UNIVERSITY OF NOTTINGHAM

Part-based Tracking with Cascaded Regression of Neighbours

by

Xiaomeng Wang

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Faculty of Science
School of Computer Science

March 2018

Declaration of Authorship

I, Xiaomeng Wang, declare that this thesis titled, 'Part-based Tracking with Incrementally Learned Cascades' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ________________________________

Date: ________________________________

“The intuitive mind is a sacred gift and the rational mind is a faithful servant.”

Albert Einstein

UNIVERSITY OF NOTTINGHAM

Abstract

Faculty of Science
School of Computer Science

Doctor of Philosophy

by Xiaomeng Wang

Visual tracking aims to detect the location of a possibly moving target by extracting local appearance features and matching them between consecutive images to obtain accurate estimates of target location. Tracking of generic objects is one of the most active topics in computer vision. Despite the large body of work addressing this problem, robust visual tracking of generic objects is still a challenging problem, as the performance of a visual tracking algorithm is affected by many factors, such as non-rigid object deformation, partial or full occlusion of the target, illumination variation, scale variation, etc. Especially, many objects in the real world have a complex appearance and articulated structure. The combination of rigid motion and non-rigid object deformation results in complex appearance changes, making general object tracking a particularly hard problem.

Recently, part-based trackers are preferred in tracking with occlusion and non-rigid deformation because part-based models, which represent the target as a connected set of components, each describing a section of the object, can provide more flexible and robust object appearance models. However, there are four main problems with current part-based trackers: 1) current part-based trackers rely on a response map estimating the likelihood that any given location in an image represents the target (part); 2) the spatial information utilised by current part-based models is limited and inflexible; 3) there is no way of jointly learning shape and appearance for current part-based trackers; 4) a more complex motion model is required, with parts' motion having separate factors.

To address these four problems, this thesis proposes a novel approach to part-based tracking by replacing local matching of an appearance model by direct prediction of the displacement between local image patches and part locations. This thesis proposes to use cascaded regression (SDM) with incremental learning on deeply learned features to track generic objects without any prior knowledge of an object's structure or appearance. This thesis exploits the spatial constraints between individual parts and those between parts and the object as a whole by implicitly learning the shape and deformation parameters of the object in an online fashion. A multiple temporal scale motion model is integrated to initialise the cascaded regression search close to the target and to allow it to cope with occlusions. Experimental results clearly demonstrate the value of the method's components, and comparison with the state-of-the-art techniques in the CVPR 2013 Visual Tracker Benchmark shows that the proposed TRIC-track tracker ranks first on the full dataset.

To address the problems of low efficiency and limited samples in SDM in TRIC-track, this thesis introduces Continuous Regression to model-free visual tracking. It is found that the Taylor expansion is not able to accurately approximate image features of sample space with a high variance in visual tracking. This problem is alleviated by introducing

Locally Continuous Regression strategy, proposed in this thesis. It unifies sampling-based regression with Continuous Regression in an efficient manner by running Continuous Regression on a few sample locations spread around the target, and relating those sampled locations to each other. Locally Continuous Regression is integrated into the main framework of TRIC-track and shows six times computational cost improvement without sacrificing the performance, compared to its sampling-based counterpart.

Acknowledgements

First of all, I would like to express my sincerest gratitude to my supervisors Prof. Tony Pridmore and Dr Michel Valstar. Thank you very much for your guidance and support throughout last four years. You have always provided enlightening discussion and encouragement whenever I was struggling during my PhD study. You told me to slow down and concentrate on my goal when I blindly went too far in a certain direction. I have benefited greatly from your expertise, wisdom and professional experiences, and this will be a treasure for my whole life. In addition to my research, you have helped me as much as you can in many other aspects, such as my scholarship application, being teaching assistant and the job interview. I always feel so lucky and grateful for having you two as my supervisors.

I am very grateful to Dr Brais Martinez. You gave me a lot of advice on my research and my personal development. We had many valuable discussions during my second and third year, which made important contributions to the development of my paper. You taught me to be patient, which is a key thing I learnt during my PhD. You also selflessly shared your knowledge with me in all aspects and you are a really good mentor for me. I would like also thank Kike. You were always available for discussion whenever I met problems. Thank you very much for your assistance and patience in helping me with my last chapter. I am really appreciated that you taught me to listen to the opinions of others.

Many thanks go to all people in Computer Vision Laboratory. All of you are so friendly and helpful. Everyone is collaborative and would love to help each other whatever problems they have. I have always benefited from your valuable experience. Especially, I would like to thank my office mates and friends. Tuan, Haris and Stephan, thank you very much for giving a lot of advice when I started my PhD. Without your help, I could not have got used to a very different life so quickly and started my research so smoothly. I am also grateful for your assistance with your expertise in visual tracking. Amy, Jim and Timur, thank you for organising many activities for our boring life and bringing a lot of fun to our office. Thank you for creating such a friendly and pleasant atmosphere in our office.

Special thanks go to my family, my parents and my sister. You and your characteristics influenced me so much and took me this far. Your encouragement and unconditional support gave me the strength to overcome any problem. I am particularly thankful to Bo for your company and support during this long journey. You helped me through the hardest moments and gave me a lot of joy.

This research would not have been possible without the financial support from the University of Nottingham Vice-Chancellors Scholarship and China Scholarship Council.

Contents

List of Figures

List of Tables

List of Experiments

Abbreviations

A-BHMC	**A**daptive **B**asin **H**opping **M**onte **C**arlo
B-A-BHMC	**B**oundingbox **A**daptive **B**asin **H**opping **M**onte **C**arlo
B-MCMC	**B**oundingbox **M**arkov **C**hain **M**onte **C**arlo
CNN	**C**onvolutional **N**eural **N**etworks
CR	**C**ontinuous **R**egression
DDP	**D**irect **D**isplacement **P**rediction
LCR	**L**ocally **C**ontinuous **R**egression
LC-TRIC	**T**racking by **L**ocally **C**ontinuous **R**egression with **I**ncrementally **L**earned **C**ascades
SDM	**S**upervised **D**escent **M**ethod
SF-A-BHMC	**S**tick **F**igure **A**daptive **B**asin **H**opping **M**onte **C**arlo
SF-MCMC	**S**tick **F**igure **M**arkov **C**hain **M**onte **C**arlo
SR	**S**ampling-based **R**egression
TRIC	**T**racking by **R**egression with **I**ncrementally **L**earned **C**ascades

To my beloved parents, my lovely sister and my dearest fiance.

Chapter 1

Introduction

Tracking of generic objects is one of the most active topics in computer vision. The visual object tracking (VOT) challenge, which has been held from 2013 to 2016, has received results of many trackers evaluated on the benchmark every year (Kristan et al., 2013, 2014, 2015, 2016b). For example, VOT2015 presents results of 62 state-of-the-art trackers. It aims to detect the location of a possibly moving target by extracting local appearance features and matching them between consecutive images to obtain accurate estimates of the target location, often helped by similar estimates of the object's motion. The parameters describing a target object can vary, but often include position, size, orientation and velocity. Tracking information can subsequently be used to reason about the target's behaviour or complete other tasks that require knowledge of the object's state. Visual tracking technology is related to image processing, pattern recognition, artificial intelligence, automatic control and many other areas (Maggio & Cavallaro, 2011).

It has extensive applications in areas such as visual (robot) navigation (Davison, Reid, Molton, & Stasse, 2007), surveillance (Hampapur et al., 2005; Anjum & Cavallaro, 2008; Chen, Lin, & Chen, 2011), traffic monitoring (J. Zhou, Gao, & Zhang, 2007; Morris & Trivedi, 2008), medical imaging (Notomi et al., 2005; Mountney & Yang, 2008), human-machine interfaces (K. S. Huang & Trivedi, 2003; Poole & Ball, 2005; Hayashi, Agamanolis, & Karau, 2008), etc.

Visual tracking methods are extensively applied in the area of robotics. Robots use the information obtained from cameras mounted on their bodies to interact with or navigate in the environment. Visual tracking technology is the basis for human-robot interaction via gesture recognition, environment exploration and mapping. 3D localisation information can be generated by tracking the position of salient image features such as corners and edges (see Figure 1.1). Information of 3D position can be used to construct the structure of surrounding environment.

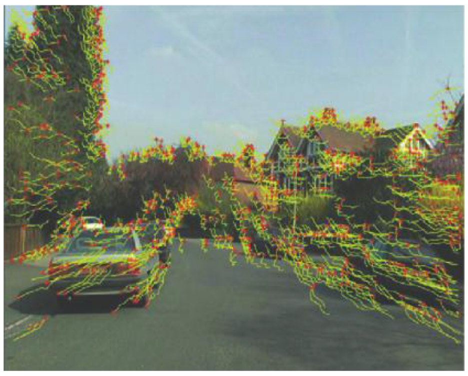


Figure 1.1: An example of a camera tracker that uses the information obtained by tracking image patches. The figure is from (Maggio & Cavallaro, 2011).

In automated video surveillance systems, tracking can work as a pre-processing stage which facilitates subsequent behaviour classification algorithms. It can also work as a forensic tool in post-processing of videos. The IBM Smart Surveillance System (S3) is shown in Figure 1.2 as an example of video surveillance systems (Maggio & Cavallaro, 2011).

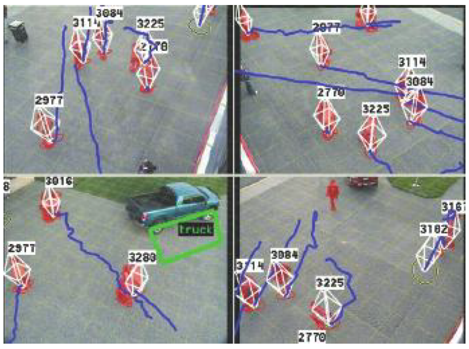


Figure 1.2: An example of object tracking in surveillance applications. The figure is from (Maggio & Cavallaro, 2011).

Visual tracking has been extensively applied to medical systems to aid the diagnosis and to speed up the operation (Maggio & Cavallaro, 2011). Tracking human motion can be used in analysing the gait of a patient to assess the condition of their joints (see Figure 1.3(a)) and in analysing the motion of athletes to improve athletes' performances. Visual tracking can also estimate the position of specific soft tissues or of instruments such as needles and bronchoscopes during surgery. In biological research, one can analyse the influence of particular drugs through tracking the motion of non-human organisms (see Figure 1.3(b)).

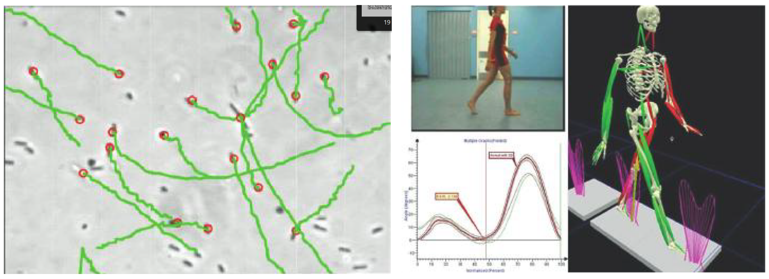

(a) (b)

FIGURE 1.3: Examples of visual tracking for medical and sport analysis applications. (a) Automated tracking of the position of Escherichia coli bacteria. (b) Motion capture is used to analyse the gait of a patient. This figure is from (Maggio & Cavallaro, 2011).

In video conference, webcams can be shipped with tracking software to localise and follow the face of a user. In lecture room, PTZ cameras can be combined with visual tracking technology to follow the position of a lecturer. The trajectory information in real-time is used to guide the pan, tilt and zoom parameters of the camera. Visual tracking is also changing the way how we control machines. Players used to press a button on the controller to play games. Nowadays, users can perform more intuitive gestures in front of a camera to play interactive games.

FIGURE 1.4: The example of visual tracking in an interactive game. The camera uses tracking technology to capture the gestures that are converted into gaming actions. This figure is from (Maggio & Cavallaro, 2011).

During recent decades, visual tracking has been deeply studied and there are many algorithms proposed to resolve this problem. Many of them have made great contributions to the field of visual tracking (Ross, Lim, Lin, & Yang, 2008; Mei & Ling, 2009; Kalal, Matas, & Mikolajczyk, 2010; Babenko, Yang, & Belongie, 2011; Hare, Saffari, & Torr, 2011). Despite the large body of computer vision work addressing this problem, robust visual tracking of generic objects is still a challenging problem (H. Yang, Shao, Zheng, Wang, & Song, 2011; Kristan et al., 2015, 2016a). The performance of a visual tracking algorithm is affected by many factors (Yilmaz et al., 2006), such as non-rigid object deformation, partial or full occlusion of the target, illumination variation, scale variation, viewpoint change, motion blur, background clutter, etc. Especially, many objects in the real world have a complex appearance and articulated structure. The combination of rigid motion and non-rigid object deformation results in complex appearance changes, making general object tracking a particularly hard problem.

According to Yilmaz, a tracking system contains three main components (Yilmaz et al., 2006): (1) an appearance model which describes image information of the target and gives the likelihood that the target is located at a particular location, (2) a motion model to transmit previous target state to current processing time step, and (3) a search strategy to find the target location from its hypotheses.

Traditional tracking methods (Ross et al., 2008; Mei & Ling, 2009; Babenko, Yang, & Belongie, 2009; Kwon & Lee, 2010; Kalal, Mikolajczyk, & Matas, 2012) tend to define a tracking target using a bounding-box and then extract features within the bounding box to represent the target's appearance (template). The target location is obtained through searching the image and finding the image information which is the most similar to the template. These methods model the holistic appearance of the target, and they lose local spatial information of targets which leads to a lack of flexibility when handling non-rigid objects. Recently, part-based trackers are preferred in tracking with occlusion and non-rigid deformation because part-based models (Adam et al., 2006; M. Yang et al., 2007; Kwon & Lee, 2009; L. Zhang & van der Maaten, 2013), which represent the target as a connected set of components, each describing a section of the object, can provide more flexible and robust object appearance models. The appearance of a part is usually more consistent and specific than the appearance models defined by a simple bounding box around the whole. In addition, parts can exhibit their own modes of variation over time in appearance as well as motion. For example, the legs of a walking person will exhibit their own motion patterns. Besides, part-based tracking methods are capable of handling (self)-occlusions which make object detection and tracking much harder for traditional methods. When the target is partially hidden from view by other objects, tracking can proceed using the visible parts. Part-based models are flexible in describing the structure of the target and how it changes over time, which facilitates tracking objects with articulated motion.

1.1 Observations

Part-based models have been successfully applied to object detection and recognition (L. Zhu, Chen, Yuille, & Freeman, 2010; Felzenszwalb, Girshick, McAllester, & Ramanan, 2010; Lin, Hua, & Davis, 2009; Amit & Trouvé, 2007; Schnitzspan, Roth, & Schiele, 2010; Pedersoli, Vedaldi, & Gonzlez, 2011). However, it is hard to apply the part-based models proposed for object detection and recognition to visual tracking. Object detection and recognition usually work on the image information in a single frame while tracking needs to update appearance models online. In addition, explicitly tracking each part would require individual training and initialisation for each part (Yao et al., 2013). Thus, applying part-based models directly to visual tracking will increase the complexity of the appearance model and exponentially increase time cost. Recently, some researchers have proposed several part-based tracking methods. However, there are four main problems with current part-based trackers, which are explained as follows:

- Current part-based trackers rely on a response map estimating the likelihood that any given location in an image represents the target (part) (Adam et al., 2006; Shahed Nejhum et al., 2008; Kwon & Lee, 2009; L. Zhang & van der Maaten, 2013; T. Zhang et al., 2014; Liu et al., 2015). While approaches to the computation of this fitness vary from simple template matching to complex machine-learning-based methods, all assign a single scalar value to a queried location. Tracking then becomes a problem of determining what area(s) to search in, often guided by some motion model(s), how to construct an appearance model that can capture changes in the object's image properties, and how to deal with local optima.

  While a template likelihood strategy may appear to be the logical solution to the part-based visual tracking problem, its view of the image as a set of independent, identically distributed target locations introduces a number of inherent drawbacks. Firstly, incorrect optima occur when changes in the target's appearance make it look less like the original template than some background image patch(es), or when there are multiple identical objects (parts) in the scene. Secondly, for the template strategy to work, an entire region of interest needs to be searched. This can be computationally intensive, but methods (e.g. gradient descent) introduced to speed up the search can also converge to local optima. Thirdly, in a template likelihood strategy local information (be it part of the object foreground or the background) only serves to inform the tracker whether it is the target or not. Once the optimal location has been sampled, sampling additional non-optimal local patches does not improve the confidence that this is indeed the target location. It can only make matters worse: increasing the size of the search area only increases the likelihood of encountering an incorrect optimum.

- In recent years, some researchers have attempted to apply part-based or fragment-based appearance models to capture the spatial information of the object (Adam et al., 2006; Shahed Nejhum et al., 2008) in tracking. However, the spatial information utilised with their models is limited and inflexible. Thus, they still cannot handle non-rigid object tracking situations which have severe or complex movements. An extra component, shape, is necessary for general object tracking. This would effectively extend Yilmaz's description of tracking systems to four main components, when speaking of part-based tracking.
- Another major issue is that there is no way of jointly learning shape and appearance for current part-based trackers. Template likelihood approaches to part-based tracking cannot directly use the appearance of one part to determine the location of another. Although an explicit shape model can be learned to constrain the expected relative positions of parts, such a shape fitting step is effectively bolted-on on top of independent, individual local part trackers.
- Parts can have separate motions over time. For example, the legs of a walking person will exhibit their own motion patterns. This situation is not considered in current part-based trackers. Thus, more complex motion models are required, with parts' motion having separate factors.

1.2 Contribution

The main objective of this PhD project is to develop a part-based general tracking method, which is able to correctly track an object of interest without the need to train a specific model offline, i.e., we seek an online method. In light of the work in (Adam et al., 2006; Kwon & Lee, 2009; Martinez et al., 2013; Xiong & De la Torre, 2013), this PhD project builds a robust part-based appearance model for visual tracking. It is comprised of a few local parts and flexible and inter-constrained topological relationships among the parts. The part contains the local information of tracked target appearance. The topological relationships can accurately represent the spatial distribution of the parts. Thus, this model can keep the spatial information by using the topological relationships among them. In conclusion, this thesis makes the following contributions:

- The thesis exploits the potential benefits of the part-based appearance model and verifies the advantages of the part-based model over the global appearance model. Also, the thesis explores the structure of the parts to seek a structure which is able to model the geometric relationship between parts.

- The thesis proposes a visual tracking method in which the local fitness-based approach is replaced by direct displacement-based tracking. Specifically, the tracker predicts the two-dimensional displacement vector between the centre of a sampled image patch and the target (part) location using regressors (see Figure 1.5). In doing so, local patches contribute to the solution by directly 'voting' for the target (part) location.

- This thesis implicitly models the shape of a target using local evidence from multiple parts and global information from a bounding box part. While template-based approaches need to model part appearance and shape fitness separately, this direct displacement prediction by regression tracker implicitly learns the shape and possible deformations of an object. It does so by tracking each part using not only the local evidence for that part, but also evidence provided by neighbouring parts and the object as a whole.

- The thesis adapts the framework of the supervised descent method (SDM) (Xiong & De la Torre, 2013) to the problem of online tracking of generic objects. While SDM has been used for what is essentially structured object detection, it has never been used for online model-free tracking. The key difference between detection of a known object and generic object tracking is that the appearance and structure models of the former can be learned offline on potentially hundreds of thousands of images, while the models for the latter must be initialised on a single frame. It is shown that it is possible to learn the cascade models on-the-fly without strong supervision.

- This thesis introduces Continuous Regression (Sánchez-Lozano, Tzimiropoulos, Martinez, De la Torre, & Valstar, 2016) to replace the sampling-based regression, i.e. the supervised descent method (SDM), in model-free tracking. With Continuous Regression, the shape displacement is regarded as a continuous variable, the feature space is approximated by its first-order Taylor expansion. Only the feature in the ground truth target location needs to be sampled. However, it is then observed that the Taylor expansion is only a good feature approximation in a relatively small region around the target. This region is too small to enable tracking. Therefore, this thesis proposes Locally Continuous Regression, which unifies sampling-based regression with continuous regression by repeating Continuous Regression in a few regions around the target in an efficient manner. It shows six times speed improvement without sacrificing performance of the tracker.

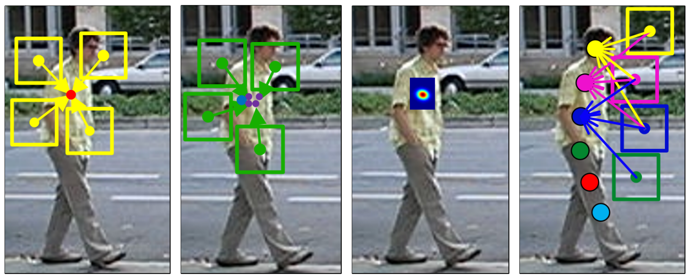


FIGURE 1.5: (a) Training a direct displacement regressor with four examples. (b) Testing a regressor. Four test patches sampled around the initial location (blue dot) provide predictions (purple dots). (c) Evidence aggregation map. (d) Location-based initialisation and implicit shape model.

1.3 Thesis Organisation

The rest of the thesis is organised as follows. The related literature review is introduced in Chapter 2, in which the related concepts and main challenges of visual tracking are introduced in Section 2.1; the main components of visual tracking are described in Section 2.2, Section 2.3 and Section 2.4; tracking methods with structured part-based models are presented in Section 2.5. Preliminary experiments exploring the ability of part-based tracking mechanism are presented in Chapter 3. A stick figure initialisation is introduced in Section 3.1 and followed by experiments assessing its effectiveness. The direct displacement prediction method and point-based initialisation are explained in Section 3.2 and followed by experimental research on the regression technique which is able to directly predict the point's location. The simple direct displacement prediction tracker is introduced and evaluated in Section 3.3. The proposed part-based tracking method, TRIC-track, is introduced in Chapter 4. The overview of the TRIC-track is presented in Section 4.1. The limitations of previous methods and their relation to TRIC-track are explained in Section 4.2. The technical details of different algorithm components of TRIC-track are explicitly described in Section 4.3. The TRIC-track is evaluated in Section 4.4. The proposed tracking framework is further improved by replacing sampling-based regression with proposed Locally Continuous Regression, which is explained in Chapter 5. Section 5.1 presents the motivation of Continuous Regression (CR) and its limitations. Section 5.2 explains the proposed Locally Continuous Regression which overcomes the problem of CR. Section 5.3 explicitly explains the integration of Locally Continuous Regression into the main framework of TRIC-track. The proposed tracker is evaluated in Section 5.4. Chapter 6 concludes the thesis and proposes the possible future work.

1.4 Related Publications

Based on the work of the thesis, I have published four papers as follows:

1. Xiaomeng Wang, Michel Valstar, Brais Martinez, Muhammad Haris Khan, Tony Pridmore, “TRIC-track: Tracking by Regression with Incrementally Learned Cascades”, 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 4337-4345.

2. Matej Kristan, Xiaomeng Wang, Michel Valstar, Brais Martinez, Muhammad Haris Khan, Tony Pridmore et al., “The Visual Object Tracking VOT2015 Challenge Results”, 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, 2015, pp. 564-586.

3. Matej Kristan, Xiaomeng Wang, Michel Valstar, Brais Martinez, Muhammad Haris Khan, Tony Pridmore et al., “The Visual Object Tracking VOT2016 Challenge Results”, 2016 European Conference on Computer Vision Workshop (ECCVW), Amsterdam, 2016, pp. 777-823.

4. Xiaomeng Wang, Michel Valstar, Wout Elferink, Brais Martinez, Tony Pridmore, “Part-based Tracking with Cascaded Regression of Neighbours Using Learned Features”, (submitted to TPAMI).

Chapter 2

Related Work

2.1 The Visual Tracking Problem

With the development of high-powered computers, the emergence and extensive application of video cameras and the demand for artificial intelligence, visual tracking has attracted a great deal of interest from researchers all over the world. It is the fundamental problem in video analysis, which includes detection of the target of interest, tracking the target, identifying its state frame by frame, analysis of its state and understanding its behaviour. Thus, the visual tracking problem is the basis of many tasks (Maggio & Cavallaro, 2011), such as vehicle navigation (Davison et al., 2007), intelligent traffic monitoring (J. Zhou et al., 2007; Morris & Trivedi, 2008), automated surveillance (Anjum & Cavallaro, 2008; Hampapur et al., 2005; Chen et al., 2011), human-computer interaction (K. S. Huang & Trivedi, 2003; Poole & Ball, 2005; Hayashi et al., 2008), motion-based recognition (Saleemi, Shafique, & Shah, 2009; Monti & Regazzoni, 2010; Xu et al., 2012), and a variety of medical applications (Notomi et al., 2005; Mountney & Yang, 2008).

Given a target state in the first frame of a video sequence, either automatically or manually, a visual tracking problem is usually to identify the target state in the subsequent frames by estimation. The target state may include location, size, orientation, scale, velocity, shape and so on. Although the video sequence could be the combination of the output of multiple cameras, the scope of this dissertation will be limited to scenarios captured by a single camera.

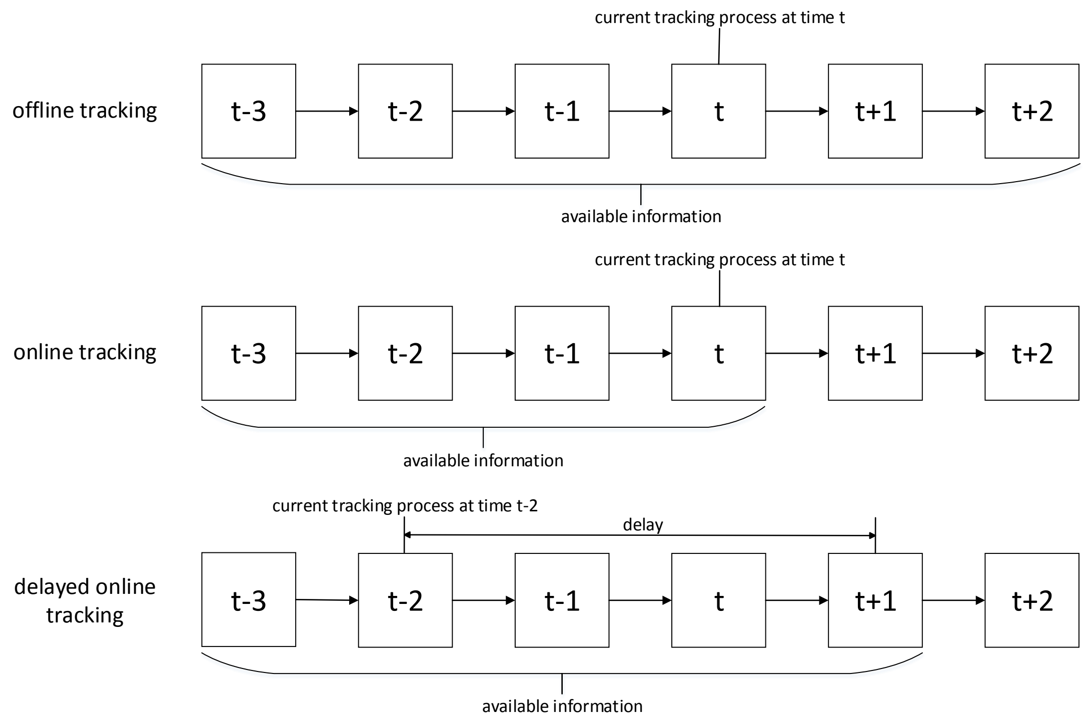


FIGURE 2.1: The differences of available information for offline, online and delayed online tracking. For example, the current tracking process is at time t for offline tracking and online tracking. The available information for offline tracking includes the whole image sequence while the available information for online tracking is all the information up to current process time t. For delayed online tracking, although current tracking process is at time t-2, the tracker has access to the information up to time t+1 obtained by adding the time delay to current process time t-2.

2.1.1 Online and Offline Tracking

Tracking approaches can be divided into online and offline tracking. The online methods are only be able to process the image information of the current and the tracked frames of a video sequence to estimate the current target state, and the tracking process is sequential; while offline methods have the access of all the image information of a video sequence and the tracking process is usually non-sequential. Please note that real-time tracking is one specific class of online tracking methods which is able to process available image information to estimate the target state before the next frame arrives.

