Dinh et al. [2011]; Possegger et al. [2015] are regions that have similar appearance as the target, and supporters Grabner et al. [2010]; Dinh et al. [2011] are regions or features around the target with consistent co-occurrence and motion correlation in a short time span. These methods require careful maintaining models for distracters and supporters.

Li et al. [2011], showed that the high-order contextual information from samples can increase the robustness of the classifier to noise. The high-order context is defined as a group of samples having some common properties. Each sample in the high-order context is influenced by other samples in the same high-order context. For their tracker, the similarity measure depends on not only two individual samples but also their corresponding contexts. Even though the high-order context provides complementary information to counteract the impact of noise, it still lacks a mechanism to incorporate background context.

The idea of splitting the data into groups and training a separate classifier for each group to handle the large intra-class variability is proved to be successful, mainly based on boosting algorithms in image classification and object detection Torralba et al. [2007]; Kim and Cipolla [2008]; Godec et al. [2010]; Saffari et al. [2010a]. In particular, Godec et al. [2010] introduced a set of virtual classes generated by a context-driven clustering to cope with the intra-class variability in object detection. They used an online multi-class classifier to initiate and update new virtual classes, and then label a given patch by one of the virtual classes.

Our method does not require explicit labeling of the background into multiple classes. Instead of using a multi-class structure, our tracker operates in a more efficient and consistent manner when it maintains the set of object-versus-contextual clusters. Thus, it is a binary labeling scheme. In addition, not having to explicitly label for multiple background classes enables our method to construct more discriminative models that significantly improve the tracking performance.

Another related work is the distance metric learning that seeks an effective and discriminative metric space where both intra-class compactness and inter-class separability are maximized. Li et al. [2016] proposed a metric-weighted linear representation of appearance to capture the interdependence of different feature dimensions and developed two online distance metric learning methods using proximity comparison information and structured output learning. Similarly, Li et al. [2012] observed that different visual metrics should be optimally learned for different candidate sets in the context of human reidentification problem, which is to match persons observed in non-overlapping camera views. This approach selects and reweights the training samples according to their visual similarities with the query sample and its candidate set. In contrast, our work does not handle the discriminative metric space. Instead, we explore the contextual information for the background samples via explicitly grouping them into clusters. We deploy a top-level SSVM to fuse the discriminative information from the individual clusters' SSVMs, which can be considered relevant to the metric learning concept.

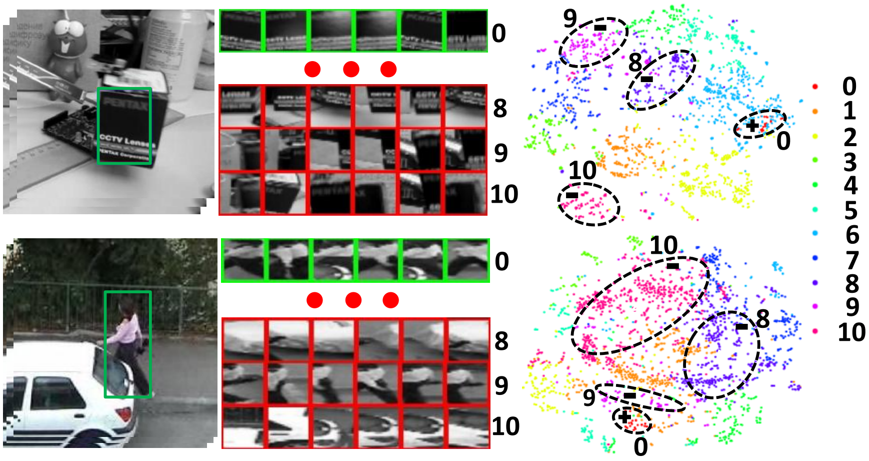


Figure 3.2: Two instances of contextual clusters. Middle column: one foreground cluster (green) and ten contextual clusters (red). Each row corresponds to a separate cluster. Last column: the 2D layout of samples in principal components from the t-SNE dimension reduction van der Maaten and Hinton [2008] for visualization. 0: foreground. 1-10: color coded background clusters. As visible, there is a significant variance in the background samples, which may hinder the performance of a monolithic binary classifier.

3.3 Tracking with Multiple Clusters

The basic idea of tracking-by-detection is to establish object correspondence between consecutive frames using an object detector. Many recent state-of-the-art trackers are often based on this scheme Wu et al. [2015]; Hare et al. [2011]; Babenko et al. [2009]; Li et al. [2013c], resulting in improved accuracy and robustness of tracking performance. One reason is that an online updated classifier helps to address challenging situations such as appearance variations, partial occlusions, and background clutters in a single, unified manner.

Given an estimated object bounding box B^*_{n-1} in a previous frame $n-1$, the tracker proceeds to find a new object location B^*_n at the current frame n through a dynamically maintained and updated classification confidence function F as follows:

$$B^*_n = \arg\max_{B_n \in \mathcal{S}_n(B^*_{n-1})} F_{n-1}(B_n), \tag{3.1}$$

where $\mathcal{S}_n(B^*_{n-1})$ denotes the set of candidates in frame n, sampled around the previous object location B^*_{n-1} within a search radius. For example, the search radius of 30 pixels was used in Hare et al. [2011].

As mentioned above, to efficiently maintain and update a classification confidence function $F_{n-1} \to F_n$ is key to the success. In this regard, previous work (e.g. Babenko et al. [2009]) often use multiple-instance-learning to compose the positive and negative training samples (this can be also viewed as the online labeling task as in Hare et al. [2011]) against label noise issue. However, many of these methods make the implicit assumption that the background (and the context) conform to a single, monolithic, possibly homogeneous class, which is rarely the case.

In contrast, our method does not assume the background samples have an identical distribution, or they belong to a single semantic class. We argue that, the appearance variance of the background can be better modeled by a committee of foreground-versus-contextual-cluster classifiers. Noticing in practice most tracking failures occur either when a background element such as background clutter distracts the tracker or when the tracker slightly drifts and starts accumulating error until a total breakdown, here we propose constructing fine-grained boundaries using multiple classifiers on contextual clusters.

3.3.1 Fine-Grained Classifiers

To better capture the latent data distributions of the negative samples (i.e. background clusters), which can be rather complex, we use unsupervised clustering with temporal continuity priors.

A simple way to perform this is to use k-means algorithm to label each negative sample as one of K clusters at every frame by initiating the iterations with the previously estimated clusters centers. Alternatively, Hough-forest based clustering Gall et al. [2011] may be used. This method employs a random forest to cluster patches that have consistent appearance (and spatial displacement). Yet another solution is a graph mode-seeking method Li et al. [2011], which can automatically discover the distribution modes, i.e. dense subgraphs, of a graph characterized by a baseline kernel. In this work, we suggest the k-means algorithm mainly due to its computational simplicity.

An illustration of the clustering result is given in Figure 3.2. Here, the negative and positive sample set descriptors (480-dimensional intensity histogram features) are mapped down to a 2D space for visualization. We use a dimension reduction technique, t-distributed stochastic neighbor embedding (t-SNE) van der Maaten and Hinton [2008], which computes a mapping of distances while preserving the overall global structure. Notice that, each cluster of the background samples portrays *hard negative* patterns. Even though the background samples are collected from different frames, they exhibit patterns that can be clustered into a few consistent patterns, which will leverage the discriminative power of the corresponding classifiers.

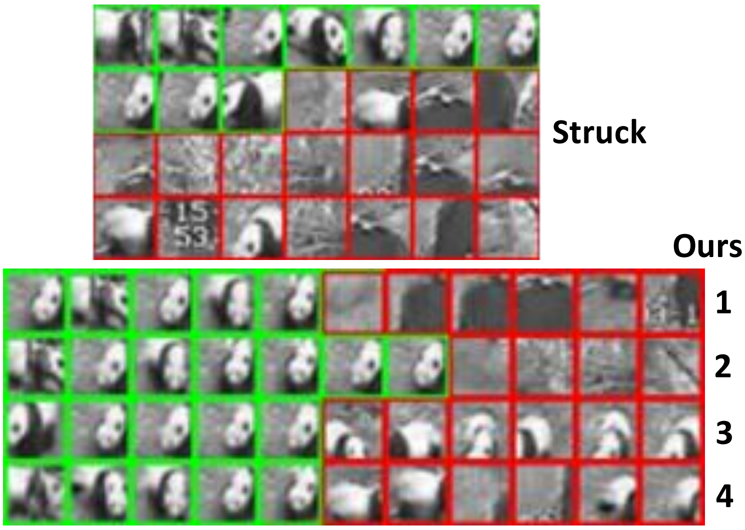


Figure 3.3: Conventional single SSVM (top, from Struck Hare et al. [2011]) versus the proposed multiple SSVMs of the contextual clusters (bottom, each row corresponds to one contextual SSVM). Green: positive support vectors. Red: negative support vectors. Notice that the burden of the classification task is reduced significantly for each contextual SSVM.

We select SSVM as the foreground-versus-contextual cluster classifier, nonetheless our method can be extended to any object model easily. SSVM is shown to provide better object localization and tracking performance than other variants of SVM Blaschko and Lampert [2008]; Tsochantaridis et al. [2005].

Suppose the negative samples $\{\mathcal{B}_i \backslash B_i^* : i = 1, \ldots, n-1\}$ from $n-1$ previous frames are grouped into K contextual clusters $\{\{\mathcal{B}_i^k : i = 1, \ldots, n-1\} : k = 1, \ldots, K\}$, where $\mathcal{B}_i^k$ denotes the negative samples belonging to the k-th cluster, $\{B_i^* : i = 1, \ldots, n-1\}$ is the set of positives, and $\mathcal{B}_i$ is the set of all positive and negative samples at frame i. We separately train K classifiers to obtain confidence functions for each pair of the negative cluster $\{\mathcal{B}_i^k\}$ and the positive set $\{B_i^*\}$, which have the form of:

$$F_{n-1}^k(B_n) = \sum_{B_{i,j}^k \in \mathcal{V}_{n-1}^k} w_{i,j}^k \Phi(B_{i,j}^k, B_n) \quad k = 1, \ldots, K, \tag{3.2}$$

where $\mathcal{V}_{n-1}^k$ is the support vector set of the k-th SSVM after the training process, and $w_{i,j}^k$ is a scalar weight associated with the support vector $B_{i,j}^k \in \mathcal{B}_i^k \cup B_i^*$ indexed by j from frame i. The kernel $\Phi(B_{i,j}^k, B_n)$ calculates the affinity between two feature vectors extracted from $B_{i,j}^k$ and B_n, respectively.

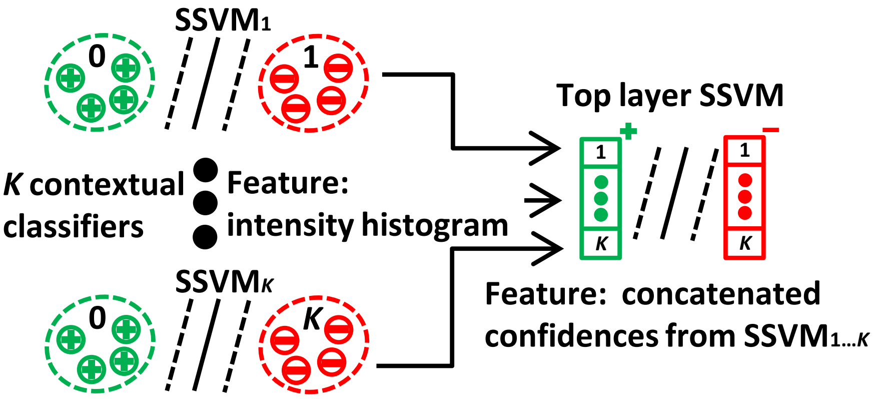

SSVM1
0
1
K contextual classifiers
Feature: intensity histogram
SSVMK
0
K
Top layer SSVM
+
−
1
K
1
K
Feature: concatenated confidences from SSVM1...K

Figure 3.4: Hierarchy of classifiers in the proposed tracker. First layer: K separate SSVMs trained using the positive samples and the K contextual negative sample sets. Second layer: a single SSVM is trained to fuse the classification confidences. All SSVMs are updated online to adapt object appearance and background changes.

Here, both $\mathcal{V}_{n-1}^{k}$ and $w_{i,j}^{k}$ are learned using the online SSVM algorithm "Larank" Bordes et al. [2007, 2008], which is shown to be an efficient SSVM solver Hare et al. [2011]. As the image feature, we employ intensity histograms from a spatial pyramid Lazebnik et al. [2006] to represent the image patch in $\Phi(B_{i,j}^{k}, B_n)$ capturing the discriminative cues between the foreground and background patches. In the experiment section, we test different 2D kernels including linear, radial basis function (RBF) and intersection.

We ask the question whether a strong SSVM using RBF kernel with a large number of support vectors, most of which correspond to the previously estimated negative samples, would achieve the same performance. To our observations, simply inflating the number of support vectors does not generate a proportionally more accurate classifier since it either tends to overfit data or fails to model essential differences between the object and background samples. As shown in Table 3.3, a single very strong classifier results in only marginal improvement on the performance if any. In Figure 3.3, we give an example of the differences between the support vectors maintained by the tracker that uses a single strong SSVM Hare et al. [2011] and by our proposed contextual SSVMs. It is apparent that the burden of the classification task is reduced for each contextual SSVM in our method, comparing to the single SSVM.

3.3.2 Confidence Combination

There are numerous strategies to combine multiple confidence functions including max or average pooling Chatfield et al. [2011], voting, and multiple kernel learning. These, however, are not capable of learning an adaptive discriminative model for each video sequence.

Instead, we treat each confidence function as a feature generator, and use an additional top layer SSVM as illustrated in Figure 3.4 to learn the optimal combination of multiple confidence function results of K contextual clusters.

In this stage, the negative samples $\{\mathcal{B}_i \backslash B_i^* : i = 1, \ldots, n-1\}$ and the positive samples $\{B_i^* : i = 1, \ldots, n-1\}$ are used to train a top layer discriminant function $F_{n-1}(B_n)$ as:

$$F_{n-1}(B_n) = \sum_{B_{i,j} \in \mathcal{V}_{n-1}} w_{i,j} \Psi(B_{i,j}, B_n). \tag{3.3}$$

The difference between $F_{n-1}(B_n)$ and $F_{n-1}^k(B_n)$ is the design of the feature for kernel function $\Psi(B_{i,j}, B_n)$. This feature concatenates the classification confidences from the K SSVMs into a K-dimensional vector. Different choices of kernels are tested in the experimental section. The overall tracking algorithm is summarized in Algorithm 1.

3.3.3 Online Update with Temporally Consistent Clustering

In object tracking, the training data for the object model is given only in the first frame. The SSVM framework Bordes et al. [2007, 2008] selects a triplet $\{i, B_{i,+}^k, B_{i,-}^k\}$ and optimizes their corresponding coefficients $w_{i,+}^k$ and $w_{i,-}^k$ using an SMO-style step Platt [1999]. The main step is to choose the negative support vector $B_{i,-}^k$ by

$$B_{i,-}^k = \arg\max_{B_i \in \mathcal{B}_i^k} L(B_i, B_i^*) + F_{n-1}^k(B_i), \tag{3.4}$$

where the loss function $L(B_i, B_i^*) = 1 - (B_i \cap B_i^*)/(B_i \cup B_i^*)$ defines on the bounding box overlap. Optimizing (3.4) corresponds to finding such a negative training sample that locates far from the positive one (high $L(B_i, B_i^*)$) yet presents close appearance (high $F_{n-1}^k(B_i)$). We use the C++ implementation from Hare et al. [2011] for it.

To avoid independently re-clustering and re-optimizing over the K separate SSVMs at every frame, we benefit from the k-means initialization. First, we run the k-means multiple times to obtain a consistent clustering. At every new frame, we recycle the previous clusters' centers to initialize k-means clustering. Since only a portion of the previously clustered samples change after clustering, we keep the unchanged support vectors and avoid re-optimizing the SSVMs. To be specific, we use the *processold* step in Hare et al. [2011] to add an extra number of negative support vectors, replacing those lost due to the re-clustering procedure if necessary.

Keeping all available training samples is not computationally and memory-wise efficient, thus we employ the budget management method used in Wang et al. [2010]. This allows at most a fixed-number (100 in all experiments) of maintained support vectors. Once this number is exceeded, we remove the most insignificant support vectors that induce the smallest changes to the classification boundary.

Algorithm 1. Two-layer SSVM based Tracker using Multiple Background Clusters

Tracking
Require: K confidence functions F^k_{n-1}, the top layer discriminant function F_{n-1}, previous model B^*_{n-1} and object location

1. Generate K confidence scores for each candidate in the search radius $\mathcal{S}_n(B^*_{n-1})$ in the current image: $F^k_{n-1}(B_n) = \sum_{B^k_{i,j}\in\mathcal{V}^k_{n-1}} w^k_{i,j}\Phi(B^k_{i,j}, B_n)$.

2. Compute aggregated confidence score: $F_{n-1}(B_n) = \sum_{B_{i,j}\in\mathcal{V}_{n-1}} w_{i,j}\Psi(B_{i,j}, B_n)$.

Return: New location : $B^*_n = \arg\max_{B_n\in\mathcal{B}_n} F_{n-1}(B_n)$.

Update
Require: Support vector sets and the corresponding weights of K contextual cluster SSVMs $\mathcal{V}^k_{n-1}$, and of the top layer SSVM $\mathcal{V}_{n-1}$, the new positive sample B^*_n and negative samples $\mathcal{B}_n \backslash B^*_n$.

1. Run k-means initialized with the previously estimated clusters centers to obtain the new contextual clusters: $\{\{\mathcal{B}^k_i : i = 1, \ldots, n\} : k = 1, \ldots, K\}$.

2. Update the contextual SSVMs: $\mathcal{V}^k_n \leftarrow \mathcal{V}^k_{n-1}$ and the corresponding weights $w^k_{i,j}$ as in Section 3.3.3.

3. Update features for the top layer SSVM: $[F^1_n(B_i), \ldots, F^K_n(B_i)], B_i \in \mathcal{B}_i, \forall i \in \{1, ..., n\}$, using the updated contextual SSVMs from 2.

4. Train the top layer SSVM: $\mathcal{V}_n \leftarrow \mathcal{V}_{n-1}$ and the corresponding weights $w_{i,j}$ using online optimization with the features from step 3.

Return: Support vectors $\{\mathcal{V}^k_n \mid k = 1, \ldots, K\}$, $\mathcal{V}_n$ and the corresponding weights.

3.4 Experiments

3.4.1 Standard Benchmark Evaluation

We evaluate our method on three recent benchmark datasets: OTB Wu et al. [2013], TB50 Wu et al. [2015] and VOT2014 Kristan et al. [2014]. These datasets provide a large number of sequences depicting a wide spectrum of challenging tracking scenarios.

OTB contains 50 video sequences with fully bounding box annotations. The total number of frames is more than $29,000$, and for each sequence, the number varies from tens to thousands, e.g. *deer* (71 frames), *skiing* (81 frames), *dog*1 (1350 frames), *doll* (3872 frames), etc.

In comparison to the OTB dataset, TB50 Wu et al. [2015] contains more challenging sequences. Samples can be seen in Figure 3.7. Many of the TB50 sequences depict strong motion blur (e.g. *blurBody*), fast object motion (e.g. *dragonbaby*), and intermittent occlusions (e.g. *skating2*). As visible in Figure 3.5, there is a big performance gap for all trackers between OTB and TB50.

Both benchmarks additionally annotate each sequence globally with various visual attributes. Some common attributes available in the benchmarks are:

- Fast Motion - the motion of the ground truth is larger than t_m pixels ($t_m = 20$).
- Motion Blur - the target region is blurred due to the motion of target or camera.
- Deformation - non-rigid object deformation.
- Occlusion - the target is partially or fully occluded.

In the benchmarks, individual sequences are not per-frame annotated. For example, a sequence has the *occlusion* attribute if the target is occluded at any frame in the sequence. Although many factors could contribute to the performance, these attributes help us to diagnose the weaknesses and strengths in a more detailed way.

The sequences embodied in the VOT2014 benchmark are selected from widely used datasets in literature, including the Amsterdam Library of Ordinary Videos for tracking (ALOV++) Smeulders et al. [2014]; ALOV300 and OTB. It comprises a set of 25 sequences, which cover various real-life visual phenomena. The duration of these sequences are relatively short in order to keep the computational load of experimental evaluations reasonably low. Unlike OTB and TB50, VOT2014 labels each frame in each sequence with five visual attributes. It also features a reinitialization evaluation scheme. After the tracker loses the target object during tracking, which is the case when the overlap measure with the ground truth becomes zero, the tracker is reinitialized five frames after the failure. This scheme measures the robustness of trackers by counting how many times they fail in a sequence.

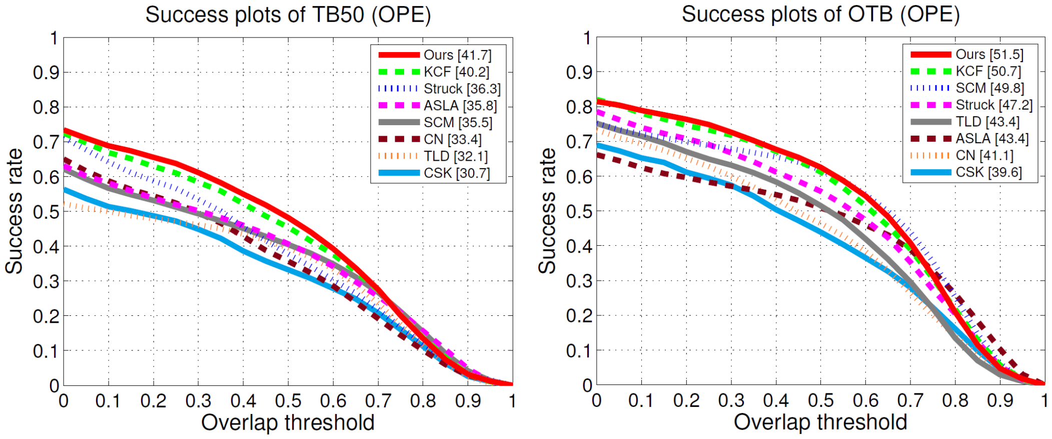


Figure 3.5: *Success ratio* plots on the TB50 and OTB datasets. Trackers are ranked by the Area Under Curve (AUC) of the *success ratio* plots. As visible, our method (red) achieves the best performance on both datasets.

Evaluation Metrics:

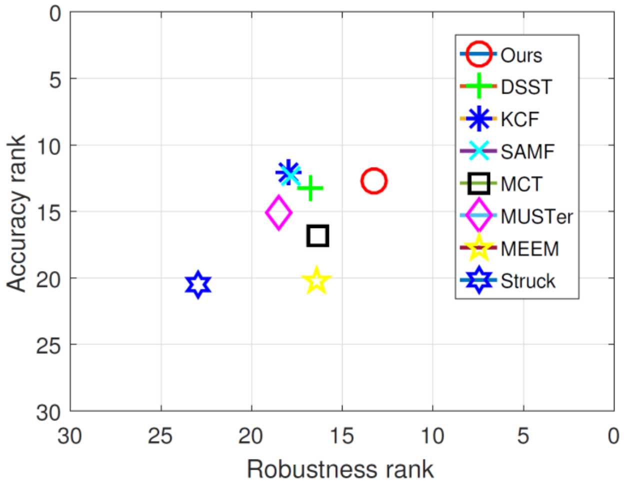
We use the metrics and the source code provided by these benchmarks. On OTB and TB50, the performance is evaluated using the *precision score* and *success ratio* metrics. The *precision score* calculates the rate of frames whose center location is within a certain threshold distance with the ground truth. Here, a commonly used threshold is 20 pixels as recommended by the benchmark protocol. This metric emphasizes how well a tracker is able to clasp the target. The *success ratio* calculates the same ratio based on bounding box overlap threshold $(B^* \cap B_{gt})/(B^* \cup B_{gt})$, where B^* and B_{gt} are the estimated and ground truth bounding boxes, respectively. This metric indicates how well a tracker adapts and covers the target. A typical value is 0.5 as used in object detection evaluation Everingham et al. [2015].

We employ the one-pass evaluation (OPE) that takes the ground truth at the first frame as the initialization bounding box then run trackers until the last frame.

For VOT2014, the benchmark provides a ranking based on the *robustness* performance measure. As mentioned above, the *robustness* measures how many times the tracker loses the target (failures). The ranking scheme considers the statistical significance of performance differences to ensure an objective comparison, e.g., trackers are equally ranked if there is only a negligible difference from a practical point of view. We also calculate the ranking result based on the *accuracy* metric, which measures how well the bounding box predicted by the tracker overlaps with the ground truth bounding box. We test all trackers 15 times on each sequence to obtain reliable statistics on performance measures.

Table 3.1: Area Under Curve (AUC) of *success ratio* plots and *precision scores* (at 20 pixels threshold) on TB50 and OTB benchmark datasets for the one-pass evaluation (OPE). fps: frames-per-second. Best in bold. Our performance on TB50 when we adapt to scale changes is even higher; **42.0**/**62.5**.

Datasets	Ours	Struck	KCF	SCM	TLD	CN	ASLA	CSK
TB50 (50)	**41.7**/**61.2**	36.3/49.9	40.2/61.1	35.5/47.8	32.1/45.0	33.4/42.2	35.8/46.2	30.7/41.8
OTB (50)	**51.5**/72.5	47.2/65.3	50.7/**72.9**	49.8/64.8	43.4/60.1	41.1/55.3	43.4/60.1	39.6/54.1
fps	2.3	4.8	70.9	0.3	8.8	27.2	3.8	18.6



(a)

	Robustness Rank
Ours	**13.22**
DSST	16.75
KCF	17.95
SAMF	17.81
MCT	16.34
MUSTer	18.49
MEEM	16.42
Struck	22.98

(b)

Figure 3.6: (a) Accuracy-robustness ranking plots of our method and top ranked methods on VOT2014. Our tracker provides the best trade-off between accuracy and robustness; (b) Robustness performance on VOT2014.

Benchmark Results:

The tracking results for the benchmark datasets are presented in Table 3.1, Figure 3.5 and Figure 3.6.

As shown, our method outperforms all other trackers including more recent approaches CN and KCF on both TB50 and OTB in both the precision score and the Area Under Curve (AUC) of the success plot. On VOT2014, our method achieves the best robustness rank among all state-of-the-art. It exhibits consistent performance for all three benchmarks as well.

Attributes (TB50)	Ours	Struck	KCF	SCM	TLD	CN	ASLA	CSK
FM (25)	**41.6/59.5**	34.4/42.5	39.0/54.0	25.2/29.6	35.6/46.5	30.9/35.2	25.0/26.0	26.4/33.7
MB (19)	**42.4/59.2**	30.9/35.5	40.6/56.4	21.7/25.1	39.3/49.7	31.1/36.0	23.3/25.5	29.8/36.4
DEF (23)	**43.6/65.0**	32.5/41.5	39.8/58.2	28.5/40.3	24.8/33.4	32.1/35.9	34.7/46.9	25.8/33.4
IPR (29)	**41.4**/58.0	34.3/45.2	38.7/**58.7**	34.5/46.2	33.1/45.8	36.4/48.5	33.9/43.9	29.8/40.6
OPR (32)	**40.2/60.3**	35.3/49.2	39.5/59.8	35.5/49.1	29.0/41.3	32.6/42.4	38.0/49.2	26.2/36.4
OV (11)	**42.4/68.5**	33.9/46.1	32.8/44.1	27.9/35.8	30.8/41.6	30.6/35.5	31.0/38.3	21.3/25.6
OCC (29)	**40.5/62.0**	35.6/49.8	39.5/60.4	34.8/48.0	27.4/39.5	32.0/40.7	36.7/48.5	26.5/37.3
BC(20)	39.0/55.2	36.5/47.6	**41.7/62.3**	36.2/46.6	29.5/39.9	35.6/43.7	39.8/50.0	34.3/46.7
SV (38)	35.9/54.7	34.0/47.5	35.2/**56.5**	**37.0**/49.7	30.0/42.4	32.4/35.9	35.8/46.3	26.7/36.6

Table 3.2: Area Under Curve (AUC) of *success ratio* plots and *precision scores* (at 20 pixels threshold) on TB50 dataset attributes. FM: fast motion, MB: motion blur, DEF: deformation, IPR: in-plane rotation, OPR: out-of-plane rotation, OV: Out-of-view, BC: background clutters, SV: scale variation. Best results are shown in bold.