Offline methods are not able to perform tracking in real time based on their nature. For interactive applications and those which have a requirement for high processing speed, real-time tracking would be necessary. For applications where the videos are collected for later processing, offline methods are more preferable for the reason that all the video information is available to offline methods and the additional information, future data, can be utilised to improve the robustness of tracking. By allowing a delay of a short time between image acquisition and estimation of target state, online tracking methods can have access to a small number of future frames which is similar to the property of offline methods to a degree. The differences in available image information among online, offline and delayed online tracking methods are illustrated in Figure 2.1.

2.1.2 The General Framework of Tracking

The general framework for visual tracking is to represent the appearance of a tracked target in the given frame, usually referred to as the appearance model, then search from the target candidate space given the previous target states. A similarity measure is used to determine the candidate which matches the appearance model most as the current target state. Thus, there are three main components included in a tracking framework, appearance model, motion model and search strategy.

The appearance model usually takes two roles. One is the target representation, which emphasises the construction of a robust target descriptor. The other is statistical modelling, applying statistical learning strategies, to establish a mathematical model to predict how likely the target is located at a specific position based on the local image information. The motion model is to propagate the target state over time or provide prior probability that the target is located at a specific position, and is usually based on the tracked states of the target. The search strategies decide the way of finding the optimal target hypothesis, and can be roughly split into deterministic and stochastic approaches (S. K. Zhou, Chellappa, & Moghaddam, 2004).

2.1.3 Some Challenges

Visual tracking remains a challenging problem because it is influenced by a number of factors. Firstly, the target appearance can experience significant variations due to photometric factors such as illumination change. Secondly, geometric changes, such as pose variation, non-rigid deformation and scale variation, can lead to complex variations of target appearance as well. Other factors include partial or full occlusion, motion blur, non-linear motion, in-plane rotation and out-of-plane rotation, background clutter, low resolution, etc.

Nowadays, tracking algorithms mainly work on real world videos which have complex conditions for tracking. It's natural that a real world scene has several challenging factors at the same time. For example, in a basketball match scene, a basketball player would display articulated deformation frequently and it's very likely that the player will be fully or partially occluded by other players. There are many other players, who are in the same team of the tracked player, wearing the similar basketball shirt. They are background distractors when tracking a specific basketball player. The player would probably perform out-of-plane rotation during the match. In a surveillance video, usually in low resolution, a person walks across the street. The camera is fixed for surveillance, thus the person would show scale variation in his or her appearance. Again, a person is articulated and would display non-rigid deformation. The light changes in the street may lead to illumination variation in the person's appearance. In a car driving scene, with the camera monitoring the car, the car moves fast which would produce motion blur. The car may go up and down because of the road condition, leading to in-plane rotation. Other cars or bridges over the road would occlude the tracked car either fully or partially. The appearance of the car could display scale variation when the camera is fixed. It is obvious that the real world videos always have a combination of many challenging conditions for visual tracking problem.

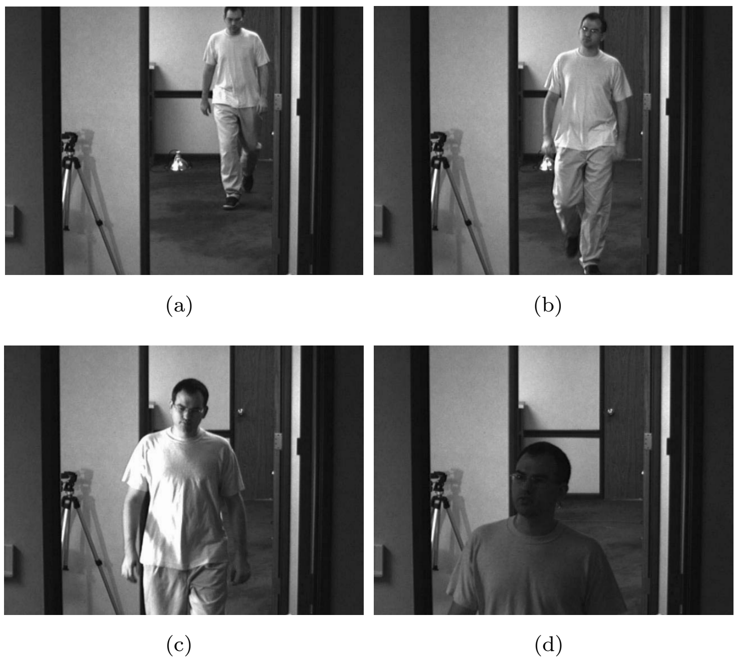


Figure 2.2: The influence on a target's appearance due to illumination variations. (a) The target is diffusely lit from above. (b) The target passes through the shadow. (c) The target is strongly lit from the side. (d) The target is in a darker region. (Kale & Jaynes, 2006)

.

Illumination variation is one of the most common problems in visual tracking. The object's colour is frequently affected by illumination changes in the real-world because of the properties of the incident light, e.g. colour temperature and intensity. For example, a target under sunlight may look really different than it does under strip lighting or moonlight. This brings problems to tracking as colour is the most common feature used when representing a target's appearance. In addition to fixed light sources, there are some moving light sources, such as car headlights. Relative movement between the light sources and the target can change the direction of illumination and reflection. An example of the influence of illumination changes on a target's appearance (Kale & Jaynes, 2006) is shown in Figure 2.2.

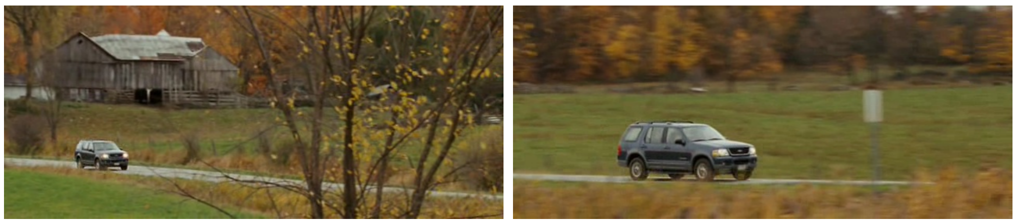

(a) (b)

FIGURE 2.3: The influence on a target's appearance due to scale variations. (a) The car looks smaller when it's far away from the camera. (b) The car looks bigger when it's close to the camera. These images are from (Wu et al., 2013).

Scale variation is another common reason for poor tracking performance. Changes in relative distance between the camera and the target result in scale variation of the target. An example of how scale change affects the appearance of a target is shown in Figure 2.3.

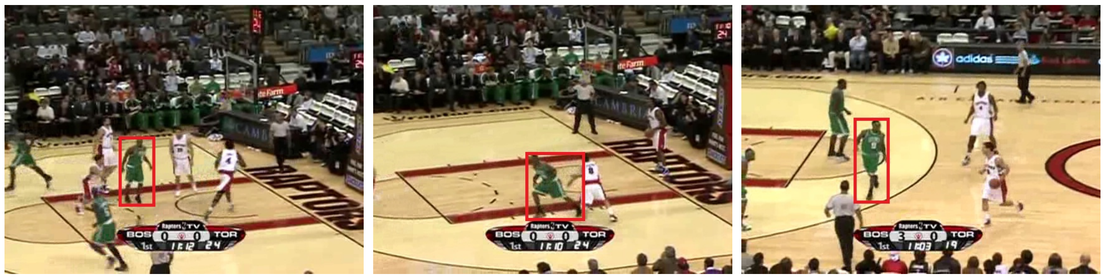

(a) (b) (c)

FIGURE 2.4: The influence on a target's appearance due to non-rigid deformations. The shape of the basketball player changes significantly because of the non-rigid deformation. These images are from (Wu et al., 2013)

.

Non-rigid deformation significantly affects the appearance of the target. Unlike rigid objects, such as cars, non-rigid objects, such as humans and animals, are prone to display complex shape variations, which introduces great difficulties to the modelling of target appearance, especially for traditional bounding box representations of a target. It is common to define a target using a bounding box, which means the image within the bounding box is the target while the outside area is the background. A non-rigid object is often articulated as well. The rigid bounding box representation lacks the flexibility of modelling the non-rigid deformation of the target within it. Even for part-based representations of the target, the modelling of the non-rigid deformation, such as the relative positions of the parts, remains challenging. As an example, Figure 2.4 shows how non-rigid deformation affects the target appearance.

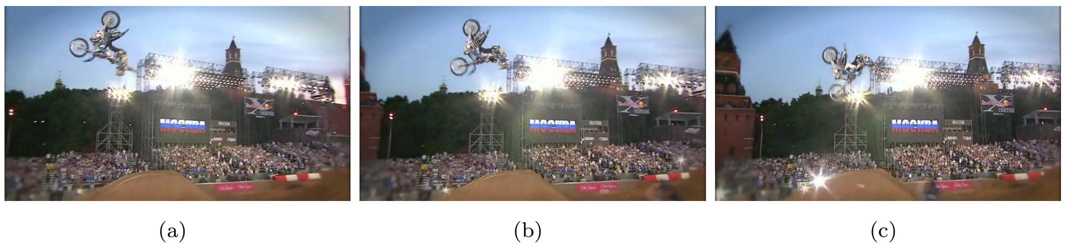


FIGURE 2.5: The example of in-plane rotations of a target. These images are from (Wu et al., 2013)

.

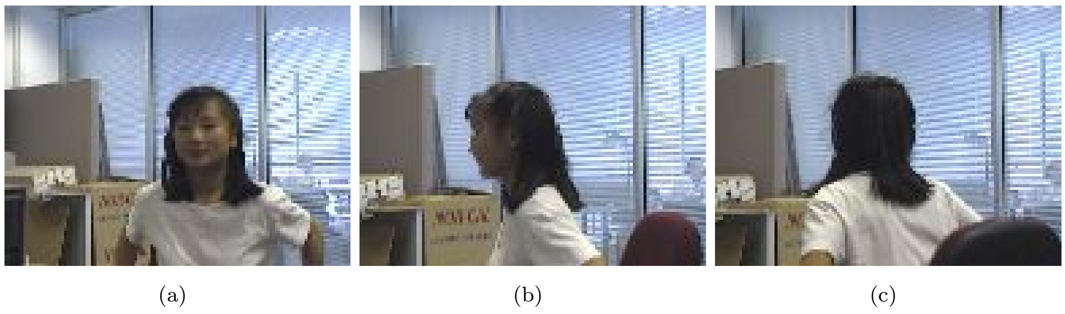


FIGURE 2.6: The example of out-plane rotations of a target. These images are from (Wu et al., 2013)

.

In-plane rotation and out-of-plane rotation are another two major reasons of complex variations of the target appearance. The plane refers to the plane of the image. In-plane rotation means that the rotation of a target happens in the image plane while the out-of-plane rotation of a target occurs across the image plane. In-plane rotation is relatively easy to solve because the target is still visible and can be matched by an appearance template processed with the same rotation transformation. Out-of-plane rotation makes tracking hard because the original appearance of the target can be partially or totally invisible. Figure 2.5 and Figure 2.6 provide examples of in-plane and out-of-plane rotations of a target.

A common factor which affects the appearance significantly is occlusion, which includes self-occlusion and inter-occlusion. Self-occlusion is that a target can be occluded by a part of itself, usually partially, due to articulated deformation. Inter-occlusion is that a target can be partially or fully occluded by other moving or stationary object(s). Thus,

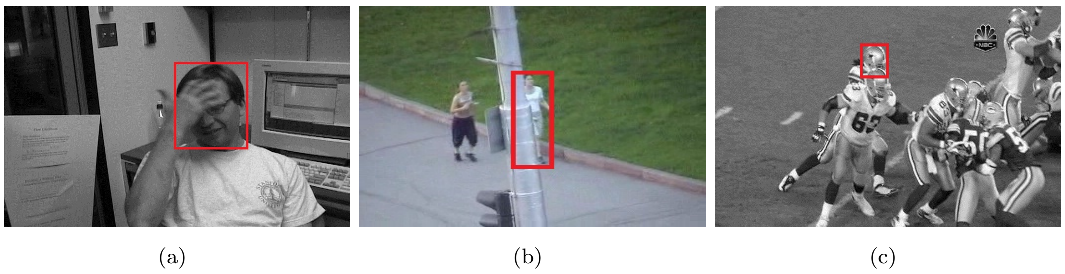


FIGURE 2.7: The example of occasions when a target is occluded. (a) A face is occluded due to the movement of the hand. (b) A running woman is occluded by a lamppost. (c) A football player is occluded by another player. These images are from (Wu et al., 2013)

.

the target can be fully or partially invisible for a period of time. The abilities to use the available target information when partial occlusion happens and to recover tracking when full occlusion ends are critical for a robust tracker. Figure 2.7 illustrates how a target could be occluded by itself or another object.

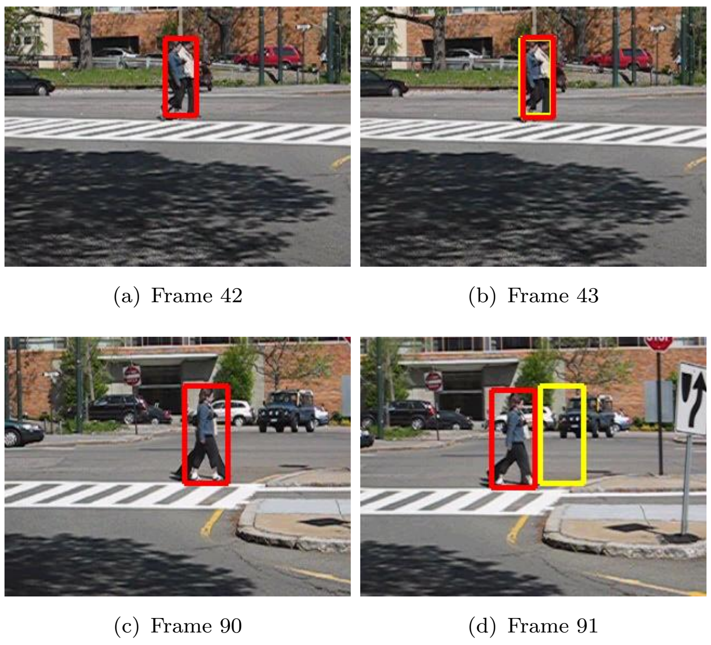


FIGURE 2.8: The example of the smooth and abrupt motion variations of a target. (a) Frame 42. (b) A smooth motion happens between frame 42 and frame 43. (c) Frame 90. (d) An abrupt motion happens between frame 90 and frame 91. The yellow rectangle shows the target location of the previous frame while the red one shows the current target location. These images are from (Wu et al., 2013)

.

A target's movements in a scene can be complex so that its motion is an important component that should be considered in tracking. Basically, the target can move smoothly or abruptly. Smooth movement enables a tracker to search the target state from a local space while abrupt movement usually needs the tracker to enlarge the search space to locate the target, which introduces difficulty and time consumption to tracking. Besides, the camera's movement can cause abrupt motion variation of a target in an image. An example of smooth and abrupt motion variations of a target is demonstrated in Figure 2.8.

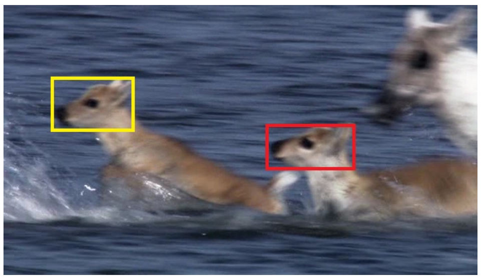


FIGURE 2.9: The example of the background clutter in tracking. The red one shows the target while the yellow rectangle shows the distractor. The image is from (Wu et al., 2013)

.

Background clutter is another circumstance that degrades tracking. Background clutter occurs when a part or several parts of the background bear similar appearance, colour or texture, to the target. These parts of the background are called distractors, causing the confusion between the distracting background and the real target during tracking. Distractors appear frequently when the similar objects are in the background or the texture of the background is complex. Figure 2.9 shows an example of background clutter in tracking.

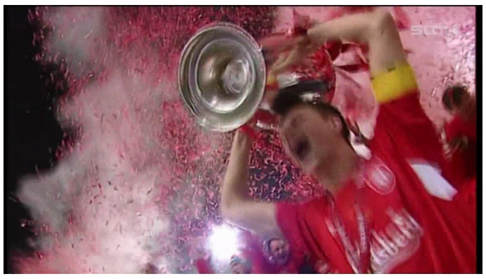


FIGURE 2.10: The example of the motion blur in tracking. The image is from (Wu et al., 2013)

.

Motion blur introduces difficulty to tracking by making the target appearance blurry. It occurs due to high speed movement of the camera or the target itself. An example of motion blur is illustrated in Figure 2.10.

A robust tracking algorithm should take all of the factors into account and try to find a balance among them to achieve decent tracking performance. Generally, a tracking system should be robust to complex context variation and adaptive to the change of the target(H. Yang et al., 2011). Specifically, to be robust, a tracking system should successfully trace the target even when experiencing various and complex conditions such as illumination variation, partial or full occlusion, cluttered backgrounds and so on; while adaptability means that the tracking algorithm is able to adapt to the variation of the target itself, such as articulated deformation, scale variation and out-of-plane rotation. High efficiency also needs to be considered for visual tracking application processing live videos. Although for general online tracking methods real-time processing ability is not required, high processing speed is preferable.

The main issues for the state-of-the-art tracking algorithms include but are not limited to: the initialisation of tracking, occlusion handling and the drifting problem produced by an insufficient or wrong update of the target model. As the initialisation of tracking, it is usually essential to establish an appropriate descriptor to represent the target, which is vital to the following processes as it directly relates to tracking performance (X. Li et al., 2013). The descriptor ranges from the simple histogram to complex features such as HOG and Haar-like features. The use of convolutional neural network (CNN) features has become a recent trend. Occlusion in visual tracking is nearly inevitable. A part of a target could be occluded by other parts of the target or other objects. As part-based modelled object appearance (Adam et al., 2006; Shahed Nejhum et al., 2008; Godec, Roth, & Bischof, 2011; Cehovin, Kristan, & Leonardis, 2011) is robust to partial occlusion by its property, it becomes a natural way of addressing the occlusion problem. Because of the continuously changing target appearance and environmental conditions, the initial target appearance model can be reliable for only a short period of time. Keeping using the original model would not be able to handle the variations of appearance, which leads to the drift problem. However, incorrect updating of the model may bring background information into the template, resulting in drift as well.

2.2 Object Representation

Because of the challenging factors described above, establishing a robust representation of the target is critical. It should be discriminative enough to distinguish between the target and the background, and it should be robust enough to handle the variability of the target, such as illumination change, pose variation, scale variation and partial occlusion. Objects are usually represented by their shapes and appearances (Yilmaz et al., 2006). This section will introduce the typical methods of shape representation in tracking first, and then describe the common methods of appearance representation of the object, which are usually combined with shape representation.

2.2.1 Shape Representation

It is necessary to give a brief introduction to the ways of describing the target shape. The representation of appearance features is usually related to the shape representation of the object, which means the appearance features are calculated based on the area defined by the shape representation or these two kinds of representations are usually combined together in tracking. The classical methods for shape representation are shown in Figure 2.11.

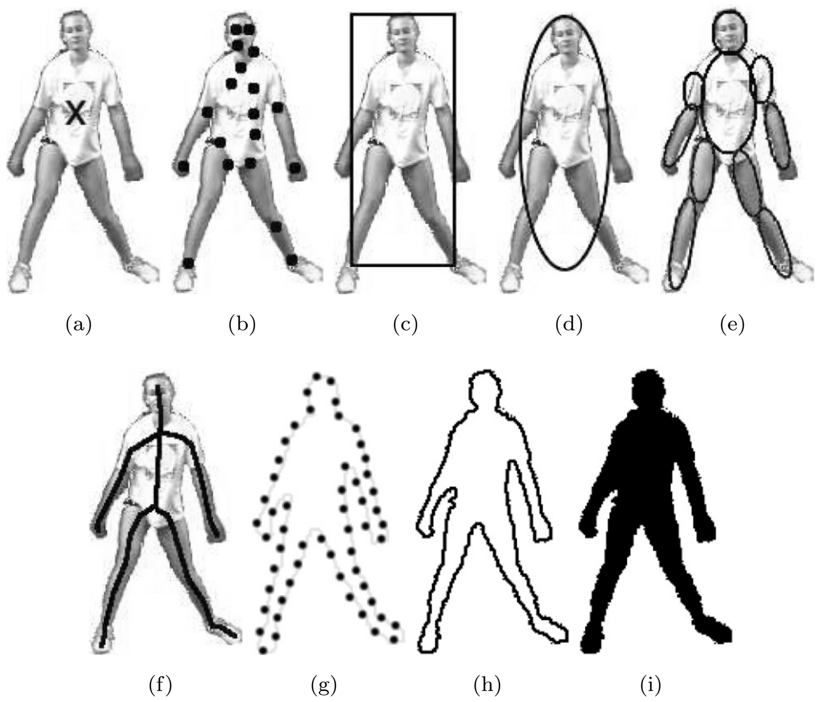


FIGURE 2.11: Object representations (Yilmaz et al., 2006). (a) Centroid. (b) Multiple points. (c) Rectangular patch. (d) Elliptical patch. (e) Part-based multiple patches. (f) Object skeleton. (g) Control points on object contour. (h) Complete object contour. (i) Object silhouette.

- Points. A single point, the centroid (Figure 2.11(a)) (Veenman, Reinders, & Backer, 2001), or multiple points (Figure 2.11(b)) (Serby, Meier, & Gool, 2004) are used to represent the target when the target covers a small area of the image.
- Basic geometric shapes. The object is represented by a basic geometric shape, mostly is a bounding box (Figure 2.11(c)), i.e. rectangle, or eclipse (Figure 2.11(d)) (Comaniciu, Ramesh, & Meer, 2003). Basic geometric shapes can represent rigid

objects. However, they are frequently used to represent non-rigid objects as well, which either inevitably introduces background information into the target representation or would not be able to fully cover the target. Currently, the bounding box is still the most common object representation method.

- Object contour and silhouette. Articulated objects can be represented either by the object contour (Figure 2.11(g), 2.11(h)), the boundary of the object, or the object silhouette (Figure 2.11(i)), the area within the boundary (Yilmaz, Li, & Shah, 2004). The contour and silhouette can be obtained by segmentation.
- Articulated models. Articulated objects can also be represented by articulated models (Figure 2.11(e)), which consist of several rigid parts and relations among the parts. The rigid parts can be described by small basic geometric shapes, such as rectangles or eclipses. The relations among the parts are usually relative geometric positions between them. Most articulated models are used to track humans (Ramanan, Forsyth, & Zisserman, 2007). Recently, the part-based models are applied to general visual tracking (L. Zhang & van der Maaten, 2013; Liu et al., 2015) as well.
- Skeletal models. Both rigid and non-rigid objects can be represented by skeletal models (Yilmaz et al., 2006), which are obtained through applying medial axis transformation to the target silhouette (Ballard & Brown, 1982).

2.2.2 Appearance Description

Object appearance can be affected by many conditions, such as illumination variations, occlusion, deformation and viewpoint changes, which has motivated the design of different image features to cover target appearance properties under different conditions. The features employed for appearance description are critical to visual tracking because a discriminative feature usually facilitates the identification of the target, while a less discriminative feature may confuse other objects (similar to the target) with the real target. Feature selection is usually associated with the object representation (Yilmaz et al., 2006). For instance, for histogram-based appearance representations, colour is usually selected as the feature.

- Colour features. The apparent colour of the target is affected mainly by two physical factors (Yilmaz et al., 2006): (1) the spectral power distribution of the illuminant and (2) the surface reflectance properties of the object. A colour space represents the visual perceptions of the human mathematically, which allows us to analyse and manage the colour (Maggio & Cavallaro, 2011). The RGB (red, green, blue) colour space is commonly used to represent colour in image processing. Other colour spaces applied in visual tracking include $L^*u^*v^*$, $L^*a^*b^*$ and HSV (hue, saturation and value) colour space.

- Gradient features. Local intensity changes hold important information about the target appearance (Maggio & Cavallaro, 2011). These changes happen within the object area and at the boundary between the object and the background. Gradient features describe the statistical summarisation of the gradients (H. Yang et al., 2011). For instance, Lowe et al. (Lowe, 2004) present the famous scale-invariant feature transform (SIFT) descriptor for object recognition. Bay et al. (Bay, Ess, Tuytelaars, & Van Gool, 2008) introduce speeded up robust features (SURF) used as an interest point descriptor. It is scale and rotation invariant and several times faster than the SIFT. Dalal and Triggs (Dalal & Triggs, 2005) propose the histogram of oriented gradient (HOG) descriptor for pedestrian detection. Zhu et al. (Q. Zhu, Yeh, Cheng, & Avidan, 2006) present a speeded-up algorithm for human detection based on HOG descriptor.

- Texture features. Texture measures the intensity variation of an image patch and quantifies properties such as smoothness and regularity (Yilmaz et al., 2006). Gabor wavelet (Manjunath & Ma, 1996) is a common texture feature. Frequency and orientation representations of Gabor filters are considered appropriate for texture representation and discrimination. Ojala et al. (Ojala, Pietikainen, & Maenpaa, 2002) propose the local binary patterns (LBP) descriptor which is an efficient texture descriptor. LBP is tolerant of illumination variations and simple to calculate.

- Deep learned features. Feature learning (Bengio, Courville, & Vincent, 2013) techniques can be applied to make a system automatically discover the representations needed for computer vision tasks. This replaces hand-crafting of features and allows a machine to learn the features which suits a specific problem. In recent papers it is shown that features learned through Convolutional Neural Networks (CNN) are more discriminative than conventional features such as HOG and SIFT (Fischer, Dosovitskiy, & Brox, 2014; Hou, Zhang, & Zhou, 2015; Zagoruyko & Komodakis, 2015). The CNN features are usually extracted from a specific layer after passing and processing the raw data through several layers in a deep network (Ma, Huang, Yang, & Yang, 2015; L. Wang, Ouyang, Wang, & Lu, 2015).

As described in Section 2.1.2, the appearance model takes two roles. One is to represent the target, the other is to predict the likelihood of a target candidate locating at a specific position. Next, this thesis is going to introduce the most common appearance models in tracking. There are two main categories of appearance models, generative models and discriminative models. As the most common manner of shape representation of the object is the basic geometric shape, the following section will mainly introduce appearance models established based on a basic geometric shape, i.e. the rectangle or eclipse, which is usually called the bounding box.

2.2.3 Appearance Models

2.2.3.1 Generative Models

Generative models focus on the object itself and only model the appearance of the target, ignoring the background information (Black & Jepson, 1998; Balan & Black, 2006; Ross et al., 2008). Tracking is then interpreted as finding the object which is most similar to the appearance model. However, the correctness of the learned model is very hard to verify in practice (X. Li et al., 2013). Moreover, when trying to find global optima in parameter estimation e.g. by expectation maximisation, local optima are often obtained instead (X. Li et al., 2013). To handle variations in target appearance over time, generative models are usually updated online, by incrementally learning visual information from the object, i.e. the foreground, to adapt a generative model learned at the beginning of tracking to the current tracking state.

Some generative methods are based on templates (Lucas & Kanade, 1981; Comaniciu et al., 2003; Matthews, Ishikawa, & Baker, 2004; Adam et al., 2006; Kwon & Lee, 2010). The template is learned at the beginning of tracking while the target appearance usually varies significantly over time during tracking. Thus, a fixed template is not able to handle the variations, which leads to the drift problem. Updating the template is critical for tracking; however, incorrect updating could bring background information into the template, so that the template would degrade gradually and drift away from the real target. To handle the updating problem, Matthews et al. (Matthews et al., 2004) propose a strategy of updating the template when necessary or re-using the pre-learned templates to reduce the drift. Kwon et al. (Kwon & Lee, 2010) propose a decomposition model which consists of several basic appearance models, each of which is responsible for a kind of variation.

Subspace learning (Black & Jepson, 1998; Ross et al., 2008) is also common in generative models. (Black & Jepson, 1998) proposes a method of encoding the known appearance of the target into an offline learned appearance model using the eigenbasis. (Ross et al., 2008) develops a low-dimensional subspace learned incrementally during tracking to prove its robustness to appearance changes of the target.