Our method of using multiple backgrounds also significantly improves its baseline tracker (Struck). On the OTB dataset, our improvement is significant; 4.3% for the AUC and 7.2% for the precision score. On the more challenging TB50 dataset, we achieve even a greater improvement; 5.4% for the AUC and 11.3% for the precision score. On VOT2014, our method boosts the robustness rank from 22.98 to the best score 13.22. These results demonstrate that our multiple-contextual-clusters method remarkably benefits discriminative classification schemes for tracking.

Sample tracking results of our method and the top performing state-of-the-art trackers are given in Figure 3.7 for qualitative analysis. As visible, our method tracks the target objects accurately over many various challenging scenarios, where all others fails (e.g., Struck, SCM, TLD, etc.).

To demonstrate that simply increasing the complexity of a single classifier is not an effective model for tracking and thus cannot achieve a better performance as our method, we evaluated the performance of Struck Hare et al. [2011] with increased number of support vectors. The results are given in Table 3.3 where Struck$_{500}$ denotes the singe SSVM based tracker using a maximum of 500 support vectors. The original Struck uses 100 support vectors. It is apparent that insignificant improvement is obtained by increasing the number of support vectors albeit considerable computational expense.

Performance on Attribute Categories:

To obtain a better understanding, we evaluated the performance of our method on the attribute categories of TB50. Comparative results are given in Table 3.2.

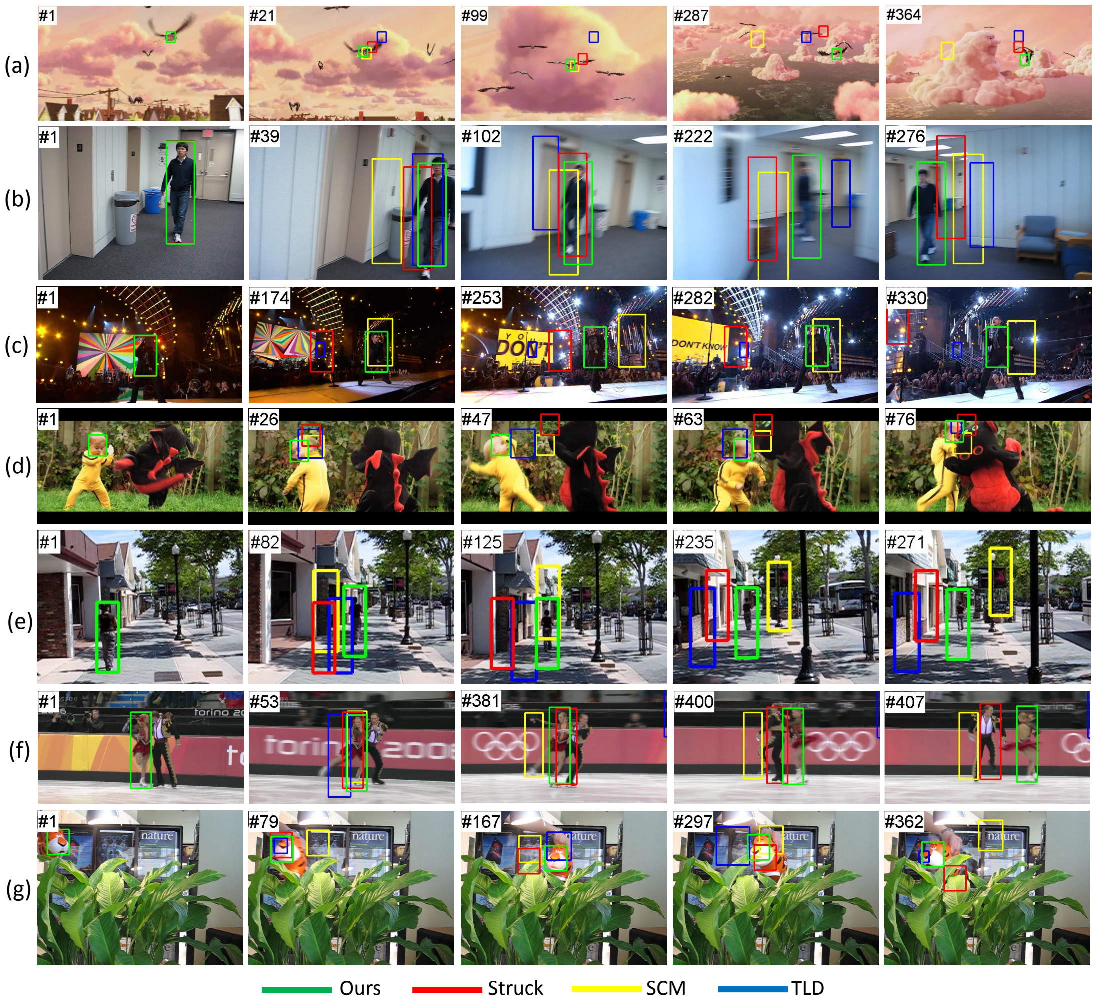


Figure 3.7: Qualitative comparisons with the state-of-the-art trackers on videos from TB50. (a) Bird1; (b) BlurBody; (c) Singer2; (d) DragonBaby; (e) Human9; (f) Skating2; (g) Tiger2. Our method attains robust tracking performance in challenging scenarios including fast motion, motion blur, deformation, and occlusion. Notice that, the object window size in each video is fixed.

Table 3.3: Struck Hare et al. [2011] performance on TB50 for different maximum number of support vectors.

	Ours	Struck$_{100}$	Struck$_{200}$	Struck$_{500}$
AUC/PS	**41.7/61.2**	36.3/49.9	35.9/50.6	36.4/50.9
fps	2.3	4.8	4.3	3.7

For most attributes, such as *motion blur*, *fast motion* and *deformation*, our method achieves superior performance. For *motion blur* and *fast motion*, the performance improvement comes from the fact that our method elegantly instantiates specific trackers for the contextual cluster of hard negative samples for the shifted version on the object window, which provides enhanced localization accuracy. For *deformation*, our method allows efficiently distributing the burden of modeling the foreground object variations over multiple classifiers, which would be difficult for a single SSVM to distinguish. For *background clutter* and *scale change*, we have still significantly better results than the base tracker, i.e. Struck.

Implementation Details and Variants:

Our method uses the intersection kernel for confidence function of the contextual cluster SSVMs, the linear kernel for the top layer discriminant classifier, and intensity histogram as low-level features.

We employ the motion model that applies a 2D translation $\{(u, v)|u^2 + v^2 < r^2\}$ for simplicity. During tracking we apply a search radius $r = 30$ pixels and during updating the classifier we take a larger radius $r = 60$ to incorporate possible nearby hard negatives in the negative samples and to ensure robustness. We sample candidate object locations on a polar grid (5 radial and 16 angular divisions, giving 81 locations).

The classification models for both the contextual cluster SSVMs and the top layer discriminant SSVM are online updated every 5 frames to trade off between computational efficiency and robustness. The algorithm parameters involved in online updating SSVM using "LaRank" Bordes et al. [2008] are set similar to Hare et al. [2011] for a fair comparison.

As feature, we operate with concatenated 16-bin intensity histograms from a spatial pyramid of 4 levels. At each pyramid level l, the underlying patch is divided into $l \times l$ cells, resulting in a $D = 480$ dimensional feature vector $[h_B^1, ..., h_B^D]$. We also tested the variants using different image features such as Haar wavelets and raw image patch. For the features we analyzed:

Table 3.4: Different low-level features. Results on TB50.

	Histogram	Haar	Raw Intensity
AUC/PS	**41.7/61.2**	39.8/56.1	40.1/58.6
fps	2.3	3.5	3.1

- Haar feature - 6 different types of Haar-like feature arranged on a grid at 2 scales on a 4 × 4 grid, resulting in 192-D features, with each feature normalized to give a value in the range $[-1, 1]$.

- Raw patch - Raw pixel features obtained by scaling a patch to 16 × 16 pixels and taking the greyscale value (in the range $[0, 1]$). This gives a 256-D feature vector.

The comparison of features are available in Table 3.4. Remarkably, the raw intensity feature performed better than the Haar feature. One explanation is that the Haar feature is not sensitive enough to the discriminative yet fine-grained appearance details.

To further evaluate our method, we examine the effectiveness of the top layer SSVM by replacing it with commonly used pooling methods. As discussed in Section 3.3.2, the incorporated top layer SSVM is for combining the confidence function results from the contextual cluster SSVMs. As an alternative, we test three different pooling methods to fuse the confidence scores: mean pooling, median pooling and maximum pooling. The results are shown in Table 3.5. As we can see from the results, all pooling methods cause inferior performance compared to ours. This is expected as the incorporated top layer SSVM learns in an online fashion to trust which contextual cluster classifier instead of blindly and heuristically choosing one.

Table 3.5: Use of different pooling schemes instead of the top layer discriminant SSVM. Results on TB50.

	Ours	mean	median	max
AUC/PS	**41.7/61.2**	39.8/57.3	39.3/58.2	40.6/59.3
fps	2.3	3.1	3.0	3.1

We also analyzed the alternative kernel combinations for the contextual SSVMs and the top layer SSVM. The joint kernel function $\Phi(B^k_{i,j}, B_n)$ (3.2) is implemented using the intersection kernel:

$$\Phi(B^k_{i,j}, B_n) = \frac{1}{D} \sum_{d=1}^{D} \min(h^d_{B^k_{i,j}}, h^d_{B_n}).$$

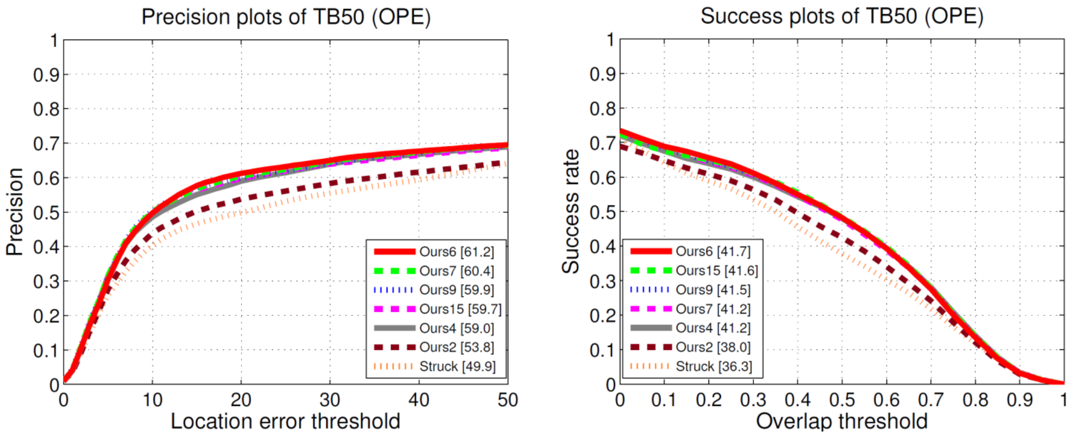


Figure 3.8: *Success ratio* and *precision score* plots of our method with different number of clusters. All our variants are better than Struck.

We use the linear kernel for the top layer discriminant function $\Psi(B_{i,j}, B_n)$ (3.3), which computes the inner products. Results are shown in Table 3.6. In this experiment, we set $\sigma = 0.1$ for the Gaussian kernel. We observed that the linear kernel generates inferior results when used in the contextual SSVMs, however it gives the best accuracy when used in the top layer SSVM. This is possibly due to the fact that the feature complexity is significantly different between these two layers.

Table 3.6: Different kernels. Results on TB50.

Contextual SSVMs	Linear 38.1/52.3	Gaussian 41.1/60.6	Intersection **41.7/61.2**
Top Layer SSVM	Linear **41.7/61.2**	Gaussian 40.3/58.7	Intersection 40.7/59.1

For k-means, the cluster number is set to $K = 6$ for all experiments. We also tested variants using different cluster numbers. The results can be seen in Figure 3.8. As visible in the graphs, our method is robust against the cluster number changes, and always better than using a single cluster. This validates the use of multiple clusters, and multiple classifiers, for the background samples.

We additionally investigated combining the spatial coordinates of samples with the visual features to enforce spatial consistency of samples within each cluster. We observed that this does not improve the performance. Besides, a heuristic imposition of spatial closeness of samples within the clusters escalates maintenance issues of clusters, in particular when the object motion causes the background to change.

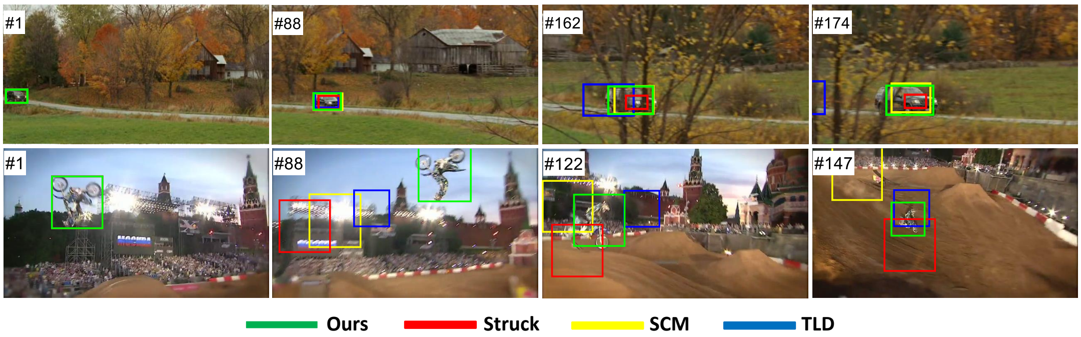


Figure 3.9: Size change adaptation: sample results of our method and the state-of-the-art trackers on videos from TB50. Top row: CarScale; Bottom row: MotorRolling.

Size Adaptation

Our method uses a simple fixed object bounding box representation through the tracking process as Struck Hare et al. [2011] and KCF Henriques et al. [2015]. Yet, it is straightforward to extend our method to adapt scale and aspect ratio changes by modifying the motion model from the 2D translation $\{(u, v)|u^2 + v^2 < r^2, r = 30\}$ to a 3D or 4D motion models (with scale and aspect ratio changes: step 0.1, range $[0.8, 1.2]$). The results are reported in Table 3.7 and sample detections are depicted in Figure 3.9.

Table 3.7: Adaptation of size change. Results on TB50.

	Ours (fixed)	Scale	Scale+As.Ra.
AUC/PS	41.7/61.2	**42.0/62.5**	41.5/60.2
fps	2.3	1.1	0.4

As visible, scale adaption further improves the AUC/PS on TB50. Yet, this increases the computational cost. By adapting scale, the performance may potentially improve for the *scale variation* category. This can be validated from Table 3.2, where SCM (size adapted) gives better scores for the *scale variation* category. However, for attributes such as *occlusion* and *deformation*, trackers with fixed object size tend to perform more robustly.

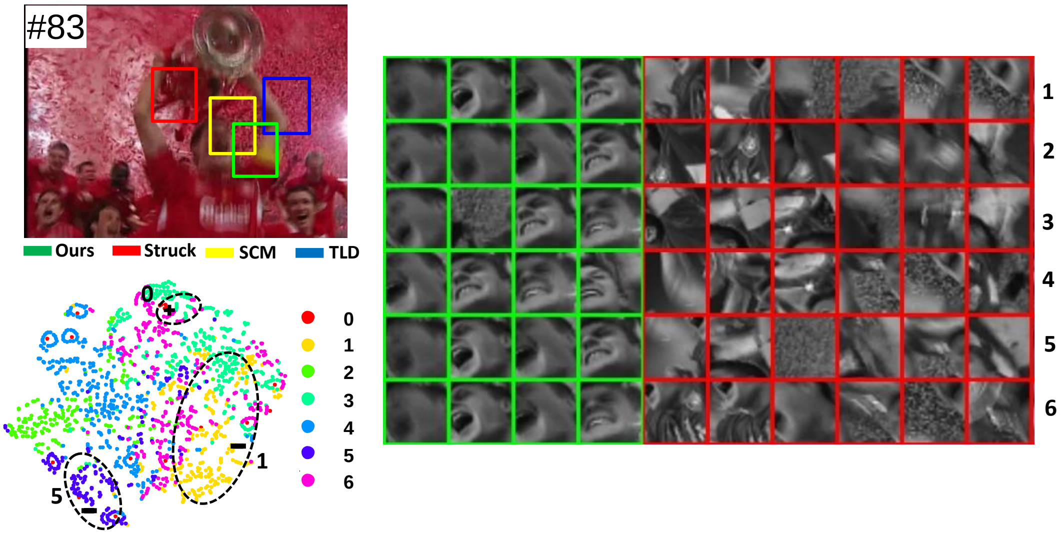


Figure 3.10: A failure example from sequence 'Soccer' (TB50). Upper right: 2D layout (t-SNE) of *k*-means results of training samples. 0: foreground. 1-6: color coded background clusters. Bottom: each row corresponds to one contextual SSVM. Green: positive support vectors. Red: negative support vectors.

Possible Failure Cases:

As we can see from Table 3.2, our method performs superior in most benchmark attributes, however it is among the second best trackers for the *scale variation* and *background clutter* after KCF. One reason for this is that we employed fixed bounding box sizes, which may have limited its capacity to acquire correct foreground models when the target object undergoes drastic scale changes. For the *background clutter*, the reason could be that there is no apparent distribution of multiple clusters exhibited as shown in Figure 3.10. In this case, *k*-means may fail to extract effective contextual clusters as illustrated in the 2D layout of the clusters and support vectors of the contextual SSVMs. Notice that, *k*-means has a random nature that may lead to this.

Nevertheless, our method of incorporating multiple contextual background clusters is always better than Struck for all attributes. This corroborates the robustness of our hierarchical SSVM structure regardless of unstable clustering results of *k*-means. We argue that all clusters are subsets of the background samples, and even potentially irregular clusters contribute to foreground-background classification task, thus their responses do not deteriorate the second layer's prediction capacity.

3.4.2 Computational Complexity

Our method is implemented in C++ and experiments are carried out on an Intel Core i7 3.40GHz PC with 4GB memory. Computational time is reported in Table 3.1. The speed of our method is 2.3 fps on average without any optimization. The overall computational cost is comparable to existing methods. In addition, it is not increased significantly in comparison to the single SSVM (e.g. Hare et al. [2011]) despite we use additional SSVMs. The reason is that the most time-consuming part in our method is in the optimizing (3.4) stage, i.e. exhaustively searching over the negative sample space to find a negative support vector as shown in Section 3.3.3. In our method, for each foreground-versus-contextual cluster SSVM, this search space is greatly reduced.

3.5 Summary

We presented a tracking method that tackles the object detection task by designating multiple classifiers where each targets discriminating a different background cluster from object samples, and combining their responses into a top layer identifying to which pattern of classifier responses indicate object. This significantly reduces the burden on the classifier, allows learning of fine-grained yet important decision boundaries, and lends itself to efficient and accurate adaption to object and background changes.

By explicitly grouping the negative samples into multiple clusters, building multiple foreground-versus-cluster SSVM classifiers, and employing another single SSVM to learn the best combination of the confidences generated from the respective contextual classifiers, the proposed method achieves superior discriminative power as verified on standard benchmark datasets.

Chapter 4

Tracking Objects Everywhere with Instance-Specific Proposals

After addressing the binary foreground-versus-background classification problem in the previous chapter, we now focus on another common limitation of the existing trackers in this chapter. Most existing tracking-by-detection methods employ a local search window around the predicted object location in the current frame, assuming the previous location is accurate, the trajectory is smooth, and the computational capacity permits a search radius that can accommodate the maximum speed yet small enough to reduce mismatches. These may not be valid always, in particular for fast and irregularly moving objects.

4.1 Introduction

Model-free object tracking, which aims to track arbitrary objects based on a single bounding-box annotation, has gained significant attention recently with numerous approaches Hare et al. [2011]; Henriques et al. [2015] proposed and several large benchmark datasets Wu et al. [2015]; Smeulders et al. [2014] released. Most of these methods, however, require a search window centered at the previous object location to select candidate patches, partly due to computational complexity. This is sometimes referred as the motion model Wang et al. [2015b], and it is implicitly assumed that the object is correctly tracked in the previous frames and the object motion is not large. Even though this simplification works in some situations, it also introduces serious difficulties especially when the object undergoes deformations and occlusions (which may cause drift), or when the object and camera motion puts the object beyond the search window radius.

One important reason that the existing trackers avoid employing a wider search radius is the potential distractions from the background Dinh et al. [2011]; Possegger et al. [2015]. It is not a trivial task to update a discriminative classifier when the negative sample space grows greatly with the samples coming from the extended search radius. In Henriques et al. [2015], extended set of training data is obtained by implicitly including all shifted versions of the given samples within the circulant matrices. However, it is impractical to apply the same trick for the negative samples, especially for the ones far away from the object.

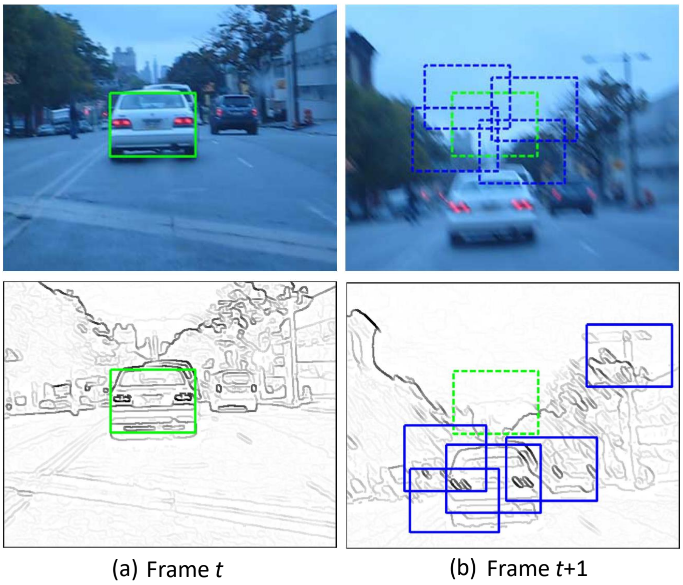


Figure 4.1: **Top row:** Most existing tracking-by-detection methods examine hypothesis locations within a local and heuristically defined search window around the last detected location. **Bottom row:** Our tracker seeks high-quality hypotheses over the entire image using instance-specific edge-box locations.

To overcome this, in this work we introduce a proposal generation procedure for handling the problem of sample selection, both for the object detection and the model update stages. Generally, the motion model limits the search radius and the applied sampling schemes disregard the contents presented on them. Instead of working within a limited search radius, we generate a small yet high-quality set of proposals efficiently in entire frame by using simple bottom-up, edge-based features Zitnick and Dollár [2014] as shown in Figure 4.1. Intuitively, edge information provides valuable guidance for object tracking since objects may often be identified by their silhouettes. In addition, concentrating on image regions where edge information is eminent allows efficient selection of more object-like proposals.

Our method can incorporate any existing object model including simple template matching models, e.g. normalized cross correlation (NCC) and sophisticated classifiers, e.g., structured support vector machines (SSVM). Using the object model, we adapt the edge-based features used in proposal generation. In an online fashion, we learn how to re-rank the proposal by a linear support vector machine, trained on the current proposals, with a crafted feature vector. Our proposal scheme, thus, generates windows that suggest certain similarity to the tracked object. This allows taking advantage of objectness to regulate the proposal selection in a temporally coherent manner instead of treating objectness as yet another cue by (linearly) combining the original tracking response with some objectness score. Since we adapt the generic edge-based objectness measure to the specific object, this selection is superior to replacing the search window with simple objectness responses.

Furthermore, for the chosen object model, we explore the best combination of global proposals provided by instance specific edge-based features and local candidates sampled around the previous location for model update (e.g., for negative support vectors in case of SSVM). We also adapt the size and scale to obtain the best proposals.

The benefits of our proposal generation is threefold:

- Our method can execute global search over entire image. Thus, it can track objects without making any assumption on object motion.

- The high-quality proposals increase the tracking accuracy since they allow including better hard negatives into training set, hence reduces drift.

- It adapts the specific object, thus provides better object model update (than generic proposals).

We validate the above arguments with two object models (from NCC tracker and Struck) and show that the incorporation of instance-specific proposals has potential to improve most detection-by-tracking approaches.

Our method is conceptually simple, easy to implement, and most importantly, provides the best results (at the time of submission) in comparison to all state-of-the-art trackers. Our method ranks as the top tracker on VOT2014 Kristan et al. [2014] benchmark as well as on OTB Wu et al. [2013] and TB50 Wu et al. [2015] datasets in comparison to the latest state-of-the-art including MEEM Zhang et al. [2014a], KCF Henriques et al. [2015], Struck Hare et al. [2011], and over twenty other methods.

4.2 Related Work

Providing an inclusive overview of the object tracking literature is outside the scope and capacity of this paper. We refer readers to the excellent surveys on object tracking. Here, we only compare with some relevant algorithms. We briefly examine different search schemes and then summarize recent object proposal methods.

Search Schemes in Tracking

There is a wide-spectrum of styles to select which windows will be tested in a current frame to locate the target object and also update its model.

Single Window Search - Several trackers use the local window around the former object location to find the object in the current frame. Examples include the tracking on Lie groups Tuzel et al. [2008], which applies iteratively a feature-motion regressor to estimate object window in the next frame, and the mean-shift tracker Comaniciu et al. [2003], which uses gradient-based local optimization to determine the mode of the underlying similarity distribution.

Particle-based Search - In recent years, tracking algorithms Ross et al. [2008]; Zhong et al. [2012]; Jia et al. [2012] based on particle filtering has been extensively studied. Particle filters apply importance sampling on the previous particle states (e.g. candidate locations) within mostly a mixed number of candidates. On the negative side, the random sampling is blind to the underlying texture, edgeness, and other spatial information.

Searching for the Hard Negatives - It is worthwhile to mention that tracking-by-detection, which allows an online trained classifier Avidan [2004]; Saffari et al. [2010b] as an object model to distinguish the object from its surrounding background, has recently become particularly popular. Rather than explicitly coupling to the accurate estimation of object position, Babenko et al. [2009] limits its focus on increasing the robustness to poorly labeled samples. Hare et al. [2011] proposes directly predicting the change in object location between frames by an online structured output SVM. Even though it produces comparably accurate tracking, it uniformly samples the state space to generate positive and negative support vectors. Such a brute force approach on a larger search window is computationally intractable.

Objectness in Object Detection

As shown in Hosang et al. [2014]; Zitnick and Dollár [2014], use of proposal has significantly improved the object detection benchmark along with the convolutional neural nets. Since, a subset of high-quality candidates are used for detection, object proposal methods improve not only the speed but also the accuracy by reducing false positives. The top performing detection methods Girshick et al. [2014]; Wang et al. [2013] for PASCAL VOC Everingham et al. [2015] use detection proposals.

Edge Box - Zitnick and Dollár [2014] proposes object candidates based on the observation that the number of contours wholly enclosed by a bounding box is an indicator of the likelihood of the box containing an object. Edge Box is designed as a fast algorithm to balance between speed and proposal recall. Its 1-D feature generates remarkably accurate results.

BING - Cheng et al. [2014] made a similar observation that generic objects with well-defined closed boundary can be discriminated by looking at the norm of gradients.They further designed a feature called binarized normed gradients (BING), which can be used for efficient objectness estimation and requires only a few atomic operations.

Objectness as Supportive Cue for Tracking

A straightforward strategy, i.e., linear combination of the original tracking confidence and an adaptive objectness score based on BING Cheng et al. [2014] is employed in Liang et al. [2016]. In Huang et al. [2015], a detection proposal scheme is applied as a post-processing step, mainly to improve the tracker's adaptability to scale and aspect ratio changes. These methods are substantially different from our work, where we adapt objectness to specific object using a separate classifier and generate high-quality proposal to regulate the tracking process.