Inspired by the recent advances in sparse representations (Donoho, 2006; Wright, Yang, Ganesh, Sastry, & Ma, 2009), many corresponding applications have been proposed in tracking (Mei & Ling, 2009; H. Li, Shen, & Shi, 2011). (Mei & Ling, 2009) represents the target sparsely using a linear combination of target templates and trivial templates. The target is then found by solving an ℓ_1-minimisation problem, and is the candidate with the smallest projection error. The proposed tracker demonstrates robustness in scenarios with occlusions, illumination changes and pose variations. However, this tracker shows high computational complexity, which limits its application in real-time scenarios. To solve this problem, (H. Li et al., 2011) adopts an orthogonal matching pursuit algorithm to achieve real-time processing with competitive tracking accuracy.

As generative models are established without taking the appearance of the background into account, they are prone to failure when coming across background clutter or similar objects moving around (L. Zhang & van der Maaten, 2013); while discriminative models are capable of handling these cases because of their classifiers learned to separate the target with the background. (Kalal et al., 2010) indicates that the performance of discriminative trackers are better than generative trackers'. (Avidan, 2004; Cristinacce & Cootes, 2008) have obtained similar results. (Minka, 2005) has provided theoretical support for this statement which is, in a discriminative task such as object tracking, the discriminative models always have better performance over their corresponding generative models (L. Zhang & van der Maaten, 2013).

2.2.3.2 Discriminative Models

Discriminative models model the appearance of both the target (the foreground) and the background to best separate the target from the background, and can be interpreted as a binary classification process, i.e. tracking-by-detection. The classifier is trained with positive and negative training samples either online or offline, which means that the performance of discriminative models is mainly determined by the training samples and the method of training. Many discriminative trackers then work on discovering the more informative features which can better distinguish the object from the background.

Collins et al. (Collins, Liu, & Leordeanu, 2005) develop a method by adaptively selecting the colour features which are the most discriminative, and so can best distinguish the target from the background. Avidan (Avidan, 2007) trains an ensemble of weak classifiers online and combines them into a strong classifier to do tracking. Grabner and Bischof (Grabner & Bischof, 2006) propose an online feature selection method based on Adaboost, which selects and maintains the best discriminative features from a pool of feature candidates.

All the methods described above show robustness to target variations and high efficiency; however, every time the classifier is updated there is a chance that the updating may introduce irrelevant information into the classifier, which would cause drift and tracking failure over time. It happens because that the errors brought in by updating gradually degrade the discriminative ability of the classifier trained before the start of tracking.

To deal with the updating problem, the basic idea is to assist an online learning classifier with a reliable classifier: the one trained at the beginning of tracking. Grabner et al. (Grabner, Leistner, & Bischof, 2008) make the attempt by proposing a semi-supervised method based on online boosting to handle the drift problem. Specifically, the fixed classifier and the online classifier are combined to decide when to update.

Label noise is considered to be another reason for drift problem. The establishment of discriminative classifiers relies heavily on the training samples. Besides, the classifier uses the positive and negative samples associated with the current tracking state to update itself. Thus, imperfect tracking can generate poorly labelled training examples, which gradually corrupts the classifier and causes drift. Bakenko et al. (Babenko et al., 2009) propose an algorithm using online multiple-instance learning (MIL) to overcome the samples' ambiguity. Compared with traditional supervised learning, the classifier is updated with a positive bag consisting of several positive samples. As is known to all, the objective of classification is to correctly label the samples, while the objective of tracking is to accurately locate the target. Based on the understanding that the objectives for classification and tracking are not consistent, Hare et al. (Hare et al., 2011) present a structured output prediction method to predict the target transformation instead of predicting the labels, which eliminates the influence of intermediate classification.

To reduce the drift problem, Kalal et al. (Kalal et al., 2012) develop a framework which divides the tracking problem into tracking, learning and detection. In addition, they propose a P-N learning method to exploit the underlying structure of the unlabelled data to train a binary classifier, which is able to estimate and then correct the errors of the tracker to alleviate the drifting. Wang et al. (S. Wang, Lu, Yang, & Yang, 2011) seek an effective image representation by applying superpixels to build a discriminative appearance model. The robust object representation allows the tracker to identify the target from the background. Zhang and Van der Maaten (L. Zhang & van der Maaten, 2013) present an online structured support vector machine (SVM) framework to learn the appearance model and the spatial constraints of the target jointly.

2.3 Motion Models

Methods of object representation are reviewed in the previous section, and this section is going to introduce motion models. The motion model restricts the search space for the target. Specifically, when an unknown frame is acquired, the motion model provides hypotheses of the possible target location, which can also be the prior probability that the object is present at a certain location. Effective motion models can guide the search towards the correct state of the target.

2.3.1 Single Motion Models

As visual tracking covers tracking of a great range of objects, it is difficult to constrain the motion too much in practice, and simpler motion models are usually used. The common motion models are a random-walk (RW) (Chang & Ansari, 2005; Perez, Vermaak, & Blake, 2004) model or a nearly constant velocity (NCV) model (Shan, Tan, & Wei, 2007; Pernkopf, 2008). Assuming that the target's velocity is a white noise sequence, the RW model characterises the target's motion as temporally completely uncorrelated. On the contrary, the NCV model is based on the assumption that velocity is temporally strongly correlated and the variations in velocity occur due to the white noise of the acceleration. The RW model works in describing abrupt motion of the target, while the NCV model describes correctly the target's motion when the target moves towards a certain direction and the target is shot with a fixed camera (Kristan, Kovacic, Leonardis, & Pers, 2010).

However, the target usually shows complex movement variations over time due to articulated deformation, rotation, fast movement, camera movement, etc. A single motion model (RW or NCV) is not able to cover the complex motion variation of the target in general visual tracking. This situation can be improved by increasing the process noise of a RW or a NCV model. An obvious drawback is that this approach is computationally expensive, and increases the search space, making it more likely that the tracker will be affected by clutter.

2.3.2 Improved Single Motion Models

To handle the increasing variance of the estimation when using a simple motion model (RW or NCV) with large process noise, the approach of providing an efficient and informed proposal distribution was proposed. Specifically, Okuma et al. (Okuma, Taleghani, De Freitas, Little, & Lowe, 2004) introduced a method to guide a particle filter by Adaboost detection models. The proposal distribution is composed of information from Adaboost detectors and a standard autoregressive motion model. By combining the two methods together, their framework is able to quickly detect and track targets against a dynamically changing background (Okuma et al., 2004). Kristan et al. (Kristan et al., 2010) proposed a two-stage dynamic model to cover common motions observed in tracking persons. This dynamic model is composed of two models: a liberal and a conservative model. The liberal model covers larger perturbations of the target, while the conservative model covers smaller perturbations of the target and constrains the liberal model. This method fails when the target exhibits frequent non-constant motions (Khan, 2015), which happens frequently.

2.3.3 Multiple Motion Models

To cover different kinds of target motion during tracking, an interacting multiple model (IMM) was proposed by (Bar-Shalom, Kirubarajan, & Li, 2002). The idea is that multiple trackers, each with a different motion model, are used at the same time to track the target. After the determination of how accurately each model describes the target's current motion, predictions from different trackers are combined accordingly (Kristan et al., 2010). The implementation of IMM based on particle filtering can be found in papers (McGinnity & Irwin, 2000; Blom & Bloem, 2007). This approach, however, significantly increases the complexity of the tracker, as the tracking process has to be performed over each of the motion models. Kwon and Lee (Kwon & Lee, 2011) proposed to construct the motion model distribution from the recent sampling history and the motion models are then randomly sampled from the distribution. Recent sampling history needs to be clustered and each tracker sample needs to be evaluated by the recent tracking history, which is a cumbersome mechanism for a real-time visual tracking method. To capture the complex motion variation of the target in visual tracking, Khan et al. (Khan, Valstar, & Pridmore, 2015) proposed to learn motion models over multiple temporal scales (MTS). The predictions from the motion models are pooled over multiple prediction scales, which provides a sufficient and efficient search space for tracking. Additionally, this MTS motion model is online learned for general tracking.

2.3.4 Approaches not Considering a Motion Prior

The straightforward tracking method without motion models usually adopts an exhaustive sliding window. In this case, the target is searched from the whole image frame, without any assumption made about the motion of the target. Full search makes the tracker robust to abrupt motion of the target and camera movement. The obvious drawback of this method is its heavy computational consumption. Instead of considering the full search space of a frame, a smaller local region around the previous target's location can be fully searched, which is a local sliding-window approach (Babenko et al., 2011; Hare et al., 2011). This mechanism makes a balance between computational efficiency and robustness to abrupt motion of the target. However, it may stop at the local optima and lead to poor performance.

2.4 Search Strategies

The methods of motion models are reviewed in the previous section, and this section is going to introduce search strategies. The search strategy aims at inferring a solution to the tracking problem using the appearance models and/or any other available image information. Usually, it is conducted by searching the space of possible hypotheses to find the most likely configuration for the current target.

Search strategies are generally classified as probabilistic or non-probabilistic. Probabilistic search strategies utilise probability distributions of random variables to model the uncertainty within the models (e.g. the target state) and the measurements (e.g. the observation). The target state is then inferred through estimation of the probability distributions. Non-probabilistic search strategies are usually deterministic, but can share some characteristics of probabilistic models (Smith, 2007).

2.4.1 Non-probabilistic Search Strategies

In most cases, non-probabilistic search strategies formulate tracking as an optimisation problem, in which the tracking problem is then represented by a cost or error function and solved by maximising or minimising the function. The main benefits of using non-probabilistic methods are good convergence properties and computational efficiency.

EigenTracking (Black & Jepson, 1998) is a famous early tracking method applying optimisation. The target is described by a small set of primitive 'basis images' and the target is located by minimising an error function with least-squares approximation. The error function measures the dissimilarity between the basis vectors and the image data.

Mean shift is probably the most widely known optimisation algorithm in tracking. It was proposed by Fukunaga and Hostetler (Fukunaga & Hostetler, 1975) in 1975 working as a general method to find the mode of a density function, then it is introduced into tracking task by Comaniciu et al. (Comaniciu, Ramesh, & Meer, 2000). Specifically, in their work, the target is located by seeking the most probable target state which matches the target model best through mean shift iterations, in which the matching function is defined by a metric derived from the Bhattacharyya coefficient.

2.4.2 Probabilistic Search Strategies

Probabilistic search strategies show three main advantages over the non-probabilistic techniques. First, probabilistic methods demonstrate flexibility. Probabilistic models used to track one specific class of objects can be adapted to track another. The flexibility is also because one component of the probabilistic model can be replaced with another without influencing the whole design. Second, they display generality, as an inference method designed to track a single target can be applicable to other problems related to sequential data. Last but not least, probabilistic models are able to systematically process the unpredictable target configurations and noise which occur frequently in real-world, through the maintenance of multiple hypotheses.

The most popular probabilistic method in tracking is recursive Bayesian estimation, also known as Bayes filter, which obtains the solution of the tracking problem by recursively predicting with previous information and correcting with incoming measurements. Specifically, in this algorithm, the target states and observations are represented by random variables, for example, noted with X and Y separately. Given previous observations up to time t, denoted with $Y_{1:t} = \{y_1, ..., y_t\}$, and the previous tracked target state at time $t-1$, X_{t-1}, the predicted current target state X_t at time t can be defined by:

$$p(X_t|Y_{1:t}) = \frac{p(Y_t|X_t) \int_{X_{t-1}} p(X_t|X_{t-1}) p(X_{t-1}|Y_{1:t-1}) dX_{t-1}}{p(Y_t|Y_{1:t-1})}, \tag{2.1}$$

where $p(Y_t|X_t)$ represents the observation model and the $p(X_t|X_{t-1})$ is known as the state evolution or motion model. $p(X_{t-1}|Y_{1:t-1})$ is the posterior distribution at time $t-1$. Tracking at time t is started by transferring the posterior distribution of the previous time step $p(X_{t-1}|Y_{1:t-1})$ to the current time step with state evolution $p(X_t|X_{t-1})$. This step forms a prior distribution of target state X_t. When the current observation Y_t arrives, the prior distribution is corrected with the observation model $p(Y_t|X_t)$, producing the posterior distribution $p(X_t|Y_{1:t})$. $p(Y_t|Y_{1:t-1})$ can be replaced with a constant C to guarantee that the posterior probability sums to one over the state space.

If the state evolution is linear with additive Gaussian noise and the observation model is Gaussian, the recursive Bayesian estimation can be resolved directly by a Kalman filter. This was proposed by R.E.Kalman (Kalman, 1960) in 1960, offering an optimal solution to the state estimation problem when the above assumptions are satisfied. However, the assumption is typically not true in visual tracking. For example, the likelihood distribution arising from image features are usually non-Gaussian. To handle these situations, the particle filter is presented to solve the state estimation problem overcoming the limitations.

The particle filter, also known as the sequential Monte Carlo (SMC) methodology, can approximate the recursive Bayesian estimation of Equation 2.1 by a set of weighted particles. The prediction and update steps of recursive Bayesian estimation, Equation 2.1, are transferred to be propagation and weighting of particles separately. Please refer to (Arulampalam, Maskell, Gordon, & Clapp, 2002) for more details. Generally, the particle filter is able to give a good and efficient approximation of the Bayesian filtering. However, with the increase of the dimension of the state space, it turns out that seeking a good proposal distribution is difficult. The particle filter method would acquire an exponentially large number of samples to cover the high-dimensional space. To handle the efficiency problem, Markov chain Monte Carlo approaches are proposed.

Markov chain Monte Carlo (MCMC) methods are basically sampling algorithms which construct a Markov chain and then sample from its probability distribution. MCMC methods are used to approximate Bayesian filtering in tracking. During sampling, structural knowledge of the state space is incorporated so that the particles are generated more frequently in important regions of the underlying posterior to improve the efficiency.

2.5 Tracking Methods with Structured Part-based Models

There are many successful applications of part-based models in the field of object detection and recognition (Mohan, Papageorgiou, & Poggio, 2001; Agarwal, Awan, & Roth, 2004; Mikolajczyk, Schmid, & Zisserman, 2004; Schmid & Mohr, 1997). However, directly applying these part-based models in visual tracking still faces some issues. To achieve accurate tracking with part-based models, each part should be trained and initialised explicitly, which is complicated and hard to handle in an online manner (Sigal, Bhatia, Roth, Black, & Isard, 2004). Part-based models have prominent advantages over global appearance models on self-occlusion and inter-object occlusion. They are able to model both flexible and articulated targets. In terms of visual tracking, part-based models mostly have been applied to human body tracking where the models are already trained or built for this specific purpose (Sigal et al., 2004; Andriluka, Roth, & Schiele, 2008; Ramanan et al., 2007). The research of this thesis focuses on tracking with model-free part-based models. The term 'model-free' means that the model is not predefined for a specific class of objects, e.g. eyes, nose and mouth for face tracking and hands for hand tracking.

Imposing shape structure on a part-based object model is typically done by combining a part-based appearance model with a global shape model. Given a set of location hypotheses for each part (e.g. particles in particle filtering or a region of interest in a sliding-window search), the appearance model gives the likelihood of each hypothesis. The final hypothesis finds a compromise between maximising the individual responses and minimising the penalties imposed by the shape model. Since the number of possible configurations grows exponentially with the number of parts and part hypotheses, it is common to resort to an efficient shape model, e.g. deformable part models (DPM) (Felzenszwalb et al., 2010), which allows exact inference. These models are often extended for tracking-specific purposes, most commonly using temporal consistency or online/incremental adaptation of the models.

2.5.1 State-of-the-art Part-based Tracking Methods

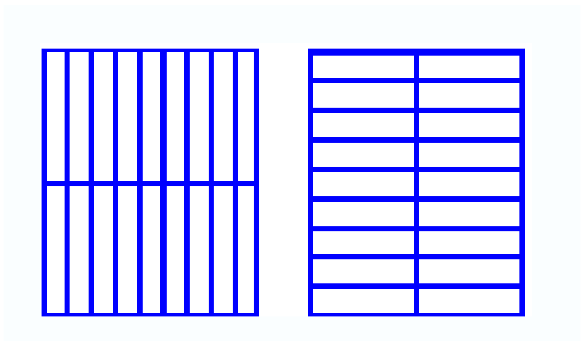


FIGURE 2.12: The patches used in (Adam et al., 2006).

To address model-free tracking with a part-based model, Adam et al. (Adam et al., 2006) propose 'FragTrack', in which vertical and horizontal patches decompose the target template in a grid, as shown in Figure 2.12. The location of each patch is defined by a vector linked to the target centre. Horizontal patches are of one tenth of the template's height and half the template width. Vertical patches are similarly defined. FragTrack utilises around 36 patches in total. In this case, the appearance model is a generative template model represented by multiple image fragments. The feature used is the histogram of an image fragment. Patch hypotheses are obtained through exhaustive search of a local region based on the previous target location within a radius r. The search strategy is a non-probabilistic search strategy using an error function. Specifically, the dissimilarity between the histogram of a patch template and the histogram of each patch hypothesis provides a voting-map of target position corresponding to the patch. The voting-maps from all the patches are then combined to form a single voting-map to locate the target. The spatial relationships between fragments and the target template centre are fixed and not updated, though, so that the fragments are not able to model non-rigid targets, which can move flexibly, well.

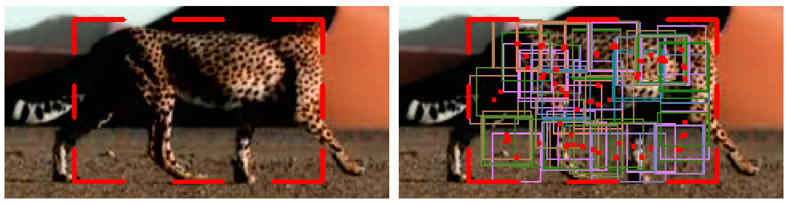

(a) (b)

FIGURE 2.13: Early election of the attentional region pool (M. Yang et al., 2007). (a)Initialisation. (b) The pool of attentional regions (ARs).

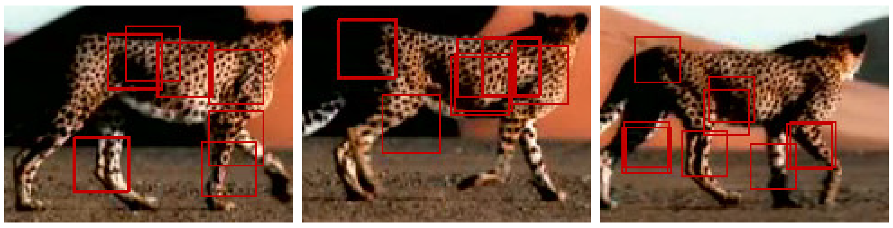

(a) (b) (c)

FIGURE 2.14: Examples of late selection of discriminative ARs of different frames (M. Yang et al., 2007).

Yang et al. (M. Yang et al., 2007) propose an attentional visual tracking (AVT) method using spatially attentional patches (as shown in Figure 2.13). Recently, research indicates that selective attention may be performed in both the early and late stages of visual processing, but it works with the different perceptual load in the two stages (Palmer, 1999). The evolution may determine innate principles for early selection while learning through experiences determines the late selection. In the AVT approach, there is a two-stage appearance model, including the attentional regions (ARs) and discriminative attentional regions (D-ARs). The ARs form a generative appearance model of the target, while the D-ARs obtained through a discriminative learning form a discriminative appearance model of the target. Because the prior knowledge of the dynamics of the ARs is generally unavailable, an exhaustive search of the motion parameter space is used, during which locality-sensitive hashing (LSH) (Datar, Immorlica, Indyk, & Mirrokni, 2004) is used to reduce the computation. So the search strategy is a non-probabilistic matching approach. Specifically, based on the two rules of the selective attention in human visual processing, AVT extracts a few attentional regions (ARs) representing the target area in the early selection process. The ARs are selected with the condition number to have properties for good localisation, which means they must be both informative and stable. All the ARs are of the same size and shape, which are 30×30 squares in this paper. The matching set of each AR is then obtained by examining motion hypotheses within its candidate region, which represents the probability distribution of target location given the relationship between current AR and the target. The target location is then voted for by fusing the probability distribution of all ARs. After that, by discriminative learning on the historical data, a subset of discriminative attentional regions (D-ARs) are obtained in the late selection stage, as shown in Figure 2.14. These D-ARs are allocated larger weights in voting in the next frame and reflect the dynamical update ability to adapt to the environment change. The patches (ARs and D-ARs) obtained by this way can contain salient and discriminative regions of the targets. Representing the target with a pool of attentional regions makes the AVT method robust to appearance variations due to small deformation, lighting changes and partial occlusions. However, the ARs have significant overlaps and spatial relations among ARs are not utilised. AVT needs a large number of ARs to make a reliable prediction, so it is only suitable for tracking large targets. Also, it inevitably includes the background in the target information and tends to be problematic when handling non-rigid objects.

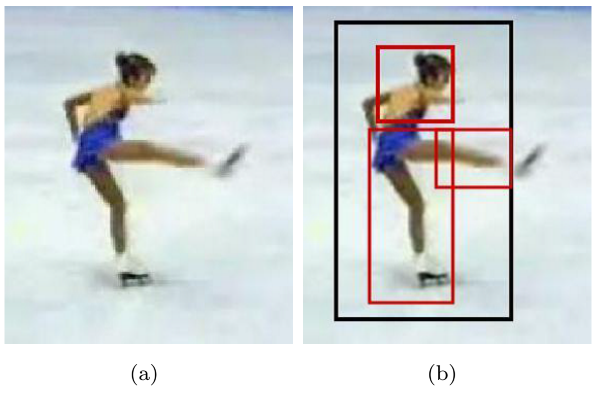


FIGURE 2.15: (a) An articulated target. (b) The blocks representing its shape (red rectangles). The black rectangle is the tracking window W (Shahed Nejhum et al., 2008).

Neihum et al. (Shahed Nejhum et al., 2008) propose a more flexible structure using multiple rectangular blocks to model the constantly changing shape of the target (as shown in Figure 2.15). Given the contour of the object, the initial window W defining the target area, K rectangular blocks $B_i(1 \leqslant i \leqslant K)$ and the weights λ_i of blocks are determined. The union of blocks covers most of the target, which may have small overlaps, and they inevitably include foreground and some background pixels at the same time, especially when the target is articulated. Higher λ_i is allocated to the block which contains more foreground pixels, and vice versa. In this case, the appearance model of the target is the template represented by the union of the blocks. The motion prior is not considered and the target is searched for in the entire image. The search strategy is a non-probabilistic method using a similarity function. Specifically, the weighted sum of histogram similarities of blocks between the template and the candidate are used to locate the target. The number of blocks K and the size of each block are fixed during tracking. After locating the target, its contour is segmented and then the position of each block is adjusted locally on-the-fly based on the updated contour to maximally cover the moving target. By this configuration, spatial information of the target is loosely encoded in this block structure. Nevertheless, the fixed size and number of the block are quite limited in tracking articulated targets.

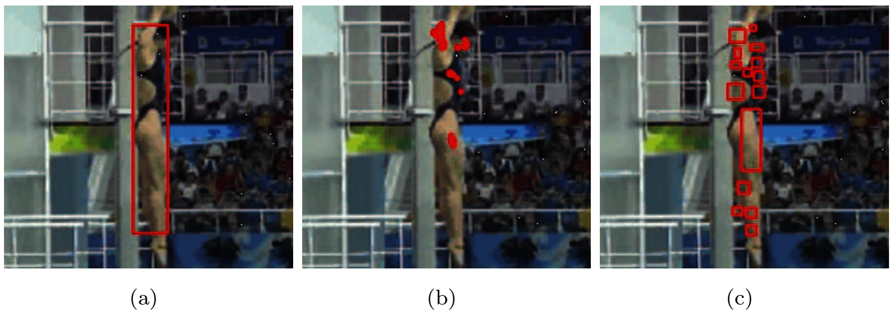
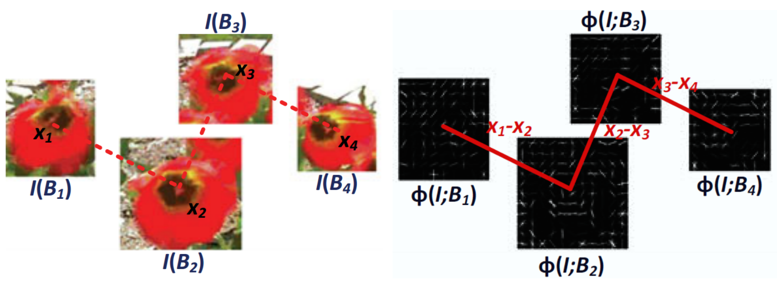


FIGURE 2.16: Example of patch initialisation process in (Kwon & Lee, 2009). (a) Given bounding box. (b)Good points with small condition number. (c)Chosen patches.

Kwon and Lee (Kwon & Lee, 2009) propose a patch-based appearance model. All the patches within the bounding box are connected with the centre of the bounding box enclosing the target. The position of patches are initialised by the condition number of the Hessian matrix to be good for image alignment and the size is set randomly. Each local patch is only dependent on the centre of the target in this model. Given the bounding box, the first local patch's centre position is selected as the point with the lowest condition number K value, which means the Hessian matrix is the most numerically stable. The centre position of the second patch is determined by the point with the second lowest K value. Applying the criterion that new patches are non-overlapping with existing patches, initialisation of patches is repeated until there is no space to add a new patch or the number of patches reaches a predefined value. The whole initialisation process of patches is shown in Figure 2.16. In this method, the appearance model is a generative appearance model represented by an assembly of patches. The motion model is basically a proposal density given by a transition model. The search strategy is a probabilistic search strategy, Bayesian filter. Specifically, they (Kwon & Lee, 2009) incorporate the star model into a Bayesian filtering framework and an MCMC-based search, in which the likelihood is defined by the product of the photometric and geometric likelihood of all patches. Patches can be flexibly moved, deleted or added in an online update step, which is determined by landscape analysis of the patch. The patches have no relationships between each other and the patch doesn't relate to a specific target area. All the patches are only dependent on the centre position of the target, which constitutes a loose target structure and doesn't represent the target's geometric information clearly. Thus this model, though robust to a certain degree, tends to suffer the part drifting problem during tracking.



(a) (b)

FIGURE 2.17: The configuration used in (L. Zhang & van der Maaten, 2013). (a) $I(B)$ is the image within the bounding box and x is the centre of each bounding box. (b) $\Phi(I, B)$ is the HOG feature extracted from previous configuration and the red solid lines represent the relative location between tracked objects.

Alternatively, Zhang et al. (L. Zhang & van der Maaten, 2013) propose the structure preserving object tracking method, i.e. the SPOT tracker, verifying the importance of structural constraints between objects or within an object in tracking. The appearance model in their method is a classifier trained on HOG features. The motion prior is not considered and a sliding window approach is used. The search strategy is a non-probabilistic optimisation method minimising the structured SVM loss. Specifically, the initial configuration is a set of bounding boxes, defining the tracked objects, and the spatial constraints among them, as shown in Figure 2.17. The size and number of bounding boxes are fixed during tracking. They use a star model (Felzenszwalb et al., 2010) and a minimum spanning tree model (X. Zhu & Ramanan, 2012) to represent the spatial relations among the tracked objects separately. The star model is defined by allocating a dummy object centred on the area covered by all the objects and the spatial constraints are displacements from this dummy object to other objects. In the minimum spanning tree model, the spatial constraints are defined by the tree model minimising the sum of weights of all edges of the tree obtained through searching from all possible tree models. Experimental results (L. Zhang & van der Maaten, 2013) show that the SPOT tracker with the minimum spanning tree model has better performance. The SPOT tracker utilises an online structured SVM framework to find a configuration best matching the appearance model without stretching or compressing the 'springs' between the tracked objects too much. The weight of each bounding box and the spatial structure are updated every frame after locating the tracked objects. The SPOT tracker is designed for tracking multiple objects at the same time. The paper also shows that the SPOT tracker can improve the performance of tracking the single target by adding object parts, and the number of parts is fixed to 2. This paper shows that spatial constraints are crucial for model-free tracking.

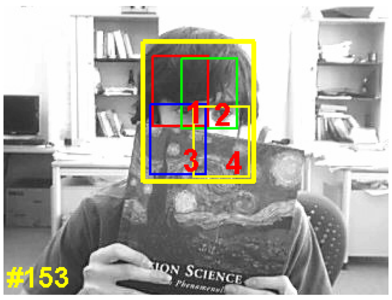


Figure 2.18: An example of tracking result of parts and the bounding box in (Yao et al., 2013). The yellow rectangle shows the bounding box of the object and four small rectangles show the bounding box of parts.