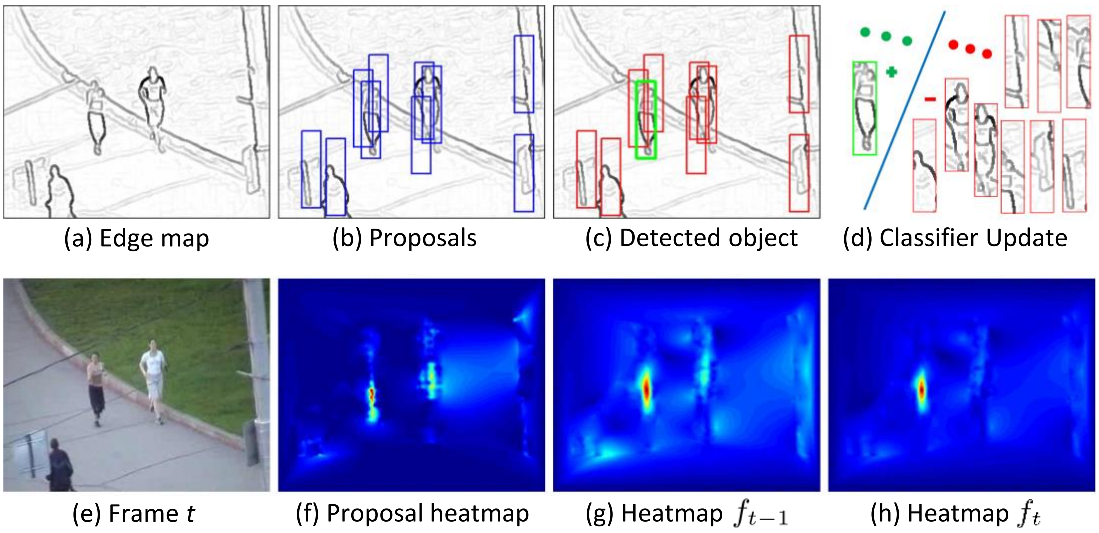


Figure 4.2: Framework of the proposed method. First column: (a) Edge map extracted from the current frame (e); Second column: (b) Object proposals in blue bounding boxes (Section 4.3.3) and (f) corresponding heatmap of instance specific proposals; Third column: (c) Detection results on proposals (green is detected as object) and (g) detection heatmap (by the proposed EBT classifier); Fourth column: (d) EBT is updated using the proposals and (h) detection heatmap with updated EBT. Notice that spurious hypotheses (bright regions in (g)) are suppressed significantly by treating them as negative samples.

4.3 Global Tracking with Proposals

4.3.1 Pipeline

A typical tracking-by-detection framework is composed mainly of motion model, observation model and model updater Wu et al. [2013]; Smeulders et al. [2014]; Wang et al. [2015b]. Motion model generates a set of candidates which might contain the target in the current frame based on the estimation from the previous frame. Observation model judges whether a candidate is the target based on the features extracted from it. Model updater online updates the observation model to adapt the change of the object appearance.

Suppose the object location is initialized manually at the first frame $t = 1$ and B_t is its bounding box at frame t. Then, given an observation model, i.e., a classification function f_{t-1} trained on the previous frames, the current location of the object is estimated through:

$$B_t^\star = \arg\max_{B_t \in \mathcal{B}_t} f_{t-1}(B_t), \tag{4.1}$$

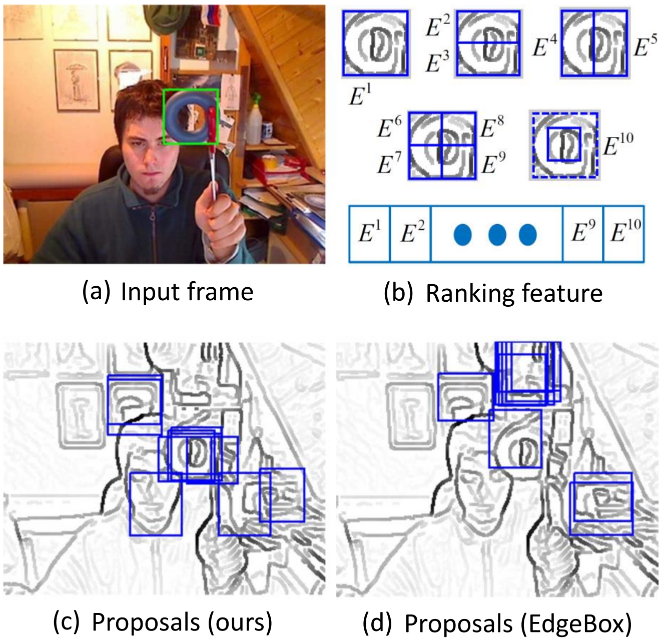
where $\mathcal{B}_t$ is a set of samples generated by the motion model at the current frame. To select samples, traditional trackers use heuristic search windows around the previously estimated object location for computational and accuracy reasons. For example, a search radius of 30 pixels is used in Hare et al. [2011].

Each sample is labeled by a classifier that models the object. The update routine will then revises its model $f_{t-1} \to f_t$ with the new location of the object to adapt possible appearance changes. It is not trivial to design a robust updating scheme Matthews et al. [2004]; Wang et al. [2015b]. As there is only one reliable example, the tracker must maintain a trade-off between adapting to new but possibly noisy examples collected during tracking and preventing the tracker from drifting to the background.

4.3.2 Our Method

The method proposed in this paper uses a similar framework as introduced in Section 4.3.1, yet we made two critical changes to the motion model. The first change is that we recognize not all candidate bounding boxes $B_t \in \mathcal{B}_t$ should be treated equally (as the traditional trackers often do) since those boxes possess different *object-like* appearance, i.e. *objectness* Krizhevsky et al. [2012]; Carreira and Sminchisescu [2012] characteristics, which should be taken into account. Secondly, we do not constrain the search radius to a small window that causes throwing so much available image information away.

To execute our changes, we take advantage of the sparse, simple, yet critical edge information. The current frame I_t is processed into an edge map as shown in Figure 6.2. Then, we employ an instance specific proposal method (explained in Section 4.3.3) build on top of the object proposal algorithm Zitnick and Dollár [2014] to produce a number of candidate bounding boxes (Figure 6.2 and 4.3) denoted as $\mathcal{B}_t^E$. Notice that, we impose a smooth size change constraint to the bounding boxes between consecutive frames.



(a) Input frame

(b) Ranking feature

(c) Proposals (ours)

(d) Proposals (EdgeBox)

Figure 4.3: Instance specific proposals. (a) Input frame (ground truth is the green bounding box); (b) 10-dimensional feature vector for ranking of the bounding boxes; (c) Top proposals using the proposed method; (d) Top proposals from Zitnick and Dollár [2014]. As shown, the instance specific proposals are far more precise.

Suppose the bounding box set generated by sampling only around the previous object location as $\mathcal{B}_t^R$ (as in traditional methods). Now we have two different sets of candidates, i.e., $\mathcal{B}_t^E$ and $\mathcal{B}_t^R$. The first one possesses object regularity while the second one is with no discriminative information. As shown in the experimental section 4.5.2, the choice of using only the proposals $\mathcal{B}_t^E$ generates the best results, better than combining them together. This confirms our argument that object proposals not only reduce the candidate sample space but also reduce spurious false positive and improve tracking accuracy. Our tracker will not drift to a textureless region like other trackers due to the *objectness* constraint.

During the update stage, we also have different options for using $\mathcal{B}_t^E$ and $\mathcal{B}_t^R$. As validated in the experimental part 4.5.2, the combination of using both of them to choose negative support vectors results in the best performance. This can be easily explained: $\mathcal{B}_t^E \backslash B_t^\star$ only represents other good *object-like* regions. By putting them as negative support vectors, we would only increase the discriminative power among *objects-like* candidates. However, the negative sample space contains a lot more other negative samples. Thus, the advantageous option is to augment $\mathcal{B}_t^E \backslash B_t^\star$ with $\mathcal{B}_t^R$ in order to achieve the best discriminative ability.

4.3.3 Instance Specific Proposals

Objectness attempts to generate quickly as few as possible hypotheses yet cover all of the objects present in an image. Take EdgeBox Zitnick and Dollár [2014] for example - it generates a pool of bounding boxes $\{B_{t,i}\}$ uniformly sampled in a sliding window manner, then ranks and extracts the top H candidates with the highest *objectness* score $E_{t,i}$, represented by:

$$\mathcal{B}_t^{EB} = \{B_{t,i}|E_{t,i}\}_H. \tag{4.2}$$

$E_{t,i}$ is basically a weighted and normalized number of contours wholly enclosed by the bounding box $B_{t,i}$. This feature can be calculated very efficiently in real-time. We refer Zitnick and Dollár [2014] for more details.

Instead of directly applying the computed proposals $\mathcal{B}_t^{EB}$ for tracking, we argue that the object instance level properties should be taken into account. As such, there is a strong object prior in terms of its geometric structure of contours and size in contrast to object detection where the goal is to locate all instances of all object classes in the image. EdgeBox generates proposals that favors bounding boxes with many internal contour segments, thus it is likely to miss the target in a cluttered background as shown in Figure 4.3.

To this end, we incorporated an online updated linear SVM Wang et al. [2010] classifier f_{t-1}^R to re-rank proposals and determine the top H proposals based on their classification scores:

$$\mathcal{B}_t^{E} = \{B_{t,i}|f_{t-1}^R(B_{t,i})\}_H, \tag{4.3}$$

with a 10-dimensional feature vector $\{E_{t,i}^1, \ldots, E_{t,i}^{10}\}$ as shown in Figure 4.3. This feature characterizes the spatial structure of edge information. It concatenates EdgeBox scores corresponding to Haar wavelet like partitioning of the bounding box $B_{t,i}$. Notice that, only the bounding boxes whose initial *objectness* scores are above a threshold, i.e., $\mathcal{B}_t^{EB_T} = \{B_{t,i}|E_{t,i} > e_T\}$ (in all experiments $e_T = 0.005$) are accepted into the classifier for re-ranking to save computing time.

The re-ranking classifier is initialized using the top EdgeBox proposal (top 200 in all experiments) and then online updated at every 5 frames with the same number of proposals. The estimated position gives the positive sample and bounding boxes which overlap the estimation less than 0.5 are assigned as negative ones. We use the implementation and parameters as in Zhang et al. [2014a].

4.3.4 Candidate Classification

We use the following decision function to estimate the new location of the object (Figure 6.2):

$$B_t^\star = \arg\max_{B_t \in \mathcal{B}_t} f_{t-1}(B_t) + s(B_t, B_{t-1}^\star). \tag{4.4}$$

Here $s(B_t, B_{t-1}^\star)$ is a term representing the motion smoothness between the previous object location and the candidate box. This is important in our formulation as we are testing candidates all over the image, though not penalizing it too much. We use a simple function in this paper: $s(B_t, B_{t-1}^\star) = w_s \exp(-\frac{1}{2\sigma^2}\|c(B_t) - c(B_{t-1}^\star)\|^2)$, where $c(B_t)$ is the center of bounding box B_t, $w_s = 0.1$ and σ is set as the diagonal length of the initialized bounding box.

4.4 Proposed Trackers

Two core object models are integrated in the proposal tracker. The first one (called as EBT to indicate its relation to EdgeBox) follows a popular structured support vector machine (SSVM) framework Hare et al. [2011], which shows good performance on several benchmarks Wu et al. [2013]; Smeulders et al. [2014]. We additionally incorporated a much simpler, normalized cross correlation (NCC) template matching, called as NCC$_{\mathbf{EB}}$, to investigate how much additional performance improvement our method is able to provide.

4.4.1 EBT Tracker

Suppose the support vector set maintained by the SSVM as $\mathcal{V}_{t-1}$ and the classification function can be written as a weighted sum of affinities Blaschko and Lampert [2008]; Hare et al. [2011]:

$$f_{t-1}^S(B_t) = \sum_{B_{t-1}^i \in \mathcal{V}_{t-1}} w_{t-1}^i k(B_{t-1}^i, B_t), \tag{4.5}$$

where w_{t-1}^i is a scalar weight associated with the support vector B_{t-1}^i. Kernel function $k(B_{t-1}^i, B_t)$ calculates the affinity between two feature vectors extracted from B_{t-1}^i and B_t respectively. The classifier is updated in an online fashion using Bordes et al. [2007] with a budget Wang et al. [2010]. Intersection kernel is used and other parameters are set same as Hare et al. [2011].

To take advantage of the small set of proposals, we use histogram features obtained by concatenating 16-bin intensity histograms from a spatial pyramid of 5 levels and RGB channels separately. At each level L, the patch is divided into $L \times L$ cells, resulting in a 2640-D feature vector, comparing to the 480-D feature used in Hare et al. [2011], while running at a similar speed. The performance gain of using the richer feature is demonstrated in the experimental section 4.5.2.

4.4.2 NCC$_{\mathbf{EB}}$ Tracker

The classification function for the normalized cross correlation can be written as:

$$f^N_{t-1}(B_t) = \rho(B_t, B_{Temp}), \tag{4.6}$$

where ρ calculates the normalized cross-correlation coefficient Briechle and Hanebeck [2001] between the candidate patch and the object template. This procedure can be accelerated using the fast Fourier transform (FFT) trick. We compared the proposed NCC$_{\mathbf{EB}}$ tracker with instance-specific proposals and fixed template with: (1) NCC, an implementation from Kristan et al. [2014], uses local exhaustive search, and has no update; and (2) IMPNCC, an improved NCC version from Kristan et al. [2014], uses local exhaustive search, online update, and Kalman Filter Kalman et al. [1960] for trajectory smoothness.

4.5 Experiments

In the first part, we compare our method with the state-of-the-art trackers on benchmark datasets for a general performance evaluation. We also test on fast-motion related categories to put it under the spotlight to understand how well our method can handle the challenging scenarios such as fast moving objects, randomly moving objects, and tracking under low-frame-rate. In the second part, we analyze different components of our method.

4.5.1 Full Benchmark Evaluations

Our method is tested on three large datasets: OTB Wu et al. [2013], TB50 Wu et al. [2015] and VOT2014 Kristan et al. [2014]. The first two of these datasets are composed of around 50 sequences each. They are annotated with ground truth bounding boxes and various visual attributes. TB50 is an upgraded version of OTB and contains much more challenging sequences. VOT2014 dataset selectively collects 25 sequences from various datasets and allows the tracker to re-initialize once the tracker drifts away from the object.

We compare against the existing algorithms on respective benchmarks and additionally two recent works: KCF Henriques et al. [2015] and MEEM Zhang et al. [2014a]. Evaluation metrics and code are provided by the respective benchmark. For OTB and TB50, we employ the one-pass evaluation (OPE) and use two metrics: *precision plot* and *success plot*. The former metric calculates the rate of frames whose center location is within a certain threshold distance with the ground truth. The latter one calculates a same ratio but based on bounding box overlap threshold.

Table 4.1: Area Under Curve (AUC) of *success plot* and Precision Score (20 pixels threshold) reported on various datasets (AUC/PS) corresponding to the one-pass evaluation (OPE).

	Pro. EBT	KCF	MEEM	Struck	SCM	ASLA	TLD	CXT	CSK
OTB	**58.1/84.8**	51.7/74.2	56.4/82.5	47.2/65.3	49.8/64.8	43.4/52.9	43.4/60.1	42.3/57.0	39.6/54.1
TB50	**49.6/73.9**	40.2/61.1	47.9/72.3	36.3/49.9	35.5/47.8	35.8/46.2	32.1/45.0	32.1/43.2	31.4/43.0

Table 4.2: Performance on VOT2014.

	Final Rank	Acc. Rank	Rob. Rank
Proposed EBT	**13.03**	15.81	10.24
PLT_{14}	13.75	16.66	10.84
PLT_{13}	14.26	18.59	**9.92**
DGT	14.54	15.48	13.61
DSST	15.25	13.40	17.09
KCF	15.25	**12.20**	18.29
SAMF	15.47	12.79	18.15
MEEM	18.95	21.15	16.76
Struck	22.83	22.30	23.36
Proposed $\text{NCC}_{\textbf{EB}}$	27.27	24.20	30.35
MIL	27.69	31.24	24.14
FSDT	27.86	25.97	29.75
IMPNCC	27.99	26.05	29.94
CT	28.26	29.14	27.38
FRT	28.64	25.02	32.26
NCC	29.30	22.32	36.28

Parameters For EdgeBox proposals, the sampling step of sliding window is set at $\alpha = 0.85$ since we aim for a high accurate localization. The minimal and maximal areas are 0.5 and 2 of the area of the previous estimated bounding box respectively. Non-maximum suppression parameter is fixed at $\beta = 0.8$. The maximum number of proposal is 200 (more discussion in Section 4.5.2).

4.5.1.1 Benchmark Results

The results are summarized in Table 4.2, 4.1 and Figure 4.4. Our EBT tracker ranks as the best tracker on VOT2014 as shown in Table 4.2. We use the original VOT protocol. EBT achieves the best overall performance in all datasets[1]. It consistently outperforms the state-of-the-art trackers and improves the base Struck tracker by a large margin. A few examples can be found in Figure 4.5.

[1] As stated in FAQ of the official VOT website, the rankings would not be identical to the Table 1 in the 2014 paper.

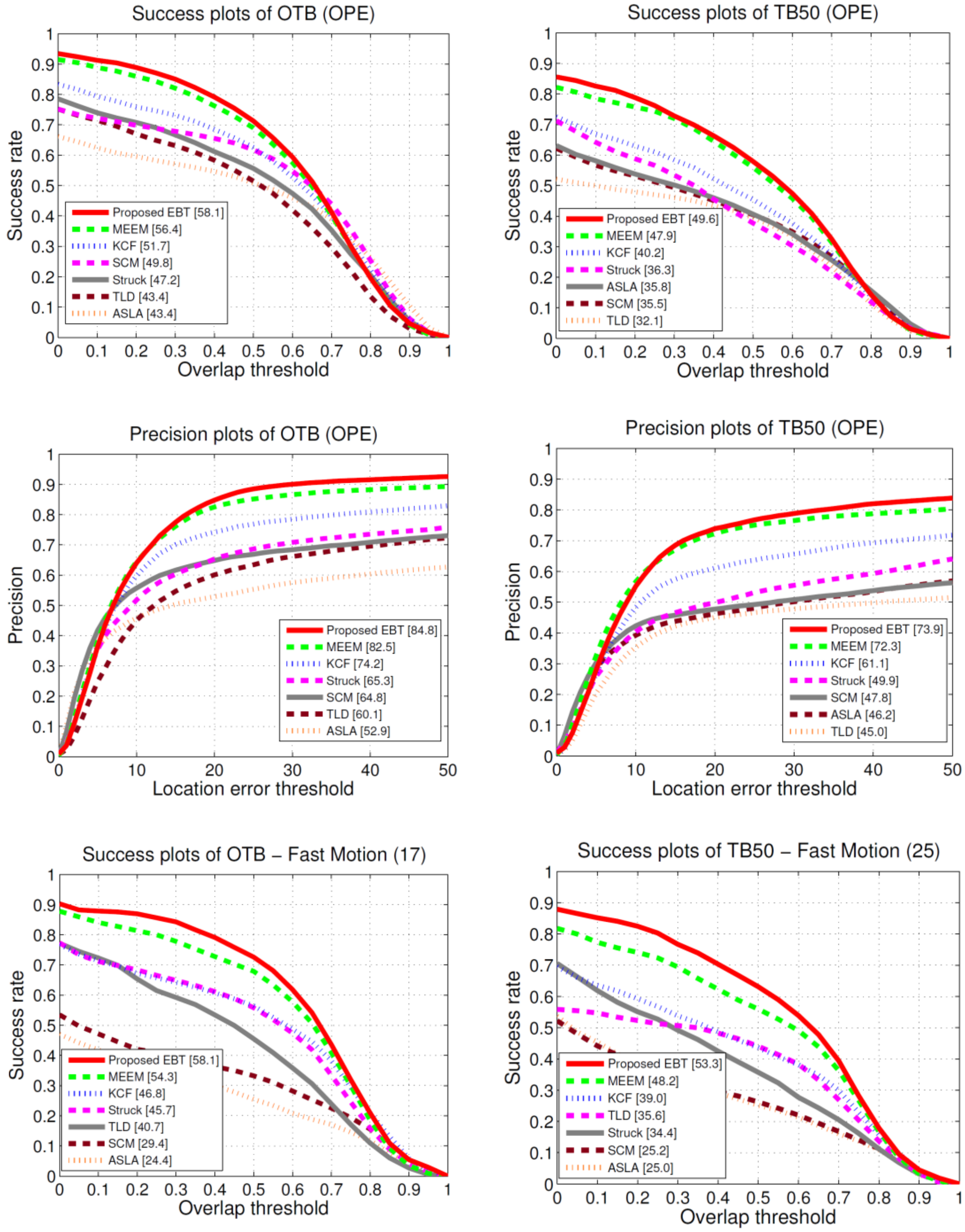


Figure 4.4: *Success plot* and *precison plot* on two large benchmarks: OTB and TB50. Algorithms are ranked by the area under the curve and the precision score (20 pixels threshold). Our method achieves consistently superior performance.

Table 4.3: Area Under Curve (AUC) of *success plot* and Precision Score (20 pixels threshold) reported on various fast-motion related categories (AUC/PS). FM: fast motion, MB: motion blur, MC: moving camera. fps: frames per second.

Attributes	Pro. EBT	KCF	MEEM	Struck	SCM	ASLA	TLD
FM (17) (OTB)	**58.1/77.8**	46.8/61.0	54.3/71.4	45.7/59.6	29.4/32.9	24.4/24.6	40.7/53.2
MB (12)	**58.3/77.1**	50.8/66.0	53.0/68.0	42.6/54.0	29.5/33.3	25.1/26.8	39.0/49.0
FM (25) (TB50)	**53.3/74.5**	39.0/54.0	48.2/68.4	34.4/42.5	25.2/29.6	25.0/29.6	35.6/46.5
MB (19)	**54.9/78.5**	40.6/56.4	52.8/72.9	30.9/35.5	21.7/25.1	23.3/25.5	39.3/49.7
MC (22) (ALOV300)	**60.9/68.4**	56.4/62.9	57.2/65.1	44.9/44.8	35.7/37.9	38.6/38.8	56.1/67.9
fps	4.4	**70.9**	7.1	4.8	0.3	3.8	8.8

Even the proposed NCC$_{\mathbf{EB}}$ tracker using only template matching manages to improve the simple NCC tracker significantly and outperforms several other trackers including the IMPNCC tracker, which has incorporated sophisticated mechanisms in comparison to ours and NCC. This result is not surprising since the incorporation of objectness has proven to be a successful strategy in single image object detection Girshick et al. [2014]; Wang et al. [2013]; Everingham et al. [2015]. We believe that our method is a counterpart in the tracking domain as no existing tracking methods successfully adopted such objectness schemes before, to the best of our knowledge.

4.5.1.2 Tracking Fast Objects

Since our method searches over the entire image, it is suitable for tracking fast moving objects, which could move outside of the search radius of the traditional trackers. As shown in Table 4.3, our method outperforms other trackers in the fast-motion related categories as well.

We also tested our method on an extra category *Moving Camera* from ALOV300 Smeulders et al. [2014]. This category contains many sequences that depict camera shake, sudden object motion, and abrupt jumps. ALOV300 provides a high number of short sequences with 14 visual attributes. The main source of their data is real-life videos from YouTube.

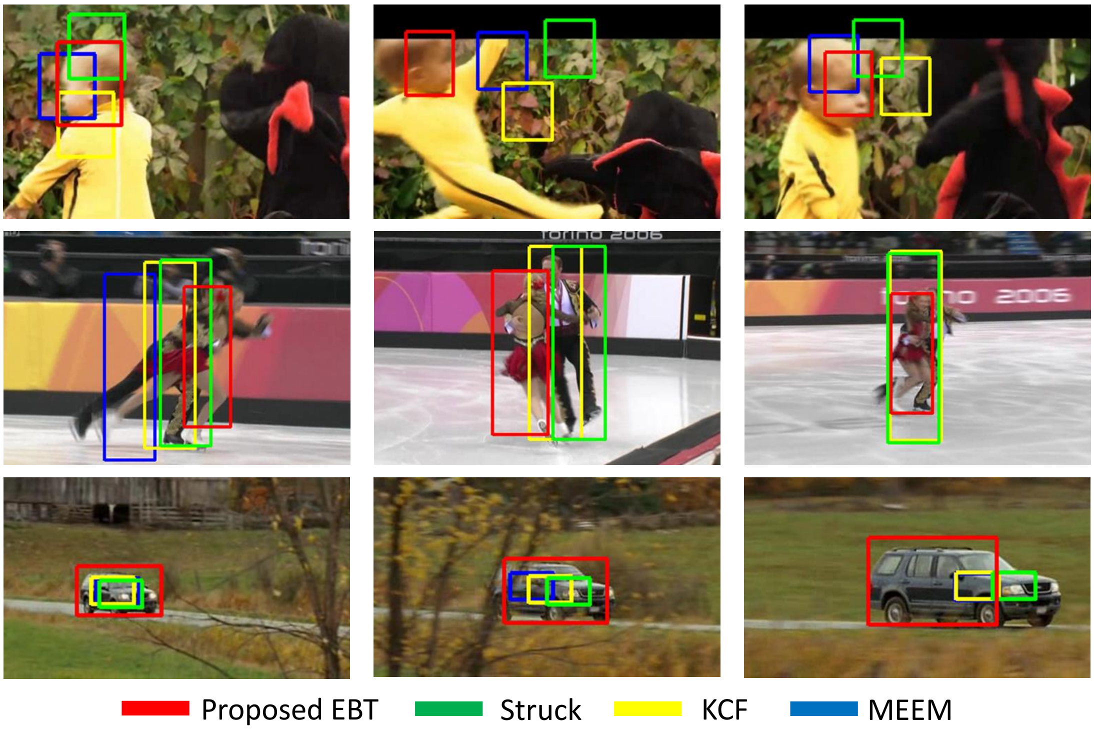


Figure 4.5: Qualitative comparisons with the state-of-the art trackers on the *DragonBaby*, *Skating2*, and *CarScale* videos. Our method exhibits robustness in challenging scenarios such as fast motion, occlusion, and scale changing.

Tracking under Ultra-Low-Frame-Rate We additionally created a dataset, called as VOT2014+ by temporally sampling sequences at every 20 frames on VOT2014, thus, it contains 20× faster moving objects. Our method is tested against with other top-ranked trackers, KCF and MEEM. Even though both MEEM and KCF rapidly failed, our tracker retained **very high** performance scores (see Table.4.4).

Table 4.4: Performance on the low-fps dataset.

	Pro. EBT	KCF	MEEM
VOT2014	**46.7/65.9**	38.9/53.7	44.5/62.3
VOT2014+	**43.7/58.5**	28.4/34.1	37.5/47.7

Table 4.5: Results for different combinations of $\mathcal{B}_t^E$ and $\mathcal{B}_t^R$.