Yao et al. (Yao et al., 2013) propose a structured learning method with the unknown parts of a tracked target modelled using latent variables. The part-based appearance model in this paper is to use a structured SVM with latent variables. The motion model is only based on the previous target location. The search strategy is an optimisation approach. Specifically, a target is represented with a few parts, each of which has a weight. An online structured support vector machine method with latent variables is adopted to learn the weight parameters of a target and its parts. The target is then discriminated from the background with the weight parameters. The target location is given by maximising the classification score in the vicinity around the prediction of the target from the previous frame. A two-stage training method is employed in their tracking approach, in which the parts parameters are predicted in the first stage and the total object and correlation parameter are estimated in the second stage. The whole object is defined by the bounding box. A part is represented by a smaller box, a part box, similarly. The parts of the object are defined heuristically based on the ratio of width and height of the object. If the ratio is larger than 0.5, the object is divided into four parts equally (two rows and two columns), or else it's three parts. This method shows the flexibility of the appearance representation using part-based models to a certain degree. Figure 2.18 shows an example of a tracking result from one video frame (Yao et al., 2013).

Cehovin et al. (Cehovin et al., 2013) present a coupled-layer visual model to handle the rapid and significant variations of target appearance. The coupled-layer model consists of a local layer and a global layer. The local layer is a geometrical constellation of visual patches representing local visual properties of the target, while the global layer describes the visual information of the whole target using a probabilistic model. The appearance model in this tracker is a coupled-layer model containing both global and local target appearance information. The motion model is a Kalman filter with a NCV dynamic model. The search strategy is a Bayesian filter. Specifically, the target's velocity is estimated by the Kalman filter and the local-layer patches are initialised with the NCV

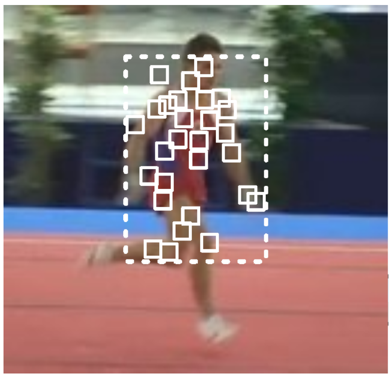


FIGURE 2.19: Illustration of the configuration of the proposed coupled-layer visual model in (Cehovin et al., 2013). The white dashed rectangle shows the global layer of the object, and the small white rectangles show the patches.

model. The locations of local-layer patches are adapted through the Bayesian framework. The target's centre is then calculated from the locations of local-layer patches. The patches which do not respond to the target are removed from the local visual model. The remaining stable patches are used to update the visual features of the global layer. After that, the allocation of the new patches is determined by the global layer when necessary. The couple-constrained updating of the visual model realised by feedback loops between the global and the local layer increases the robustness of the proposed tracker. Figure 2.19 demonstrates the configuration of the coupled-layer visual model in (Cehovin et al., 2013).

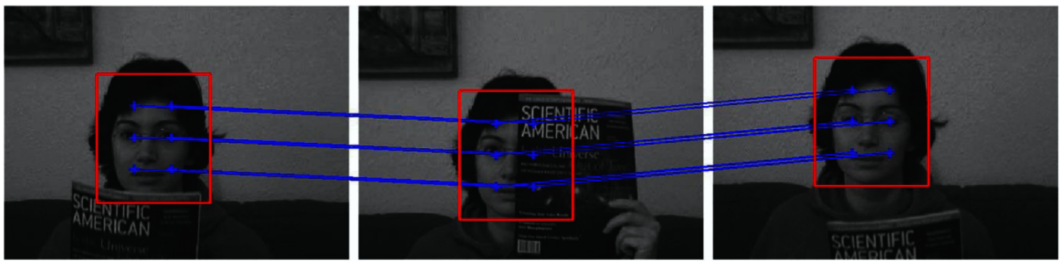


FIGURE 2.20: The correspondences of parts are established from multiple frames in (T. Zhang et al., 2014). The blue cross marks denote the positions of parts, and the blue lines represent their correspondences. The final tracking results are denoted with red bounding boxes.

Zhang et al. (T. Zhang et al., 2014) propose a robust part matching method to handle the partial occlusion problem through the exploitation of the confidence score of individual parts. The appearance model is a template represented by a low-rank matrix formed by intensity vectors of corresponding parts in multiple frames (see Figure 2.20). The motion model is a particle filter sampling at and around the parts of the previous tracking results. The search strategy is a matching approach. Different from other matching methods, the matching of parts in this paper is conducted among multiple frames, which enables a globally consistent property through the sequence. Specifically, the part correspondences in multiple frames are obtained through a locality-constrained low-rank sparse learning approach using optimisation of partial permutation matrices, during which the parts are matched jointly to cover the occlusion of part appearance caused by occlusion. The parts are obtained through dividing each template into regular grids, from which the number of parts is 3×2 or 2×3 depending on the target's aspect ratio. The number of incoming frames to be tracked is set to be 3. This method displays robustness to partial occlusion.

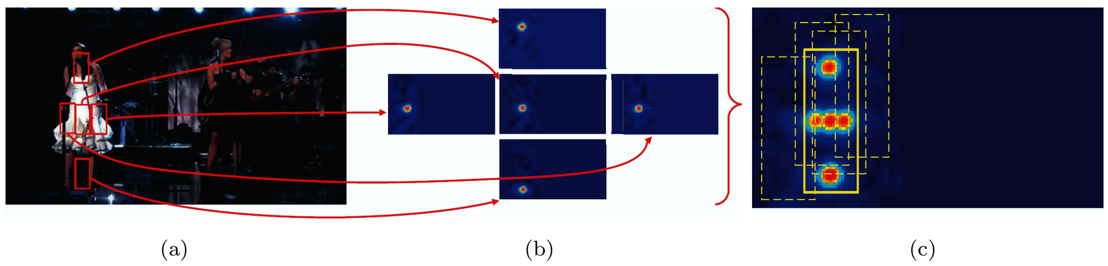


FIGURE 2.21: The framework of (Liu et al., 2015). (a) The parts' configuration of a target. Each part is tracked individually. (b) The corresponding confidence map of each part. (c) The combined confidence map and Bayesian framework are applied to locate the target. The yellow windows are candidate windows and the solid one is the tracking result.

Recently, research on correlation filters has attracted much attention and has been frequently applied in tracking (Bolme, Beveridge, Draper, & Lui, 2010; Rodriguez, Boddeti, Kumar, & Mahalanobis, 2013; Danelljan, Khan, Felsberg, & v. d. Weijer, 2014; J. F. Henriques, Caseiro, Martins, & Batista, 2015) because of its high efficiency and competitive performance. Liu et al. (Liu et al., 2015) propose a part-based real-time tracker based on the kernelized correlation filter (KCF) (J. F. Henriques et al., 2015) as the part classifier, as shown in Figure 2.21. The number of parts is fixed to 5 in this paper. The size of the part is defined between 1/4 and 1/6 of the object size. Each part is tracked separately using the KCF tracker, i.e. the part classifier, which is the discriminative appearance model in this paper. The motion parameter space is a larger search window based on previous target location. The target is tracked by correlating the filter over the search space, and the location with the maximum value in the correlation response map would be the new predicted target location. Specifically, a correlation response map is obtained for each part to locate the part and the confidence maps of parts are combined to infer the target location. When combining the response maps for all the parts, the peak-to-sidelobe ratio (PSR) and temporal smoothness of the confidence map are considered to decide the weight of each map, which is to make sure the reliable parts are given more weight and to reduce the influence of occluded parts. Similarly, when updating the classifier coefficients and the learned target appearance, the same rule is followed so that each part tracker is updated adaptively based on its weight. The robust part tracker would be updated more with higher learning rate. To locate the target, a Bayesian framework is applied to the combined confidence map during which the spatial constraint mask is applied to enforce the structural relationships between parts. After that, the candidate window which has the highest likelihood within the confidence map is selected as the target location. The size of each part tracker is fixed while the target size is determined by the Bayesian framework to enable variation of scale and rotation. This method is proved to perform competitively, which relies on its multiple part configuration and adaptive updating.

2.6 Datasets for Evaluation of Visual Tracking Methods

It has become a standard of evaluating trackers on publicly-available sequences with a standard ground truth labelling (Wu et al., 2013; Kristan et al., 2013, 2014, 2015, 2016b) which provide convenience for comparison. The datasets which are used most extensively include the CVPR2013 Benchmark (Wu et al., 2013) and the VOT2015 Benchmark (Kristan et al., 2015). ‘Amsterdam Library of Ordinary Videos(ALOV)’ is another very large dataset. However, some sequences contain ambiguously defined targets such as fireworks, which makes the dataset inappropriate for tracking evaluation.

2.6.1 CVPR2013 Benchmark

To provide a benchmark to evaluate the state-of-the-art tracking methods Wu et al. proposed a benchmark in the 2013 IEEE Conference on Computer Vision and Pattern Recognition(Wu et al., 2013). It includes 50 video sequences (including 51 tracking instances, see Figure 2.22) and 29 state-of-the-art trackers. This benchmark provides videos with challenging conditions such as scale variation, occlusion, deformation, fast motion, illumination variation, in-plane rotation, out-of-plane rotation, background clutter and so on. Thus, this tracker can avoid over-fitting to a small subset or one specific attribute. All videos are manually tagged with what the main challenges of the video are. Usually, each sequence is annotated with several visual attributes. The benchmark can report tracker performance with respect to each attribute separately. The list of the attributes is shown in Figure 2.23.

The ground truth provided by this benchmark is the bounding box. The performance of the various trackers is measured using precision (Babenko et al., 2011; J. a. F. Henriques, Caseiro, Martins, & Batista, 2012; Wu et al., 2013) and success plots in this benchmark. The precision plot measures the percentage of frames whose estimated location is within the given threshold distance of the ground truth (Wu et al., 2013). The success plot measures the percentage of frames for which the overlap divided by the union of the

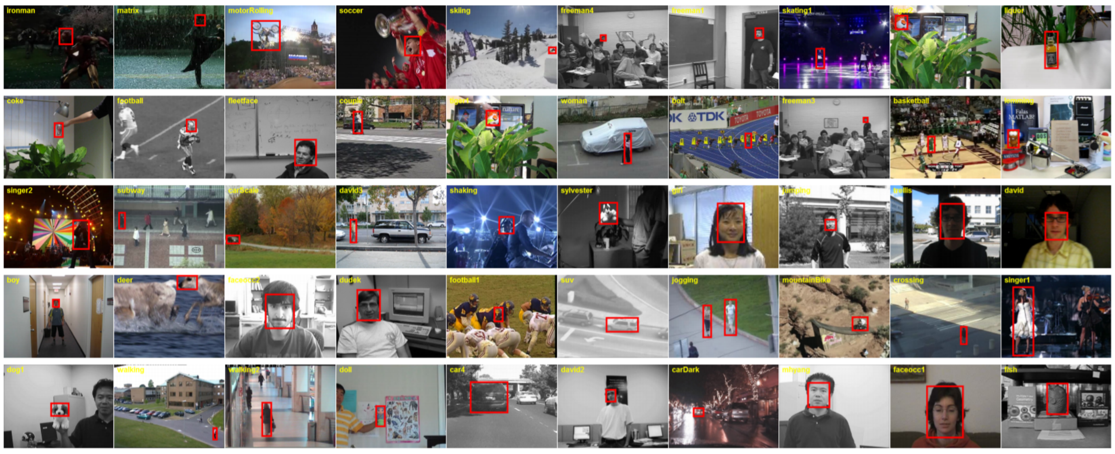
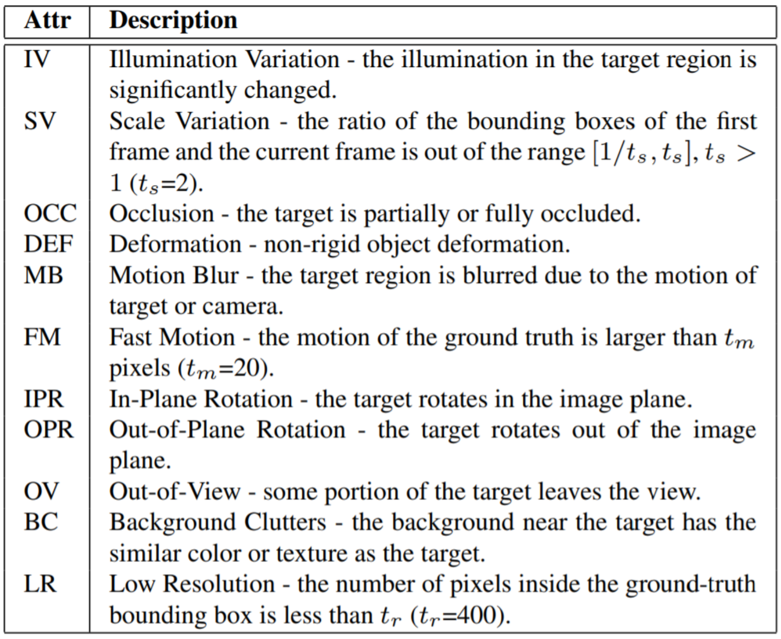


FIGURE 2.22: Tracking sequences for evaluation. The first frame with the bounding box of the target object is shown for each sequence. The sequences are ordered based on the ranking results (See (Wu et al., 2013)): the ones on the top left are more difficult for tracking than the ones on the bottom right. Note that two targets are annotated for the jogging sequence. This figure is from (Wu et al., 2013).



Attr	Description
IV	Illumination Variation - the illumination in the target region is significantly changed.
SV	Scale Variation - the ratio of the bounding boxes of the first frame and the current frame is out of the range $[1/t_s, t_s]$, $t_s > 1$ (t_s=2).
OCC	Occlusion - the target is partially or fully occluded.
DEF	Deformation - non-rigid object deformation.
MB	Motion Blur - the target region is blurred due to the motion of target or camera.
FM	Fast Motion - the motion of the ground truth is larger than t_m pixels (t_m=20).
IPR	In-Plane Rotation - the target rotates in the image plane.
OPR	Out-of-Plane Rotation - the target rotates out of the image plane.
OV	Out-of-View - some portion of the target leaves the view.
BC	Background Clutters - the background near the target has the similar color or texture as the target.
LR	Low Resolution - the number of pixels inside the ground-truth bounding box is less than t_r (t_r=400).

FIGURE 2.23: List of the attributes annotated to test sequences. The threshold values used in this benchmark are also shown. This figure is from (Wu et al., 2013).

predicted and ground truth bounding boxes exceeds a given threshold ratio which varies from 0 to 1. This benchmark reports on one-pass evaluation (OPE), i.e. the tracker is run throughout the whole video initialised only with the ground truth in the first frame. To rank the performance, this benchmark uses the precision obtained for a location error threshold of 20 pixels as the precision score for precision plot. For success plot, the area under curve (AUC) is used as the success score.

2.6.2 VOT2015 Benchmark

The dataset consists of 60 sequences (see Figure 2.24) which are selected from a large pool of sequences combined from existing datasets CVPR2013 benchmark (Wu et al., 2013) (50 sequences) and ALOV (Smeulders et al., 2014) (315 sequences), PTR (Vojir, Noskova, & Matas, 2014) and other sources, which makes sure that the Visual Object Tracking 2015 (VOT2015) (Kristan et al., 2015) dataset is a representative set of challenging sequences. The ground truth provided by the dataset is slightly more flexible; rotated bounding boxes described by the coordinates of their four corner points. However, the bounding boxes are still not able to fully exclude background pixels. The dataset tries to make the bounding box contain at most about 30% background pixels. Each frame of the dataset is labelled with five visual attributes, which are occlusion, illumination change, motion change, size change and camera motion. Any frame which doesn't show any of these five attributes is labelled as unassigned.

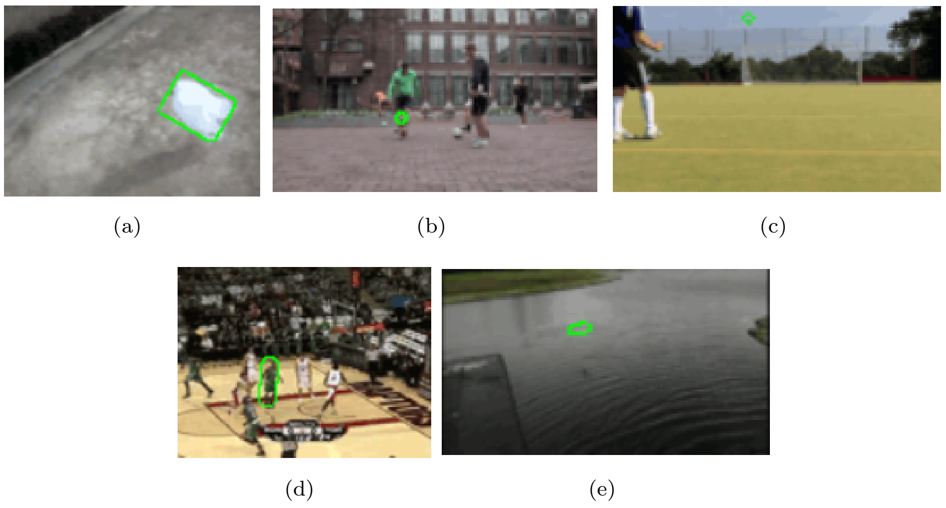


Figure 2.24: Examples of tracking sequences in (Kristan et al., 2015).

The main difference between the evaluation mechanism of the VOT2015 and that of the CVPR2013 benchmark is that VOT2015 allows for re-initialisation. In VOT2015, re-initialisation happens when the overlap between the estimated bounding box and the ground truth bounding box of the target reduces to zero.

Performance on the VOT2015 dataset is measured by two weakly correlated measurements, *accuracy* and *robustness* (Kristan et al., 2015). The accuracy represents how well the predicted bounding box overlaps with the groundtruth bounding box. While robustness means the number of tracking failures when the tracker loses the target. Each tracker evaluated on the VOT2015 dataset is performed on each sequence 15 times to guarantee good statistics. The VOT2015 benchmark (Kristan et al., 2015) also introduces the *expected average overlap* as a new metric to rank tracking algorithms; it combines the raw values of per-frame accuracies and failures in a principled manner and it provides a clear interpretation of the accuracy and the robustness. The expected average overlap estimates how accurate the estimated bounding box is after a certain number of frames are processed since initialisation.

2.7 Conclusion

This section has reviewed the main concepts and challenges in visual tracking, and then explained the three main components of tracking: appearance model, motion model and search strategy. Especially, this section has analysed the algorithm components, advantages and drawbacks of the state-of-the-art tracking methods with structured part-based models (Adam et al., 2006; M. Yang et al., 2007; Shahed Nejhum et al., 2008; Kwon & Lee, 2009; L. Zhang & van der Maaten, 2013; Yao et al., 2013; Cehovin et al., 2013; T. Zhang et al., 2014; Liu et al., 2015).

- These part-based methods rely on a template likelihood strategy to estimate target location. Background is often included in the target template, which tends to deteriorate the template when continuously varied background appear during tracking.

- It is found that these part-based methods are capable of handling partial occlusions to a degree, and they are robust to small deformations. However, they still cannot handle non-rigid objects with severe or complex movements. Most of these methods adopt rigid structures or a star model, and there are no spatial constraints between adjacent parts. Rigid structures and the star model are sub-optimal in modelling target structure when the target is deformable and articulated, as adjacent parts of a target are more likely to move toward the same direction with the same speed. Utilising the spatial relations between parts will be beneficial to tracking. It means that an extra component, shape, is necessary for general object tracking.

- Among these methods, the part-based model proposed by (Kwon & Lee, 2009) has the most flexible shape, and it includes the least background in target. It can be used as a starting point to design an improved part-based model in this thesis.

Chapter 3

Potential Benefits of Part-based Models

After theoretical analysis of the advantages and problems of current part-based models in tracking, the next step is to perform practical analysis of part-based models in tracking. According to the literature review on part-based visual tracking, the part-based model of the tracker Adaptive Basin Hopping Monte Carlo(A-BHMC)(Kwon & Lee, 2009), has the most flexible model and is closest to this research's goal. A-BHMC is different from the rigid definitions of other part-based models used in tracking (Adam et al., 2006; M. Yang et al., 2007; Shahed Nejhum et al., 2008; L. Zhang & van der Maaten, 2013; Yao et al., 2013). The part-based model of A-BHMC consists of small rectangles, only containing a little background, and the shape of the part-based model of A-BHMC tends to vary with the whole target shape. Thus, the A-BHMC tracker was used as a starting point for further comparative experimental research.

3.1 Initialising Part-based Trackers

When designing a part-based model, the first issue is how to initialise the parts, which defines the parts used during tracking. The initialisation of parts is crucial, because it is the first step in tracking and all the information used in the following automatic tracking is obtained in this step. Any error introduced during the initialisation stage will significantly affect the accuracy of subsequent tracking.

As shown in Figure 2.16(c), the parts initialised in the A-BHMC tracker are many distributed small rectangles (Kwon & Lee, 2009). Their initial positions are decided based on the condition number K of the Hessian matrix, as described in Section 2.5.

This makes it hard to decide the number of parts, and the parts are very likely to include background areas that do not provide target information.

Boykov and Jolly (Boykov & Jolly, 2001) propose an initialisation method in which the user manually draws strokes with the mouse to label the pixels of the object and the background as clues for image segmentation. Moschini and Fusiello (Moschini & Fusiello, 2008) and Guo et al. (Guo, Xu, & Tsuji, 1994) use a stick figure for tracking human bodies. The stick figure comprises stick components representing human body parts and joints linking these components together (as shown in Figure 3.1 (Moschini & Fusiello, 2008)).

Inspired by these works, this thesis proposes an initialisation method for model-free online tracking using stick figures. This method allows the automatic growth of parts from key points which are selected from skeleton lines drawn by the user. The parts obtained by this initialisation method are shown in Figure 3.2(e).

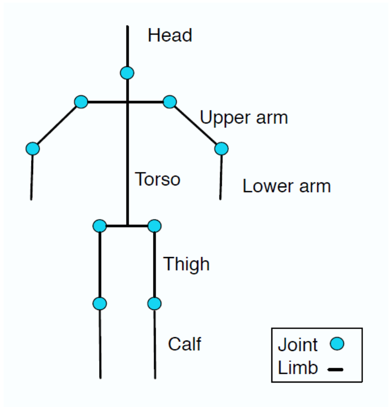
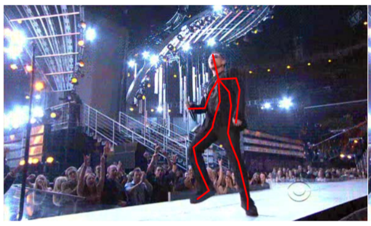
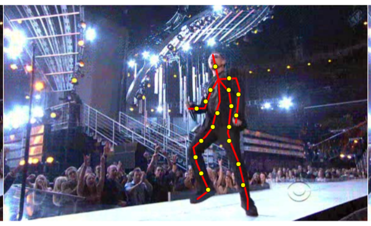


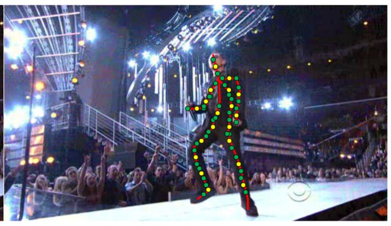
FIGURE 3.1: The stick figure body model employed in (Moschini & Fusiello, 2008).

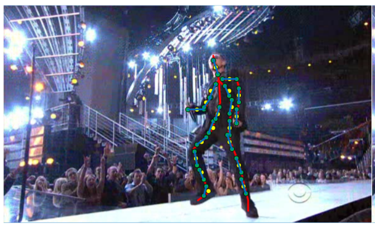
Parts are grown based on the stick figure as follows:

- The user draws free-form lines to label the tracking target, forming a stick figure of the target, which is similar to giving it a skeleton structure (shown as the red lines in Figure 3.2(a)).
- Given the stick figure, the intensity gradient values of all pixels on the stick figure are calculated. The point with the highest gradient value is then selected as the first base point. The point with the second highest gradient value is selected as the second base point. In total 30 base points are chosen in this way. These base points should keep a certain distance between each other because they are intended to form the centres of potential parts. When examining a point with a high gradient value, the distances from the point to all existing base points are measured. If the minimum of these distances is less than 10 pixels, the examined point will be skipped. The base points selected are the yellow dots shown in Figure 3.2(b).


(a) Drawing lines


(b) Base points


(c) Extended points


(d) Base lines

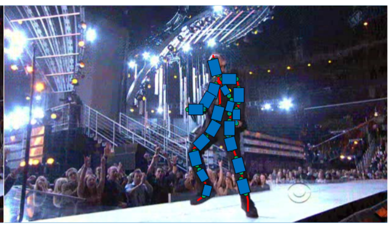

(e) Grown parts

FIGURE 3.2: The initialisation of parts using the proposed stick figure method. (a) The drawing lines are in red. (b) The yellow points are the base points with a high gradient value. (c) The green points located at both sides of a yellow point are the extended points. (d) The blue lines linking two extended points of a yellow base point are the base lines. (e) The blue parts are grown from the base lines.

- Along the stick figure, for each base point, find a pair of points which are each four pixels away from that base point in opposite directions along the skeleton. These pairs of points are called extended points, shown as the green dots in Figure 3.2(c).

- Link each pair of extended points with a line segment. These line segments are base lines (shown as the blue line segments in Figure 3.2(d)).

- For each base line, examine its neighbouring parallel line segments, and check their HSV histograms, as shown in Figure 3.3. If the HSV histogram similarity (measure by the Bhattacharyya distance) between a neighbouring line segment and the base line is higher than a predefined threshold σ ($\sigma = 0.65$ in this experiment, determined empirically and fixed during the whole experiment), the line segment is regarded as a section of the part being grown from the base line. Only neighbouring parallel line segments located at most ten pixels away from the base line are considered.

Thus every part grows to be a small rectangle gradually, as shown in Figure 3.2(e). After growing the parts, initialisation is finished.

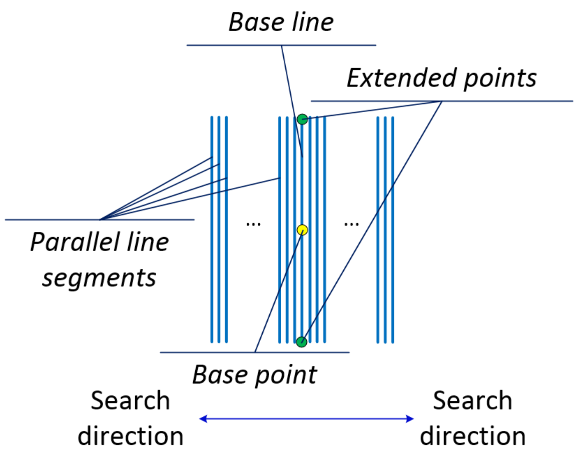


FIGURE 3.3: The illustration of searching similar line segments of a base line.

3.1.1 Tightly Initialised Targets Improve Tracking Performance

The traditional way of initialisation in tracking is that a user draws a bounding box to locate the target area in the first frame. The ground truth of existing datasets for visual tracking (such as the CVPR2013 benchmark (Wu et al., 2013), VOT2014 and VOT2015 benchmark (Kristan et al., 2014, 2015)) is presented as parameters of the bounding box. Bounding-box-based initialisation is accepted as a proper initialisation method for visual tracking. It is however claimed in this thesis that stick figure initialisation is more appropriate, as the target initialised through the proposed stick figure method includes less background and represents target appearance information more accurately.

To examine the benefit of this stick figure initialisation method, a comparative experiment is designed to compare the performance of three different initialisation methods; bounding box initialisation, the part-based initialisation used in the A-BHMC tracker and the proposed stick figure initialisation.

Inspired by A-BHMC (Kwon & Lee, 2009), which is a part-based improvement of the MCMC method, a standard MCMC method was implemented using Matlab. The different initialisation methods would then be added on top of the standard MCMC method and tested separately.

Parts were initialised with either the Hessian matrix method of A-BHMC or the proposed stick figure method. The target location is the centre of the smallest bounding box enclosing all parts. Note that the part's location only depend on the target location. The proposals made by MCMC are based on the target location as well. The width and height of the distributed parts are several pixels, usually less than the width or height of the whole object.