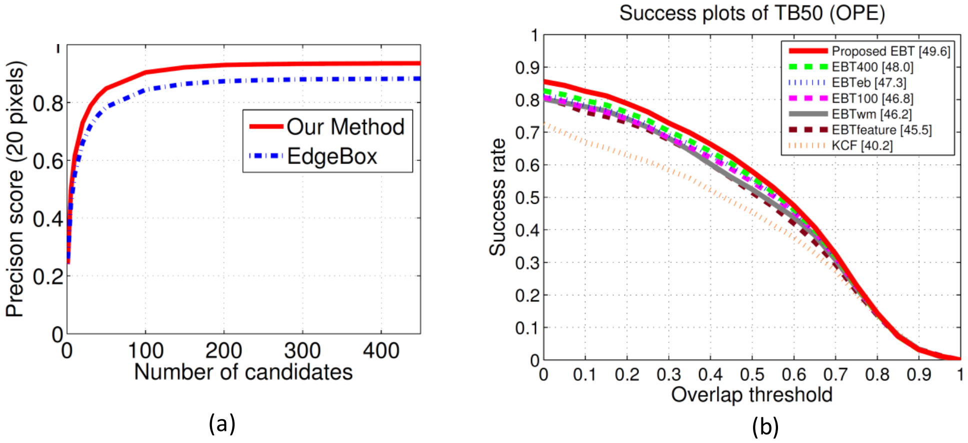
TB50	(Test) $\mathcal{B}_t^R$	$\mathcal{B}_t^E$	$\mathcal{B}_t^E + \mathcal{B}_t^R$
$\mathcal{B}_t^R$ (Update)	41.1/58.7	44.7/64.2	42.7/59.4
$\mathcal{B}_t^E$	40.1/56.3	46.5/68.6	43.0/61.8
$\mathcal{B}_t^E + \mathcal{B}_t^R$	39.2/56.5	**49.6/73.9**	43.2/63.6

4.5.2 Further Remarks

Combination of $\mathcal{B}_t^E$ and $\mathcal{B}_t^R$ As discussed in Section 4.3.2, we tested different combinations of the hypothesis proposals $\mathcal{B}_t^E$ and candidate bounding boxes $\mathcal{B}_t^R$ sampled around the previous object location within a radius. The results are shown in Table4.5. For combinations which use only $\mathcal{B}_t^R$ in the testing stage, we apply an exhaustive sampling within a 30-pixels radius to achieve a comparable result. For the others which use $\mathcal{B}_t^R$, we only generate 80 samples uniformly within a 30-pixels radius. Our main discussion about these results can be found in Section 4.3.2. We observed the combination of using samples from the hypothesis proposals and local region in update stage and samples only from the proposed locations in the test stage performs the best.

Number of Proposals To quantitatively compare the proposed instance specific proposals and the one using Edge Box Zitnick and Dollár [2014], we analyzed the upper bound performance with respect to varying number of proposals as shown in Figure 4.6 (a). A variant denoted as EBTeb using EdgeBox proposals instead of ours is also tested and available in Figure 4.6 (b). Both results show that the proposed re-ranking method outperforms the one directly applies EdgeBox. We also tested the variants using different number of proposals. EBT100 and EBT400 use 100 and 400 respectively, comparing to the proposed EBT that uses 200. Our observations are, using insufficient number of proposal leads to a bad coverage of the false positives as well as the object, while using a large number of proposals attracts spurious candidates.

Richer Features and Motion Constraint EBTfeature denotes the variant using a lower dimensional 480-D feature. This version has lower performance than the one uses 2640-D feature as expected. More details about the feature can be found in Section 4.4.1. EBTwm denotes the variant without using the smoothness term $s(B_t, B_{t-1}^\star)$ in Function 4.4. The success rate dropped due to the fact that the motion in the tracking sequences is not completely random.



(a)

(b)

Figure 4.6: (a) The performance bounds for using EdgeBox proposals and the proposed instance-specific proposal method on TB50. The best candidate in each frame is used for calculating the performance. (b) *Success plot* of variants of the proposed method on TB50. Details can be found in Section 4.5.2.

Table 4.6: Performance when BING is used instead of Edge Box.

	Struck	BING-VOC	BING-Adapt
TB50	**36.3/49.9**	30.8/47.6	33.7/48.0

Proposals using BING We evaluated another popular object proposal method, BING Cheng et al. [2014], for proposals. Two ways of incorporation were tested. The first one (BING-VOC) uses the pretrained model on VOC dataset Everingham et al. [2015], while the second one (BING-Adapt) relearns the model using the first frame of each sequence. We tested these two variants on TB50. Results are in Table 4.6. Both performances are worse than the baseline Struck. This is expected. As shown Hosang et al. [2014]; Zitnick and Dollár [2014], BING results in a relatively low recall of the objects, which is one reason for its mediocre performance.

Computational Speed The computational speed of the proposed is comparable to the state-of-the-art trackers even though we can track over the entire image. The proposal part takes less than 100 milliseconds and the overall tracking speed is available in Table 4.3.

4.6 Summary

This work presented a robust method that can locate objects that are moving randomly and very fast, as well as perform tracking under extremely low-frame rates. To the best of our knowledge, our tracker achieves the **best results on all** common benchmark **datasets** including OTB Wu et al. [2013], TB50 Wu et al. [2015], VOT2014 Kristan et al. [2014] and ALOV300 Smeulders et al. [2014] at its submission time.

Chapter 5

Affine Tracking on Lie Groups using Structured SVM

Previous two chapters addressed two limitations of conventional "tracking-by-detection" based trackers. In this chapter, we advance the state-of-the-art by tracking image regions that undergo affine transformations such as translation, rotation, scale, dilatation, and shear deformations that span the six degrees of freedom of motion, considering the importance of the object motion to visual tracking [Smeulders et al., 2014].

5.1 Introduction

Tracking affine transformations of image regions is essential in many applications from pose estimation to object recognition, and still one of the most challenging tasks in computer vision. In addition to critical problems such as appearance changes, lighting variations, indiscriminate backgrounds and occlusions that arise in tracking translational motion of an image window, tracking affine motion confronts with a higher dimensional parameter space that blows up the computational complexity and non-Euclidean manifold structure of motion matrices that leads into inaccurate distance computations when they are flattened.

Existing methods often attempt to solve the affine motion tracking problem in vector space and can be roughly categorized into state-space estimation [Lucas and Kanade, 1981; Vetter and Poggio, 1997; Blake and Isard, 1996; Boykov and Huttenlocher, 2000; Albrecht et al., 2008], template alignment Cootes et al. [1998]; Baker and Matthews [2001, 2004]; Matthews and Baker [2004] and feature correspondence Ozuysal et al. [2007]; Wagner et al. [2008]; He et al. [2009] approaches. State-space estimators assume affine tracking as a Markovian process and construct a probability density function of object parameters, which is a normal distribution in case of Kalman filtering and a multi-modal distribution for particle filtering. In theory, particle filter can track any parametric variation including affine motion. However, its dependency to random sampling induces degenerate likelihood estimations especially for the higher dimensional parameter spaces. In template alignment, parametrized motion models are estimated using appearance and shape models that are usually fitted by nonlinear optimization. One shortcoming of these algorithms is that they require computation of partial derivatives, Jacobian, and Hessian for each iteration, which makes them impractical. Feature point based methods mainly differ in the type of features and descriptors used for matching the object model to the current frame. Their shortcoming is that in many cases only little texture is present on the object.

It is worthwhile to mention that tracking-by-detection, which allows an online trained classifier Avidan [2004]; Grabner et al. [2006]; Saffari et al. [2010a] as an object model to distinguish the object from its surrounding background, has recently become particularly popular. Most tracking-by-detection update the classifier by a set of binary labeled training samples that are obtained using heuristics such as the distance of a sample from the estimated object location. One implication of this is that slight inaccuracy during tracking can lead to poorly labeled samples, thus, tracking failure. Rather than explicitly coupling to the accurate estimation of object position, Babenko et al. [2009]; Masnadi-Shirazi et al. [2010]; Saffari et al. [2010b] limit their focus on increasing the robustness to poorly labeled samples.

As a remedy, Hare et al. [2011] proposed directly predicting the change in object location between frames by an online structured output SVM. Even though Hare et al. [2011] produces comparably accurate tracking for translational motion, for affine motion it has two major drawbacks. Since it strictly depends on a bounding box overlap based loss function in its compatibility function, it can not distinguish rotations and complex affine deformations. Besides, it uniformly samples the state space to generate positive and negative support vectors. Such a brute force approach on a high dimensional search space is computationally intractable.

Unlike the prevalent practice, the set of 2D affine transformations do not constitute a vector space, but rather an analytical manifold $\mathcal{M}$ that has the structure of a Lie group $\mathrm{Aff}(\mathbb{R}^2)$. Existing methods for the most part disregard this manifold structure and flatten the topology in a vector space. Vector forms cannot globally parameterize the intrinsic topology on $\mathcal{M}$ in a homogeneous fashion, thus fail to accurately evaluate the distance between affine motion matrices causing unreliable tracking performance.

There are only a few relevant work for parameter estimation on Lie groups, e.g. Drummond and Cipolla [2000] for tracking an affine snake and Bayro-Corrochano and Ortegon-Aguilar [2004]; Tuzel et al. [2008]; Kwon et al. [2009] for tracking a template. However, Bayro-Corrochano and Ortegon-Aguilar [2004] fails to account for the noncommutativity of the matrix multiplications thus the estimations are valid only around the initial transformation. Tuzel et al. [2008] learned the correlation between affine motions and the observed descriptors using a regression model on Lie algebra. Inherent topology is considered by Kwon et al. [2009] where a conventional particle filter based tracker where the state dynamics are defined on $\mathcal{M}$ using a log-Euclidean metric. However, none of these methods incorporate an efficient mechanism to incorporate object appearance changes.

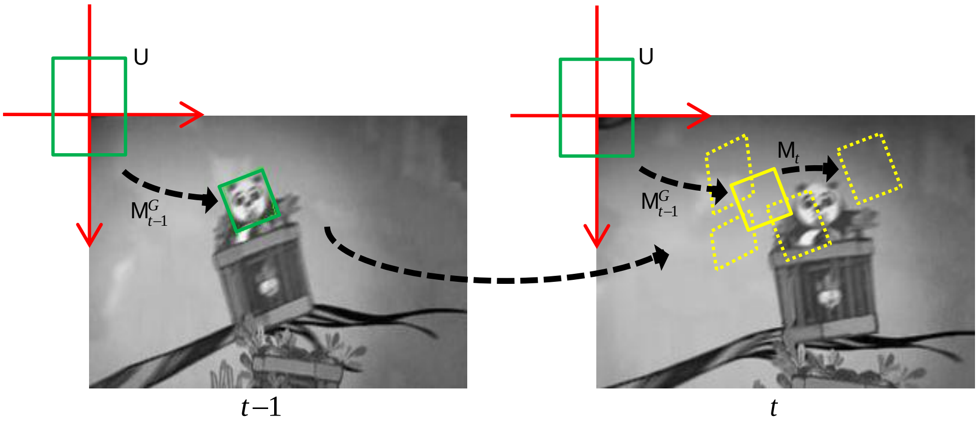


Figure 5.1: Predict the location of object region in frame t based on the location obtained in the previous frame $t-1$.

To overcome the shortcomings of the existing methods, here we propose a novel affine tracking-by-detection method, Lie-Struck, that takes advantage of the intrinsic topology using a geodesic distance when it compares two motion matrices. Our method incorporates Lie group structure $\mathrm{Aff}(\mathbb{R}^2)$ presented in Tuzel et al. [2008] into a structured output SVM classifier introduced in Hare et al. [2011] to directly estimate 2D affine transformation using an appearance based prediction function. Unlike Tuzel et al. [2008], Lie-Struck can efficiently learn object's temporal appearance changes. Unlike Hare et al. [2011], our method can accurately track affine transformations. We demonstrate that these combined motion and appearance model structures significantly improve the tracking performance while an incorporated particle filter mechanism keeps the computational complexity minimal. Experimentally, we show that our method consistently outperforms the state-of-the-art trackers on various scenarios even when the object undergoes challenging aspects such as occlusion and motion blur.

We also introduce a manually annotated affine tracking dataset since most existing datasets have only rectangle ground truth regions, thus are not suitable for performance evaluation of affine trackers.

5.2 Lie-Struck Formulation

2D affine motion matrices constitute Lie group $\mathrm{Aff}(\mathbb{R}^2)$ with the structure of a differentiable manifold $\mathcal{M}$ such that the group operations, multiplication and inverse, are differentiable maps. The structure of $\mathrm{Aff}(\mathbb{R}^2)$ is a 6 dimensional manifold with the 3×3 affine transformation matrix as:

$$\mathsf{M} = \begin{pmatrix} \mathsf{A} & v \\ 0 & 1 \end{pmatrix} \tag{5.1}$$

where, A is a 2×2 matrix (for rotation, scale, sheer), and $v \in \mathbb{R}^2$ (for translation). Here, the tangent space $\mathcal{T}_\mathsf{I}\mathcal{M}$ to the identity element I of the group forms a Lie algebra. The distances on the manifold $\mathcal{M}$ are measured by the lengths of the curves connecting the points, and the minimum length curve between two points is called the geodesic. From I there exists a unique geodesic starting with point m. The exponential map, $\exp : \mathcal{T}_\mathsf{I}\mathcal{M} \to \mathcal{M}$ maps the point m on the tangent space to the point reached by this geodesic. Let $\exp(\mathsf{m}) = \mathsf{M}$, then the length of the geodesic is given by $\rho(\mathsf{I}, \mathsf{M}) = \|\mathsf{m}\|$. The inverse mapping is given by $\log : \mathcal{M} \to \mathcal{T}_\mathsf{I}\mathcal{M}$. Using the logarithm map and the group operation, the geodesic distance between two group elements is measured by

$$\rho(\mathsf{M}_1, \mathsf{M}_2) = \|\log(\mathsf{M}_1^{-1}\mathsf{M}_2)\|. \tag{5.2}$$

The exponential and logarithm maps of a motion matrix are given by

$$\exp(\mathsf{m}) = \sum_{n=0}^{\infty} \frac{1}{n!}\mathsf{m}^n \quad \log(\mathsf{M}) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n}(\mathsf{M} - \mathsf{I})^n. \tag{5.3}$$

In case M_{t-1}^G represents the affine transformation from the unit rectangle (for normalization purposes, we map back onto unit rectangle U when we compute image features) to the object region in frame $t-1$, the *incremental* motion M_t is defined as

$$\mathsf{M}_t^G = \mathsf{M}_t \mathsf{M}_{t-1}^G, \tag{5.4}$$

where M_t^G is the object box parallelogram in the frame t.

Inspired by Hare et al. [2011], we treat the affine tracking-by-detection problem as learning a prediction function $f : x \to \mathcal{M}$ where x is the feature vector extracted from the object region. The prediction function f is determined in a structured output SVM framework Blaschko and Lampert [2008]. Let $F : x \times \mathcal{M} \to \mathbb{R}$ be a discriminant function that maps both an affine motion matrix and the feature corresponding to its region in the image to a scalar label. Here, we assign the discriminant function as it as

$$\mathsf{M}_t = f(x_t(\mathsf{M}_{t-1}^G)) = \arg\max_{\mathsf{M} \in \mathcal{M}} F(x_t(\mathsf{M}_{t-1}^G), \mathsf{M}), \tag{5.5}$$

where M_{t-1}^G is the location of the object region in frame $t-1$ as we stated above. Here, M_{t-1}^G transforms the unit rectangle U to the parallelogram that bounds the target region in frame $t-1$.

The discriminant function F measures the compatibility between the feature x and incremental affine motion M pairs $(x_t(\mathsf{M}^G_{t-1}), \mathsf{M})$. In other words, F has a higher score when the affine transformation M leads into a more accurate location of object region in frame t. By exhaustively searching over all possible transformations $\mathsf{M} \in \mathcal{M}$ near the object region in the previous frame, the target M_t can be obtained as the maximizer of $F(x_t(\mathsf{M}^G_{t-1}), \mathsf{M})$.

As the structured output SVM formulations [Blaschko and Lampert, 2008], we express the discriminant function in the form of

$$F(x_t(\mathsf{M}^G_{t-1}), \mathsf{M}) = \langle \boldsymbol{w}, \phi(x_t(\mathsf{M}^G_{t-1}), \mathsf{M})\rangle, \tag{5.6}$$

where $\phi(x_t(\mathsf{M}^G_{t-1}), \mathsf{M})$ is a raising function from the joint (feature, motion) space to a transform space. The specific form of ϕ is not necessarily to be defined explicitly by taking advantage of the kernel-based method Tsochantaridis et al. [2005] (Section 5.3). The linear coefficient vector $\boldsymbol{w}$ can be learned through an incrementally obtained set of example pairs

$$\mathcal{S}_t = \{(x_1(\mathsf{M}^G_0), \mathsf{M}_1), \ldots, (x_t(\mathsf{M}^G_{t-1}), \mathsf{M}_t)\}, \tag{5.7}$$

Learning procedure is then minimizing the following convex objective function:

$$\begin{aligned} &\min_{\boldsymbol{w}} \;\; \frac{1}{2}\|\boldsymbol{w}\|^2 + c\sum_{i=1}^{t} \xi_i, \\ &\text{s.t.} \quad \xi_i \geq 0, \forall i \\ &\qquad \langle \boldsymbol{w}, \delta\phi(x_i(\mathsf{M}^G_{i-1}), \mathsf{M})\rangle \geq \mathcal{L}(\mathsf{M}_i, \mathsf{M}) - \xi_i, \forall i, \forall \mathsf{M} \neq \mathsf{M}_i, \end{aligned} \tag{5.8}$$

where $\delta\phi(x_i(\mathsf{M}^G_{i-1}), \mathsf{M}) = \phi(x_i(\mathsf{M}^G_{i-1}), \mathsf{M}_i) - \phi(x_i(\mathsf{M}^G_{i-1}), \mathsf{M})$ and c is the blending weight for the (soft-margin) errors. Thus optimization of (5.8) finds such $\boldsymbol{w}$ that enables discriminant function (6.1) to produce lower values for $\mathsf{M} \neq \mathsf{M}_i$, by a margin depends on a loss function $\mathcal{L}(\mathsf{M}_i, \mathsf{M})$. This loss function should satisfy $\mathcal{L}(\mathsf{M}_i, \mathsf{M}) = 0$ iff $\mathsf{M} = \mathsf{M}_i$ and decrease towards 0 as M and M_i become more similar.

5.2.1 Loss Function

Loss function $\mathcal{L}(\mathsf{M}_i, \mathsf{M})$ plays an important role in optimizing (5.8), as it quantifies the loss associated with a prediction M, if the true output value is M_i Tsochantaridis et al. [2005]. It allows to address the issue raised in the previous works that all negative samples being treated equally Hare et al. [2011]. Thus, the standard zero-one loss function typically used in classification is not appropriate for this problem and we introduce three loss function forms in this paper.

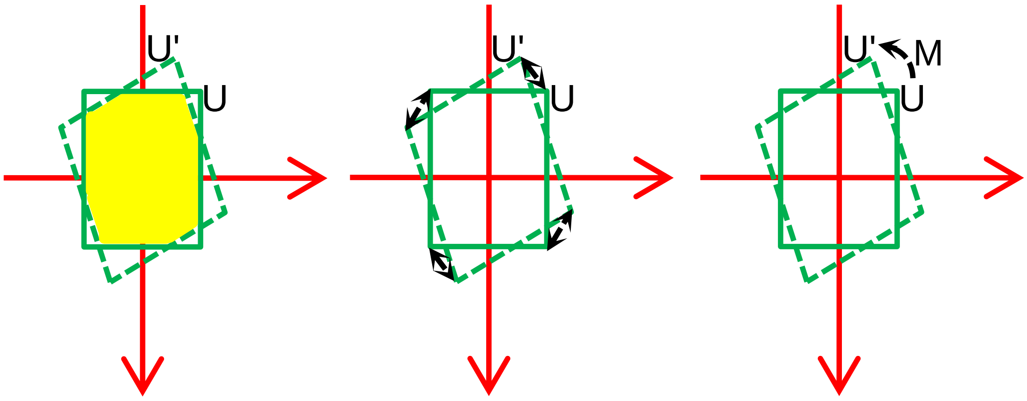


Figure 5.2: An illustration of three ways to design the loss function. From left to right: overlap rate based, average vertex distance based and geodesic distance based.

• Conventional: Based on Overlap Rate

For affine tracking, one can use the loss function based on the bounding box overlap rate as in Hare et al. [2011]:

$$\mathcal{L}_o(\mathsf{M}_i, \mathsf{M}) = 1 - \mathcal{O}_{\mathsf{M}^G_{i-1}}(\mathsf{M}_i, \mathsf{M}), \tag{5.9}$$

$$\mathcal{O}_{\mathsf{M}^G_{i-1}}(\mathsf{M}_i, \mathsf{M}) = \frac{(\mathsf{M}_i\mathsf{M}^G_{i-1}) \cap (\mathsf{M}\mathsf{M}^G_{i-1})}{(\mathsf{M}_i\mathsf{M}^G_{i-1}) \cup (\mathsf{M}\mathsf{M}^G_{i-1})}. \tag{5.10}$$

Function (5.10) measures the degree of overlap between two parallelograms in frame i: $\mathsf{M}_i\mathsf{M}^G_{i-1}$ and $\mathsf{M}\mathsf{M}^G_{i-1}$, as illustrated on the left of Figure 5.2.

The overlap rate based loss function is used in Hare et al. [2011] for the translational motion. This treatment is, however, more of a coarse heuristic approach than a rigorous mathematical method for our problem, as the overlap rate measurement (5.10) can be very ambiguous when representing the similarity between two affine transformations M_i and M. Take a unit rectangle for example, rotating it by 90 degrees in any direction produces 100% overlap rate, which indicates the loss function of Hare et al. [2011] based on overlap rate is not suitable for affine tracking.

- **Average Vertex Distance based Loss Function**

Instead of using the overlap rate measurement (5.10), the following average vertex distance based loss function can be applied:

$$\mathcal{L}_v(\mathsf{M}_i, \mathsf{M}) = 1 - \exp(-\tau_v \rho_v^2(\mathsf{M}_i, \mathsf{M})), \tag{5.11}$$

where τ_v is a constant and $\rho_v(\mathsf{M}_i, \mathsf{M})$ is the mean distance of four pairs of corresponding vertices as shown on the middle of Figure 5.2:

$$\rho_v(\mathsf{M}_i, \mathsf{M}) = \frac{1}{4}\sum_{j=1}^{4} \|\mathsf{v}_i^j - \mathsf{v}^j\|^2, \tag{5.12}$$

where $\{\mathsf{v}_i^1, \mathsf{v}_i^2, \mathsf{v}_i^3, \mathsf{v}_i^4\}$ and $\{\mathsf{v}^1, \mathsf{v}^2, \mathsf{v}^3, \mathsf{v}^4\}$ represent the four corresponding vertices of parallelograms $\mathsf{M}_i\mathsf{M}_{i-1}^G$ and $\mathsf{M}\mathsf{M}_{i-1}^G$ respectively.

Compared to the overlap rate based loss function (5.9), the average vertex distance based loss function gives a more reliable loss measurement between two transformations: M_i and M. The spatial order of the four corresponding vertices is taken into account and there is no more obvious rotation ambiguity, though it is still in the image space and not handled principally.

- **Geodesic Distance based Loss Function**

In order to obtain a loss function which correctly measures the difference between two affine transformations, we propose a geodesic distance based function:

$$\mathcal{L}_g(\mathsf{M}_i, \mathsf{M}) = 1 - \exp(-\tau_g \rho_g^2(\mathsf{M}_i, \mathsf{M})), \tag{5.13}$$

where τ_g is a constant and $\rho_g(\mathsf{M}_i, \mathsf{M})$ is the geodesic distance of two affine transformations. For loss function (5.13), any transformation $\mathsf{M} \neq \mathsf{M}_i$ can thus be correctly assigned with a mathematically well-defined loss. Here, the geodesic distance distance $\rho_g(\mathsf{M}_i, \mathsf{M})$ is $\|\log(\mathsf{M}_i^{-1}\mathsf{M})\|$. Following Rossmann [2002], we use a first order approximation:

$$\rho_g(\mathsf{M}_i, \mathsf{M}) \approx \|\log(\mathsf{M}) - \log(\mathsf{M}_i)\|. \tag{5.14}$$

To visually demonstrate the differences among the overlap rate based, average vertex distance based and geodesic distance based loss functions, we apply a pure rotation transformation on a unit square as shown on the left of Figure 5.3. The corresponding loss measurements are calculated then we illustrate them on the right of Figure 5.3. It clearly shows that the principled geodesic distance is linear to the rotation angle while other two measurements are not.

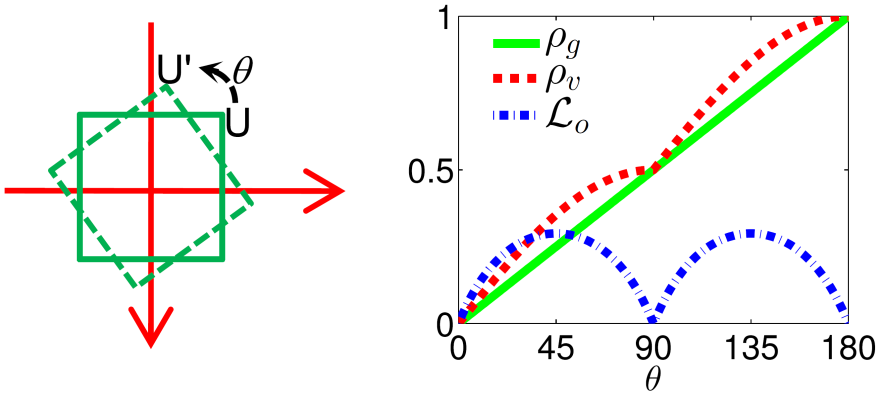


Figure 5.3: The overlap rate based, average vertex distance based and geodesic distance based loss measurements under pure rotation transformation.

5.3 Tracking Procedure

We summarize the basic tracking steps of the proposed algorithm below:

Given the bounding box of object in the first frame as M_0^G.
Set the training example pair set as $\mathcal{S}_0 = \{\emptyset\}$.
For each frame t in the sequence, do the following steps.

1. Add the current example pair into the training set as $\mathcal{S}_t = \{(x_1(\mathsf{M}_0^G), \mathsf{I}), \ldots, (x_t(\mathsf{M}_{t-1}^G), \mathsf{M}_t)\}$.
2. Learn discriminant function F by optimizing (5.8) with training set $\mathcal{S}_t$.
3. Estimate the location of object in frame $t+1$: $\mathsf{M}_{t+1} = \arg\max_{\mathsf{M} \in \mathcal{M}} F(x_{t+1}(\mathsf{M}_t^G), \mathsf{M})$.

During tracking, two major problems need to be carefully considered. Training stage: minimization of the convex learning function (5.8). As the training set $\mathcal{S}_t$ is incrementally obtained, re-optimizing function (5.8) independently every time after obtaining a new sample pair will be time consuming. In other words, training with all possible motion hypotheses in the affine motion manifold $\mathcal{M}$ would make a brute-force optimization intractable.

Testing stage: optimizing the objective function (5.5). As $\mathcal{M}$ denotes a matrix Lie group the set of all affine transformations, which is 6 degrees of freedom, an efficient searching approach over $\mathcal{M}$ needs to be employed in order to achieve a practical processing speed. In the following section, we explain how to optimize (5.8) given a set of example pairs $\mathcal{S}_t$.

5.3.1 Minimizing the Objective Function

The approach proposed in Bordes et al. [2008] is used for optimizing the function (5.8) as suggested in Hare et al. [2011]. Here we provide the version on the manifold following the equivalent dual form problem:

$$\begin{aligned}\max_{\beta} \quad & -\sum_{i,\mathsf{M}} \mathcal{L}(\mathsf{M}_i, \mathsf{M})\beta_i^{\mathsf{M}} \\ & -\frac{1}{2}\sum_{i,\mathsf{M},j,\tilde{\mathsf{M}}} \beta_i^{\mathsf{M}}\beta_j^{\tilde{\mathsf{M}}}\langle\phi(x_i(\mathsf{M}_{i-1}^G),\mathsf{M}),\phi(x_j(\mathsf{M}_{j-1}^G),\tilde{\mathsf{M}})\rangle, \\ \text{s.t.} \quad & \beta_i^{\mathsf{M}} \le c\Delta(\mathsf{M}_i,\mathsf{M}), \forall i, \forall \mathsf{M} \\ & \sum_{\mathsf{M}} \beta_i^{\mathsf{M}} = 0, \forall i\end{aligned} \tag{5.15}$$

where $\Delta(\mathsf{M}_i, \mathsf{M}) = 1$ if $\mathsf{M}_i = \mathsf{M}$ and 0 otherwise, c is the same as in (5.8). The discriminant function $F(x_i(\mathsf{M}_{i-1}^G), \mathsf{M})$ can also be represented in the dual form as

$$\sum_{j,\tilde{\mathsf{M}}} \beta_j^{\tilde{\mathsf{M}}}\langle\phi(x_i(\mathsf{M}_{i-1}^G),\mathsf{M}),\phi(x_j(\mathsf{M}_{j-1}^G),\tilde{\mathsf{M}})\rangle. \tag{5.16}$$

We refer those pairs $(x_i(\mathsf{M}_{i-1}^G), \mathsf{M})$ for which $\beta_i^{\mathsf{M}} \neq 0$ as support vectors as Bordes et al. [2007, 2008]. Only the support vector $(x_i(\mathsf{M}_{i-1}^G), \mathsf{M}_i)$ will have $\beta_i^{\mathsf{M}} > 0$, while any other support vector will have $\beta_i^{\mathsf{M}} < 0$, $\mathsf{M} \neq \mathsf{M}_i$. They are referred as positive and negative support vectors respectively. Function (5.16) is computed using a joint kernel function as:

$$\begin{aligned}&\langle\phi(x_i(\mathsf{M}_{i-1}^G),\mathsf{M}),\phi(x_j(\mathsf{M}_{j-1}^G),\tilde{\mathsf{M}})\rangle \\ &= K(\hat{x}_i(\mathsf{M}\mathsf{M}_{i-1}^G), \hat{x}_j(\tilde{\mathsf{M}}\mathsf{M}_{j-1}^G)).\end{aligned} \tag{5.17}$$

Here, $\hat{x}_i(\mathsf{M}\mathsf{M}_{i-1}^G)$ is the feature vector (HOG, Haar) extracted from the parallelogram $\mathsf{M}\mathsf{M}_{i-1}^G$ in frame i (Figure 5.1). The kernel K can be any kernel such as Gaussian.

Optimizing function (5.15) is then composed of mainly two basic operations: select a triplet $\{i, \mathsf{M}_+, \mathsf{M}_-\}$ and optimize its corresponding coefficients $\beta_i^{\mathsf{M}_+}$ and $\beta_i^{\mathsf{M}_-}$ using an SMO step Platt [1999]. The parameter i in the triplet is randomly selected, and for a given i, M_+ and M_- are chosen with respect to the gradient of the function (5.15):

$$\partial_i(\mathsf{M}) = -\mathcal{L}(\mathsf{M}_i, \mathsf{M}) - F(x_i(\mathsf{M}_{i-1}^G), \mathsf{M}). \tag{5.18}$$

For example, M_- can be chosen by $\mathsf{M}_- = \arg\min_{\mathsf{M}\in\mathcal{M}}\partial_i(\mathsf{M})$, i.e., finding the most important negative sample in frame i: the one has high compatibility value of F while possesses a big difference with M_i.

5.3.2 Online Update

As we mentioned at the beginning of Section 5.3, training example set $\mathcal{S}_t$ is incrementally obtained and re-optimizing function (5.15) for every frame will be time-consuming. Thus, we propose the following approach.

Given support vectors $\mathcal{V} = \{(x_i(M^G_{i-1}), \mathsf{M}) \mid \beta^{\mathsf{M}}_i \neq 0, i = 1, \ldots, t-1\}$ and a new example pair $(x_t(\mathsf{M}^G_{t-1}), \mathsf{M}_t)$ at frame t.

1. Select a triplet $\{t, \mathsf{M}_+, \mathsf{M}_-\}$, where $\mathsf{M}_+ = \mathsf{M}_t$ and $\mathsf{M}_- = \arg\min_{\mathsf{M}\in\mathcal{M}} \partial_t(\mathsf{M})$.
2. Optimize the triplet $\{t, \mathsf{M}_+, \mathsf{M}_-\}$ obtained in step 1 using SMO, if the resulted coefficients $\beta^{\mathsf{M}_+}_t$ and $\beta^{\mathsf{M}_-}_t$ are not zero, add them into $\mathcal{V}$.
3. Select the triplet $\{i, \mathsf{M}_+, \mathsf{M}_-\}$ (for random i): $\mathsf{M}_+ = \arg\max_{\mathsf{M}\in\mathcal{M}_i} \partial_i(\mathsf{M})$ and $\mathsf{M}_- = \arg\min_{\mathsf{M}\in\mathcal{M}_i} \partial_i(\mathsf{M})$, where $\mathcal{M}_i = \{\mathsf{M} \in \mathcal{M} | \beta^{\mathsf{M}}_i \neq 0\}$.
4. Optimize the triplet $\{i, \mathsf{M}_+, \mathsf{M}_-\}$ obtained in step 3 using SMO, if $\beta^{\mathsf{M}_+}_t$ or $\beta^{\mathsf{M}_-}_t$ is zero, remove them from $\mathcal{V}$, else update the corresponding coefficient in $\mathcal{V}$.
5. Repeat step 3 to step 4 N_o times.
6. Repeat step 1 to step 5 N_a times.

The main difference with the method proposed in Hare et al. [2011] is that we do not revisit previous frames for adding new samples as negative support vectors any more but only in the current frame (corresponds to step 1). This strategy is widely used in sparsity-based tracking methods Xing et al. [2013]; Zhong et al. [2012]; Ross et al. [2008], as they argue that recent observations will be more indicative. Note that step 3 will not add new support vectors, but can remove existing support vectors depending on the result of the SMO optimization Platt [1999]. In our paper, $N_o = 10$ and $N_a = 10$.

5.3.3 Efficient Tracking

Another issue we mentioned at the beginning of Section 5.3 is how to efficiently optimize the objective function (5.5) in the testing stage. Similar difficulty exits when to add new support vector in Section 5.3.2 solving $\arg\min_{\mathsf{M}\in\mathcal{M}} \partial_t(\mathsf{M})$.

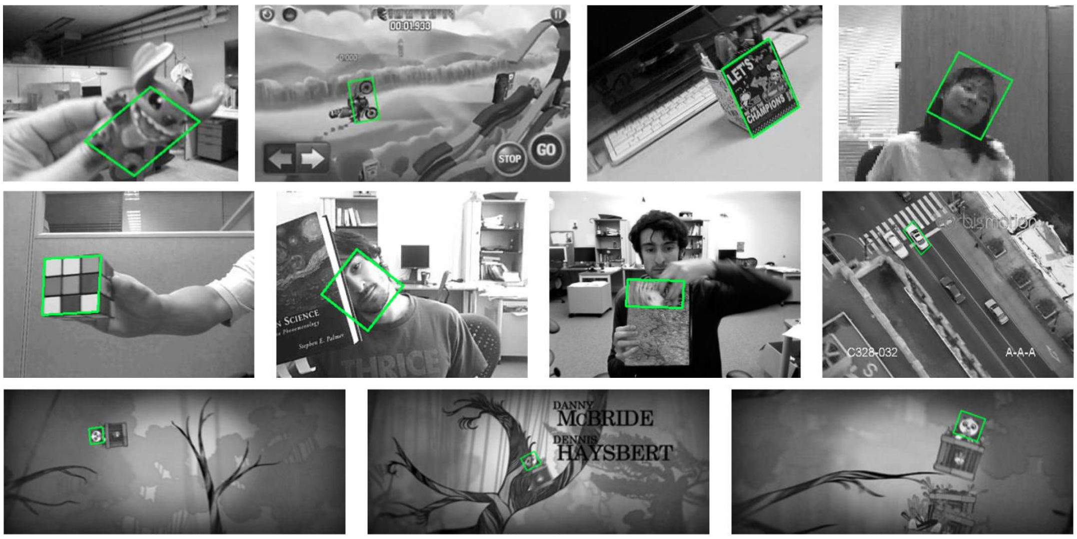


Figure 5.4: Sample frames from our newly-constructed affine tracking dataset. From left to right, top to bottom: toy, bike, vase, girl, panda, panda, panda, cube, faceO, cliff and car.

Two sampling strategies are proposed: uniform sampling and particle filter sampling. The former uniformly samples points on the manifold $\mathcal{M}$. For particle filter sampling, advanced methods such as Kwon et al. [2009] can be employed. Here, we treat 6 affine parameters independently and model them with 6 Gaussian distributions as in Zhong et al. [2012]; Ross et al. [2008]; Xing et al. [2013].

The number of support vectors has to be limited, as the computational and memory costs increase with the number of support vectors and the number of training examples can be huge in the tracking procedure. Employing the method proposed in [Wang et al., 2010], we remove support vector $(x_r(\mathsf{M}^G_{r-1}), \mathsf{M})$ that results in the smallest change to the coefficient vector $\boldsymbol{w}$, as measured by $\|\Delta \boldsymbol{w}\|^2$. when the budget is exceeded.

5.4 Experiments

5.4.1 Datasets and Evaluation Method

Since there are no existing annotated datasets specified for affine tracking, we collect nine image sequences and manually annotate them frame by frame. Most sequences in the proposed dataset are subjected to some challenging aspects such as motion blur and occlusion. This dataset will be publicly available and we summarize those aspects as follows: • toy: out-of-plane rotation. • bike: out-of-plane rotation, background clutters. • girl, faceO: occlusion. • panda: motion blur, occlusion. • cliff: background clutters, motion blur. • vase, cube, car: no challenging aspects involved.

Additionally, sequences from VOT2014 Challenge Kristan et al. [2014] are employed to evaluate our proposed method. They provided per-frame labeled rotated bounding box as ground truth and are not specifically designed for affine tracking purpose. We evaluate the proposed tracker on this dataset to demonstrate its performance for general tracking task.

To evaluate, traditional overlap rate (5.10) and center location error are used. We also propose the average vertex distance error ρ_v (5.12) and geodesic distance error ρ_g (5.14) to measure the error of the affine tracking results for an objective evaluation. Computational speed is evaluated as frames per second (FPS).

5.4.2 Implementation

Lie-Struck is implemented in C++ and experiments are carried out on an Intel Core i7-2600 3.40GHz PC with 4 GB memory. The joint kernel function (5.17) is implemented using the same Harr feature and Gaussian kernel as Hare et al. [2011]

$$K(\hat{x}_i(\mathsf{MM}^G_{i-1}), \hat{x}_j(\widetilde{\mathsf{M}}\mathsf{M}^G_{j-1})) = \exp(-\sigma\|\hat{x}_i(\mathsf{MM}^G_{i-1}) - \hat{x}_j(\widetilde{\mathsf{M}}\mathsf{M}^G_{j-1})\|^2),$$

$$\exp(-\sigma\|\hat{x}_i(\mathsf{MM}^G_{i-1}) - \hat{x}_j(\widetilde{\mathsf{M}}\mathsf{M}^G_{j-1})\|^2),$$

where σ is 0.2. The budget size of support vectors is set to 100. For the geodesic distance based loss function (5.13), $\tau_g = 2$ and the matrix logarithm (5.3) is implemented using the Eigen C++ template library Guennebaud et al..

The same affine motion parameters are used for all tests in our experimental evaluation. For uniform sampling, we use 81 samples for the two translation parameters as same as Hare et al. [2011], 5 samples with 4 degrees and 0.03 interval for the rotation angle and scaling parameters, 3 samples with 4 degrees and 0.03 interval for the skew angle and aspect ratio parameters. For particle filter sampling, the Gaussian parameters are $\{8, 8, 5, 0.04, 5, 0.04\}$ respectively. The proposed approach using uniform sampling is denoted as Lie-Struck$_u$, while the one uses 2000 particles as Lie-Struck$_{20}$, 1000 particles as Lie-Struck$_{10}$, and 600 particles as Lie-Struck$_6$.

Lie-Struck are compared against 2 state-of-the-art trackers that are able to perform affine tracking : SCM Zhong et al. [2012] and Lie-Tracker Tuzel et al. [2006]. For SCM tracker, particle filter number is set to 1000 and same parameters are used. For Lie-Tracker, default setting is used. T demonstrate the effectiveness of the geodesic distance based loss function (5.13), we also implement the approaches using the overlap rate based loss function (5.9) and the average vertex distance based ($\tau_v = 5$) loss function (5.11), which are denoted as O-Struck and V-Struck respectively. Struck Hare et al. [2011] is evaluated as well.

	toy	bike	vase	girl	panda	cube	faceO	cliff	car	**MeanV**	**MeanC**	**MeanO**	**MeanG**	**FPS**
Lie-Struck$_u$	**8.6**	**12.5**	9.7	**7.6**	**3.4**	6.5	7.7	**5.7**	5.5	**7.5**	**5.3**	**0.79**	**37.2**	0.4
O-Struck$_u$	9.1	81.4	9.8	28.3	85.3	11.1	9.7	6.6	5.7	27.4	21.3	0.68	55.1	0.4
V-Struck$_u$	9.6	74.7	9.8	15.4	56.1	5.1	8.9	6.1	6.1	20.4	14.9	0.74	61.2	0.4
Struck	23.1	95.5	47.1	19.3	104.3	46.3	29.9	67.9	31.6	51.7	36.5	0.39	91.5	2.7
SCM	32.7	35.8	7.4	25.3	110.1	**4.9**	161.3	65.4	66.5	56.6	47.0	0.51	95.8	0.2
Lie-Tracker	20.0	90.4	**4.2**	17.6	266.2	8.9	15.1	86.8	**4.5**	57.1	25.5	0.62	77.8	**4.3**
Lie-Struck$_{20}$	8.7	39.0	10.4	20.6	15.9	5.1	**6.2**	35.4	5.5	16.4	10.8	0.73	50.3	1.4
Lie-Struck$_{10}$	8.8	57.4	10.6	37.5	48.2	7.8	7.1	34.6	5.7	24.1	17.4	0.69	51.2	1.7
Lie-Struck$_6$	10.7	55.1	12.0	26.1	89.0	6.3	8.3	40.8	7.1	28.4	21.3	0.67	54.7	1.9

Table 5.1: The average vertex distance error is reported for each sequence on the proposed affine tracking dataset. Processing speed (FPS) and means of center location error (MeanC), overlap rate (MeanO) and geodesic distance error (MeanG) over all sequences are reported as well. Best results are marked in bold.

	ball	bicy.	car	david	drunk	jog.	moto.	polar.	skat.	sphe.	suns.	surf.	tun.	**MeanV**	**MeanO**
Lie-Struck$_u$	38.8	51.6	27.8	**25.7**	44.6	**9.6**	**107.3**	44.6	93.0	21.1	7.3	6.9	13.9	**37.9**	**0.52**
O-Struck$_u$	42.1	28.5	29.1	43.0	108.1	60.2	316.7	71.6	101.5	26.9	6.7	25.3	21.9	67.8	0.43
V-Struck$_u$	40.9	51.7	28.7	57.3	60.2	105.1	479.5	43.7	117.1	24.7	6.2	13.1	21.3	80.7	0.44
Struck	**32.8**	11.6	43.4	57.2	69.1	11.1	139.1	15.6	76.1	**19.2**	**4.2**	3.7	22.3	38.9	0.50
SCM	103.3	9.9	**19.4**	26.2	72.4	153.0	203.0	17.4	79.5	26.5	6.6	3.7	43.1	58.8	0.47
Lie-Tracker	138.7	92.8	94.4	181.9	**34.9**	10.0	377.9	**8.8**	**46.4**	664.4	88.5	4.6	80.3	140.2	0.38
Lie-Struck$_{20}$	102.0	**8.3**	26.4	27.2	61.0	122.3	112.3	35.8	89.3	36.7	7.0	**3.6**	**10.0**	49.4	0.49
Lie-Struck$_{10}$	126.3	8.5	26.8	45.6	62.8	125.4	114.8	36.2	91.7	40.9	7.9	3.8	12.2	54.1	0.48
Lie-Struck$_6$	207.1	8.6	30.8	69.2	63.6	149.3	153.2	42.1	93.6	60.2	7.9	5.3	13.0	69.5	0.46

Table 5.2: The average vertex distance errors are reported for sequences from VOT2014 Challenge dataset (more general tracking scenarios). The mean of overlap rate (MeanO) over all sequences is reported as well. Best results are marked in bold.

5.4.3 Performance Evaluation

The results of evaluation are summarized in Table 5.1 for sequences on the proposed affine tracking dataset and Table 5.2 for sequences from VOT2014 Challenge dataset. Every sequence is repeated 10 times for those stochastic methods, especially ones using particle filter sampling, then the average result is reported.

Affine Tracking: Comparing the performance between Lie-Struck$_u$ and Struck on the affine tracking dataset, it convincingly demonstrated that the combined motion and appearance model structures greatly improve the tracking accuracy. This can be further validated using Figure 5.5, in which we visualize the support vectors maintained in the respective trackers. Those positive support vectors maintained in Lie-Struck$_u$ have consistent appearance, while Struck treats every rotated object region as a new positive support vector since it is not aware of rotations. This difference is significant and it causes a big performance gap as the novelly incorporated motion model simplifies the classification task for structure SVM.

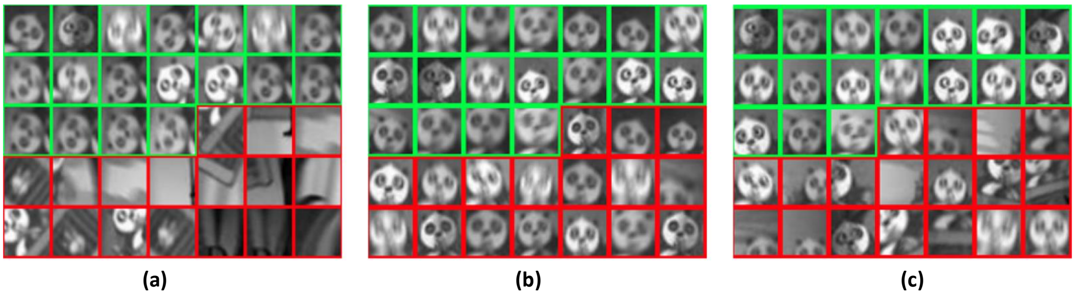


Figure 5.5: Visualization of the support vectors maintained by trackers at the frame $t = 125$ in sequence “panda". (a) Struck; (b) O-Struck$_u$; (c) Lie-Struck$_u$. Positive and negative support vectors have green and red borders respectively. Notice that Struck treats every rotated object region as a positive support vector, while Lie-Struck$_u$ generates consistent positive support vectors and distinctive negative support vectors, which indicates an simplified classification boundary.

The performance gaps among Lie-Struck$_u$, O-Struck$_u$ and V-Struck$_u$, especially for sequences such as “bike", “girl" and “panda", are huge because Lie-Struck$_u$ keeps tracking the objects while O-Struck$_u$ and V-Struck$_u$ lost the objects in the middle of sequences. The differences among them can also be demonstrated using Figure 5.5, as the negative support vectors in O-Struck$_u$ are not as distinctive from the positive support vectors as Lie-Struck$_u$, which indicates a better SVM classification boundary and more robustness from drifting away from the object region.

The incorporated particle filter mechanism has resulted comparable performance as the uniform sampling one, while the processing speed is improved. It even produced better results (“cube", “faceO") as the Gaussian distribution well captures the smooth motion of objects. Overall, it clearly demonstrated that the proposed methods outperform existing state-of-the-arts: SCM and Lie-Tracker, with comparable computational speeds.

General Tracking: Based on the experimental results (Table 5.2) for sequences from VOT2014 Challenge dataset, we can see that the proposed methods achieve competitive performance, though this dataset is not specifically designed for affine tracking purpose. Especially on sequences such as “jog." and “moto.", where affine transformations present, our methods showed clear advantages over the state-of-the-arts. In all, the resulted performance differences among Lie-Struck$_u$, O-Struck$_u$ and V-Struck$_u$ demonstrated that the principled geodesic distance based loss function outperforms other image space based loss functions in both affine and general tracking scenarios.

5.5 Summary

We proposed a novel affine motion tracking-by-detection method that took advantage of the intrinsic topology of motion manifold. We incorporated the Lie group structure into a structured SVM classifier to directly estimate 2D affine transformation using an appearance-based prediction function. We demonstrated that these combined motion and appearance model structures significantly improved the tracking performance on a newly-constructed affine tracking dataset and a challenging general tracking dataset, while an incorporated particle filter mechanism kept the processing speed fast.

Chapter 6

Model-Free Multiple Object Tracking with Shared Proposals

After presenting three studies for single object model-free online visual tracking from the perspectives of background contextual clusters, global tracking with instance-specific proposals, and Lie geometry based affine motion tracking in the previous three chapters, we introduce a novel framework for multiple object model-free tracking in this chapter.

6.1 Introduction

Single object tracking attained considerable success thanks to the advances in "tracking-by-detection" that demonstrated improved performance on standard benchmarks Yoon et al. [2016]; Bae and Yoon [2014]; Milan et al. [2016]. Compared to single-object tracking counterpart, multiple-object tracking is a more challenging task due to the frequent occlusions between the target objects and typical similarities in their motion patterns as well as visual appearances. Moreover, the background scenes also tend to be more cluttered due to the presence of other moving objects Patino et al. [2016]; Milan et al. [2016].

In model-based tracking-by-detection of multiple objects, an offline trained category-specific object detector, e.g., DPM Felzenszwalb et al. [2010] or R-CNN Ren et al. [2015], is applied at every frame to generate high quality object hypotheses, and then graph-based methods such as max-flow Leal-Taixe et al. [2014]; Milan et al. [2014] are used to solve the subsequent multi-frame multi-target association problem. These multiple object tracking methods, however, depend heavily on the performance of category-specific object detectors, which often miss objects or generate false positives that are induced by the discrepancy between the training dataset and the test conditions of individual deployments Torralba and Efros [2011].

Being constrained to a specific object class also limits the applicability of the tracker to a certain setting, for example, multiple vehicle tracking in traffic scenes. In practice, however, various applications demand tracking of different types of objects undergoing complex motions as shown in Fig. 6.1.

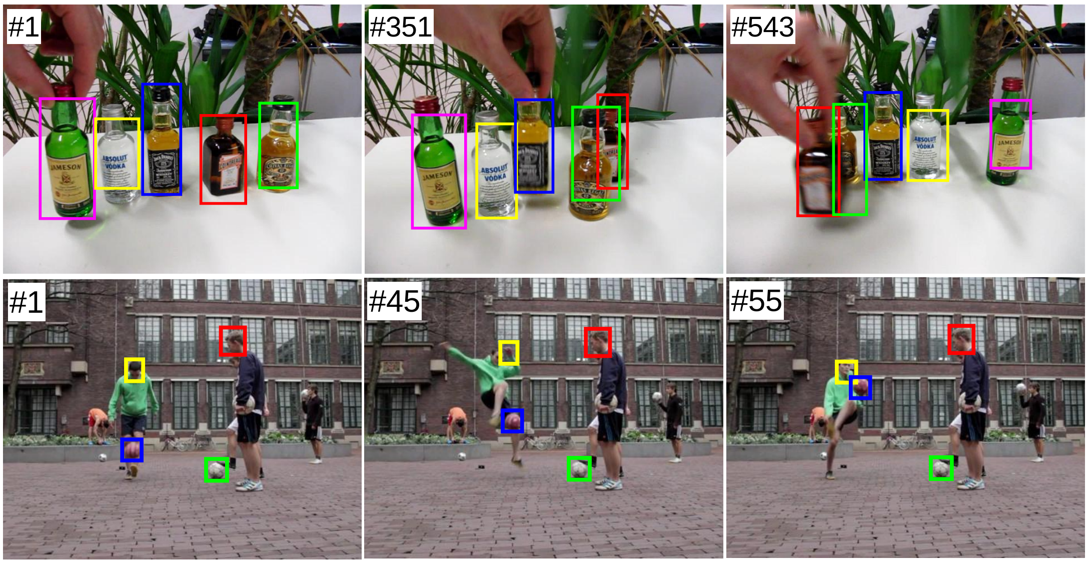


Figure 6.1: Results obtained using our model-free multiple object tracking method. Bounding boxes of the same color denote the same tracked object. After initialization, our method tracks each object without any pretrained models.

On the other end of the spectrum, “model-free" approaches aim to track arbitrary (category-independent) objects Wu et al. [2013]; Smeulders et al. [2014]; Kristan et al. [2015]; Felsberg et al. [2015]; Wu et al. [2015]. They initiate a single bounding box on the target in the first frame and then employ either a generative Ross et al. [2008]; Mei and Ling [2011]; Li et al.; Jia et al. [2012] or a discriminative Zhang et al. [2014a]; Henriques et al. [2015]; Babenko et al. [2009] strategy to train their object models online. These methods are successfully applied for single-object tracking. However, extending “model-free” methods to multiple tracking task is not a straightforward problem due to two major reasons:

- Computational efficiency – Since each tracker searches around the previous location to localize the object, the time cost is proportional to the number of objects.
- Interactions – Objects contact or occlude each other. They often have similar appearances. Blindly and independently applying single-object trackers multiple times for different targets leads to ambiguities and tracking failures.

To overcome the above challenges, we propose a model-free multiple object tracking framework based on generic object proposals. We take advantage of the proposals in both online training and testing of the tracker.

In the testing stage, a small set of object candidates are generated based on simple objectness cues first. Notice, this set is shared by *all* trackers and it provides two benefits: i) a significant reduction of the number of candidates, and ii) tracking accuracy improvement since many false positives can be eliminated at this stage. The proposals are then assigned to trackers based on the classifier confidence and temporal smoothness measures. The number of proposals can be as many as hundreds while the number of objects might be only a few. We use the Hungarian algorithm Munkres [1957]; Babenko et al. [2009] with appropriate modifications to reduce the computational cost during the data association stage. Other association methods Yoon et al. [2016]; Bae and Yoon [2014]; Milan et al. [2016] can also be used, yet we observe that the computationally efficient Hungarian method works favorably when we build discriminative classifiers based on the generated proposals.

In the training stage, we collect the proposals as *hard* negative samples instead of manual selecting around positive samples. These proposals are expected to contain the other targets and object-like background clutter. Mining explicitly for such hard negative samples and employing hard negatives in the training of individual object models significantly improves the discriminative power of the object models. We update the classifiers at certain time intervals in an online fashion to compensate for object appearances changes over time and incorporate *new* distractors. A few local candidates sampled around the previous object locations are included in the negative set to further improve tracking precision.

We focus on a challenging scenario of multi-object tracking where each object may move **very fast** in an **irregular** fashion. To our knowledge, this challenge has not been widely researched and there are only a few benchmarks (e.g. PETS Patino et al. [2016]) available for investigation. Therefore, we collected an extensive set of challenging video sequences from various sources and manually labeled the ground-truth object locations for a comprehensive experimental evaluation.

Our method is conceptually simple, easy to implement, and most importantly, achieves superior performance in comparison to several state-of-the-art techniques in terms of both tracking accuracy metrics and computational efficiency.

6.2 Related Work

Here we give a brief review to previous methods for multi-object tracking that are most related to this paper. For more comprehensive literature surveys the reader is referred to Wu et al. [2013]; Smeulders et al. [2014]; Kristan et al. [2015]; Milan et al. [2016].

Multiple Target Tracking

As aforementioned in Section 6.1, most multiple object tracking methods focus on the data association problem, assuming sufficiently long and accurate tracklets are provided by using advanced object detectors Milan et al. [2016]. For example, Dicle et al. [2013] considers motion dynamics as the major cue to distinguish different targets with similar appearance. It solves the problem as generalized linear assignment (GLA) of tracklets, which are incrementally joined forming longer trajectories based on their similar dynamics. The work in Yoon et al. [2016] observes that motion cues are not always reliable for this task, due to for example abrupt camera movement. As a remedy a structured motion constraint between objects is therefore proposed to address this issue. Bae and Yoon [2014] proposes an online discriminative appearance learning approach to handle similar appearances of different objects in tracklet association. This method is similar to our method to be described in this paper; however, in their work those negative training samples are only collected around the tracklets, while ours pivots on the hard negative ones.