	Centre errors of MCMC-tracker with different initialisations (pixels)		
Videos	Bounding box	Hessian matrix	Stick figure
Diving	34.6539	26.9265	**20.0078**
Gymnastics	**14.4942**	27.3005	14.9493
High Jump	44.0417	42.9666	**42.3133**
Boy	7.9611	8.3652	**6.0101**
MountainBike	14.3928	15.0809	**11.9879**
MotorRolling	171.7854	165.0882	**134.4405**
Ironman	214.8088	281.9014	**56.2587**

TABLE 3.1: The centre errors of standard MCMC method with bounding box, Hessian matrix and the proposed stick figure initialisation over seven videos. The values in bold are the best result for each video.

The three methods were applied to seven video sequences and their centre errors of whole objects were compared. The centre error is the Euclidean distance between the predicted target location (the centre location of the bounding box enclosing a whole object) and the manually labelled real location, namely ground truth, of the whole object. Each method was run for five times on each video. Results are shown in Table 3.1. In Table 3.1, bold figures represent the best results. It shows that, the tracker with the proposed stick figure initialisation method achieved the best results for six test videos. The proposed method also achieved second best performance on the remaining video sequence.

The experimental results clearly show that the initialisation method, indeed has an effect on tracking performance. The MCMC with stick figure initialisation has shown the best performance in the comparative experiment. One hypothesis is that the stick figure initialisation method focuses on the target area and includes little background as foreground, while bounding box and Hessian matrix initialisation methods usually treat some background areas as target. Given the common bounding-box-based ground truth, although the proposed stick figure requires a little extra effort to initialise the target, the improvement in tracking performance is unignorable. This initialisation method has been improved to be automatic, based on a segmentation technique, which is explained in section 4.4. It is also shown in section 3.2 and Chapter 4 that a direct displacement prediction technique works best in areas relatively close to the target, which verifies that it is critical the initialised tracking target focuses on the foreground area. It is confirmed that the stick figure initialisation method, which focuses on the target area, improves tracking performance.

3.1.2 Exploration Advantages and Disadvantages of the Target Parts Initialised by Stick Figure

Having examined the effect of stick figure initialisation on a standard MCMC tracker, we now explore the performance of the bounding box initialisation working with a non-part-based tracker (MCMC) and a part-based tracker (A-BHMC) , and the performance of stick figure initialisation working with the above two trackers. The advantages and disadvantages of a part-based representation determined by the stick figure initialisation, can be concluded from the experiment.

As for concrete experiment, it is conducted to compare four methods which are examining the bounding box and the stick figure initialisation with MCMC and the part-based tracker A-BHMC (Kwon & Lee, 2009), respectively. The four methods are noted with B-MCMC, SF-MCMC, B-A-BHMC and SF-A-BHMC, where B stands for bounding box and SF stands for stick figure.

This experiment tests 35 video sequences from the CVPR2013 tracking benchmark(Wu et al., 2013), namely the $Baseketball$, $Bolt$, Boy, $CarDark$, $CarScale$, $Coke$, $Couple$, $Crossing$, $David$, $David3$, $Deer$, $Doll$, $FaceOcc1$, $Football1$, $Girl$, $Ironman$, $Jogging$, $Lemming$, $Liquor$, $Matrix$, $MotorRolling$, $MountainBike$, $Shaking$, $Singer1$, $Singer2$, $Skating1$, $Skiing$, $Soccer$, $Subway$, $Tiger1$, $Tiger2$, $Trellis$, $Walking$, $Walking2$ and $Woman$ sequences. There are two targets in video $Jogging$, which are actually two running persons located at the left side and right side respectively in each frame of the video sequence. The $Jogging$ video can be denoted as $JoggingLeft$ and $JoggingRight$ to describe different tracked target. It is equivalent to 36 videos in total. Every method is run on the 36 videos respectively. For each video, each method is run for five times.

This research follows the conventional way of evaluating a tracker's performance, which is to run the tracker on the test video sequence given the initialisation of the ground truth target position in the first frame. The initialisation in bounding box form is provided by the CVPR2013 benchmark dataset. Initialisation by the stick figure method needs to be generated as explained in Section 3.1.

As standard datasets in visual tracking give bounding box ground truth, the bounding box ground truth is used in this experiment. Specifically, this experiment adopts centre position of the bounding box as the tracking result. For part-based tracker, the tracking result is also based on bounding box which is the minimal rectangle covering all distributed parts of the whole object. Table A.1, Table A.2, Table A.3 and Table A.4 show the centre error averaged over a whole video sequence for each video each run. The figure in bold is the best tracking result of the corresponding method for each video. μ is the mean of the average centre error of different runs for each video. σ is the standard

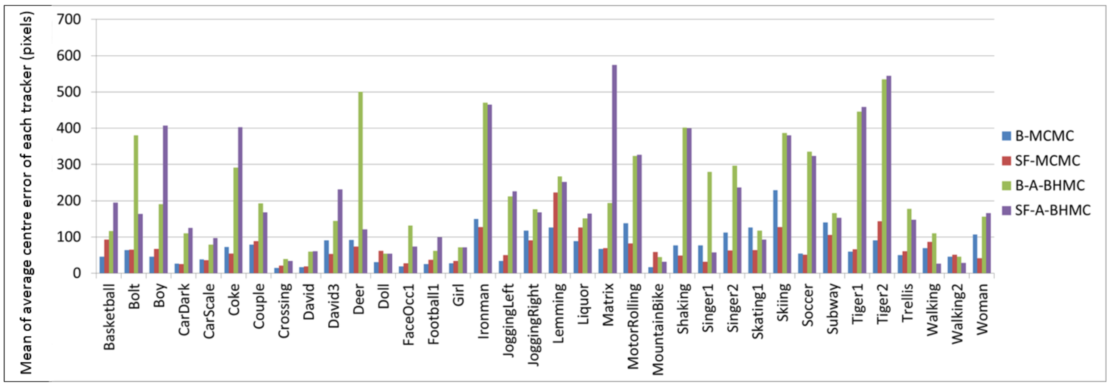


FIGURE 3.4: The values of y-axis correspond to means of the average centre error of four trackers, B-MCMC, SF-MCMC, B-A-BHMC and SF-A-BHMC, running five times on each video.

deviation of the average centre error of different runs for each video. The means (μs) of the results of four methods corresponding to different videos in above four tables are compared and shown in Figure 3.4.

According to experimental results, B-MCMC achieves the best results for 18 video sequences, and SF-MCMC obtains the best results for 16 video sequences. SF-A-BHMC achieves the best results for remaining 2 videos. It has shown that the stick figure initialisation method has competitive performance compared to the bounding box initialisation method.

In SF-MCMC, parts can rotate based on a Gaussian model, as the real target may rotate. When starting tracking using MCMC, in each frame, proposals of the position of target's centres are made based on the best prediction of the target location in previous frame. Similarly, the rotation α of all samples of a part are sampled based on the corresponding part of the best sample of the whole object in the previous frame via a Gaussian distribution:

$$\alpha(i, t+1) = \alpha(i, t) + 30 * \pi/180 * G(0, 1), \tag{3.1}$$

where $G(0, 1)$ randomly generates a number from 0 to 1 following a Gaussian distribution. i is the part index and t is the frame index. Although the part can rotate based on the corresponding part of the best sample following a Gaussian distribution, in SF-MCMC, each part's position is only dependent on the centre of the target, like the star model (Fergus, Perona, & Zisserman, 2005; Leibe, Schindler, Cornelis, & Van Gool, 2008).

Thus, in SF-MCMC, parts cannot constrain or communicate with each other. Adopting the star model to describe spatial relationships among parts does not allow us to represent the structure of the target. The flexibility of a star model leads to a very loose part structure, so that SF-MCMC sometimes performed worse than the traditional method B-MCMC. Also, as parts are grown utilising histogram similarity, the part cannot be grown to be an appropriate size when the histogram similarity is not that high locally. Thus the parts initialised would be sparsely distributed over the target area and thus not enough target information can be obtained for tracking, as shown in Figure 3.5.

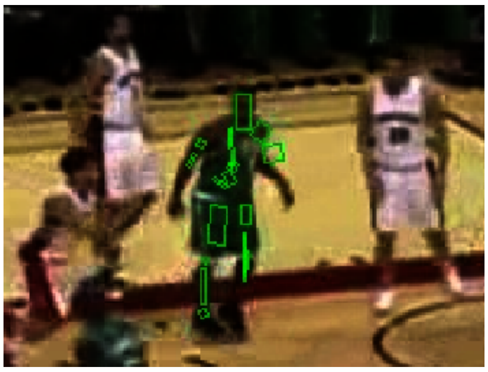
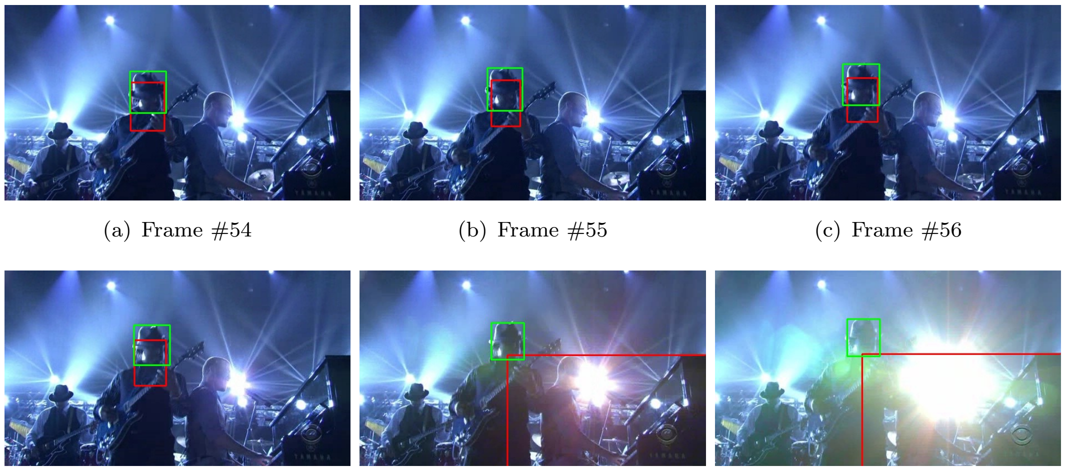


FIGURE 3.5: The initialised parts (green rectangles) for video *basketball* by the proposed stick figure method.



(a) Frame #54 (b) Frame #55 (c) Frame #56

(d) Frame #57 (e) Frame #58 (f) Frame #59

FIGURE 3.6: Illumination variation leads to loss of target in tracking. This is the result of SF-A-BHMC tracker for *Shaking* video. The green rectangle is the ground truth, while the red rectangle is the predicted target area.

The part-based methods, B-A-BHMC and SF-A-BHMC, perform worse than the other two methods except on two videos, *Walking2* and *Woman*. MCMC does not update the template while A-BHMC has the strategy of updating the template in every frame. However, in most of the test videos, tracking has problems with occlusion or illumination variation, which would introduce error to the template after updating. Once the template is updated to be a new one with a small error, the error will be accumulated and the tracker will never find the correct target again. Figure 3.6 shows the result of

SF-A-BHMC for the *Shaking* video sequence, which illustrates that illumination variation leads to loss of target in tracking because of incorrect updating. Note that there are no spatial relationships between the parts in SF-A-BHMC either.

3.1.3 Conclusion

The first comparative experiment was performed to verify the benefit of the proposed stick figure initialisation method, which determines the parts. It has confirmed that the stick figure initialisation method, which is less likely to include background in the foreground information, has better performance than bounding box initialisation and the part-based initialisation method used in A-BHMC. Both of the last two methods (bounding box initialisation and part-based initialisation method used in A-BHMC) include the background in the foreground model to different degrees. The second experiment was performed by combining the stick figure initialisation and bounding box initialisation with MCMC and A-BHMC separately. The similar performances of B-MCMC and SF-MCMC have shown that stick figure initialisation has competitive performance. However, from the experimental results, some problems which need to be addressed can be identified as well.

Firstly, the parts are not initialised properly. As shown in Figure 3.5, parts do not include background but are not grown to be an appropriate size which can fully cover the target's prior information. Secondly, the parts lack an explicit relationship between each other. Non-rigid objects usually have a flexible but connective structure. One part's movement will lead to its neighbours' movements. When one part rotates, its related parts are likely to rotate to a similar degree. Therefore, parts which are only dependent on the centre tend to move too flexibly, which may lead to drift. Thirdly, updating the appearance model sometimes introduces error to the template. The template is critical in tracking using a template likelihood strategy. Templates with even a small error are highly likely to lead to incorrect optima. In addition, the target usually changes appearance constantly throughout the video. Many factors, such as illumination, movement and scale variations, will cause appearance transformations. Thus, seeking an appearance model which is robust to the variation of target's appearance and accurate to describe the target is critical to correct tracking.

3.2 Preliminary Experiments on Direct Displacement Prediction Strategies

It is necessary to generate an approach which can effectively model the target's appearance and shape as well as correctly predict parts location. A successful method, Local Evidence Aggregation based on regression (LEAR) (Martinez et al., 2013), has shown competitive performance on facial point detection. This method relies on the ability of a local image patch evaluated by a pre-learned regressor to provide an estimate of the displacement from it to the target location, which is the Direct Displacement Prediction. In doing so, local image patches contribute to the solution by directly voting for the target (part) location.

More precisely, given a test location in an image, a feature descriptor is extracted from an image patch centred at it. The feature descriptor is evaluated by a regression method to estimate the relative position of the target facial point with respect to the image patch. The regression is performed on the image patch to predict the horizontal and vertical components Δx and Δy of the displacement vector pointing from the test location to the true target location. LEAR uses Support Vector Regression (SVR) as the regression algorithm. This algorithm learns a separate regressor for each facial point to estimate the point location of the target face. Stochastically selected local appearance information is evaluated by the regressor to give estimates of potential target location. These estimates are aggregated to form a single robust prediction of target location. In addition, LEAR (Martinez et al., 2013) uses the facial point definitions commonly used for facial expression recognition, as shown in Figure 3.7. Representing the face with facial points instead of a bounding box avoids background information. Facial points can also provide a more accurate face localisation and capture the shape of the face. Adopting this point-based definition of parts in tracking has the potential benefits of avoiding background noise and defining the target shape using points locations directly without extra effort.

LEAR is able to detect an arbitrary set of facial points with the requirement that the target points must have a distinctive local texture. This requirement is easy to achieve in facial point detection as local areas around facial points usually contain identifiable texture. Variations in the appearance of faces are also typically smaller than those of the general objects considered in visual tracking. Regressors for all the facial points in LEAR, and the facial shape model, are pre-learned from approximately 1000 manually annotated images. However, offline training is not an option for model-free visual tracking. The research question addressed here is whether LEAR's approach can be extended

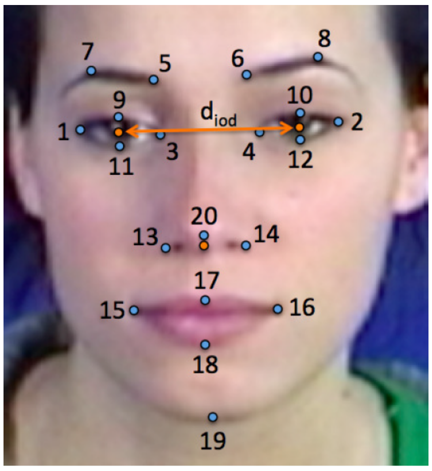


Figure 3.7: The twenty facial points used in LEAR (Martinez et al., 2013) (blue) and the centre points for the eyes and nose (orange). The inter-ocular distance d_{iod} is the Euclidean distance between the centre eye points.

to model-free tracking. This chapter now presents preliminary experiments to explore the performance of the direct displacement prediction technique in visual tracking.

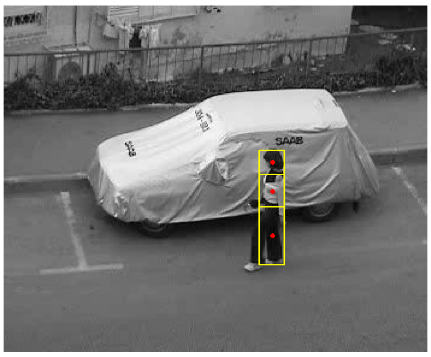


Figure 3.8: The part-based representation of a target. Yellow rectangles are different parts and red dots are their centres.

Since this research's goal is to track general (usually deformable) objects, the human is selected as an example target in preliminary experiments, as it is one of the most common classes of deformable objects. Note that the models used are not trained for any specific class of object. Specifically, the target is divided into three meaningful parts; three parts are used as a simple part-based representation of the whole target. As shown in Figure 3.8, the first part covers the head, the second part covers the upper body and the third part covers the lower body. The centres of these three parts are target points in tracking and they are tracked separately. The test videos in preliminary experiments are selected from the CVPR 2013 benchmark (Wu et al., 2013), and this benchmark provides bounding-box-based ground truth for targets. The ground truth of the target points are obtained from the bounding box ground truth using the parts' definition shown in Figure 3.8.

3.2.1 Qualitative Experiments

The initial experiment is designed to test whether the position of a general target point can be decided by predicting the direction vector of samples around the point. The idea is to train a regressor modelling the relationship between local image information and relative displacement to the target, and then examine the image information used for training to see whether the predicted displacement is consistent with the displacement used for training (see Figure 3.9). If they are consistent, it means that the target point's location can be predicted by the estimation of displacement vector through a regression technique. If not, the estimation of the displacement vector is not reliable in visual tracking.

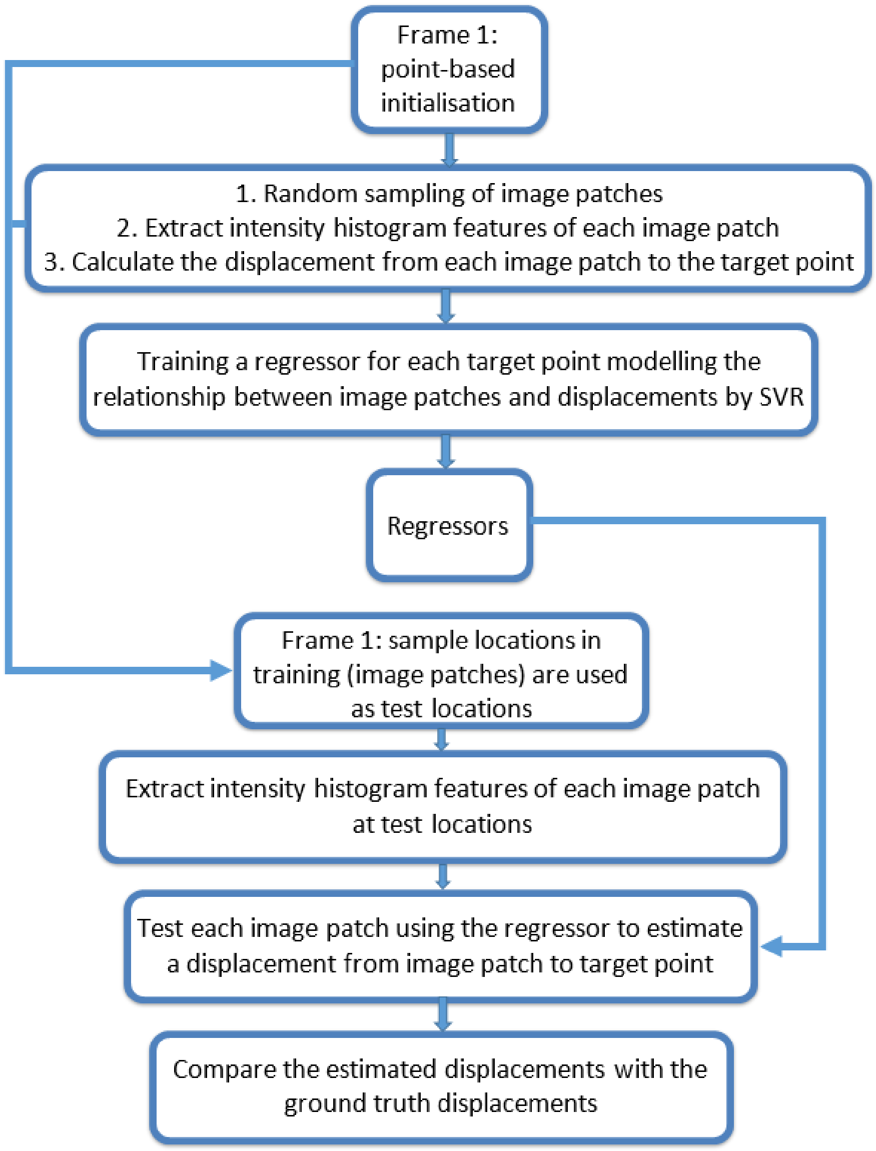

Frame 1: point-based initialisation
1. Random sampling of image patches
2. Extract intensity histogram features of each image patch
3. Calculate the displacement from each image patch to the target point
Training a regressor for each target point modelling the relationship between image patches and displacements by SVR
Regressors
Frame 1: sample locations in training (image patches) are used as test locations
Extract intensity histogram features of each image patch at test locations
Test each image patch using the regressor to estimate a displacement from image patch to target point
Compare the estimated displacements with the ground truth displacements

FIGURE 3.9: The flowchart of the initial experiment.

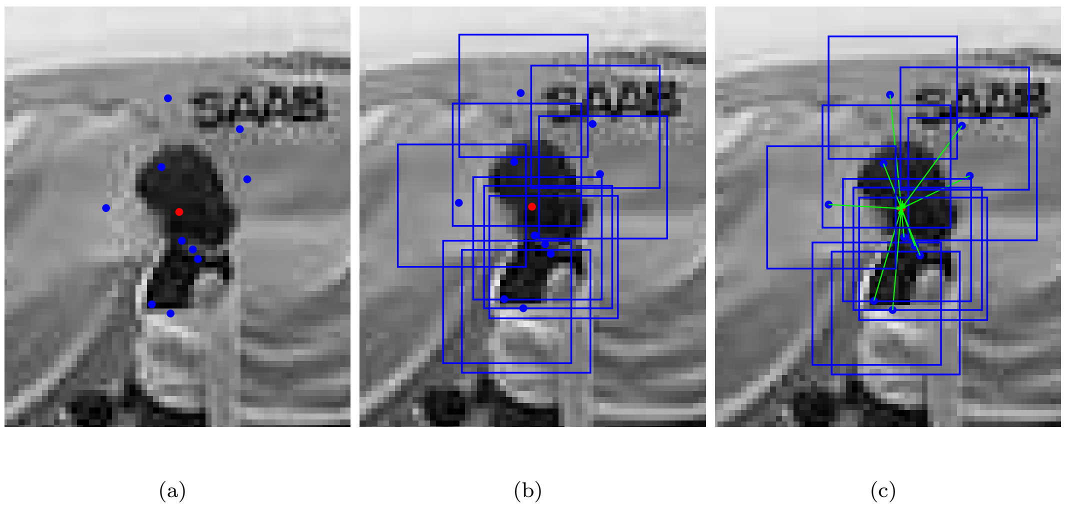


Figure 3.10: The samples around a target point and their corresponding displacements. (a) The random sampling locations around a target point. The red dot represents the target point and blue dots are sample locations. (b) Blue rectangles are image patches centred at sample locations. (c) Blue dots are centres of samples (sample locations) and green arrows represent ground truth displacements from sample locations to target.

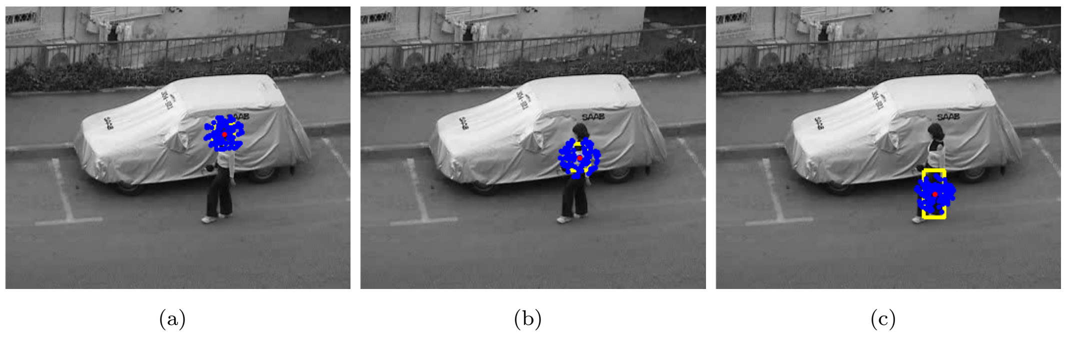


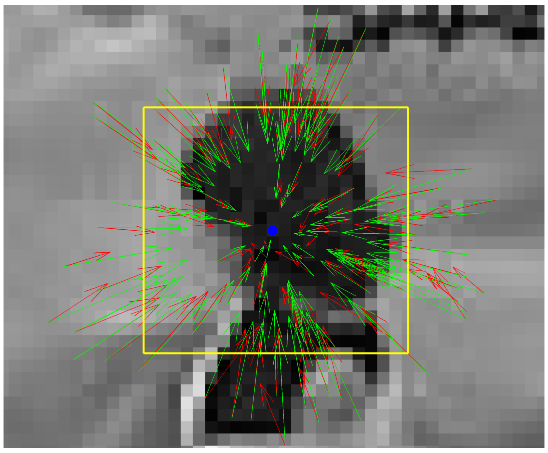
Figure 3.11: Training sample locations (blue dots) of different parts of the target. The red dots are centres of three parts of the target.

The video used in qualitative experiments is video $Woman$. The difficulty of this video is classed as mid-level, based on the ranking in the CVPR 2013 benchmark (Wu et al., 2013). Given the ground truth of the three target points in the first frame of a video sequence, for every target point, N sample locations are obtained through uniform random sampling within a circular area of radius r centred on the target point, as shown in Figure 3.10(a) and Figure 3.11. The sampling radius r is 20 pixels and the number of samples N is 100 in the initial experiment, which are decided empirically. Features are extracted from the image patch (as shown in Figure 3.10(b)) centred at every sample location. The image patch is in the same size as its corresponding target part. The feature used in the preliminary experiments is the intensity histogram. The displacement is a vector from the sample location (the centre of the image patch) to the target. The features and the displacements are used for training. Support Vector Regression (SVR) is used as the regression technique, as in LEAR, although other regression algorithms are also applicable. Taking a feature vector as input, an SVR can predict a 1-dimensional real-valued prediction as the output. To predict the displacement vector, which is a 2-dimensional vector, two regressors are trained separately for each point. Sample locations are then treated as test locations and their features are evaluated by the learned regressor to estimate displacement vectors, which ideally will point from test locations to the target point. This process is done for the three target points separately. Estimation and ground truth of displacement vectors are then compared. Comparative results of three parts of a target are shown in Figure 3.12. The results are promising, and have shown that, for all three parts, most estimated displacement vectors are consistent with their ground truth in direction and distance. The qualitative results in Figure 3.12 show that predictions from sample locations which are relatively close to the target tend to be more consistent with the ground truth than the predictions from the farther sample locations.

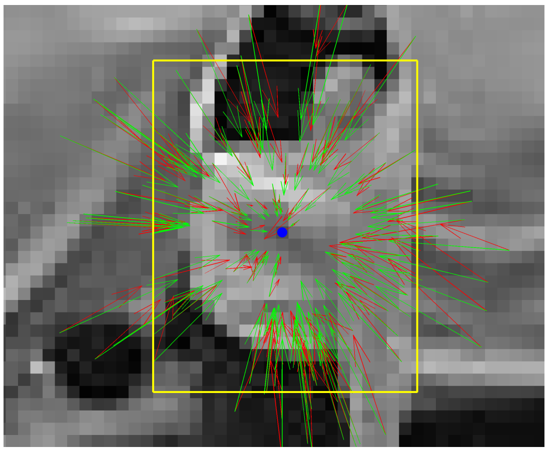
Further experiments are designed to assess the performance of the displacement-based technique. Specifically, in Experiment 3.1 the regressor is trained in the first frame, and then it is tested in frames 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 30, 40 and 50 (see Figure 3.13). This is to explore the robustness of the regressor without re-training. In Experiment 3.2 the regressor is trained in one frame and is tested in the next frame. Specifically, the regressor is trained in frames 1, 9, 14, 19, 24, 29 and 49, and tested in frames 2, 10, 15, 20, 25, 30, 50 (see Figure 3.14). This is to investigate the robustness of the regressor with correct re-training every frame.