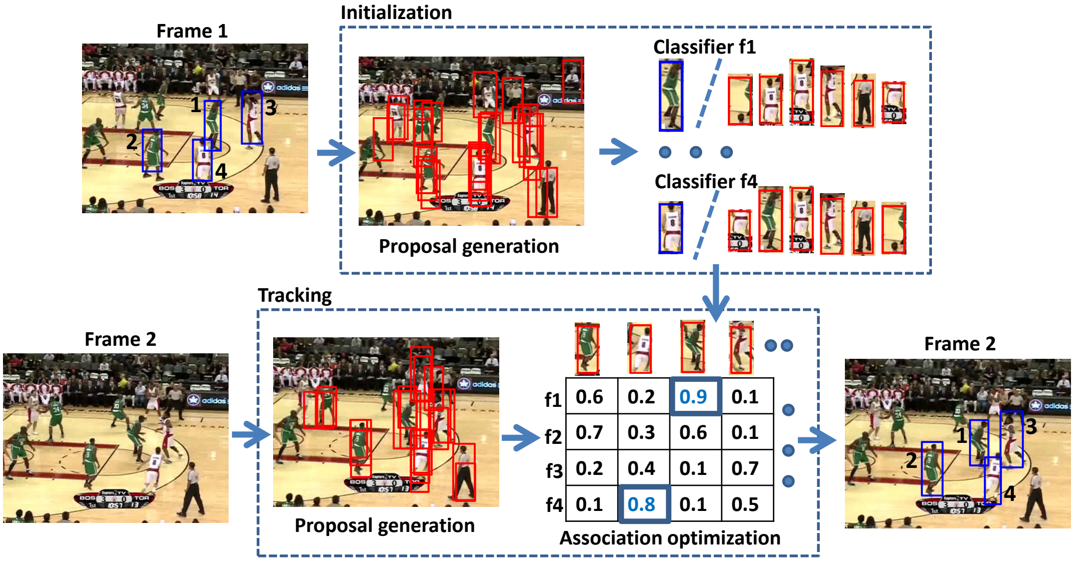
Model-Free Object Tracking

Model-free object tracking algorithms are proposed primarily for solving single object tracking applications Wu et al. [2013]; Smeulders et al. [2014]. The work in Duan et al. [2012] tries to improve the identification of a single target object by also tracking stable features in the background, thereby improving the location prior for the target object. Yang et al. [2009] proposes a context-aware tracker which considers a set of auxiliary objects as the contextual information for the foreground. These auxiliary objects must satisfy conditions such as having persistent co-occurrence with the foreground and consistent motion correlation.

The tracker in Zhang and van der Maaten [2014] is probably the most closely related work to ours. However, they assume spatial relationship between objects. For instance, nearby objects tend to move along the same direction. The appearance models of all the objects and the structural constraints between these objects are jointly trained in an online structured support vector machine framework. Our framework has no such an assumption and can track arbitrarily moving objects.

Object Proposals for Visual Tracking

As reported in Hosang et al. [2014]; Zitnick and Dollár [2014], using object proposal improves the object detection benchmark along with the convolutional neural nets. Since, a subset of high-quality candidates are used for detection, object proposal methods boost not only the speed but also the accuracy by reducing false positives. The top performing detection methods Girshick et al. [2014]; Wang et al. [2013] for PASCAL VOC Everingham et al. [2015] use detection proposals. Among the existed proposal methods, the EdgeBox method Zitnick and Dollár [2014] proposes object candidates based on the observation that the number of contours wholly enclosed by a bounding box is an indicator of the likelihood of the box containing an object. It is designed as a fast algorithm to balance between speed and proposal recall, com-



f1	0.6	0.2	0.9	0.1
f2	0.7	0.3	0.6	0.1
f3	0.2	0.4	0.1	0.7
f4	0.1	0.8	0.1	0.5

Figure 6.2: The structure of our model-free multiple object tracker. The only input is the bounding boxes at the first frame. Our method then initializes multiple classifiers for each object taking advantage of a small set of object proposals generated from the frame. In the next frame, these classifiers are used to assign confidence scores for the candidate proposals. The final trajectories are obtained after solving the optimal association problem. Note that, we also apply the proposals to online update the classifiers to make them more robust to distractors.

paring to BING Cheng et al. [2014] and region proposal network (RPN) Ren et al. [2015].

Many work exist adopting the object proposals for the model-free single object tracking. A straightforward strategy based on linear combination of the original tracking confidence and an adaptive objectness score obtained by BING is employed in Liang et al. [2016]. In Huang et al. [2015], a detection proposal scheme is applied as a post-processing step, mainly to improve the tracker's adaptability to scale and aspect ratio changes. EBT Zhu et al. [2016a] employs the EdgeBoxes method to globally track the object, disregarding potentially fast or drastic object motion. In contrast, our work utilizes the shared proposals for efficient handling of multiple trackers. Ošep et al. [2016] deals with generic object tracking for street scenes by generating multi-scale candidates from the point-density map. Tracking is performed using the pseudo-Boolean optimization (QPBO) method. In comparison, our method is applied to more generic object categories rather than street scenes. Besides, our object models is built taking advantage of the proposals, while Ošep et al. [2016] adopts a generative model using RGB feature distance.

6.3 Multiple Object Tracking with Proposals

As illustrated in Figure 6.2, our framework starts with a few manually initialized bounding boxes on the target objects to be tracked in the first frame of the video. This is similar to the single object online visual tracking task Wu et al. [2013]; Smeulders et al. [2014]; Kristan et al. [2015]. Given these initial bounding boxes, denoted as $\{B^i_{t=1}\}$, $i = 1, \ldots, N_o$, where N_o is the total number of objects, the multiple object tracking problem then aims to find the locations and bounding boxes of the multiple objects in the remainder of the video while maintaining the correct identity of each object.

Following the tracking-by-detection framework, we train the object appearance models for each object. We have an option to use either the generative or discriminative learning strategy. Recent literature on object tracking resort to the discriminative learning to maximize the inter-class separability between the object and background regions and report improved performance as the discriminative learning is more robust to distractions from the background. This property is especially important in multiple object tracking Bae and Yoon [2014]; Possegger et al. [2015] where the objects exhibit similar appearance and interact frequently, as depicted in Fig-6.2.

As explained in Section 6.1, we do not independently initialize N_o classifiers by collecting locally and densely sampled negative patches as training samples, a scheme that conventional online single object trackers typically employ.

Instead, we incorporate object proposals Hosang et al. [2014]; Zitnick and Dollár [2014] to generate a small number of shared object candidates. Notice that, we are not simply using the original object proposals either, since the sizes of the objects usually change during the tracking. We impose the proposal bounding box sizes to be within a certain range of the object sizes. More details about this can be found in Section 6.3.1.

Suppose the object proposal bounding boxes are $\{\hat{B}^j_{t=1}\}$, $j = 1, \ldots, N_p^{t=1}$, where $N_p^{t=1}$ is the total number of proposals in the first frame. We train the classifiers with the corresponding positive samples $B^i_{t=1}$ that are not in the common negative set $\{\hat{B}^j_{t=1}\}$. The initialized classifiers are denoted as

$$f^i_{t=1}(B), \;\; i = 1, \ldots, N_o, \tag{6.1}$$

We additionally select a small set of local candidates sampled around the object to further improve the discriminative power, thus the localization precision, of the classifier as Zhu et al. [2016a].

In the consecutive frame, we generate a set of proposals $\{\hat{B}^j_{t=2}\}, j = 1, \ldots, N^{t=2}_p$, to be shared and tested by all classifiers $\{f^i_{t=1}(B)\}$. Considering the temporal smoothness between the object $B^i_{t=1}$ and the proposal $\hat{B}^j_{t=2}$, (spatial distance between them), we build an association matrix that will be efficiently optimized by a modified Hungarian algorithm Munkres [1957]; Babenko et al. [2009]. The new object locations are then determined as the optimal solution of this association problem. More details about it can be found in Section 6.3.2.

To adapt the object appearance changes as well as to increase the discriminative power of the classifiers against newly appeared distractors, we incrementally update the classifiers by treating the estimated bounding box in current frame as the positive sample and object proposals as the negative samples as we did in the first frame. More information is in Section 6.3.3.

6.3.1 Object Proposal Generation

As mentioned in Section 6.2, various object proposal algorithms exist. We employ EdgeBox Zitnick and Dollár [2014] as it strikes a good balance between recall and speed. In our experimental analysis, we also test other proposal methods such as BING Cheng et al. [2014] and region proposal network (RPN) Ren et al. [2015].

Two important factors should be noticed here. The first one is the about the sizes of the generated object proposals, termed as size adaption ratio and denoted as $\alpha \in [0, 1]$. We allow the size of the proposals maximally differ the target with a bounding box intersection-over-union (IoU) Everingham et al. [2015] of ratio α. To be specific, we consider $\hat{B}^j_t$ only when

$$\max_i(\mathrm{IoU}(\hat{B}^j_t, B^i_{t-1},)) > \alpha, \quad i \in [1, \ldots, N_o] \tag{6.2}$$

This setting significantly reduces the number of proposals while permitting the object window to adapt the target size changes at the same time. We use $\alpha = 0.8$ and test other values in the experimental part.

The other factor is the maximal number of object proposals generated. EdgeBox does not output a fixed number of proposals. The number of proposals could be any depending on the threshold of the "objectness" score (set as 0.01 as recommended). An appropriate maximal number of proposals needs to be used as its lower values may result in missing the object window in the proposal set while its higher values would cause an extensive number of distractors. We set this number at 500 for all experiments. We also run test other values of the maximal number of proposals in Section 6.4.2.

Similar to Zhu et al. [2016a], we generate a fixed number of bounding boxes, $\{\tilde{B}_t^k\}_{t-1}^i$, $k = 1, \ldots, N_s$, by sampling only around the previous object location B_{t-1}^i for each object (as in traditional methods). This set $\{\tilde{B}_t^k\}_{t-1}^i$ is only tested by the corresponding classifier $f_{t-1}^i(B)$ and they are useful to smoothen the trajectory as the object proposal component works independently at each frame, which may result in temporally inconsistent proposals. Thus, a combined set of $\{\hat{B}_t^j\} \cup \{\tilde{B}_t^k\}_{t-1}^i$ is used during the test stage for the classifier $f_{t-1}^i(B)$. However, we only update the classifier when the estimated one comes from the proposal set $\{\hat{B}_t^j\}$ to attain resistance to potential corruptions. We sample $N_s = 80$ patches uniformly within a 30-pixels radius. More details are in Section 6.3.3.

6.3.2 Optimal Target Association

Given N_o targets and $(N_p^t + N_s \times N_o)$ candidates, the target association stage therefore aims to find the optimal non-repetitive N_o candidates for the N_o targets, such that the overall *gain* is maximized. Note that, the candidates $\{\tilde{B}_t^k\}_{t-1}^i$ are only allowed to link with target i, thus we set the *gain values* of linking them to other targets to zero.

The gain value $P(B_t, i)$ of linking a candidate B_t to target i is designed base on both classifier confidence score and temporal smoothness,

$$P(B_t, i) = f_{t-1}^i(B_t) + s(B_t, B_{t-1}^i). \tag{6.3}$$

$s(B_t, B_{t-1}^i)$ is a term representing the temporal smoothness between the previous target bounding box B_{t-1}^i and the candidate box B_t. We use a simple function in this paper: $s(B_t, B_{t-1}^i) = \exp(-\frac{1}{2\sigma^2}\|c(B_t) - c(B_{t-1}^i)\|^2)$, where $c(B_t)$ is the center of bounding box B_t and σ is a value controlling the impact of the temporal smoothness term. We set $\sigma = R_i$, where R_i is half of the diagonal length of the initialized bounding box B_1^i. We also test other values as in Section 6.4.2.

Once the gain values are set, the standard Hungarian algorithm Munkres [1957]; Babenko et al. [2009] can be modified to optimally solve the association problem. As $(N_p^t + N_s \times N_o)$ is usually much larger than N_o (a few hundreds vs. a few), available fast implementation Cao [2008] is too slow to be applied directly. We thus firstly find top N_o candidates for each target i locally and separately. As the global optimal assignment for that target i must be one of them, we then combine those found local candidates into a small matrix in which the optimal solution is exact the same global optimal solution to the original association problem. Notice that, the standard Hungarian algorithm solves the minimization problem, thus a simple modification is required before feeding the small matrix to it.

6.3.3 Online Updating with Proposals

To update the classifier, $f^i_{t-1} \to f^i_t$, we also generate a few local samples, $\{\tilde{B}^k_t\}^i_t$, $k = 1, \ldots, N_s$, around the estimated object location B^i_t. They are helpful to increase the discriminative power of the classifier, as the object proposals alone represent other good "object-like" regions and training with them increases the discriminative power among "objects-like" candidates, while the negative sample space contains a lot more other negative samples, thus more negative samples help. The updating procedure is applied every 5 frames to balance computational time and minimize potential drift.

As mentioned in the last paragraph of Section 6.3.1, we treat the estimated result B^i_t as an indication for model updating. This is to say, when $B^i_t \in \{\tilde{B}^k_t\}^i_{t-1}$, we assume that there is no good object proposal and the current estimation is a compromise for trajectory smoothness, thus skipping the model updating. If $B^i_t \in \{\hat{B}^j_t\}$, then it suggests a good estimation which has both desirable classifier response and high "objectness", then we update the object model $f^i_{t-1}(B)$ immediately.

6.3.4 Proposed Tracker: PMOT

Various object models can be integrated into our framework. We choose a popular structured support vector machine (SSVM) method Hare et al. [2011], as it shows good performance on several benchmarks Wu et al. [2013]; Smeulders et al. [2014]. The tracker is denoted as PMOT to reflect the concepts of shared proposals and multiple object tracking.

Denote the support vector set trained in the SSVM as $\mathcal{V}_{t-1}$, the classification function can then be expressed as a weighted sum of affinities between the candidate bounding box and the support vectors Blaschko and Lampert [2008]; Hare et al. [2011]:

$$f^i_{t-1}(B_t) = \sum_{\bar{B}^m \in \mathcal{V}_{t-1}} w^m k(\bar{B}^m, B_t), \quad m = 1, \ldots, |\mathcal{V}_{t-1}| \tag{6.4}$$

where w^m is a scalar weight associated with the support vector $\bar{B}^m$. Kernel function $k(\bar{B}^m, B_t)$ calculates the affinity between two feature vectors extracted from $\bar{B}^m$ and B_t respectively. The classifier is updated in an online fashion using Bordes et al. [2007, 2008] with a budget Wang et al. [2010]. Intersection kernel is used and other parameters are set same as Hare et al. [2011]. We use histogram features obtained by concatenating 16-bin intensity histograms from a spatial pyramid of 5 levels and RGB channels separately. At each level L, the patch is divided into $L \times L$ cells, resulting in a 2640-D feature vector.

6.4 Experiments

6.4.1 Full Benchmark Evaluations

To evaluate the performance of the proposed multiple object tracking method, we collect 10 videos from various sources, including TB50Wu et al. [2015], OTB Wu et al. [2013] and VOT2015 Kristan et al. [2015]. We denote this dataset as MOOT (Multiple Object Online Tracking) and a few samples can be seen in Figure 6.5. The number of targets in these videos ranges from 2 to 5. This dataset contains extremely challenging scenarios, including repetitive mutual occlusion (videos "liquor" and "skating2") and similar appearance among the targets (videos "bolt1", "bolt2", "football" and "basketball").

We also evaluate the proposed method on the video sequences from Performance Evaluation of Tracking and Surveillance (PETS) 2015 Patino et al. [2016]. These videos are from surveillance cameras and all targets are humans. We list the details of the four sequences in Table 6.1 with corresponding challenges featured. As we can see, all sequences contain challenging aspects, while video "A1_ARENA-15_06_TRK_RGB_2" (row 2 in Figure 6.3) is the most difficult one containing both deformation and occlusion challenges.

Compared Trackers and Evaluation Metrics. Our method (PMOT) is compared with several state-of-the-art methods. Specifically, we compare our method with SPOT Zhang and van der Maaten [2014] which addresses a similar task as ours and it deploys a structure preserving model. We also compare with several single online object trackers to corroborate the point that by sharing and building discriminative classifiers based on proposals, our method is more robust to drifting. MEEM Zhang et al. [2014a], KCF Henriques et al. [2015] and Struck Hare et al. [2011] are three top-ranked trackers in recent large benchmarks Wu et al. [2013]; Smeulders et al. [2014]; Kristan et al. [2014]; Wu et al. [2015] for single online object tracking. For all the trackers, we use their default settings and separately initialize on each object for each video. We also modify the PMOT for the single object case, denoted as PMOTsingle. This allows us to precisely analyze the improvement of adopting the proposal sharing scheme, in term of both the tracking metrics and computational efficiency.

Table 6.1: Attributes of the four video sequences from the PETS dataset.

Video	#humans	#frames	Challenge
N1_ARENA-01_02_TRK_RGB_2	3	115	Size change
W1_ARENA-11_03_ENV_RGB_3	2	107	Body deformation
W1_ARENA-11_03_TRK_RGB_1	2	101	Body deformation
A1_ARENA-15_06_TRK_RGB_2	3	121	Occlusion and body deformation

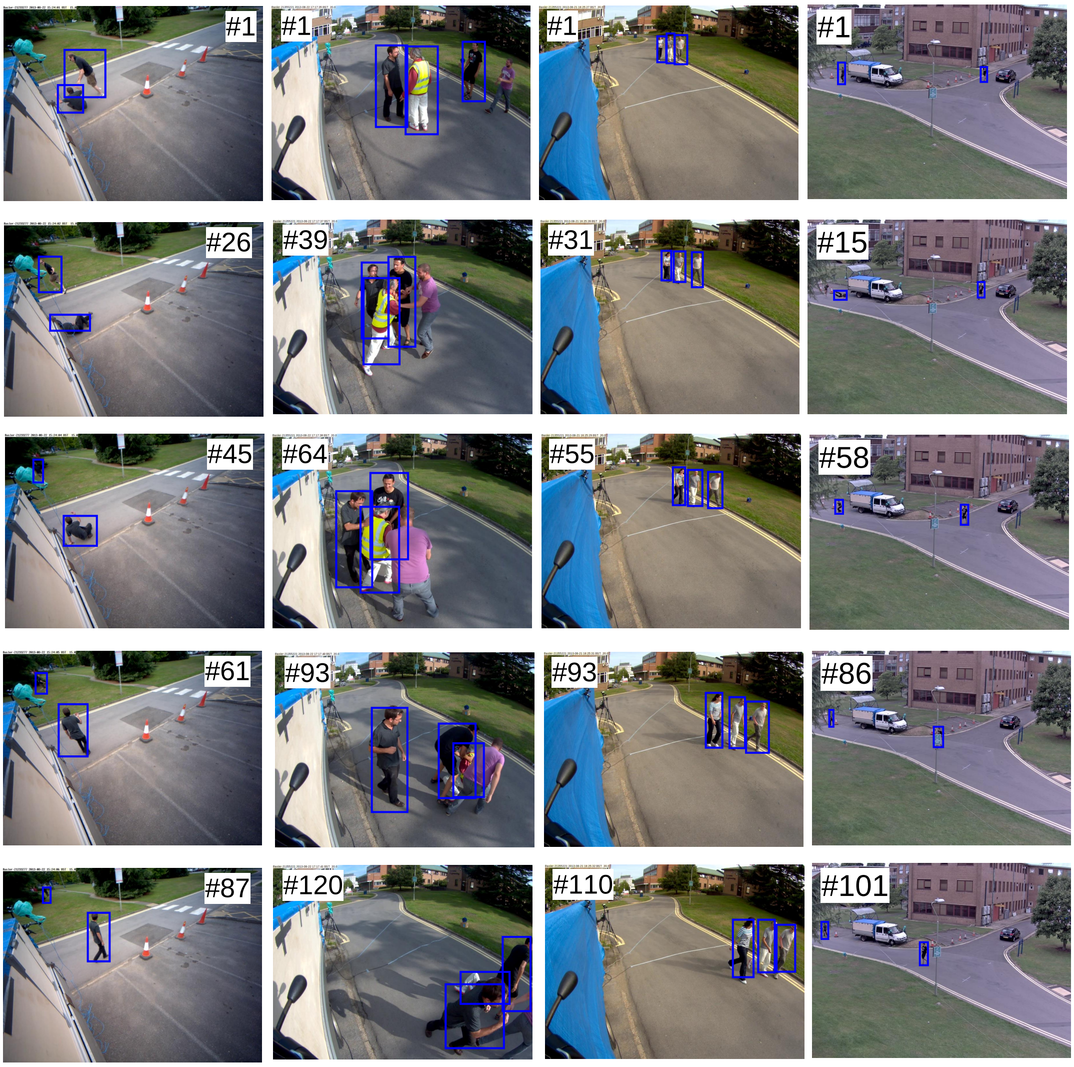


Figure 6.3: Sample sequences from the PETS benchmark dataset Patino et al. [2016] with ground truth object windows (blue).

Table 6.2: Area Under Curve (AUC) of *success plot* and *precision score* (PS) with 20 pixels threshold on the MOOT dataset for the one-pass evaluation (OPE). Cell values: AUC/PS

MOOT	Pro. PMOT	PMOTsingle	SPOT	MEEM	KCF	Struck
ball1	**66.2**/99.0	66.0/**99.3**	30.6/67.4	51.3/74.5	48.5/83.1	52.7/86.0
basketball	**61.5**/**84.0**	60.2/81.7	11.6/8.6	46.2/70.9	51.3/59.8	38.5/50.3
bolt1	**47.4**/**93.8**	36.6/71.6	0.5/0.5	23.5/50.6	34.3/70.6	33.9/73.8
bolt2	50.8/89.0	38.6/69.9	0.6/0.8	47.3/90.4	50.9/93.6	**57.4**/**97.7**
football	**62.0**/94.6	57.8/88.9	23.4/41.5	60.7/**97.0**	49.5/69.1	57.5/79.7
human4	60.7/93.5	34.5/48.5	**61.5**/**99.5**	57.4/91.2	50.2/75.7	62.7/94.7
jogging	**67.4**/**97.6**	63.8/89.7	12.3/13.5	60.6/88.4	15.5/19.9	15.0/19.7
liquor	**61.0**/**79.8**	41.6/51.0	32.8/38.2	10.6/16.8	18.8/24.6	7.2/8.9
skating1	56.5/71.2	46.5/55.4	55.5/78.4	62.2/**92.3**	**62.8**/89.6	35.9/50.0
skating2	**50.8**/**44.9**	48.1/43.7	34.6/25.8	35.9/28.4	33.7/37.1	26.7/18.2
Mean	**58.5**/**86.2**	49.5/71.5	23.7/34.1	41.7/67.7	40.5/61.6	37.5/61.4

We use the single online object tracking metrics to measure the tracking performance, similar to Zhang and van der Maaten [2014]. Evaluation metrics and code are provided by the benchmark Wu et al. [2013, 2015]. We employ the one-pass evaluation (OPE) and use two metrics: *precision plot* and *success plot*. The former one calculates the percentage (*precision score*, PS) of frames whose center location is within a certain threshold distance with the ground truth. A commonly used threshold is 20 pixels. The latter one calculates a same percentage but based on bounding box overlap threshold. We utilize the area under curve (AUC) as an indicative measurement for it.

Experimental Setting. Our tracker is implemented using C++ and MATLAB, on an i7-2600 3.40 GHz desktop with a 8 GB RAM. For the EdgeBox proposal method and SSVM applied, we use the default setting recommended by the authors, except those specified otherwise. We further discuss some parameters in Section 6.4.2

Benchmark Results. The results are summarized in Figure 6.4 and Table 6.2. We can see that the SPOT tracker achieves undesirable results, significantly lagging behind other compared methods. In term of the PS metric, it is 27.3% worse than Struck, the second worst tracker. It is not particularly surprising though, as can be seen in Figure 6.5, where we draw the visual comparison between the proposed PMOT and SPOT. It clearly demonstrates that the SPOT tracker presumes a strong spatial structure exhibited among the objects, while it does not always hold. As shown in the video "bolt1" (row 1 in Figure 6.5), the four dash-line windows (SPOT) still maintain the relative positions while drifting away the true objects. In contrast, our method robustly and consistently tracks the objects even they are not moving coherently.

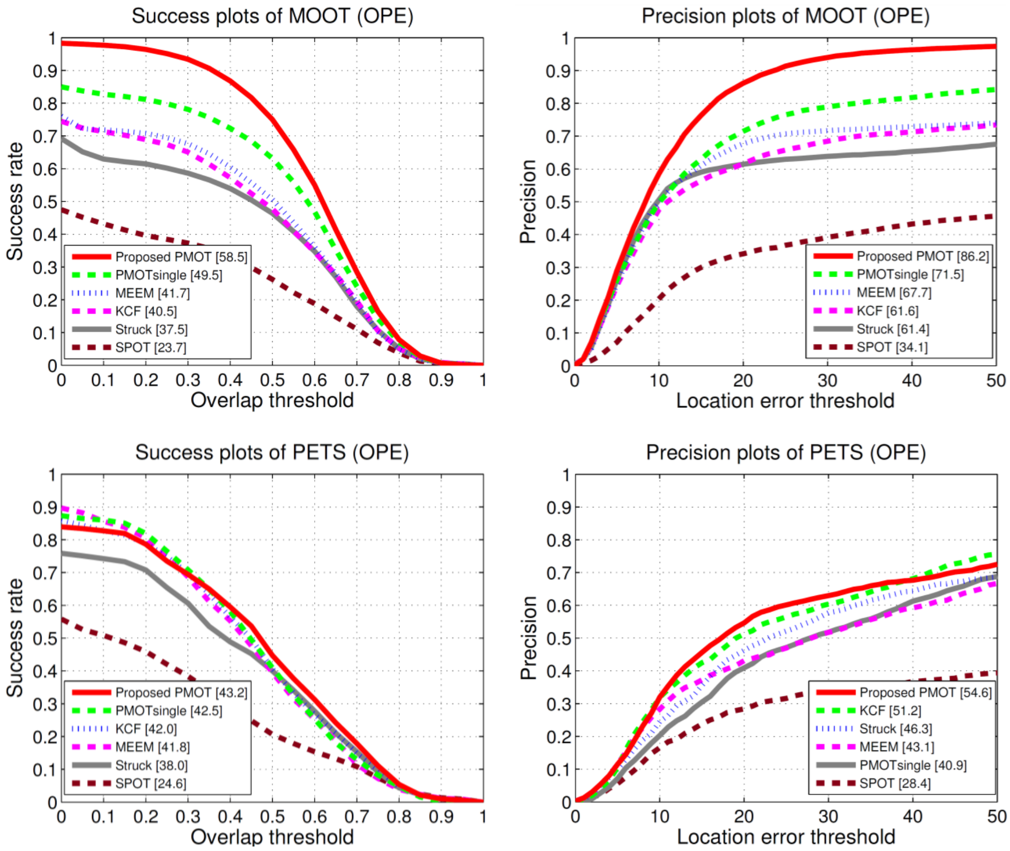


Figure 6.4: *Success plot* and *precison plot* on two datasets: MOOT and PETS. Algorithms are ranked by the area under the curve (AUC) and the *precision score* (20 pixels threshold, PS). Our method achieves consistently superior performance, especially on the more challenging MOOT dataset.

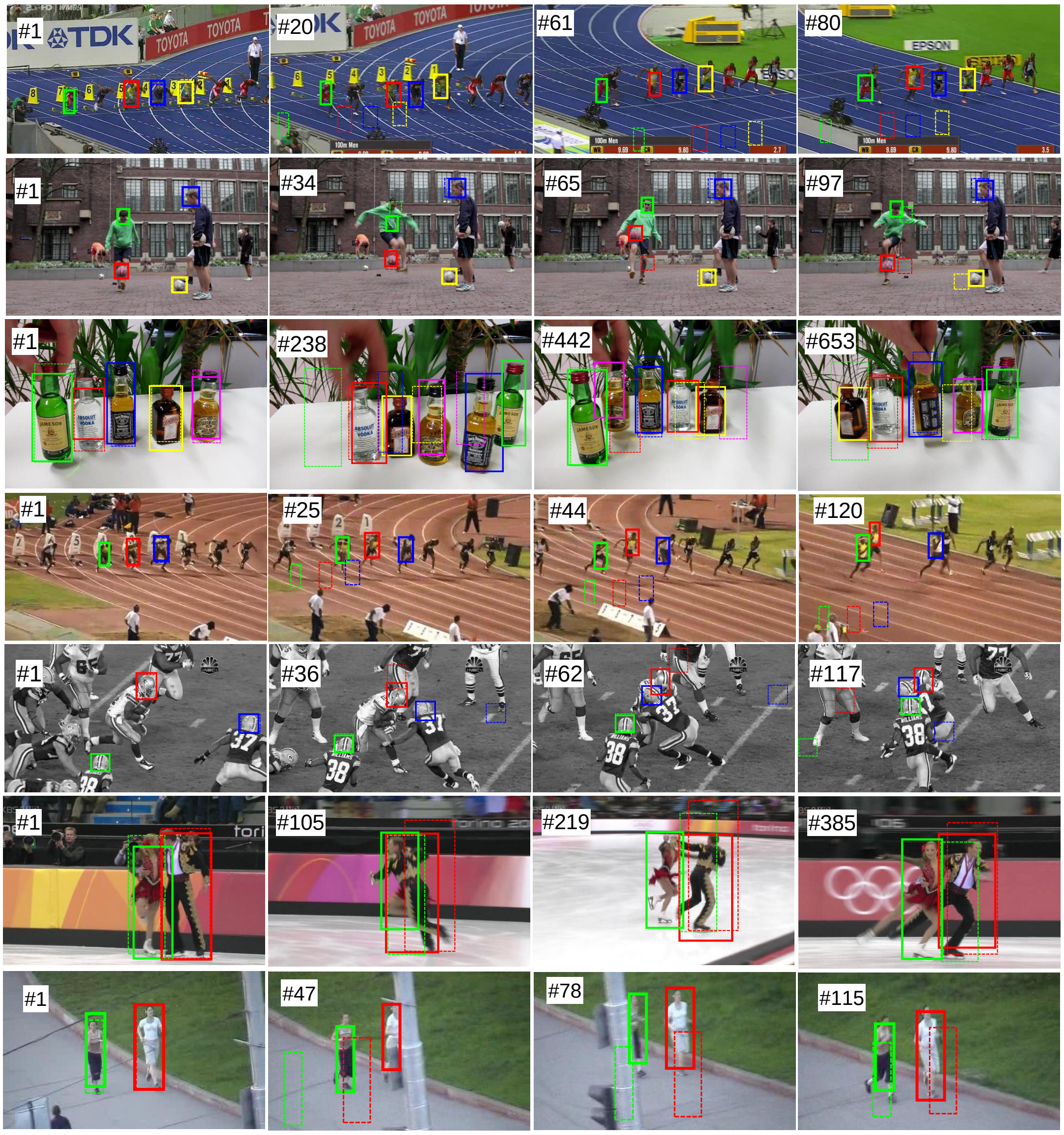


Figure 6.5: Qualitative comparisons with the proposed PMOT tracker (solid lines) against the SPOT tracker (dash lines) on videos "bolt1", "ball1", "liquor", "bolt2", "football", "skating2" and "jogging" from MOOT dataset (from top to bottom). Our method exhibits robustness in challenging scenarios such as repetitive mutual occlusions and similar target appearances.

When comparing to the single object online tracking methods, the improvement is clearly shown. On the challenging MOOT dataset, our PMOT tracker outperforms the second best tracker by a large margin, with 9% and 14.7% in term of AUC and PS respectively. We can also see the clear advantage of applying the proposal based approach. Even the single object tracking variant, PMOTsingle, outperforms the best non-proposal tracker, MEEM, by 7.8% and 3.8% in AUC and PS respectively. This is partly contributed by the online updating strategy of collecting the proposals as hard negative samples to improve the discriminative power of the classifier, hence is robust to the distractions from other objects as well as potential distractors in the background.

For the PETS dataset, we can see that the improvement of PMOT is not great, outperforming the second best tracker, by 3.4% and 0.7% in the PS and AUC metrics, respectively. This is partly due to the fact that there is no significant interactions presented among the objects on PETS, except the video "A1_ARENA-15_06_TRK_RGB_2". Therefore, our proposed multiple object tracking system is unable to take a strong advantage of the proposal sharing benefit.

6.4.2 Further Remarks

Temporal Smoothness. The smoothness term $s(B_t, B_{t-1}^i)$ (6.3) discussed in Section 6.3.2 controls the temporal consistency of the trajectory. This is especially important in our formulation as the object proposals are generated independently in each frame, which results temporal inconsistencies inevitably. We test different σ values and include the results in Table 6.3. We observe that a small σ leads to a strong smoothness constraint, which harms the performance when objects are occluded, while a large σ tends to result in unstable trajectories.

Size Adaption Ratio. The size adaption ratio α in (6.2) allows the target window to adapt the object size changes naturally once set properly. A smaller α leads to a larger set of object proposals with a more significant size variance, which harms both the computational efficiency and trajectory stability. We validate it with different values and results is in Table 6.3. It corroborates that a larger value is preferable, but the performance drops when $\alpha = 0.9$, as it constrains the sizes of object proposals too tight that it fails to adapt the object size changes.

Table 6.3: Area Under Curve (AUC) of *success plot* and *precision score* (20 pixels threshold) results of PMOT with different temporal smoothness constraints and size adaption ratios.

	Temporal Smoothness			Size Adaption Ratio		
	$\sigma = 0.5R_i$	$\sigma = R_i$	$\sigma = 2R_i$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$
AUC	51.0	58.5	56.2	49.5	58.5	57.9
PS	72.3	86.2	84.1	70.5	86.2	84.9

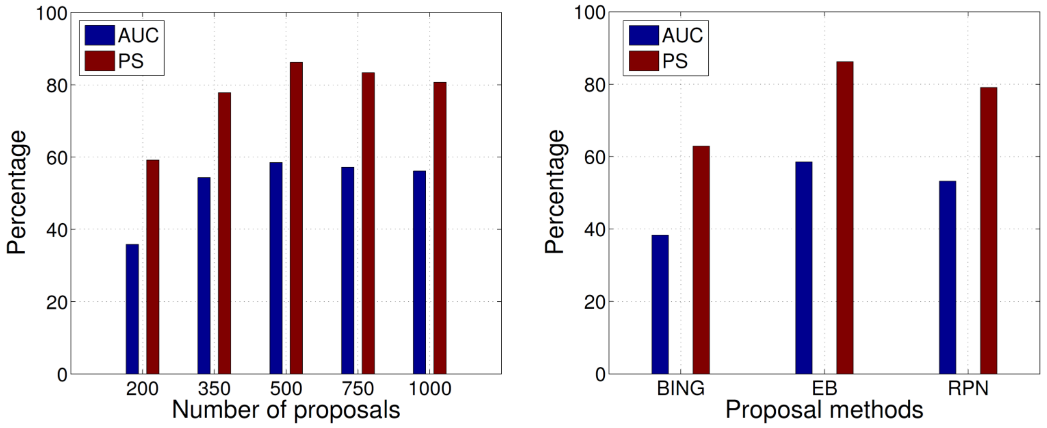


Figure 6.6: Area Under Curve (AUC) of *success plot* and *precision score* (20 pixels threshold) results of PMOT with different maximal numbers of proposals and various proposal methods.

Table 6.4: Processing times (frames per second, FPS) of PMOT on videos containing different number of objects.

	Pro. PMOT				PMOTsingle
# target	$N_o = 2$	$N_o = 3$	$N_o = 4$	$N_o = 5$	$N_o = 1$
FPS	4.1	3.3	2.6	1.9	5.3

Maximal Number of Object Proposals. We test 5 variants with the maximal object proposal number set at 200, 350, 500, 750 and 1000, respectively. The results are reported in term of AUC/PS metrics as included in Figure 6.6. As discussed in Section 6.3.1, using insufficient number of proposal leads to a bad coverage of the false positives as well as the object, while using a large number of proposals attracts spurious candidates.

Alternative Object Proposal Methods. We evaluate using other two popular object proposal methods, BING Cheng et al. [2014] and region proposal network (RPN) Ren et al. [2015], instead of EdgeBox for proposals. Results are in Figure 6.6. Both performances are worse than the EdgeBox method. This is expected. As shown Hosang et al. [2014]; Zitnick and Dollár [2014], BING results in a relatively low recall of the objects, while RPN performs undesirably for small-size objects.

Computational Efficiency. Since the object proposals are shared among the classifiers of multiple targets, we reduce the computational load by not repeating the proposal generation and feature extraction for each target. Table 6.4 shows the processing times (frames per second, FPS) for different number of targets. We categorize the test videos according to the number of targets in them. For PMOTsingle, the number of targets is always 1. As we can see, our system is computationally efficient.

6.5 Summary

We proposed a computationally efficient and accurate model-free multiple object tracking method. It takes the advantage of the object proposals and generates a small and shared set of object hypotheses in the frame. Then it initializes multiple classifiers for each target using the shared set. In consecutive frames, the application and update of the classifiers are also achieved by using the detected proposals. We evaluated our method on both PETS and a newly introduced dataset. The results show superior performance against the state-of-the-art.

Chapter 7

Conclusion

In this thesis, we addressed a specific tracking problem: model-free online visual object tracking. As we discussed in Chapter 1 and 2, it attracted a significant number of researchers in recent years along with the drastically increased volume of video contents. One important reason is that model-free trackers are convenient to be deployed in various application scenarios, e.g., for assisting drone control [TeuliÃÍre et al., 2011] and facilitating fully autonomous sports video analysis to improve players' performance [Xing et al., 2011; Lu et al., 2013; Drory et al., 2017].

To summarize our work, the rest part of this chapter is organized as follows: we firstly highlight our tracker, EBT [Zhu et al., 2016a], in the following section. Then we list the contribution and major points of our research in Section 7.1. We further discuss several promising directions based on the shortcomings of current state-of-the-art trackers as well as new developments shown in the computer vision community, in Section 7.2.

Highlight. Our proposed EBT tracker [Zhu et al., 2016a] demonstrate a significant performance boost compared to state-of-the-art trackers on four benchmark datasets: OTB [Wu et al., 2013], TB50 [Wu et al., 2015], VOT2014 [Kristan et al., 2014] and VOT2015 [Kristan et al., 2015]. Especially on the competitive VOT2015 challenge, the EBT achieved third place (out of 62 trackers) and it was the best tracker that did not exploit the learned features of deep Convolutional Neural Networks (CNNs).

The EBT also won the recent Thermal Infrared Visual Object Tracking VOT-TIR 2016 Challenge, which aims at comparing single object visual trackers on thermal infrared (TIR) sequences. VOT-TIR 2016 is the second benchmark on short-term tracking in TIR sequences. Results of 24 trackers are presented and concluded with a workshop at ECCV 2016.

In our CVPR Workshop paper [Zhu et al., 2016c], we further investigated the impact of deep convolutional neural networks in proposal-based tracking frameworks. We introduced the RPNT, that employs the region proposal network (RPN) and feature pooling network in the EBT tracking framework. This led to further improved results on the challenging PETS2016 dataset, outperforming state-of-the-art trackers, such as EBT [Zhu et al., 2016a], MUSTer [Hong et al., 2015b], MEEM [Zhang et al., 2014a] and SRDCF [Danelljan et al., 2015].

7.1 Contribution

As mentioned in Chapter 1 and 2, model-free online visual object tracking is challenging, since trackers need to address shadows, sudden appearance changes, similar targets, wild unpredicted motion and long occlusions as the video runs, given only one reliable training sample, which is the target annotation manually or automatically initialized in the first frame of the video. In this sense, model-free online visual object tracking is believed to be harder than object detection and classification [Smeulders et al., 2014], as other two tasks can usually access to a large offline training dataset and unconstrained training time for learning the object model [Russakovsky et al., 2015; Everingham et al., 2015].

Taking these challenging aspects into account, this thesis explored the model-free online visual object tracking problem in four different perspectives, i.e., tracking with background contextual clusters in Chapter 3, global proposal tracking in Chapter 4, affine motion tracking in Chapter 5 and a novel multiple object model-free online tracking framework in Chapter 6. To be specific:

Tracking with Contextual Clusters

Since most conventional tracking-by-detection approaches for visual object tracking assume that the task at hand is a binary foreground-versus-background classification problem, we proposed new multiple fine-grained foreground-versus-contextual-cluster models that provided more discriminative classifications in Chapter 3.

The novelty of our work [Zhu et al., 2017] lied in exploring multiple contextual clustering structures for visual tracking. Due to the special setting of model-free tracking problem, in particular, lacking of a large set of positive training samples, makes training a strong classier, that can overcome various challenging tracking scenarios such as occlusion and non-rigid object deformation, extremely difficult. Our method is the first in solving these difficulties by building on robust frameworks in discriminative learning and clustering. We constructed a model that achieved better representing data distribution with limited samples as proven by our experiments on large datasets.

Global Tracking based on Instance-Specific Proposals

We explored another limit existed in current state-of-the-art trackers in Chapter 4. Typical trackers assume the object location in the new frame is near the previous location and the trajectory is smooth. This assumption holds at most time and eases the difficulties of dealing a huge search space as well as potential mismatches from the background. However, it may fail due to abrupt motion and occlusion, leading to the drift of the tracker. In those cases, employing only a local search scheme is hard to recover the object and most tracking systems have to incorporate a separate re-detection component, which is difficult and tricky to design.

In contrast, our work [Zhu et al., 2016a,c] offered a solution that employed a global search strategy. Unlike conventional sampling methods, such as particle filter [Gustafsson et al., 2002] or Markov Chain Monte Carlo [Khan et al., 2004], we obtained the candidates by generating a small number of proposals similar to the object. This method brought a threefold benefit:

- Firstly, there was no more an assumption of a correct previous location, and hence we would not miss the object as long as it was present in the frame.

- Secondly, some false positives hard for the classifier might be excluded at the proposal generation stage and the small number of proposals enabled richer features and stronger classifiers, such as the deep CNNs based methods [Girshick et al., 2014; Girshick, 2015].

- Lastly, we further suppressed the mismatches from the background by treating the proposals as hard negative training samples during the online updating of the classifier.

To generate the proposals, we did not straightforwardly employ existed object proposal algorithms such as EdgeBox [Zitnick and Dollár, 2014] and BING [Cheng et al., 2014], as the proposals generated from them are generic objects and instance indistinguishable. We thus crafted a bottom-up Haar-like feature to capture the instance-level spatial information of the object. Using this feature, a light-weight online linear SVM was then deployed to select a small number of proposals from the ones generated from EdgeBox. This scheme allowed us to achieve a higher recall while using a smaller number of proposals.

Affine Motion Tracking

In Chapter 5, for tracking objects which undergo affine motion (transformation including translation, rotation, scale, dilatation and shear deformations, that span the six degrees of freedom of motion, e.g., moving cars from the camera installed in unmanned aerial vehicles), we presented a novel tracking framework [Zhu et al., 2015] which took advantage of the intrinsic Lie group structure of the 2D affine motion matrices and imposed this motion structure on a kernelized structured output SVM classifier. Furthermore, we summarize the main contributions below:

- We identified an important shortcoming of a recent state-of-the-art method. Since Struck [Hare et al., 2011] can learn the appearance change of an object, one may assume Struck can also adapt to any affine transformations. Our work proved that this was not the case.

- Our work provided affine transformation tracking in a correct and principled manner using the geodesic ally accurate formulation on Lie algebra, while Struck flattened out manifold structure of motion matrices.

Multiple Object Model-Free Tracking with Shared Proposals

In Chapter 6, we focused on developing an accurate and efficient model-free Multiple-Object Tracking (MOT) approach, by utilizing a generic object proposal method (EdgeBox [Zitnick and Dollár, 2014] in this work) to avoid any pre-trained category-specific object detector, such as DPM [Felzenszwalb et al., 2010].

Our major motivation in this work was that most conventional methods for tracking multiple objects [Milan et al., 2016] emphasized on improving the performance of category-specific object detectors as well as target (or "tracklet" [Milan et al., 2016]) association between neighboring frames. These methods were therefore heavily sensitive to the performance of the object detectors, leading to limited application scenarios.

While our MOT framework [Zhu et al., 2016b] extended a novel single object model-free tracking method [Zhu et al., 2016a,c] to the multiple object model-free online tracking problem by well handling the problems of computational efficiency and occlusion ambiguities. It incorporated category-independent object proposal algorithms [Zitnick and Dollár, 2014; Cheng et al., 2014] to generate a shared pool of target candidates in the current testing frame instead of using pre-trained object detectors. These shared proposals were further efficiently used to train more discriminative classifiers to distinguish among similar targets. A new multiple object model-free tracking dataset was further proposed to evaluate the performance of trackers, as current benchmarks [Milan et al., 2016] used only videos of pedestrians and vehicles.

7.2 Future Work

Although current "tracking-by-detection" based trackers demonstrated outstanding performance on standard benchmarks [Smeulders et al., 2014; Kristan et al., 2015] comparing to more conventional trackers, there are however several limitations:

(1) Tracking task is solved with a classifier trained using limited reliable training samples (initialization bounding boxes) [Kristan et al., 2013, 2016]. This presents a significant challenge as the training examples used to update the tracker from following frames might be corrupted and there is no supervised information to correct it.

(2) Tracking approaches lack an effective scheme for leveraging the essential motion cue, although it played a crucial role in human perceiving visual tracking [Wertheimer, 1938; Carrasco, 2011; Huk]. In the current literature, it was majorly modeled by statistical methods [Smeulders et al., 2014];

(3) Deep Convolutional Neural Networks are largely used as a feature representation for trackers proposed in recent publications [Zhu et al., 2016c; Ma et al., 2015a; Hong et al., 2015a] without taking account into the sequential (temporal) property of the video frames.

To address these limits, we provide discussions based on latest research and include them in the following sections.

7.2.1 Regularization Information for Long-Term Tracking

As we discussed in Chapter 1, only one reliable initialization sample is available at the beginning of the input video to train the tracker. This rather "ill" condition presents a severe limitation for continually tracking a target over a long period of time where the target could be occluded or undergoes significant appearance change. Some works attempt to solve these particularly challenging problems using occlusion modeling [Yang and Sundaramoorthi, 2013; Zhang et al., 2014b] or deformable object model [Godec et al., 2011; Li et al., 2015]. It is, however, extremely difficult when many problems exist in the same sequence [Smeulders et al., 2014; Wu et al., 2015].

Note that there is an effort in the tracking community focusing particularly on long-term object tracking [LTDT2014]. It aims is to attain a reliable, autonomous tracking of single or multiple objects, over long-term sequences (at least 2 minutes long at 25-30 fps, but ideally 10 minutes or longer), since few-if any-systems in the existed literature are capable of running reliably for long periods (days, weeks, or even months) without the need for human intervention to reset or re-initialize the tracker.

To alleviate such an issue, one possible way is by incorporating another long-term tracking component to suppress short-term deviation such as drastic object appearance change and occlusion. In recent literature, [Hong et al., 2015b] adopted a dual-component approach inspired by the well-known Atkinson-Shiffrin Memory Model, which consisted of short- and long-term memory stores to process target appearance memories. The integrated long-term component, which was based on key-point matching-tracking and RANSAC [Fischler and Bolles, 1981] estimation, could interact with the long-term memory and provided additional information for output control. Instead, [Ma et al., 2015b] trained an online random fern classifier to re-detect objects in case of tracking failure.

Another possible solution is to leverage the regularization prior beyond the annotation in the first frame, e.g., the category-level regularization from a specific object model, such as a pre-trained car or dog detector [Drory et al., 2017; Hall and Perona, 2014], as we mentioned Section 1.3. Furthermore, the object-level regularization from general “objectness” classifiers, such as BING [Cheng et al., 2014] and EdgeBox[Zitnick and Dollár, 2014], can also be deployed [Zhu et al., 2016a; Huang et al., 2015].

7.2.2 Temporal Information for Short-Term Tracking

In the current literature, motion cue has been mainly explored using statistical approaches such as particle filter [Gustafsson et al., 2002] or Markov Chain Monte Carlo [Khan et al., 2004]. To address this shortcoming, one can analyze the motion trajectory of the target. This strategy has been incorporated for multiple object tracking (target association) [Dicle et al., 2013; Milan et al., 2016]. However, for single object model-free tracking, not many studies can handle the scenarios such as the camera and target move wildly, thus making a direct employment of those methods difficult.

One example is the TLD tracker proposed by [Kalal et al., 2012], which applied the P-N learning to train an object detector from a labeled frame and a video stream. P-expert exploited the temporal structure in the video while assumed that the object moved along a trajectory. It remembered the location of the object in the previous frame and estimated the object location in current frame using a frame-to-frame tracker.

Optical flow is potentially a choice to provide motion input for visual object tracking, since its performance grows stronger as demonstrated on recent literature and benchmarks[Geiger et al., 2012]. Although it is widely employed by visual tracking tasks where computational cost is not an issue [Papazoglou and Ferrari, 2013; Tsai et al., 2016], optical flow is still not efficient and effective enough yet to be directly applied. One way to overcome this is to deploy the optical flow procedure only when needed as shown in [Hua et al., 2015].

7.2.3 Sequential Data based RNN Tracker

Recurrent Neural Networks (RNNs) have been successfully applied for addressing sequential data recently including video classification [Ng et al., 2015] and semantic image description generation (bidirectional Recurrent Neural Networks over sentences) [Karpathy and Li, 2015]. One highlight is the commonly used Long Short Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997; Gregor et al., 2015] units, which use memory cells to store, modify, and access internal state, allowing it to discover long-range temporal relationships. Similar to feature-pooling, LSTM networks operate on frame-level CNN activations, and can learn how to integrate information over time. By sharing parameters through time, it might capture a global description of the video’s temporal evolution.

In comparison, since online visual object tracking is operating on videos, it has potential to be solved using the RNNs architecture instead of directly feeding frames into CNNs without considering their sequential nature [Wang et al., 2015b; Wu et al., 2015]. However, the model-free visual object tracking problem possesses its own difficulties, which cannot be simply solved by straightforwardly deploying existed methods, e.g., how to train and update such a model in each new frame. One pioneering work can be found in [Gan et al., 2015].

Bibliography

ALBRECHT, T.; LUTHI, M.; AND VETTER, T., 2008. A statistical deformation prior for non-rigid image and shape registration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 73)

ALOV300. Amsterdam Library of Ordinary Videos for evaluating visual trackers robustness. http://www.alov300.org/. (cited on pages 16 and 43)

ANDRILUKA, M.; ROTH, S.; AND SCHIELE, B., 2010. Monocular 3d pose estimation and tracking by detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 6)

AVIDAN, S., 2004. Support vector tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 26, 8 (2004), 1064–1072. (cited on pages 15, 33, 35, 58, and 74)

BABENKO, B.; YANG, M.-H.; AND BELONGIE, S., 2009. Visual tracking with online multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 15, 27, 28, 33, 37, 38, 58, 74, 90, 91, 95, and 96)

BAE, S.-H. AND YOON, K.-J., 2014. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 89, 91, 92, and 94)

BAKER, S. AND MATTHEWS, I., 2001. Equivalence and efficiency of image alignment algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 73)

BAKER, S. AND MATTHEWS, I., 2004. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision (IJCV)*, 56, 3 (2004), 221–255. (cited on page 73)

BALLARD, D. H. AND BROWN, C. M., 1982. *Computer Vision*. (cited on page 1)

BALTIERI, D.; VEZZANI, R.; AND CUCCHIARA, R., 2011. 3DPeS: 3d people dataset for surveillance and forensics. In *Joint ACM Workshop on Human Gesture and Behavior Understanding*. (cited on page 22)

BAYRO-CORROCHANO, E. AND ORTEGON-AGUILAR, J., 2004. Lie algebra template tracking. In *International Conference on Pattern Recognition (ICPR)*. (cited on page 74)

Blake, A. and Isard, M., 1996. The CONDENSATION algorithm - conditional density propagation and applications to visual tracking. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on page 73)

Blaschko, M. B. and Lampert, C. H., 2008. Learning to localize objects with structured output regression. In *European Conference on Computer Vision (ECCV)*. (cited on pages 30, 39, 63, 76, 77, and 97)

Bordes, A.; Bottou, L.; Gallinari, P.; and Weston, J., 2007. Solving multiclass support vector machines with LaRank. In *International Conference on Machine Learning (ICML)*. (cited on pages 28, 29, 30, 40, 41, 63, 81, and 97)

Bordes, A.; Usunier, N.; and Bottou, L., 2008. Sequence labelling SVMs trained in one pass. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*. (cited on pages 28, 29, 30, 40, 41, 48, 81, and 97)

Bourdev, L.; Maji, S.; Brox, T.; and Malik, J., 2010. Detecting people using mutually consistent poselet activations. In *European Conference on Computer Vision (ECCV)*. (cited on page 18)

Boykov, Y. and Huttenlocher, D., 2000. Adaptive bayesian recognition in tracking rigid objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 73)

Briechle, K. and Hanebeck, U. D., 2001. Template matching using fast normalized cross correlation. In *SPIE: Optical Pattern Recognition XII*. (cited on page 64)

Brox, T. and Malik, J., 2010. Object segmentation by long term analysis of point trajectories. In *European Conference on Computer Vision (ECCV)*. (cited on pages 5 and 6)

Cai, Z.; Wen, L.; Yang, J.; Lei, Z.; and Li, S. Z., 2013. Structured visual tracking with dynamic graph. In *Asian Conference on Computer Vision (ACCV)*. (cited on page 18)

Cao, Y., 2008. Hungarian algorithm for linear assignment problems (V2.3). http://www.mathworks.com/. (cited on page 96)

Carrasco, M., 2011. Visual attention: The past 25 years. *Vision Research*, 51, 13 (2011), 1484 – 1525. (cited on page 110)

Carreira, J. and Sminchisescu, C., 2012. CPMC: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 34, 7 (2012), 1312–1328. (cited on pages 9 and 60)

Carvalho, P.; Cardoso, J. S.; and Corte-Real, L., 2012. Filling the gap in quality assessment of video object tracking. *Image and Vision Computing (IVC)*, 30, 9 (2012), 630–640. (cited on page 26)

Chatfield, K.; Lempitsky, V.; Vedaldi, A.; and Zisserman, A., 2011. The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference (BMVC)*. (cited on page 40)

Cheng, M.; Zhang, Z.; Lin, W.; and Torr, P. H. S., 2014. BING: binarized normed gradients for objectness estimation at 300fps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 9, 58, 70, 93, 95, 104, 109, 110, and 112)

Chu, D. and Smeulders, A., 2010. Thirteen hard cases in visual tracking. In *International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. (cited on pages 21 and 22)

Collins, R. T., 2003. Mean-shift blob tracking through scale space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 15)

Comaniciu, D.; Ramesh, V.; and Meer, P., 2003. Kernel-based object tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 25, 5 (2003), 564–575. (cited on pages 15 and 57)

Cootes, T.; Edwards, G.; and Taylor, C., 1998. Active appearance models. In *European Conference on Computer Vision (ECCV)*. (cited on page 73)

Cortes, C. and Vapnik, V., 1995. Support-vector networks. *Machine Learning*, 20, 3 (1995), 273–297. (cited on page 27)

Danelljan, M.; Hager, G.; Shahbaz Khan, F.; and Felsberg, M., 2015. Learning spatially regularized correlation filters for visual tracking. In *International Conference on Computer Vision (ICCV)*. (cited on pages 16 and 107)

Danelljan, M.; HÃďger, G.; Shahbaz Khan, F.; and Felsberg, M., 2014a. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference (BMVC)*. (cited on pages 15 and 16)

Danelljan, M.; Shahbaz Khan, F.; Felsberg, M.; and van de Weijer, J., 2014b. Adaptive color attributes for real-time visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 15)

Denman, S.; Fookes, C.; and Sridharan, S., 2009. Improved simultaneous computation of motion detection and optical flow for object tracking. In *International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. (cited on page 6)

Dicle, C.; Camps, O. I.; and Sznaier, M., 2013. The way they move: Tracking multiple targets with similar appearance. In *International Conference on Computer Vision (ICCV)*. (cited on pages 92 and 112)

DINH, T. B.; VO, N.; AND MEDIONI, G., 2011. Context tracker: Exploring supporters and distracters in unconstrained environments. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 17, 35, and 55)

DIVVALA, S. K.; HOIEM, D.; HAYS, J. H.; EFROS, A. A.; AND HEBERT, M., 2009. An empirical study of context in object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 17)

DRORY, A.; ZHU, G.; LI, H.; AND HARTLEY, R., 2017. Rapid automated detection and tracking of slalom paddlers using cascade classifiers and discriminative correlation filters. *Computer Vision and Image Understanding (CVIU)*, (2017). (cited on pages 3, 4, 8, 107, and 111)