Each evaluation is run five times to reduce random effects. The method and parameters of training the regressor are the same as the initial experiment described at the beginning of Section 3.2.1. After training, the regressor is tested at grid locations (with the grid spacing of four pixels) spanning the whole frame. The qualitative results of Experiment 3.1 and Experiment 3.2 , the predicted displacements and the ground truth displacements for the second part of the target, are illustrated in Figure 3.15 and Figure 3.16 for a single run of the experiment.

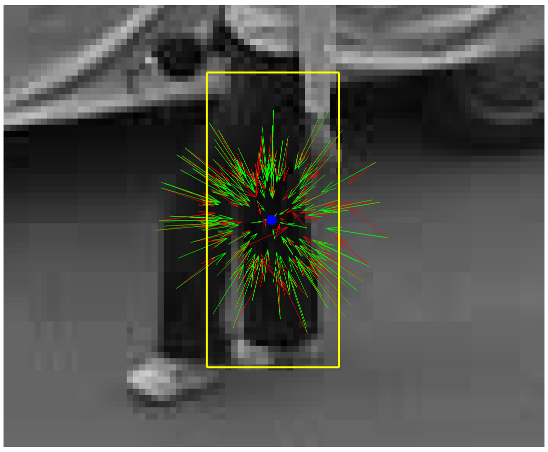
In Figure 3.15, it can be seen that, with the regressor trained in the first frame, most predictions from frames 2-10 are consistent with the corresponding ground truth. In frames 15, 20, 25, 30, 40, 50, fewer and fewer estimated displacements are consistent with their ground truth. Please note that, there are some estimated displacement vectors from the target area which are always consistent with their ground truth, as they are located on the central part of the foreground whose appearance variation is relatively



(a) The first part



(b) The second part



(c) The third part

Figure 3.12: Comparison between predictions from regressor and the ground truth of displacements from sample locations to target location. Yellow rectangles are three parts of the human. Blue dots are centres of parts, i.e. target points. Green arrows are ground truth displacements from sample locations to target location. Red arrows are predicted displacements. For clarity, arrows are scaled for display.

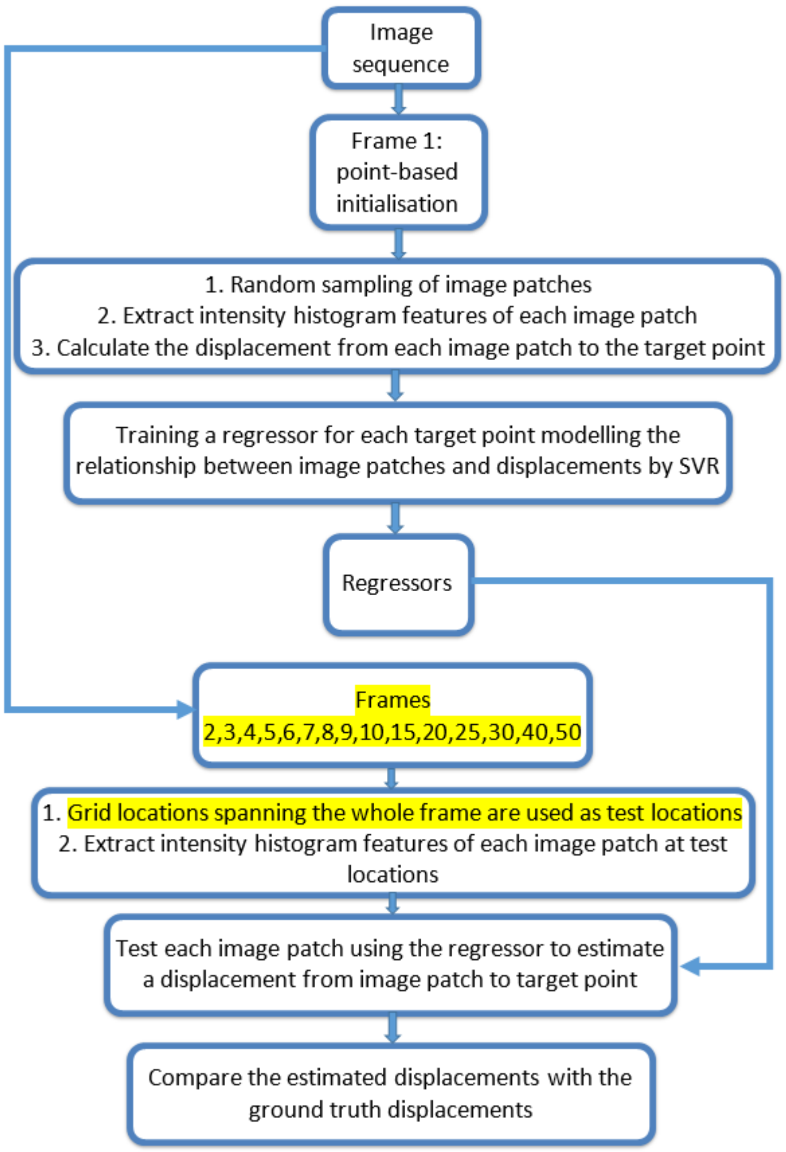


Image sequence
Frame 1: point-based initialisation
1. Random sampling of image patches
2. Extract intensity histogram features of each image patch
3. Calculate the displacement from each image patch to the target point
Training a regressor for each target point modelling the relationship between image patches and displacements by SVR
Regressors
Frames 2,3,4,5,6,7,8,9,10,15,20,25,30,40,50
1. Grid locations spanning the whole frame are used as test locations
2. Extract intensity histogram features of each image patch at test locations
Test each image patch using the regressor to estimate a displacement from image patch to target point
Compare the estimated displacements with the ground truth displacements

FIGURE 3.13: The flowchart of Experiment 3.1. The differences between the initial experiment and Experiment 3.1 are labelled with yellow background.

small. In Figure 3.16, with the regressor trained in the previous frame, most predictions near the target in each frame are consistent with their ground truth.

Experimental results show that the direct displacement prediction technique possesses a reasonable level of robustness over time. Especially with correct updating, the direct-displacement-based regression has the potential to work for a long time. However, in practice, update processes do not always have access to perfect ground truth. It is clear that predictions degrade as the test locations go further away from the target, but it is unclear how far they can go; this needs to be explored in the quantitative experiments. Test locations in the foreground can consistently give correct predictions over time. Most test locations in the background close to the target can provide correct

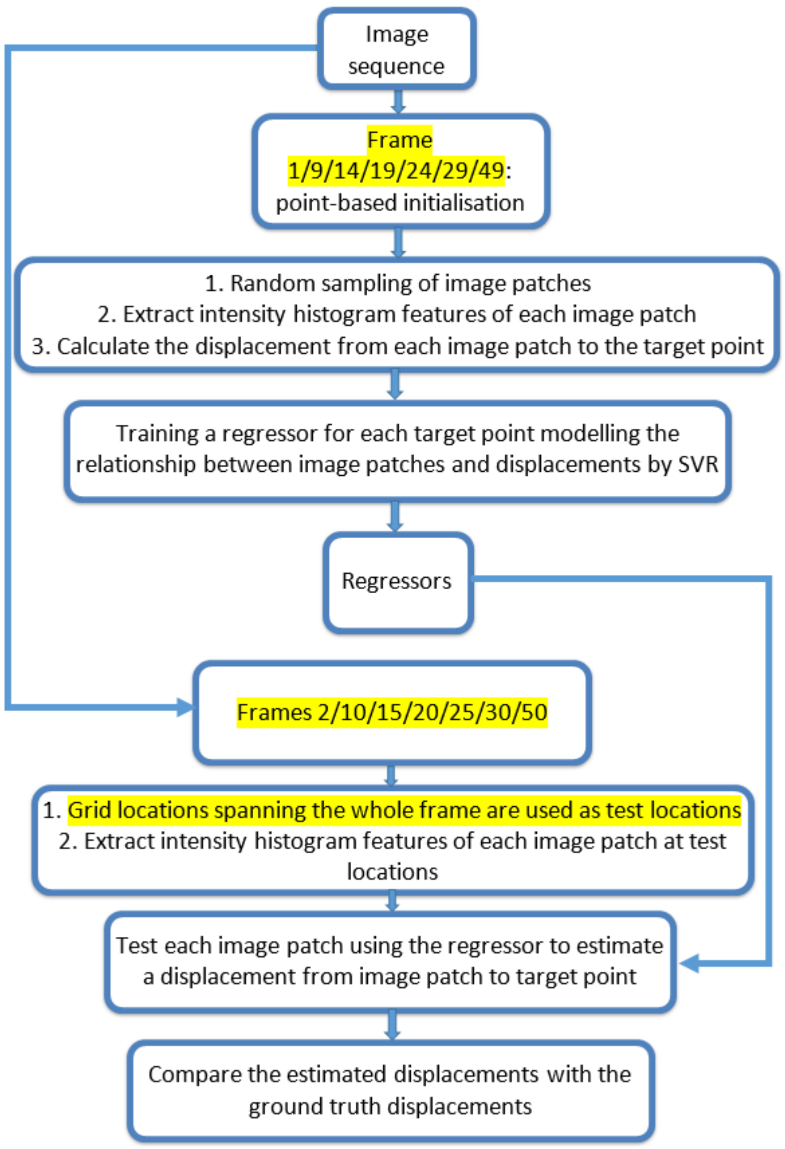
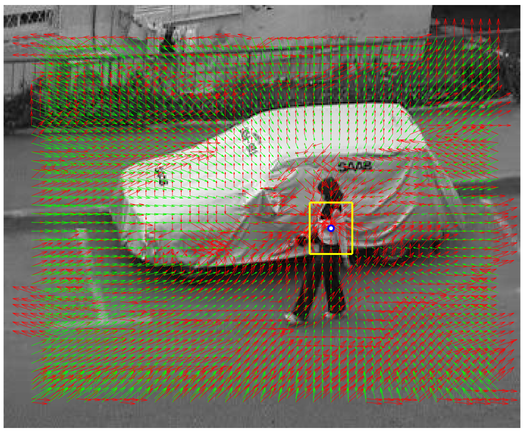
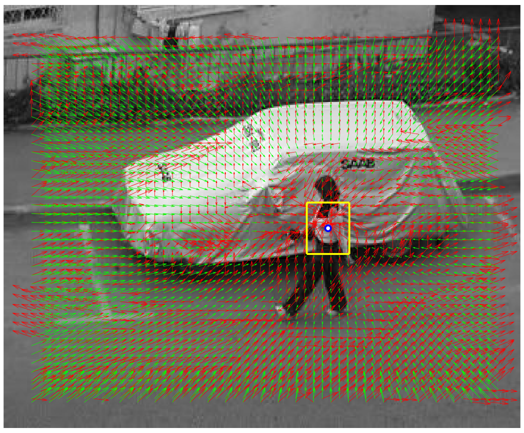


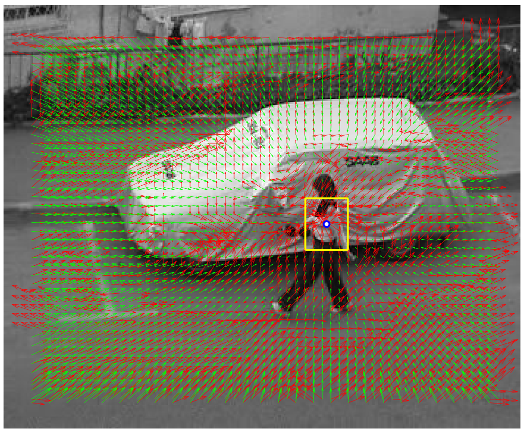
Image sequence
Frame 1/9/14/19/24/29/49: point-based initialisation
1. Random sampling of image patches
2. Extract intensity histogram features of each image patch
3. Calculate the displacement from each image patch to the target point
Training a regressor for each target point modelling the relationship between image patches and displacements by SVR
Regressors
Frames 2/10/15/20/25/30/50
1. Grid locations spanning the whole frame are used as test locations
2. Extract intensity histogram features of each image patch at test locations
Test each image patch using the regressor to estimate a displacement from image patch to target point
Compare the estimated displacements with the ground truth displacements

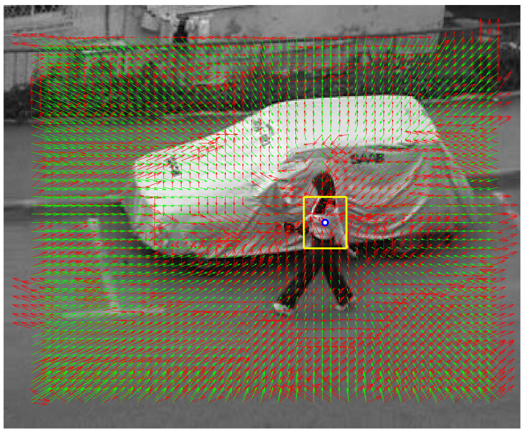
FIGURE 3.14: The flowchart of Experiment 3.2. The differences between the initial experiment and Experiment 3.2 are labelled with yellow background. Please note that the regressor trained in one frame is tested in the next frame. For example, the regressor trained in frame 14 is tested in frame 15.

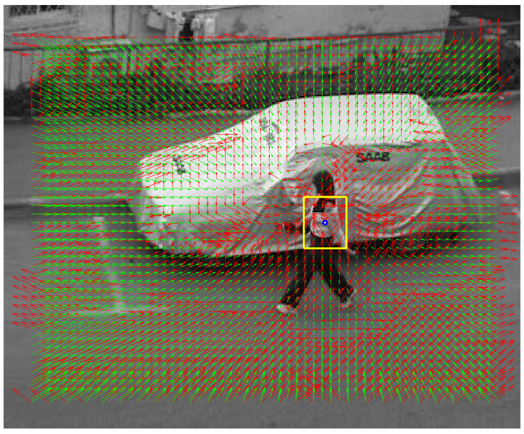
predictions, but become less reliable without proper updating. While the test locations in the background far away from the target give random predictions. Thus, to make the direct-displacement prediction work, it is critical to make sure that the test locations are close to the target and updated properly. This is an important motivation for the implicit shape model proposed in this thesis (explained in Chapter 4), which samples from the region around a target point and its two neighbouring points to make sure that most test samples are from the foreground.

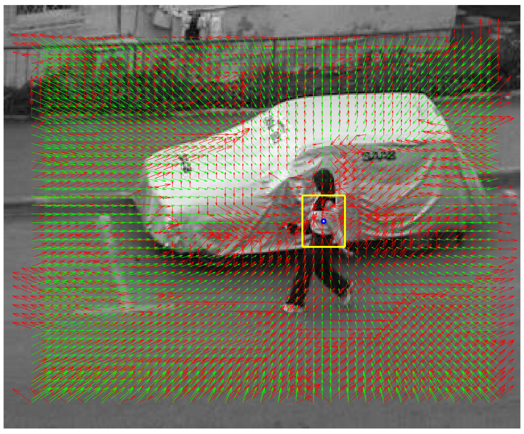

(a) Frame #2

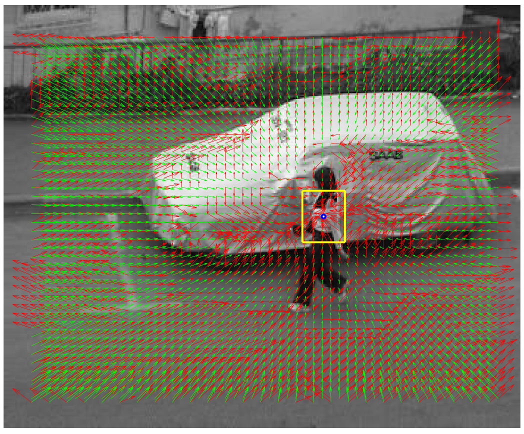

(b) Frame #3

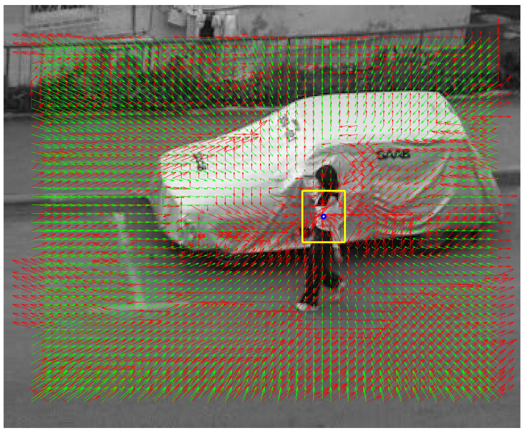

(c) Frame #4

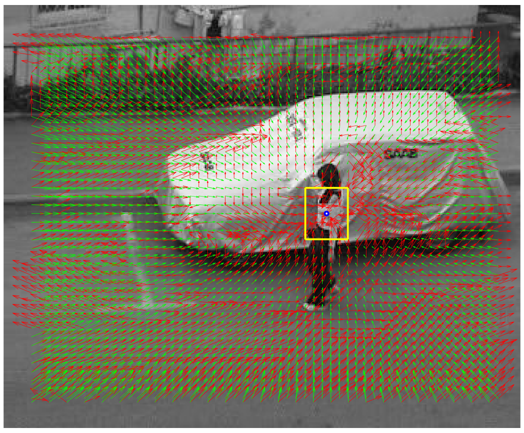

(d) Frame #5

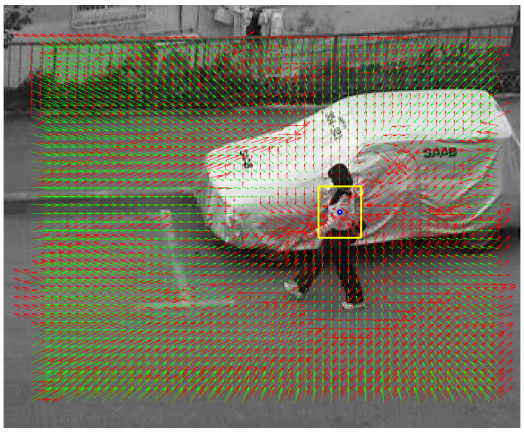

(e) Frame #6

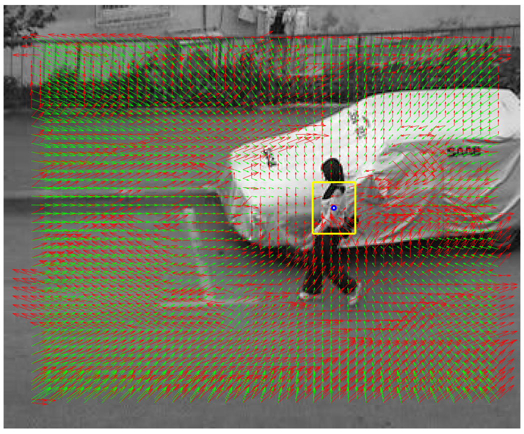

(f) Frame #7

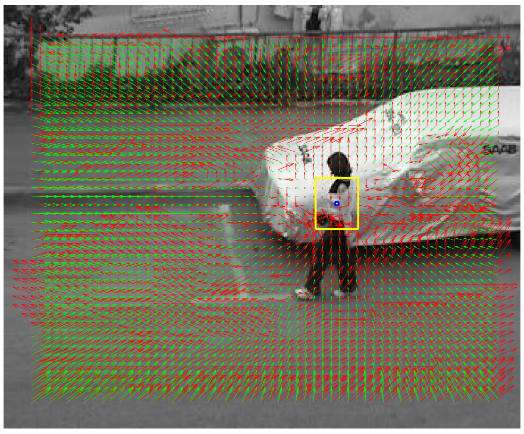

(g) Frame #8

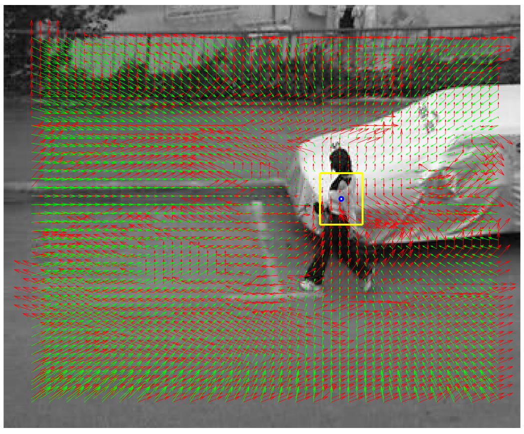

(h) Frame #9

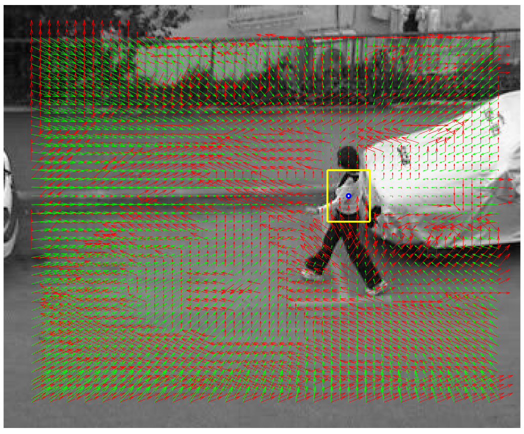

(i) Frame #10

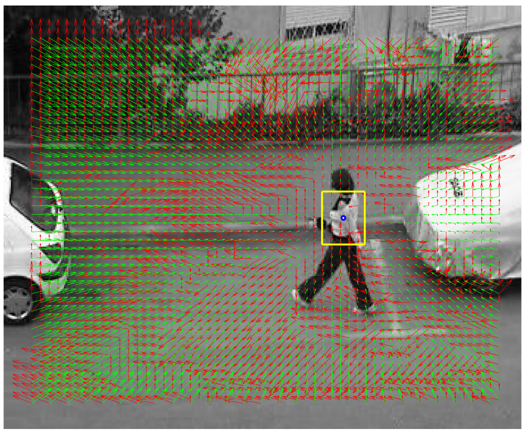

(j) Frame #15


(k) Frame #20


(l) Frame #25

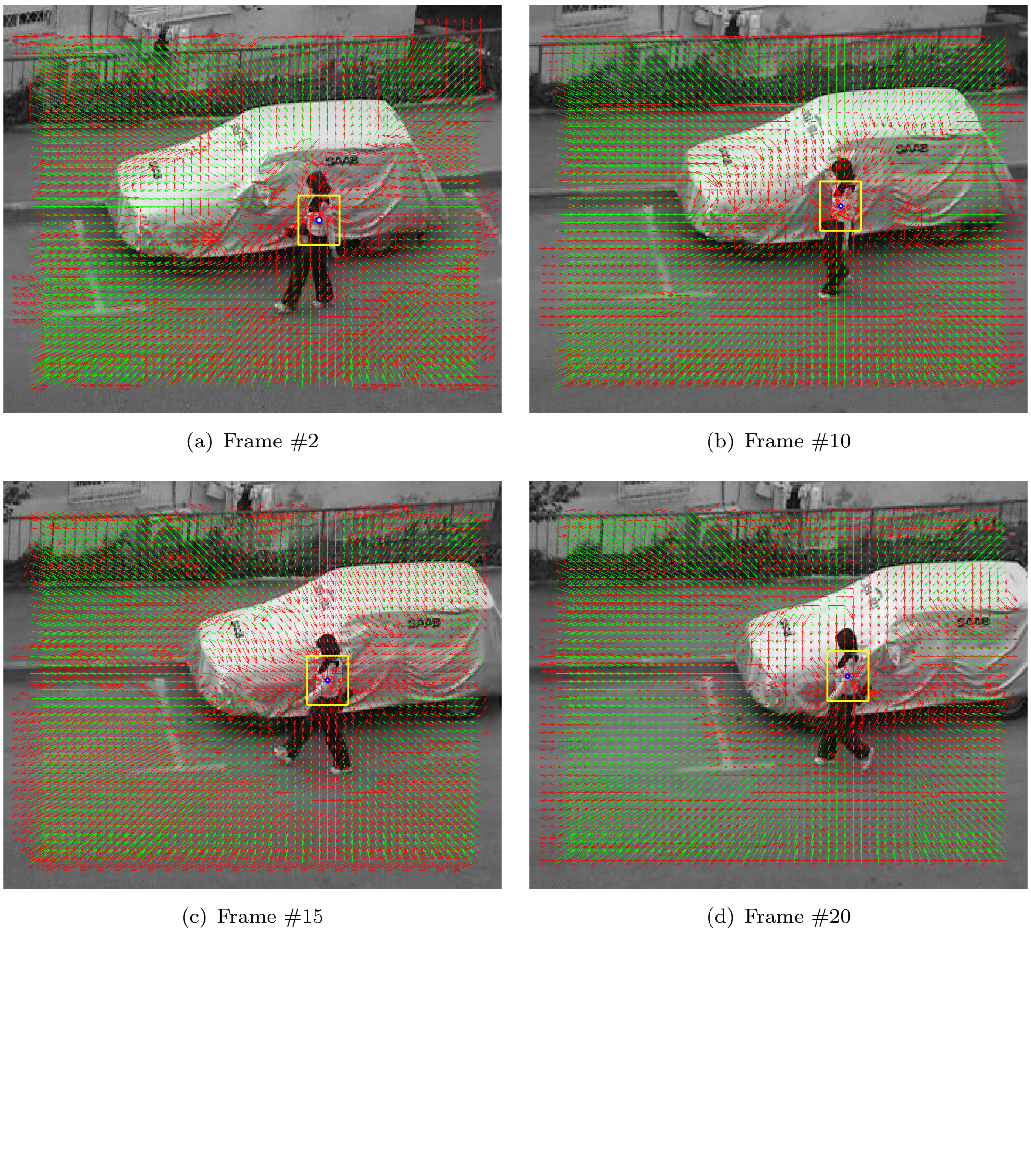

(m) Frame #30


(n) Frame #40



(o) Frame #50

FIGURE 3.15: The comparison between ground truth and estimated displacements from test locations to target location. The regressor is trained in the first frame and tested in various subsequent frames. For example, (a) shows the result in frame 2 and its regressor is trained in frame 1. The blue circle is the ground truth target location. Green arrows are ground truth displacement vectors pointing from test locations to target location. Red arrows are predictions given by the regressor trained in the first frame. Both kinds of arrows are scaled for clarity of display.

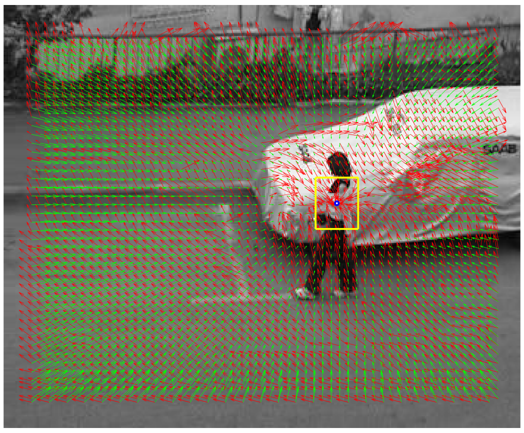


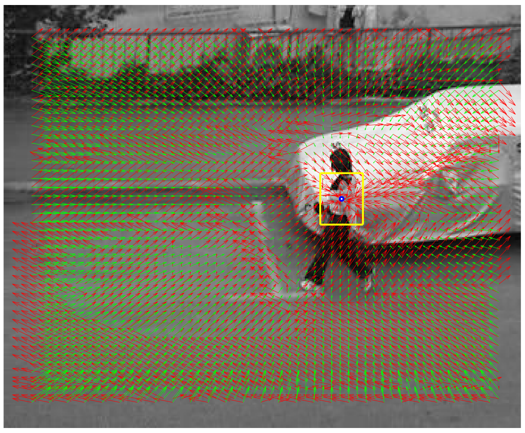
(a) Frame #2

(b) Frame #10

(c) Frame #15

(d) Frame #20


(e) Frame #25


(f) Frame #30

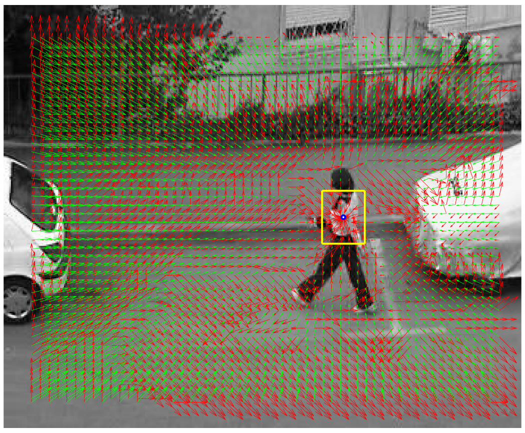

(g) Frame #50

FIGURE 3.16: The comparison between ground truth and estimated displacements from test locations to target location. The regressor is trained in one frame and tested in its next frame. For example, (a) shows the result in frame 2 and its regressor is trained in frame 1. The blue circle is the ground truth target location. Green arrows are ground truth displacement vectors pointing from test locations to target location. Red arrows are predictions given by the regressor trained in the first frame. Both kinds of arrows are scaled for clarity of display.

3.2.2 Quantitative Experiments

After qualitative analysis of the direct-displacement-based regression technique, to obtain full insight into the working of the regressor, quantitative research needs to be performed as well.

There are three goals to achieve in the quantitative research:

1. Explore the relation between the number of training samples and the accuracy of the direct displacement prediction.

2. Discover the relation between the sampling radius used in training, i.e. the radius from which training patches are sampled, and accuracy of the direct displacement prediction.

3. Investigate the relation between the sampling radius used in testing, i.e. the distance from test location to the target location, and accuracy of the direct displacement prediction.

The three goals can be achieved through two experiments. Experiment 3.3 and Experiment 3.4 are designed for the first two goals separately, and the third goal can be achieved by analysing certain experimental results selected from Experiment 3.3 and Experiment 3.4.

The designs of Experiment 3.3 and Experiment 3.4 are explained in Section 3.2.2.1. Section 3.2.2.2 gives a example of the result of one test (testing a regressor on one frame). The experimental results related to the three research goals are illustrated and analysed in Section 3.2.2.3, Section 3.2.2.4 and Section 3.2.2.5, respectively.

3.2.2.1 Design of the Quantitative Experiments

The video sequences used in the quantitative experiments are selected from the CVPR 2013 benchmark. The videos are labelled with properties they have. OCC is occlusion; DEF is non-rigid deformation; OPR is out-of-plane rotation; BC is background clutter; IV is illumination variation; SV is scale variation; MB is motion blur; FM is fast motion. Specifically, the video sequences used in the quantitative experiments and the properties they have are: *David*3 (OCC, DEF, OPR, BC), *Basketball* (IV, OCC, DEF, OPR, BC), *Jogging* (OCC, DEF, OPR), *Woman* (IV, SV, OCC, DEF, MB, FM, OPR) and *Crossing* (SV, DEF, FM, OPR, BC) from CVPR 2013 benchmark (Wu et al., 2013). These five videos are selected as the video set for three reasons. First, these five videos include nearly all the challenging aspect found in visual tracking. Second, the difficulties of these videos are classed as mid-level, based on the ranking in the CVPR 2013 benchmark (Wu et al., 2013). They are not too difficult and not too easy for tracking, which is suitable for the investigative experiments. Last but not least, the targets in these videos are articulated, and all of them have the most challenging properties for part-based tracking, either non-rigid deformation or occlusion. Results obtained from these videos are expected to give insight into the problems of part-based tracking.

The general idea of the experiment is, for each video sequence, to test the prediction accuracy of the regressor, which is trained in a frame (usually the first frame in a video sequence), with one variable (either sampling radius or sample number used in training) varying.

Based on the three research goals, two experiments are designed. The general framework is shown in Figure 3.17. Specifically, in Experiment 3.3, the sample number for training is changed to be 50, 100, 200, 300, 400 separately, while the sampling radius for training is fixed to 20 pixels. In Experiment 3.4, the sampling radius for training is changed to be 20, 30, 40, 50, 60 pixels respectively, while the sample number for training is fixed to 100. The third goal can be achieved by analysing results obtained with the specific sample number 100 and sampling radius 20 pixels.

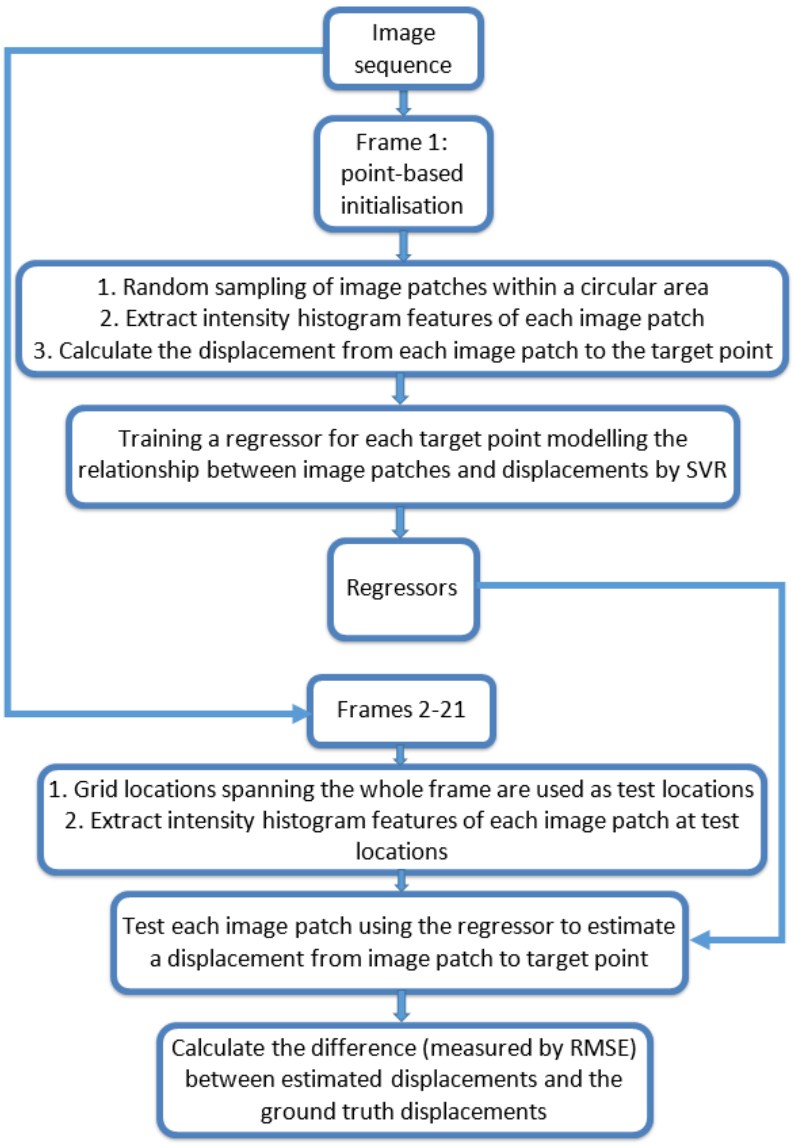


Image sequence
Frame 1: point-based initialisation
1. Random sampling of image patches within a circular area
2. Extract intensity histogram features of each image patch
3. Calculate the displacement from each image patch to the target point
Training a regressor for each target point modelling the relationship between image patches and displacements by SVR
Regressors
Frames 2-21
1. Grid locations spanning the whole frame are used as test locations
2. Extract intensity histogram features of each image patch at test locations
Test each image patch using the regressor to estimate a displacement from image patch to target point
Calculate the difference (measured by RMSE) between estimated displacements and the ground truth displacements

FIGURE 3.17: The general framework for Experiment 3.3 and Experiment 3.4. Specifically, in Experiment 3.3, the sampling radius for training is fixed while the number of samples is varying. In Experiment 3.4, the number of samples is fixed while the sampling radius for training is varying.

In both of these experiments, the first 110 frames of each video are used. The regressor is trained on every tenth frame, which are frames 1, 10, 20, 30, ..., 90. The regressor is then tested at grid locations (with the grid spacing of four pixels) spanning the whole frame (as shown in Figure 3.18) in the following 20 frames. For example, the regressor trained in frame 1 is tested in frame 2, 3, 4, 5, ..., 19, 20, 21; the model trained in frame 40 is tested in frame 41, 42, 43, 44, ..., 58, 59, 60. In this way, each video is utilised as 10 video segments, and each segment contains 21 frames in total. For each video segment, the training is performed in the frame 1 of each video segment, and the test is performed in the frames from 2-21.

The prediction accuracy is measured as the Euclidean distance between the estimated displacement vector and the ground truth displacement vector. More precisely, the root-mean-square error (RMSE) is used to evaluate the accuracy of predictions within a certain distance from the ground truth target location, as the grid test locations are used. The RMSE can be calculated by:

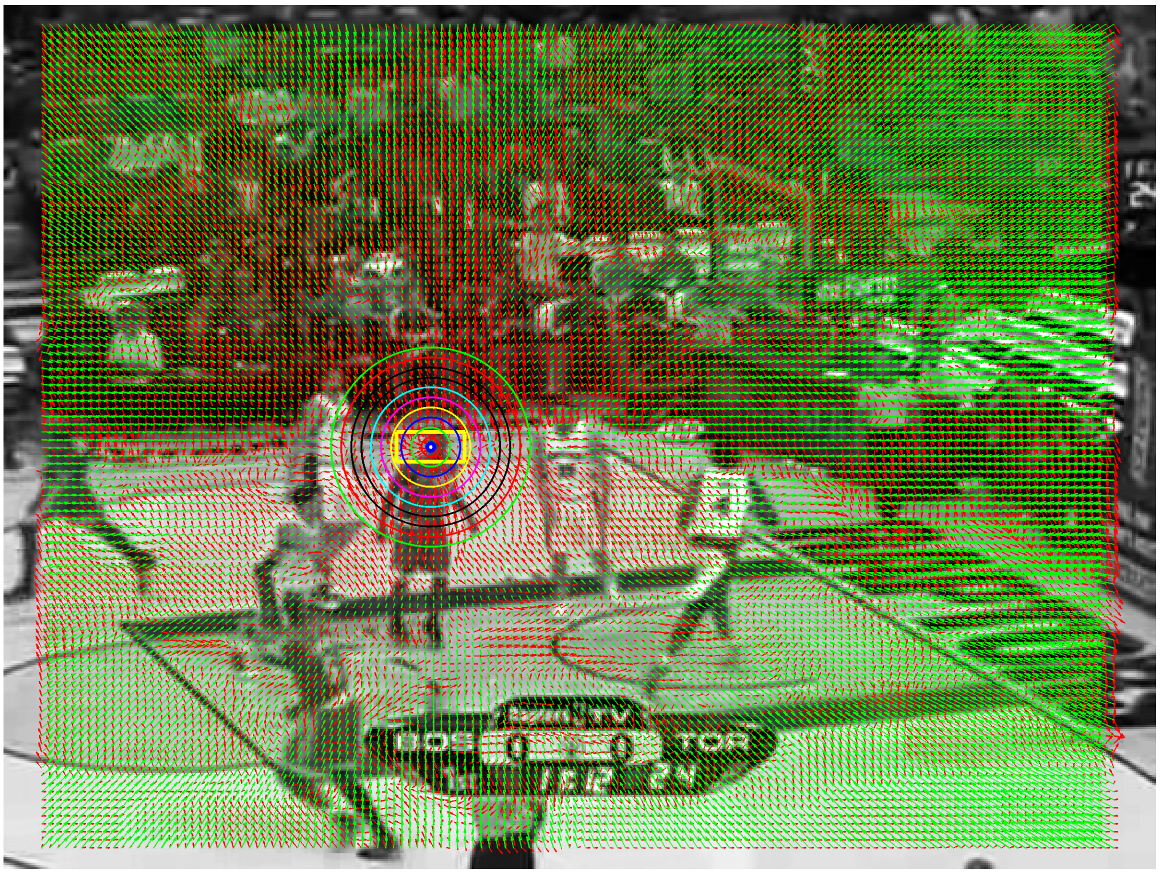
$$x_{rms} = \sqrt{\frac{1}{n}(x_1^2 + x_2^2 + ... + x_j^2 + ... + x_n^2)}, \tag{3.2}$$

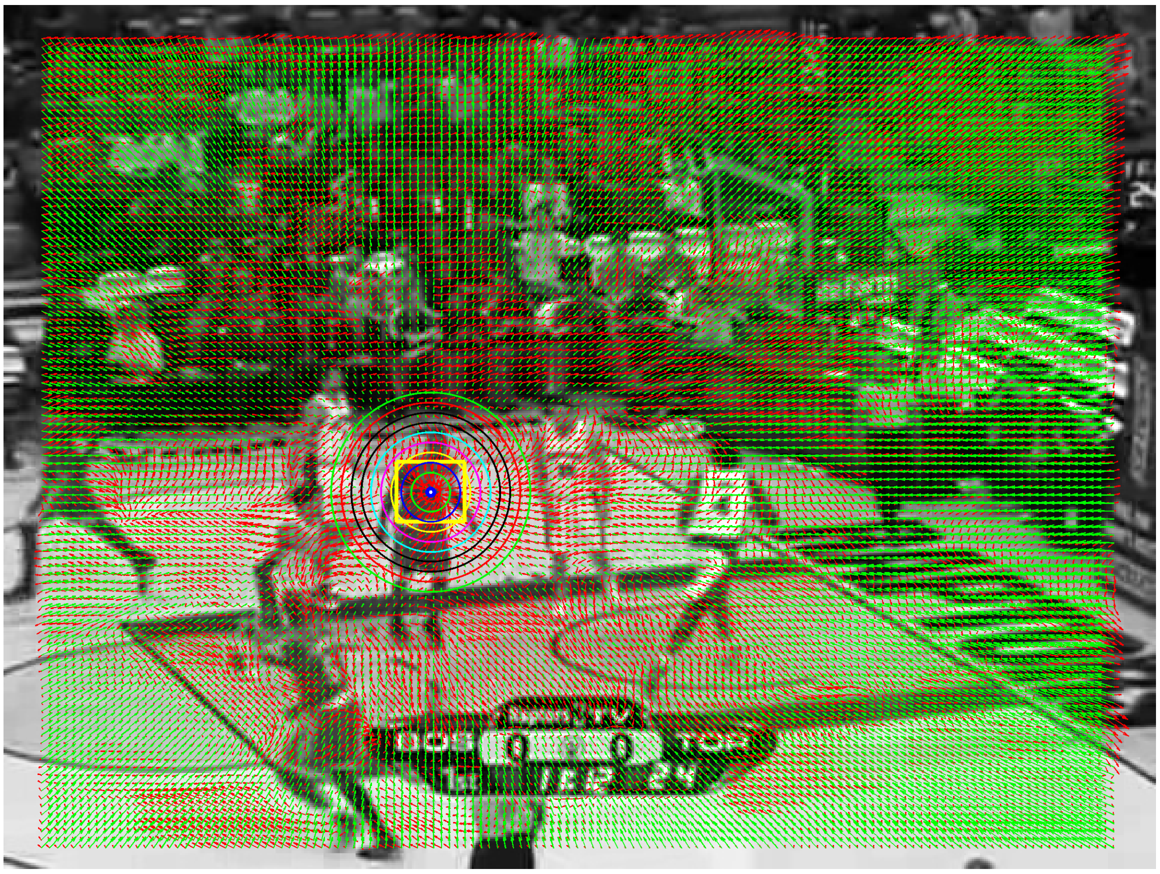
where x_{rms} is the RMSE measurement and x_j is the Euclidean distance between the predicted displacement and the ground truth displacement of j_{th} test location within a circular area. For each test (testing one model, a regressor, on one frame), the RMSE of the predictions within circles with radii 5, 10, 15, 20, 25, ..., 90, 95, 100 pixels (as shown in Figure 3.18) are calculated. Figure 3.18 gives an example of the map of estimated displacement vectors from grid locations to target location.

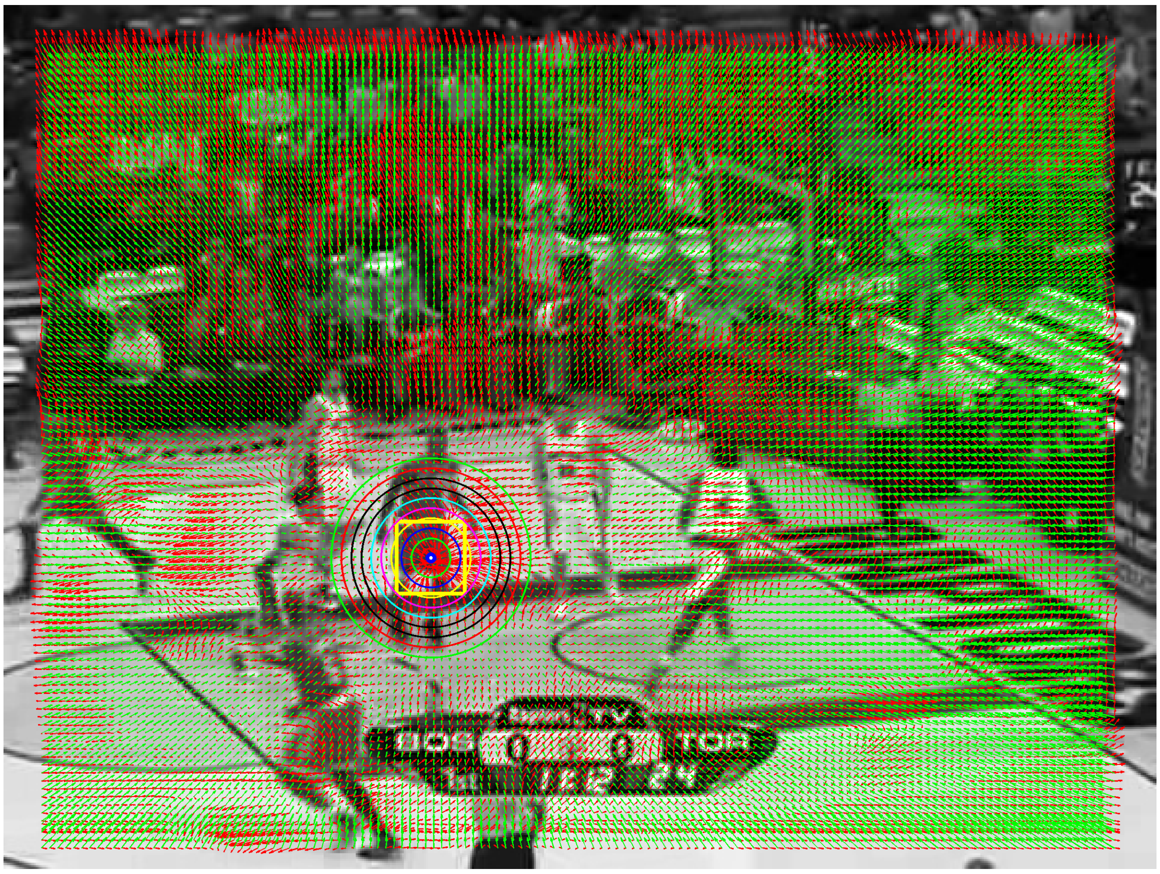
3.2.2.2 Sample Results

This section gives an example of the result of one test (testing one model on one frame). Further experimental results shown in Section 3.2.2.3, Section 3.2.2.4 and Section 3.2.2.5 are derived from the results of multiple tests.

Figure 3.19 shows the RMSE and its deviation of estimated displacement vectors within circles with different radii in the three displacement maps in Figure 3.18. The variation of the slope of the RMSE (thick black curve) shows that the increase in RMSE grows with the increase of the test radius. The variation of the deviation (the shaded area) shows that the deviation increases with the increase of the test radius. Both of these figures illustrate that patches sampled from farther test locations tend to be less reliable. Especially, compared with the other two parts, the deviation of part 1 within radius 20 pixels is bigger than those of the other two parts, which is because the image patch samples of the head part contain more background, so that the samples are less reliable.


(a) Part 1

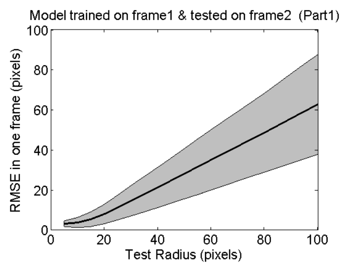

(b) Part 2



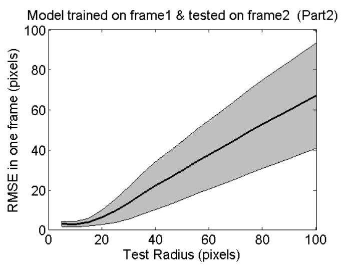
(c) Part 3

FIGURE 3.18: The estimated displacement maps of different parts of a target. (a) The displacement map for the head part; (b) The displacement map for the upper body part; (c) The displacement map for the lower body part. Green arrows are ground truth displacement vectors from test locations to target location while red arrows are estimated displacement vectors. The arrows are scaled for clarity of display. The RMSE of the predictions from sample located within circles with the radii 5, 10, 15, 20, 25, ..., 90, 95, 100 pixels are calculated. For clarity of display, only circles with radii 10, 20, 30, ..., 80, 90, 100 pixels are displayed.

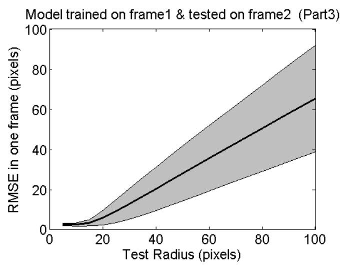
Note that only the results of the part 2 of the video *Basketball* are displayed in Section 3.2.2.3, Section 3.2.2.4 and Section 3.2.2.5 to save space. The results of other videos are very similar to the results obtained from *Basketball*. A specific video will not affect the conclusion derived from the experimental results. More importantly, results obtained from all videos were analysed, and the properties of the direct displacement prediction technique obtained through the analysis are as described in Section 3.2.2.3, Section 3.2.2.4 and Section 3.2.2.5.



(a) Part 1



(b) Part 2



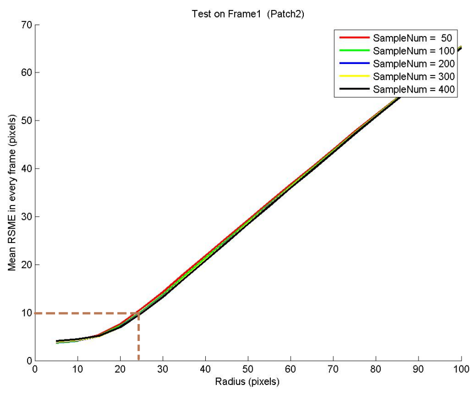
(c) Part 3

FIGURE 3.19: An example of the the RMSE and its deviation of predicted displacement vectors within circles with different radii. The middle black curve describes the RMSE values of different radii. The grey area above and below the black curve shows the deviation of the RMSE value. These are the corresponding results of the estimated displacement maps of three target parts in Figure 3.18. The regressor(model) is trained in the first frame and tested in the second frame.

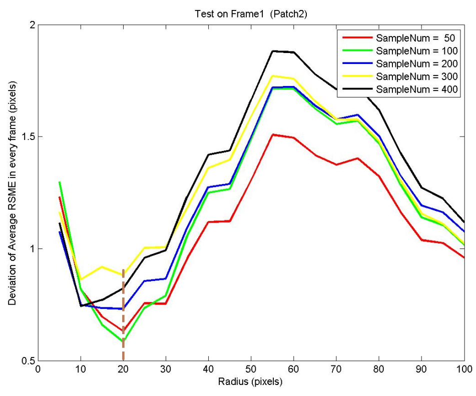
3.2.2.3 Results and Discussions of Experiment 3.3

To explore the relation between the number of samples used in training and the prediction accuracy of the regressor, Experiment 3.3 was performed.

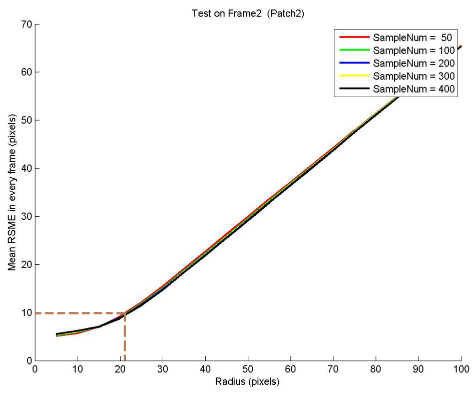
Figure 3.20 shows the averaged RMSE results obtained from part 2 of the video *Basketball* in Experiment 3.3. The averaged results are obtained by averaging the results of 10 video segments from video *Basketball*. The averaged results can provide a more reliable description of the relation than the result of a single test. Note that one test is defined as testing one model on one frame. The result of each test will be similar to the result of Figure 3.19, but with different values. Although 20 frames were tested after each training, the results of first 10 frames are displayed in Figure 3.20 to save space, which has already shown a clear relation between the number of training examples and the prediction accuracy of the regressor.