DRUMMOND, T. AND CIPOLLA, R., 2000. Application of Lie algebras to visual servoing. *International Journal of Computer Vision (IJCV)*, 37 (2000), 21–41. (cited on page 74)

DUAN, G.; AI, H.; CAO, S.; AND LAO, S., 2012. Group tracking: Exploring mutual relations for multiple object tracking. In *European Conference on Computer Vision (ECCV)*. (cited on page 92)

DUFFNER, S. AND GARCIA, C., 2013. PixelTrack: a fast adaptive algorithm for tracking non-rigid objects. In *International Conference on Computer Vision (ICCV)*. (cited on page 18)

EVERINGHAM, M.; ESLAMI, S. M. A.; GOOL, L. V.; ET AL., 2015. The Pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, 111, 1 (2015), 98–136. (cited on pages 5, 16, 18, 19, 23, 24, 25, 26, 27, 44, 58, 67, 70, 92, 95, and 108)

FELSBERG, M.; BERG, A.; HAGER, G.; ET AL., 2015. The thermal infrared visual object tracking VOT-TIR2015 challenge results. In *International Conference on Computer Vision Workshops (ICCVW)*. (cited on pages 6, 24, and 90)

FELZENSZWALB, P. F.; GIRSHICK, R. B.; MCALLESTER, D. A.; AND RAMANAN, D., 2010. Object detection with discriminatively trained part-based models. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 32, 9 (2010), 1627–1645. (cited on pages 7, 9, 18, 89, and 110)

FERRYMAN, J. AND SHAHROKNI, A., 2009. Pets2009: Dataset and challenge. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter)*. (cited on page 20)

FISCHLER, M. A. AND BOLLES, R. C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24, 6 (1981), 381–395. (cited on page 111)

GADE, R. AND MOESLUND, T. B., 2014. Thermal cameras and applications: A survey. *Machine Vision and Applications (MVA)*, 25, 1 (2014), 245–262. (cited on page 24)

Gall, J.; Yao, A.; Razavi, N.; Gool, L. V.; and Lempitsky, V., 2011. Hough forests for object detection, tracking, and action recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, (2011). (cited on page 38)

Gan, Q.; Guo, Q.; Zhang, Z.; and Cho, K., 2015. First step toward model-free, anonymous object tracking with recurrent neural networks. *CoRR*, (2015). (cited on page 113)

Geiger, A.; Lenz, P.; and Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 112)

Girshick, R., 2015. Fast R-CNN. In *International Conference on Computer Vision (ICCV)*. (cited on pages 19 and 109)

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 58, 67, 92, and 109)

Godec, M.; Roth, P. M.; and Bischof, H., 2011. Hough-based tracking of non-rigid objects. In *International Conference on Computer Vision (ICCV)*. (cited on pages 18 and 111)

Godec, M.; Sternig, S.; Roth, P. M.; and Bischof, H., 2010. Context-driven clustering by multi-class classification in an active learning framework. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. (cited on page 36)

Gomez-Balderas, J. E.; Flores, G.; García Carrillo, L. R.; and Lozano, R., 2013. Tracking a ground moving target with a quadrotor using switching control. *Journal of Intelligent and Robotic Systems*, 70, 1-4 (2013), 65–78. (cited on pages 2 and 15)

Grabner, H.; Grabner, M.; and Bischof, H., 2006. Real-time tracking via on-line boosting. In *British Machine Vision Conference (BMVC)*. (cited on pages 15 and 74)

Grabner, H.; Matas, J.; Van Gool, L.; and Cattin, P., 2010. Tracking the invisible: Learning where the object might be. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 17 and 35)

Gregor, K.; Danihelka, I.; Graves, A.; Rezende, D.; and Wierstra, D., 2015. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning (ICML)*. (cited on page 112)

Grundmann, M.; Kwatra, V.; Han, M.; and Essa, I., 2010. Efficient hierarchical graph-based video segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 18)

Guennebaud, G.; Jacob, B.; et al. Eigen v3. http://eigen.tuxfamily.org. (cited on page 84)

GUSTAFSSON, F.; GUNNARSSON, F.; BERGMAN, N.; ET AL., 2002. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing (TSP)*, 50, 2 (2002), 425–437. (cited on pages 109 and 112)

HAGER, G. D. AND BELHUMEUR, P. N., 1998. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 20, 10 (1998), 1025–1039. (cited on page 14)

HALL, D. AND PERONA, P., 2014. Online, real-time tracking using a category-to-individual detector. In *European Conference on Computer Vision (ECCV)*. (cited on pages 8 and 111)

HARE, S.; SAFFARI, A.; AND TORR, P. H. S., 2011. Struck: Structured output tracking with kernels. In *International Conference on Computer Vision (ICCV)*. (cited on pages 15, 26, 27, 28, 29, 30, 31, 37, 38, 39, 40, 41, 46, 48, 51, 53, 55, 57, 58, 60, 63, 74, 75, 76, 77, 78, 81, 82, 84, 97, 98, and 109)

HE, W.; YAMASHITA, T.; LU, H.; AND LAO, S., 2009. SURF tracking. In *International Conference on Computer Vision (ICCV)*. (cited on page 73)

HENRIQUES, J. F.; CASEIRO, R.; MARTINS, P.; AND BATISTA, J., 2012. Exploiting the circulant structure of tracking-by-detection with kernels. In *European Conference on Computer Vision (ECCV)*. (cited on page 15)

HENRIQUES, J. F.; CASEIRO, R.; MARTINS, P.; AND BATISTA, J., 2015. High-speed tracking with kernelized correlation filters. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, (2015). (cited on pages 16, 35, 51, 55, 57, 64, 90, and 98)

HÉRISSÉ, B.; HAMEL, T.; MAHONY, R.; AND RUSSOTTO, F.-X., 2010. A terrain-following control approach for a vtol unmanned aerial vehicle using average optical flow. *Autonomous Robots*, 29, 3-4 (2010), 381–399. (cited on page 2)

HOCHREITER, S. AND SCHMIDHUBER, J., 1997. Long short-term memory. *Neural Computation*, 9, 8 (1997), 1735–1780. (cited on page 112)

HONG, S.; YOU, T.; KWAK, S.; AND HAN, B., 2015a. Online tracking by learning discriminative saliency map with convolutional neural network. In *International Conference on Machine Learning (ICML)*. (cited on pages 18, 19, and 111)

HONG, Z.; CHEN, Z.; WANG, C.; ET AL., 2015b. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 15, 16, 107, and 111)

HOSANG, J.; BENENSON, R.; AND SCHIELE, B., 2014. How good are detection proposals, really? In *British Machine Vision Conference (BMVC)*. (cited on pages 58, 70, 92, 94, and 104)

HUA, Y.; ALAHARI, K.; AND SCHMID, C., 2015. Online object tracking with proposal selection. In *International Conference on Computer Vision (ICCV)*. (cited on page 112)

HUANG, D.; LUO, L.; WEN, M.; CHEN, Z.; AND ZHANG, C., 2015. Enable scale and aspect ratio adaptability in visual tracking with detection proposals. In *British Machine Vision Conference (BMVC)*. (cited on pages 58, 93, and 112)

HUK, A. Seeing motion: Lecture notes. http://www-psych.stanford.edu/~lera/psych115s/notes/lecture7/. (cited on page 110)

HUNTER, A., 2009. Canoe slalom boat trajectory while negotiating an upstream gate. *Sports Biomechanics*, 8, 2 (2009), 105–113. (cited on pages 3 and 4)

JIA, X.; LU, H.; AND YANG, M. H., 2012. Visual tracking via adaptive structural local sparse appearance model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 18, 33, 58, and 90)

JOULIN, A.; TANG, K. D.; AND LI, F.-F., 2014. Efficient image and video co-localization with frank-wolfe algorithm. In *European Conference on Computer Vision (ECCV)*. (cited on page 9)

KALAL, Z.; MIKOLAJCZYK, K.; AND MATAS, J., 2012. Tracking-Learning-Detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 34, 7 (2012), 1409–1422. (cited on pages 22 and 112)

KALMAN; RUDOLPH; AND EMIL, 1960. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, (1960). (cited on page 64)

KARPATHY, A. AND LI, F., 2015. Deep visual-semantic alignments for generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 112)

KASTURI, R.; GOLDGOF, D.; SOUNDARARAJAN, P.; ET AL., 2009. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 31, 2 (2009), 319–336. (cited on page 20)

KHAN, Z.; BALCH, T.; AND DELLAERT, F., 2004. An mcmc-based particle filter for tracking multiple interacting targets. In *European Conference on Computer Vision (ECCV)*. (cited on pages 109 and 112)

KIM, T.-K. AND CIPOLLA, R., 2008. Mcboost: Multiple classifier boosting for perceptual co-clustering of images and visual features. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on page 36)

KRISTAN, M.; MATAS, J.; LEONARDIS, A.; ET AL., 2015. The visual object tracking VOT2015 challenge results. In *International Conference on Computer Vision Workshops (ICCVW)*. (cited on pages 1, 6, 9, 13, 16, 19, 23, 24, 90, 91, 94, 98, 107, and 110)

KRISTAN, M.; MATAS, J.; LEONARDIS, A.; ET AL., 2016. A novel performance evaluation methodology for single-target trackers. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, , 99 (2016). (cited on pages 14, 23, 24, 25, 27, and 110)

KRISTAN, M.; PFLUGFELDER, R.; LEONARDIS, A.; ET AL., 2014. The Visual Object Tracking VOT2014 challenge results. In *European Conference on Computer Vision Workshops (ECCVW)*. (cited on pages 15, 19, 23, 35, 43, 57, 64, 71, 84, 98, and 107)

KRISTAN, M.; PFLUGFELDER, R.; LEONARDIS, A.; ET AL., 2013. The visual object tracking vot2013 challenge results. In *International Conference on Computer Vision Workshops (ICCVW)*. (cited on pages 1, 7, 15, 23, and 110)

KRIZHEVSKY, A.; SUTSKEVER, I.; AND HINTON, G. E., 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages 19 and 60)

KWAK, S.; CHO, M.; LAPTEV, I.; PONCE, J.; AND SCHMID, C., 2015. Unsupervised object discovery and tracking in video collections. In *International Conference on Computer Vision (ICCV)*. (cited on page 9)

KWON, J.; LEE, K. M.; AND PARK, F., 2009. Visual tracking via geometric particle filtering on the affine group with optimal importance functions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 74 and 83)

LAZEBNIK, S.; SCHMID, C.; AND PONCE, J., 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 40)

LEAL-TAIXE, L.; FENZI, M.; KUZNETSOVA, A.; ROSENHAHN, B.; AND SAVARESE, S., 2014. Learning an image-based motion context for multiple people tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 89)

LEE, D.-Y.; SIM, J.-Y.; AND KIM, C.-S., 2015. Multihypothesis trajectory analysis for robust visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 17)

LEE, Y. J.; KIM, J.; AND GRAUMAN, K., 2011. Key-segments for video object segmentation. In *International Conference on Computer Vision (ICCV)*. (cited on pages 9 and 18)

LI, F.-F.; FERGUS, R.; AND PERONA, P., 2006. One-shot learning of object categories. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 28, 4 (2006), 594–611. (cited on page 19)

LI, H.; LI, Y.; AND PORIKLI, F., 2014. Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In *British Machine Vision Conference (BMVC)*. (cited on page 18)

Li, H.; Shen, C.; and Shi, Q. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 33 and 90)

Li, W.; Zhao, R.; and Wang, X., 2012. Human reidentification with transferred metric learning. In *Asian Conference on Computer Vision (ACCV)*. (cited on pages 7 and 36)

Li, X.; Dick, A.; Shen, C.; van den Hengel, A.; and Wang, H., 2013a. Incremental learning of 3D-DCT compact representations for robust visual tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 35, 4 (2013), 863–881. (cited on page 33)

Li, X.; Dick, A.; Shen, C.; et al., 2013b. Visual tracking with spatio-temporal Dempster-Shafer information fusion. *IEEE Transactions on Image Processing (TIP)*, 22, 8 (2013), 3028–3040. (cited on page 35)

Li, X.; Dick, A.; Wang, H.; Shen, C.; and van den Hengel, A., 2011. Graph mode-based contextual kernels for robust svm tracking. *International Conference on Computer Vision (ICCV)*, (2011). (cited on pages 17, 33, 35, and 38)

Li, X.; Hu, W.; Shen, C.; et al., 2013c. A survey of appearance models in visual object tracking. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4, 4 (2013), 58:1–58:48. (cited on pages 13, 14, 35, and 37)

Li, X.; Shen, C.; Dick, A.; Zhang, Z.; and Zhuang, Y., 2016. Online metric-weighted linear representations for robust visual tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, (2016). (cited on page 36)

Li, Y.; Zhu, J.; and Hoi, S. C. H., 2015. Reliable patch trackers: Robust visual tracking by exploiting reliable patches. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 18 and 111)

Liang, P.; Pang, Y.; Liao, C.; Mei, X.; and Ling, H., 2016. Adaptive objectness for object tracking. *IEEE Signal Processing Letters*, 23, 7 (2016), 949–953. (cited on pages 58 and 93)

Liao, H.-C.; Chen, P.-Y.; Lin, Z.-J.; and Lim, Z.-Y., 2016. Automatic zooming mechanism for capturing object image using high definition fixed camera. In *International Conference on Advanced Communications Technology (ICACT)*. (cited on page 6)

LTDT2014. Long-Term Detection and Tracking. http://www.micc.unifi.it/LTDT2014/. (cited on page 111)

Lu, W.-L.; Ting, J.-A.; Little, J. J.; and Murphy, K. P., 2013. Learning to track and identify players from broadcast sports videos. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 35, 7 (2013), 1704–1716. (cited on pages 4 and 107)

Lucas, B. D. and Kanade, T., 1981. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*. (cited on pages 5, 14, and 73)

Ma, C.; Huang, J.-B.; Yang, X.; and Yang, M.-H., 2015a. Hierarchical convolutional features for visual tracking. In *International Conference on Computer Vision (ICCV)*. (cited on pages 18 and 111)

Ma, C.; Yang, X.; Zhang, C.; and Yang, M. H., 2015b. Long-term correlation tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 111)

Martin, D.; Fowlkes, C.; Tal, D.; and Malik, J., 2001. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision (ICCV)*. (cited on page 19)

Masnadi-Shirazi, H.; Mahadevan, V.; and Vasconcelos, N., 2010. On the design of robust classifiers for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 74)

Matthews, I. and Baker, S., 2004. Active appearance models revisited. *International Journal of Computer Vision (IJCV)*, 60 (2004), 135–164. (cited on page 73)

Matthews, L.; Ishikawa, T.; and Baker, S., 2004. The template update problem. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 26, 6 (2004), 810–815. (cited on pages 14 and 60)

Mei, X. and Ling, H., 2011. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 33, 11 (2011), 2259–2272. (cited on pages 15, 33, and 90)

Milan, A.; Leal-Taixé, L.; Reid, I. D.; Roth, S.; and Schindler, K., 2016. MOT16: A benchmark for multi-object tracking. *CoRR*, (2016). (cited on pages 1, 7, 89, 91, 92, 110, and 112)

Milan, A.; Roth, S.; and Schindler, K., 2014. Continuous energy minimization for multitarget tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 36, 1 (2014), 58–72. (cited on page 89)

Mottaghi, R.; Chen, X.; Liu, X.; et al., 2014. The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 17)

Munkres, J., 1957. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5, 1 (1957), 32–38. (cited on pages 91, 95, and 96)

Nam, H. and Han, B., 2016. Learning multi-domain convolutional neural networks for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 19)

Nawaz, T. and Cavallaro, A., 2013. A protocol for evaluating video trackers under real-world conditions. *IEEE Transactions on Image Processing (TIP)*, 22, 4 (2013), 1354–1361. (cited on page 26)

Ng, J. Y.; Hausknecht, M. J.; Vijayanarasimhan, S.; et al., 2015. Beyond short snippets: Deep networks for video classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 112)

Ošep, A.; Hermans, A.; Engelmann, F.; et al., 2016. Multi-scale object candidates for generic object tracking in street scenes. In *International Conference on Robotics and Automation (ICRA)*. (cited on page 93)

Ozuysal, M.; Fua, P.; and Lepetit, V., 2007. Fast keypoint recognition in ten lines of code. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 73)

Pang, Y. and Ling, H., 2013. Finding the best from the second bests - inhibiting subjective bias in evaluation of visual tracking algorithms. In *International Conference on Computer Vision (ICCV)*. (cited on page 27)

Papazoglou, A. and Ferrari, V., 2013. Fast object segmentation in unconstrained video. In *International Conference on Computer Vision (ICCV)*. (cited on page 112)

Patino, L.; Cane, T.; Vallee, A.; and Ferryman, J., 2016. PETS 2016: Dataset and challenge. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. (cited on pages 1, 6, 89, 91, 98, and 99)

Platt, J. C., 1999. *Advances in Kernel Methods*, chap. Fast Training of Support Vector Machines Using Sequential Minimal Optimization, 185–208. (cited on pages 29, 41, 81, and 82)

Possegger, H.; Mauthner, T.; and Bischof, H., 2015. In defense of color-based model-free tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 15, 17, 35, 55, and 94)

Prisacariu, V. A. and Reid, I. D., 2012. PWP3D: Real-time segmentation and tracking of 3d objects. *International Journal of Computer Vision (IJCV)*, 98, 3 (2012), 335–354. (cited on pages 6 and 8)

PÃl'rez, P.; Hue, C.; Vermaak, J.; and Gangnet, M., 2002. Color-based probabilistic tracking. In *European Conference on Computer Vision (ECCV)*, 661–675. (cited on page 15)

Ren, C.; Prisacariu, V.; Kaehler, O.; Reid, I.; and Murray, D., 2014. 3d tracking of multiple objects with identical appearance using rgb-d input. In *International Conference on 3D Vision (3DV)*. (cited on page 6)

Ren, S.; He, K.; Girshick, R.; and Sun, J., 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages 89, 93, 95, and 104)

Ren, X. and Malik, J., 2007. Tracking as repeated figure/ground segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 18)

Ross, D. A.; Lim, J.; Lin, R.-S.; and Yang, M.-H., 2008. Incremental learning for robust visual tracking. *International Journal of Computer Vision (IJCV)*, 77, 1-3 (2008), 125–141. (cited on pages 14, 33, 58, 82, 83, and 90)

Rossmann, W., 2002. *Lie groups: A introduction through linear groups*. Oxford University Press. (cited on page 79)

Russakovsky, O.; Deng, J.; Su, H.; et al., 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115, 3 (2015), 211–252. (cited on pages 1, 5, 16, 18, 19, 23, and 108)

Saffari, A.; Godec, M.; Pock, T.; Leistner, C.; and Bischof, H., 2010a. Online multi-class LPBoost. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 36 and 74)

Saffari, A.; Leistner, C.; Godec, M.; and Bischof, H., 2010b. Robust multi-view boosting with priors. In *European Conference on Computer Vision (ECCV)*. (cited on pages 58 and 74)

Santner, J.; Leistner, C.; Saffari, A.; Pock, T.; and Bischof, H., 2010. PROST Parallel Robust Online Simple Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 33)

Shi, J. and Tomasi, C., 1994. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 5)

Shitrit, H. B.; Berclaz, J.; Fleuret, F.; and Fua, P., 2014. Multi-commodity network flow for tracking multiple people. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 36, 8 (2014), 1614–1627. (cited on pages 7 and 8)

Smeulders, A. W. M.; Chu, D. M.; Cucchiara, R.; et al., 2014. Visual tracking: An experimental survey. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 36, 7 (2014), 1442–1468. (cited on pages 1, 10, 13, 14, 19, 20, 21, 22, 23, 24, 25, 26, 35, 43, 55, 59, 63, 67, 71, 73, 90, 91, 92, 94, 97, 98, 108, 110, and 111)

SZEGEDY, C.; TOSHEV, A.; AND ERHAN, D., 2013. Deep neural networks for object detection. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on page 19)

TAIGMAN, Y.; YANG, M.; RANZATO, M.; AND WOLF, L., 2014. Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 1)

TEULIÃĹRE, C.; ECK, L.; AND MARCHAND, E., 2011. Chasing a moving target from a flying uav. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. (cited on pages 2, 3, 15, and 107)

TIAN, M.; ZHANG, W.; AND LIU, F., 2007. On-line ensemble svm for robust object tracking. In *Asian Conference on Computer Vision (ACCV)*. (cited on page 35)

TORRALBA, A. AND EFROS, A. A., 2011. Unbiased look at dataset bias. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 89)

TORRALBA, A.; MURPHY, K. P.; AND FREEMAN, W. T., 2007. Sharing visual features for multiclass and multiview object detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 29, 5 (2007), 854–869. (cited on page 36)

TSAI, Y.-H.; YANG, M.-H.; AND BLACK, M. J., 2016. Video segmentation via object flow. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 112)

TSOCHANTARIDIS, I.; JOACHIMS, T.; HOFMANN, T.; AND ALTUN, Y., 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6 (2005), 1453–1484. (cited on pages 27, 28, 30, 39, and 77)

TUZEL, O.; PORIKLI, F.; AND MEER, P., 2006. Region covariance: A fast descriptor for detection and classification. In *European Conference on Computer Vision (ECCV)*. (cited on page 84)

TUZEL, O.; PORIKLI, F.; AND MEER, P., 2008. Learning on Lie groups for invariant detection and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 57, 74, and 75)

VAN DE SANDE, K. E. A.; UIJLINGS, J. R. R.; GEVERS, T.; AND SMEULDERS, A. W. M., 2011. Segmentation as selective search for object recognition. In *International Conference on Computer Vision (ICCV)*. (cited on page 9)

VAN DER MAATEN, L. AND HINTON, G. E., 2008. Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research (JMLR)*, 9 (2008), 2579–2605. (cited on pages 37 and 38)

Vetter, T. and Poggio, T., 1997. Linear object classes and image synthesis from a single example image. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 19 (1997), 733–742. (cited on page 73)

VOT-TIR2016. VOT-TIR2016 challenge. http://www.votchallenge.net/vot2016/. (cited on page 24)

VOT2016. VOT2016 challenge. http://www.votchallenge.net/vot2016/. (cited on page 13)

Wagner, D.; Langlotz, T.; and Schmalstieg, D., 2008. Robust and unobtrusive marker tracking on mobile phones. In *ACM International Symposium on Mixed and Augmented Reality (ISMAR)*. (cited on page 73)

Wang, L.; Ouyang, W.; Wang, X.; and Lu, H., 2015a. Visual tracking with fully convolutional networks. In *International Conference on Computer Vision (ICCV)*. (cited on page 18)

Wang, N.; Shi, J.; Yeung, D.; and Jia, J., 2015b. Understanding and diagnosing visual tracking systems. *International Conference on Computer Vision (ICCV)*, (2015). (cited on pages 13, 14, 55, 59, 60, and 113)

Wang, S.; Lu, H.; Yang, F.; and Yang, M.-H., 2011. Superpixel tracking. In *International Conference on Computer Vision (ICCV)*. (cited on page 18)

Wang, X.; Yang, M.; Zhu, S.; and Lin, Y., 2013. Regionlets for generic object detection. In *International Conference on Computer Vision (ICCV)*. (cited on pages 58, 67, and 92)

Wang, Z.; Crammer, K.; and Vucetic, S., 2010. Multi-class Pegasos on a budget. In *International Conference on Machine Learning (ICML)*. (cited on pages 31, 42, 62, 63, 83, and 97)

Wertheimer, M., 1938. *Laws of organization in perceptual forms*, 71–88. (cited on page 110)

Wu, Y.; Lim, J.; and Yang, M., 2015. Object tracking benchmark. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 37, 9 (2015), 1834–1848. (cited on pages 1, 6, 7, 9, 13, 14, 15, 19, 20, 21, 22, 23, 24, 25, 26, 27, 35, 37, 43, 55, 57, 64, 71, 90, 98, 100, 107, 111, and 113)

Wu, Y.; Lim, J.; and Yang, M.-H., 2013. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 1, 19, 20, 21, 35, 43, 57, 59, 63, 64, 71, 90, 91, 92, 94, 97, 98, 100, and 107)

Xing, J.; Ai, H.; Liu, L.; and Lao, S., 2011. Multiple player tracking in sports video: A dual-mode two-way bayesian inference approach with progressive observation modeling. *IEEE Transactions on Image Processing (TIP)*, 20, 6 (2011), 1652–1667. (cited on pages 2, 4, and 107)

Xing, J.; Gao, J.; Li, B.; Hu, W.; and Yan, S., 2013. Robust object tracking with online multi-lifespan dictionary learning. In *International Conference on Computer Vision (ICCV)*. (cited on pages 82 and 83)

Yan, F.; Christmas, W.; and Kittler, J., 2005. A tennis ball tracking algorithm for automatic annotation of tennis match. In *British Machine Vision Conference (BMVC)*. (cited on page 4)

Yang, F.; Lu, H.; and Yang, M.-H., 2014. Robust visual tracking via multiple kernel boosting with affinity constraints. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 24, 2 (2014), 242–254. (cited on page 33)

Yang, J. and Li, H., 2015. Dense, accurate optical flow estimation with piecewise parametric model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 5 and 6)

Yang, M.; Wu, Y.; and Hua, G., 2009. Context-aware visual tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 31, 7 (2009), 1195–1209. (cited on pages 17, 35, and 92)

Yang, Y. and Sundaramoorthi, G., 2013. Modeling Self-Occlusions in Dynamic Shape and Appearance Tracking. In *International Conference on Computer Vision (ICCV)*. (cited on page 111)

Yilmaz, A.; Javed, O.; and Shah, M., 2006. Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38, 4 (2006). (cited on page 13)

Yoon, J. H.; Lee, C. R.; Yang, M. H.; and Yoon, K., 2016. Online multi-object tracking via structural constraint event aggregation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 89, 91, and 92)

Zhang, J.; Ma, S.; and Sclaroff, S., 2014a. MEEM: Robust tracking via multiple experts using entropy minimization. In *European Conference on Computer Vision (ECCV)*. (cited on pages 17, 57, 62, 64, 90, 98, and 107)

Zhang, L. and van der Maaten, L., 2014. Preserving structure in model-free tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 36, 4 (2014), 756–769. (cited on pages 18, 92, 98, and 100)

Zhang, T.; Jia, K.; Xu, C.; Ma, Y.; and Ahuja, N., 2014b. Partial occlusion handling for visual tracking via robust part matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 111)

Zhong, W.; Lu, H.; and Yang, M.-H., 2012. Robust object tracking via sparsity-based collaborative model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 15, 58, 82, 83, and 84)

Zhu, G.; Ming, Y.; and Li, H., 2013. Object cut as minimum ratio cycle in a superpixel boundary graph. In *International Conference on Digital Image Computing: Techniques and Applications (DICTA)*.

Zhu, G.; Ming, Y.; and Li, H., 2014. Object category detection by incorporating mid-level grouping cues. In *International Conference on Image Processing (ICIP)*.

Zhu, G.; Porikli, F.; and Li, H., 2016a. Beyond local search: Tracking objects everywhere with instance-specific proposals. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Spotlight*. (cited on pages 10, 93, 94, 96, 107, 109, 110, and 112)

Zhu, G.; Porikli, F.; and Li, H., 2016b. Model-free multiple object tracking with shared proposals. In *Asian Conference on Computer Vision (ACCV)*. (cited on pages 10 and 110)

Zhu, G.; Porikli, F.; and Li, H., 2016c. Robust visual tracking with deep convolutional neural network based object proposals on PETS. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. (cited on pages 10, 18, 107, 109, 110, and 111)

Zhu, G.; Porikli, F.; and Li, H., 2017. Not all negatives are equal: Learning to track with multiple background clusters. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, (2017). (cited on pages 9, 13, and 108)

Zhu, G.; Porikli, F.; Ming, Y.; and Li, H., 2015. Lie-Struck: Affine tracking on Lie groups using structured SVM. In *IEEE Winter conference on Applications of Computer Vision (WACV)*. (cited on pages 10 and 109)

Zitnick, C. L. and Dollár, P., 2014. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision (ECCV)*. (cited on pages 9, 56, 58, 60, 61, 62, 69, 70, 92, 94, 95, 104, 109, 110, and 112)