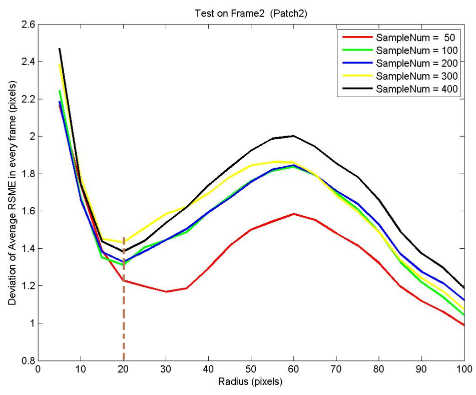
(a) Averaged results over all 1st frames after training



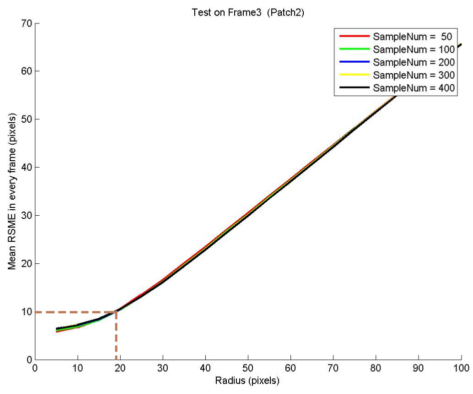
(b) Averaged results over all 1st frames after training



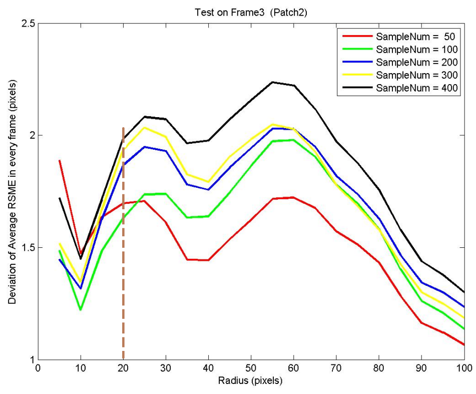
(c) Averaged results over all 2nd frames after training



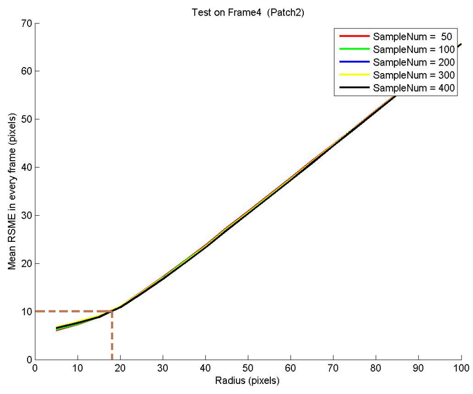
(d) Averaged results over all 2nd frames after training



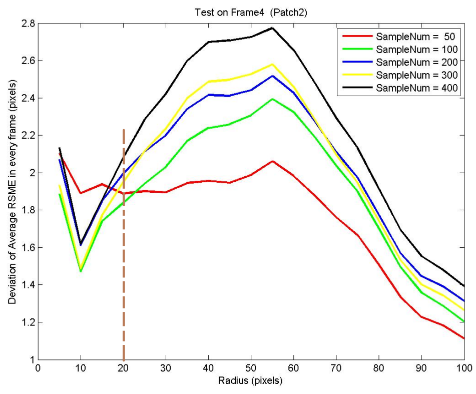
(e) Averaged results over all 3rd frames after training



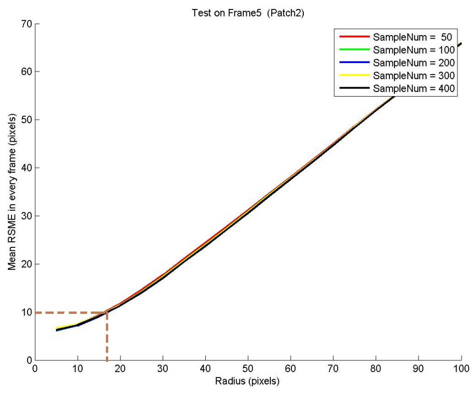
(f) Averaged results over all 3rd frames after training



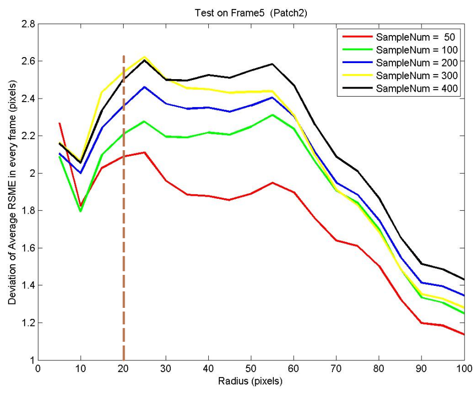
(g) Averaged results over all 4th frames after training



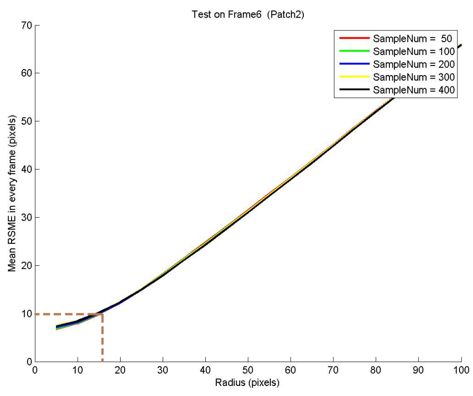
(h) Averaged results over all 4th frames after training



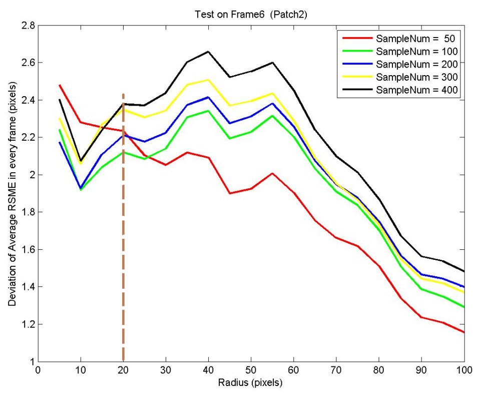
(i) Averaged results over all 5th frames after training



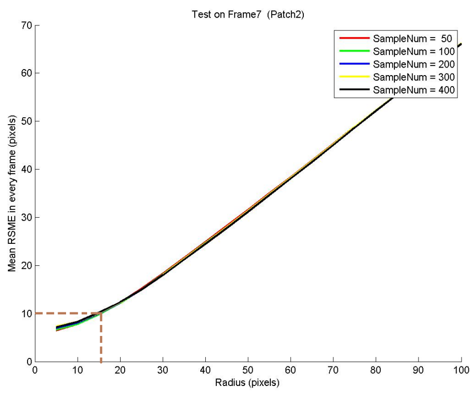
(j) Averaged results over all 5th frames after training



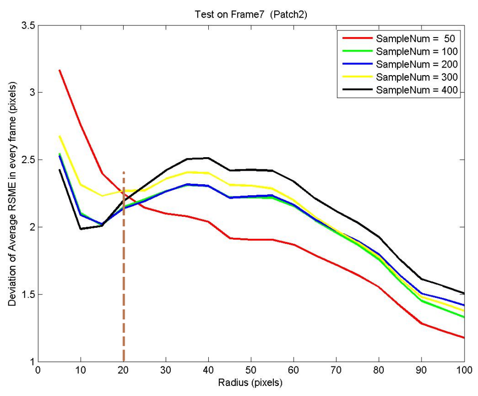
(k) Averaged results over all 6th frames after training



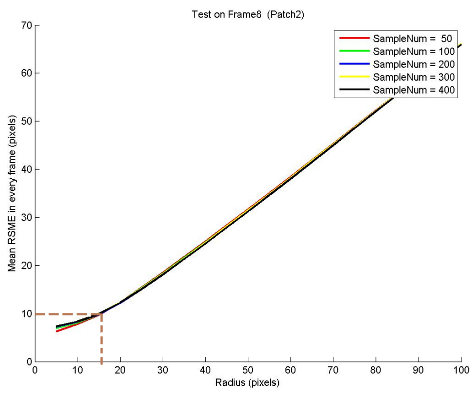
(l) Averaged results over all 6th frames after training



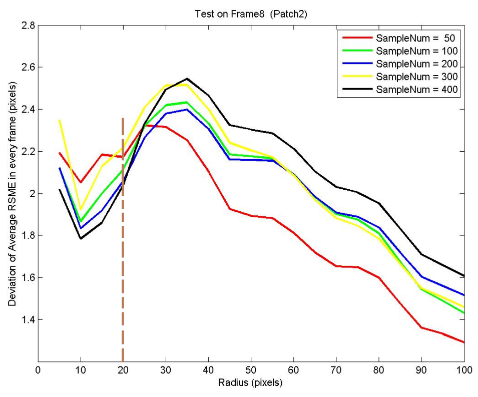
(m) Averaged results over all 7th frames after training



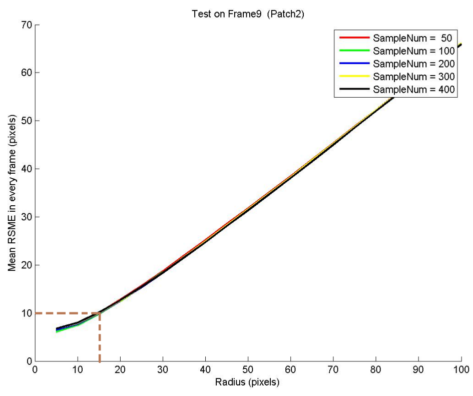
(n) Averaged results over all 7th frames after training



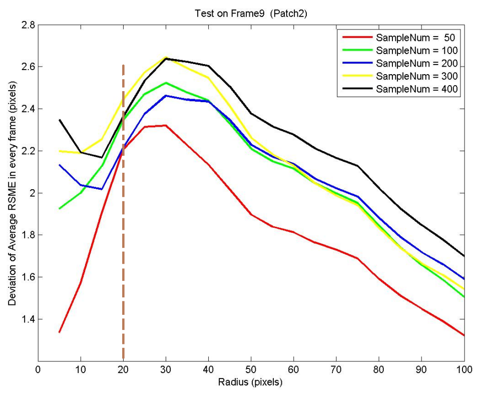
(o) Averaged results over all 8th frames after training



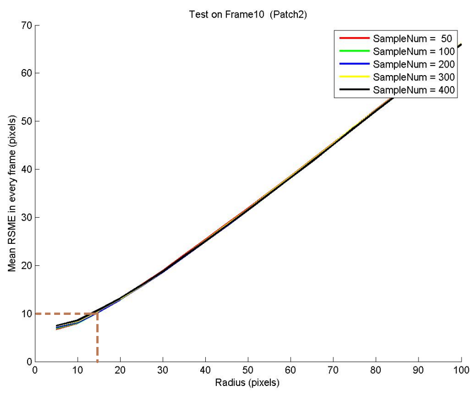
(p) Averaged results over all 8th frames after training



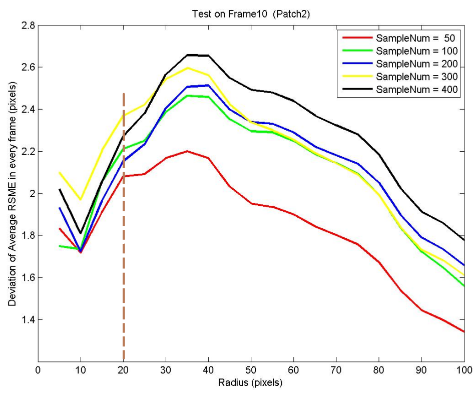
(q) Averaged results over all 9th frames after training



(r) Averaged results over all 9th frames after training



(s) Averaged results over all 10th frames after training



(t) Averaged results over all 10th frames after training

FIGURE 3.20: The mean and deviation of average RMSEs of different sample number. The x-axis is the test radius and the y-axis is the average RMSE. The RMSEs are averaged over 10 video segments. These results are for part 2 of video *Basketball*.

Figure 3.20(a), 3.20(c), 3.20(e), 3.20(g), 3.20(i),t 3.20(k), 3.20(m), 3.20(o), 3.20(q), and 3.20(s) show the mean RMSEs of different sample numbers in different frames after training. It can be seen that the mean RMSEs obtained using different sample numbers are very similar in each frame after training. This means that the sample number ranging from 50 to 400 does not have much affection on the prediction accuracy of the regressor when the training radius is 20 pixels. This situation holds for all frames (from the first to the tenth) used for testing. The mean RMSE has a relatively small increase rate with the test radius from 5 pixels to 20 pixels in the first frame after training. Although, in further frames, the radius range over which this low increase rate is observed (corresponding to a relatively flat curve in the figures of mean RMSE) gradually becomes smaller and smaller. In all frames during test, the mean RMSEs which are smaller than 10 pixels are at test locations sampled within a circular area with the radius around 20 pixels.

Figure 3.20(b), 3.20(d), 3.20(f), 3.20(h), 3.20(j), 3.20(l), 3.20(n), 3.20(p), 3.20(r), and 3.20(t) show the deviation of RMSEs of different sample numbers in different frames after training. The deviation of RMSEs obtained using larger sample numbers is generally bigger than that of RMSEs of smaller sample numbers. This holds for test locations at different distances from the target and also for different frames. In most situations, the deviation of the RMSE of the sample number 100 is consistently less than the deviations of RMSEs of other sample numbers. This is because large sample numbers (i.e. high sampling density) can cause over-fitting, so that the regressor will be less robust to unseen samples. In all test frames, with the test radius varying from 5 pixels to 20 pixels, the standard deviations of RMSEs generally decrease when the test radius increases. With the test radius varying from 20 pixels to 40 pixels, the standard deviations of RMSEs generally increase when the test radius increases. With the test radius varying from 40 pixels to 100 pixels, the standard deviations of RMSEs generally decrease when the test radius increases.

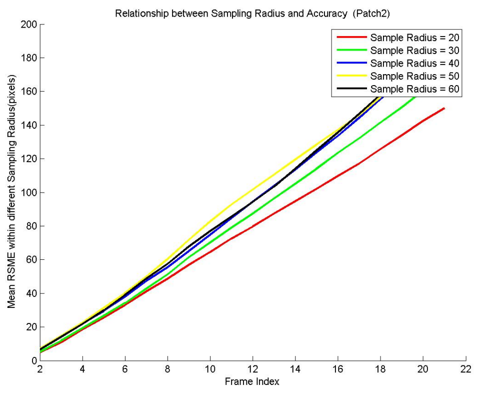
These results show that the regressor, trained with less than 100 samples and a training radius of 20 pixels, can effectively give predictions with RMSE lower than 10 pixels, within a test radius of 20 pixels. This holds for about 5 or 6 frames after the initial training.

3.2.2.4 Results and Discussions of Experiment 3.4

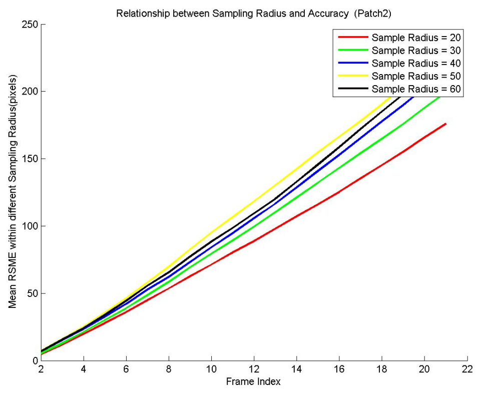
To explore the relation between the sampling radius used in training and the prediction accuracy of the regressor, Experiment 3.4 was performed. Specifically, the sampling radius for training is changed to be 20, 30, 40, 50, 60 pixels respectively, while the sample number for training is fixed to 100. The regressor is trained in frame 1 and tested in the following 20 frames in 10 video segments of each video sequence. 5 video sequences are used in total.

Figure 3.21 shows the averaged results of part 2 of the video *Basketball* in Experiment 3.4. Similarly to Experiment 3.3, the averaged results are obtained by averaging the results of 10 video segments. Each sub-figure of Figure 3.21 shows the cumulative mean RMSE results for different training radii and a specific test radius. The mean RMSE is averaged over 10 video segments. Specifically, for example, Figure 3.21(a) shows the cumulative results of mean RMSEs averaged over 10 video segments (each is from frame 2 to frame 21), with test radius equal to 5 pixels and training radius varying from 20 to 60 pixels. Figure 3.21(d) shows the cumulative results of mean RMSE averaged over 10 video segments (each is from frame 2 to frame 21), with test radius equal to 20 pixels and training radius varying from 20 to 60 pixels.

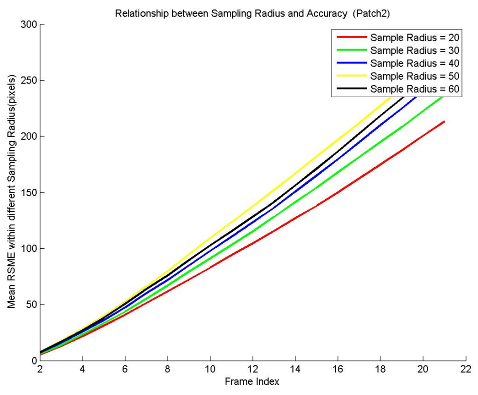
Figure 3.21 clearly shows that with the smaller test radius, the RMSEs of larger sampling radius for training have bigger errors than those of smaller sampling radius. The difference between RMSEs of different sampling radii for training becomes smaller when the test radius increases. It is obvious in sub-figures, Figure 3.21(a) - Figure 3.21(d), that the difference between RMSEs of different sampling radius for training becomes bigger over time. In these sub-figures, the slopes of the red curves is nearly the same at different frames. This shows that the samples at test locations with test radius smaller than 20 pixels are able to give correct predictions over time, given the training radius is 20 pixels. Usually 20 pixels covers a small area, which focuses on the target in most cases. It confirms that the initialisation of the target area is important for direct displacement prediction. It can be seen from Figure 3.21(e), 3.21(f) and 3.21(g), that the difference between mean RMSEs of different sampling radius gradually decreases. Figure 3.21(h), 3.21(h), and 3.21(j) show that the larger sampling radii have better performance than the smaller sampling radii when the test radius is more than 40 pixels. It's reasonable that samples at locations which are outside the training area are not able to give reliable predictions over time.



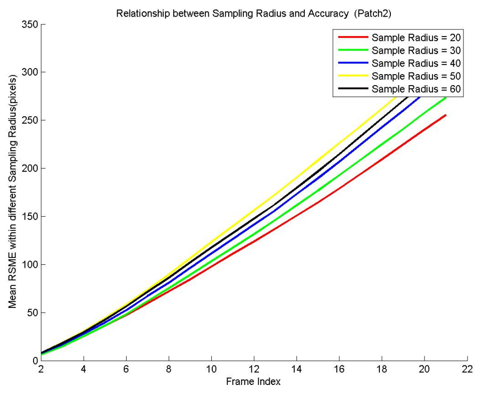
(a) Test Radius = 5 pixels



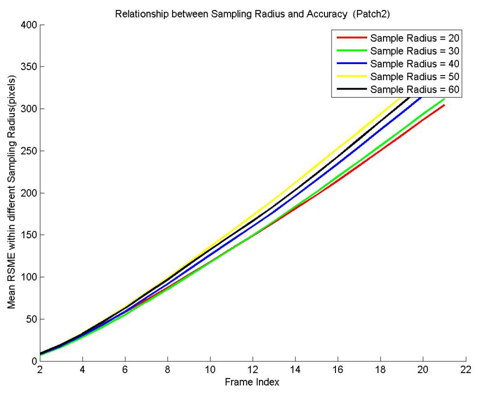
(b) Test Radius = 10 pixels



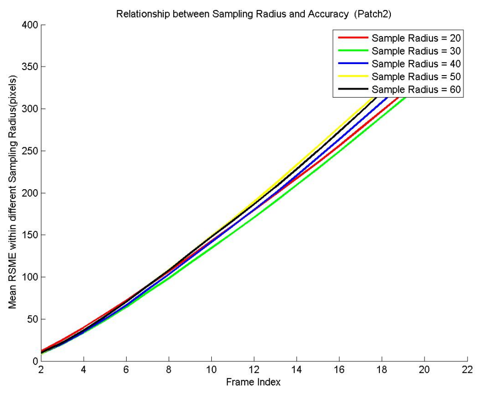
(c) Test Radius = 15 pixels



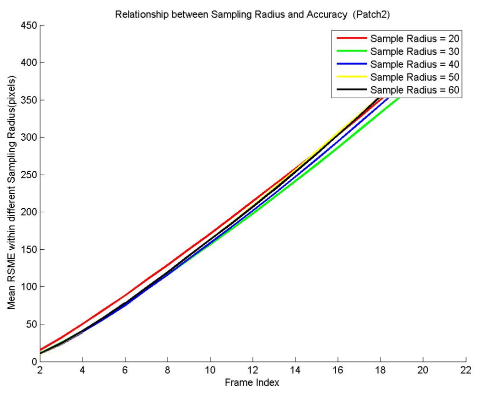
(d) Test Radius = 20 pixels



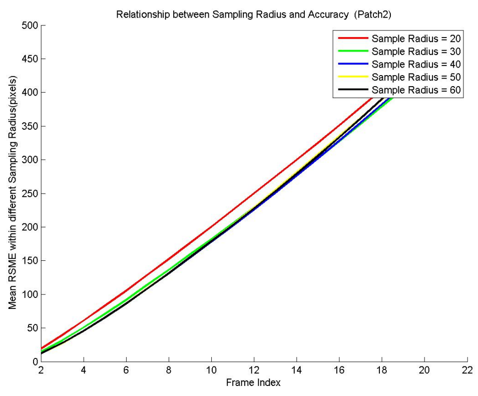
(e) Test Radius = 25 pixels



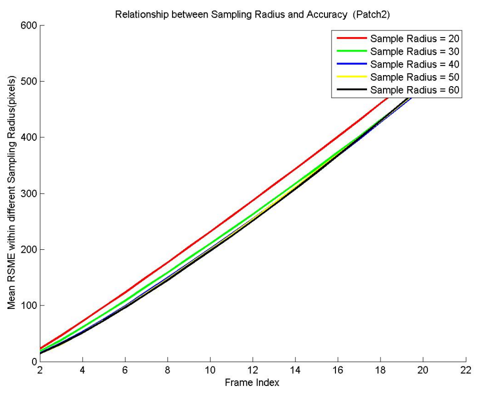
(f) Test Radius = 30 pixels



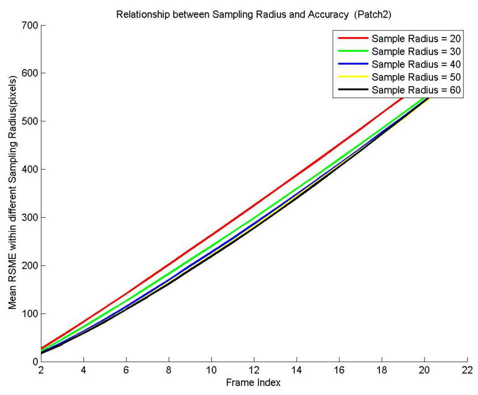
(g) Test Radius = 35 pixels



(h) Test Radius = 40 pixels



(i) Test Radius = 45 pixels



(j) Test Radius = 50 pixels

FIGURE 3.21: The mean of cumulative RMSE result of different sampling radius. The regressor is trained in frame 1 and tested in the following 20 frames (2-21) in 10 video segments. The x-axis is the frame index and the y-axis is the average RMSE. The RMSEs are averaged over 10 video segments. These results are for part 2 of video *Basketball*.

3.2.2.5 Results and Discussions Related to the Research Goal 3

To derive the relation between the sampling radius used in testing, i.e. the distance from test location to the target location, and prediction accuracy of the regressor, which is the Research Goal 3 (see the beginning of Section 3.2.2), some results are selected from experimental results of either Experiment 3.3 or Experiment 3.4. Specifically, the third goal of the quantitative research can be achieved by analysing experimental results with the specific sample number 100 and sampling radius 20 pixels, as Experiment 3.3 and Experiment 3.4 have shown that the regressor, trained with 100 samples and a training radius of 20 pixels, can effectively predict target location.

Figure 3.22 shows cumulative results of mean of RMSEs of three parts in five videos (*Basketball*, *Crossing*, *David*3, *Jogging* and *Woman*) with training radius equal to

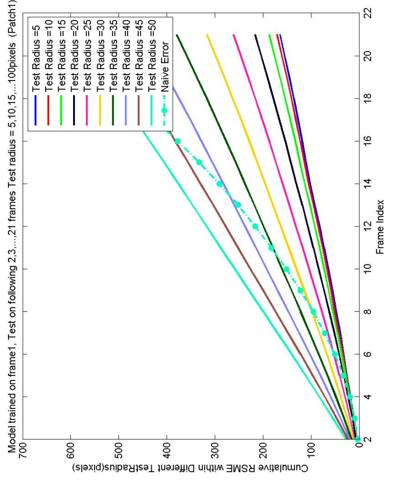
20 pixels and sample number equal to 100. Results are averaged over 10 video segments in each video sequence and each video segment contains 21 frames in total. The regressor is trained in frame 1 and tested in the following 20 frames in each video segment.

Each sub-figure shows the cumulative results for different test radii averaged over 10 video segments, with the training radius being 20 pixels and sample number being 100. For example, Figure 3.22(a) shows the cumulative result of RMSE of the first part of the target in the video *Basketball*, averaged over 10 video segments, with the test radius varying from 5 pixels to 50 pixels, training radius equal to 20 pixels and sample number equal to 100. Figure 3.22(d) shows the corresponding result of the first part of the target in video *Crossing*. The naive error is displayed in each sub-figure for reference. The naive error measures the difference between the naive result and the ground truth of the target location. The naive result of the target location is the target location without performing tracking, which means that the target location in the training frame is used as the target locations through all test frames. Results confirm that test locations far away tend to give less reliable predictions and a higher rate of increase of RMSE.

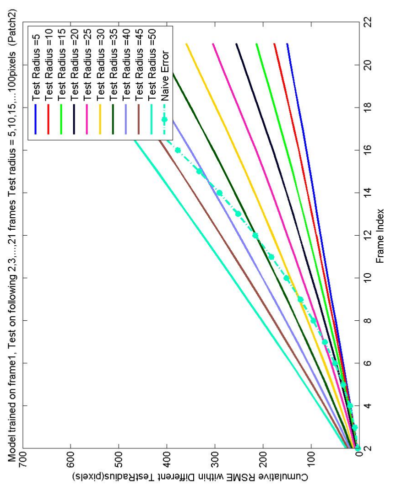
3.2.3 Conclusion

The conclusions related to the three research goals introduced at the beginning of 3.2.2 are as follows:

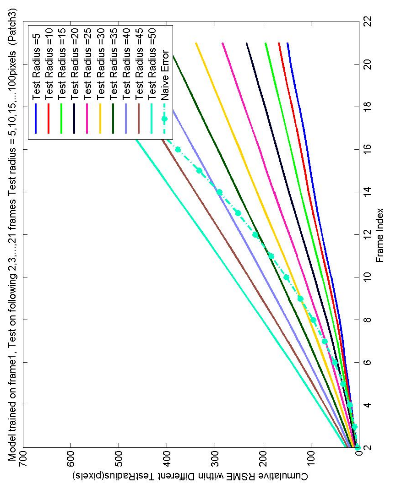
- Given a training radius equal to 20 pixels, a regressor trained with sample numbers less than 100 is able to give reliable predictions. Regressors trained on larger sample numbers tend to have an over-fitting problem.
- The samples at test locations with test radius smaller than 20 pixels are able to give correct predictions over time, given the training radius is 20 pixels. To handle larger test areas, the training radius has to be increased as well. However, when training with a larger sampling radius, the robustness of prediction increases while the accuracy of prediction decreases.
- Test locations far away tend to give less reliable predictions and have higher increase rates of prediction error.



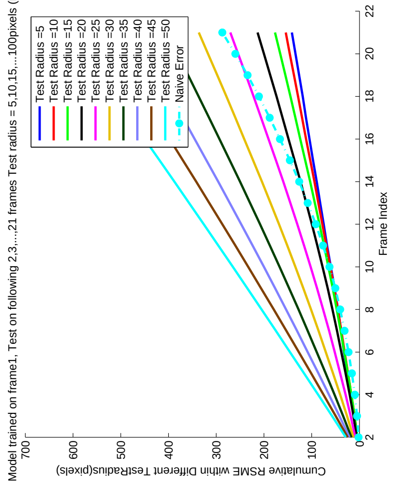
(a) Basketball part 1



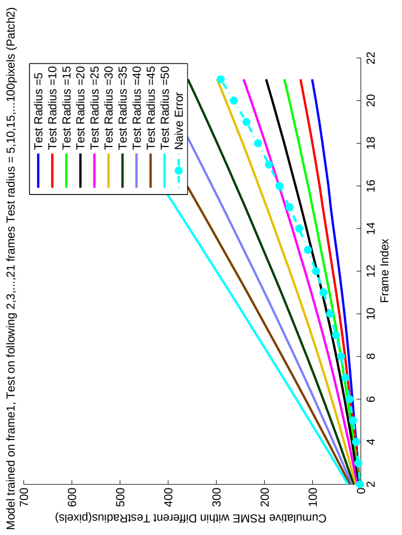
(b) Basketball part 2



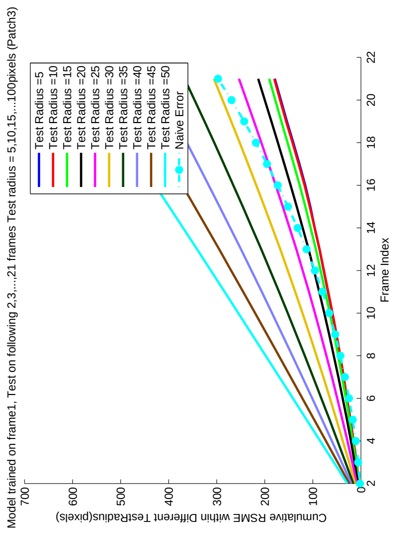
(c) Basketball part 3



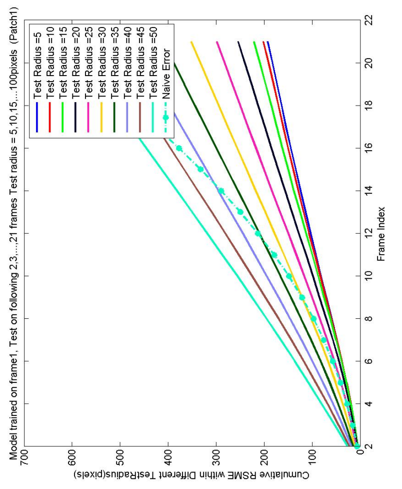
(d) Crossing part 1



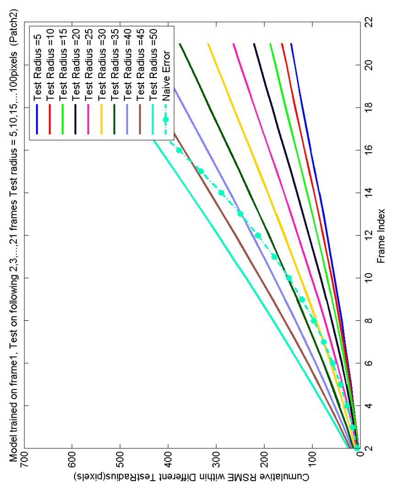
(e) Crossing part 2



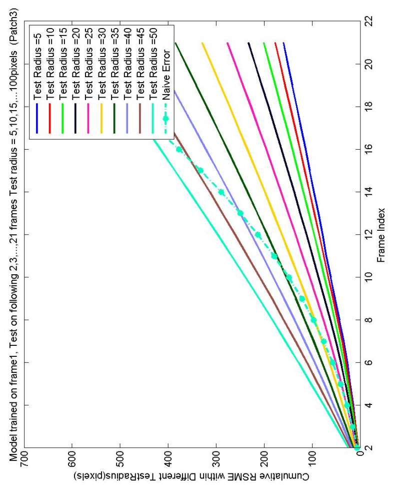
(f) Crossing part 3



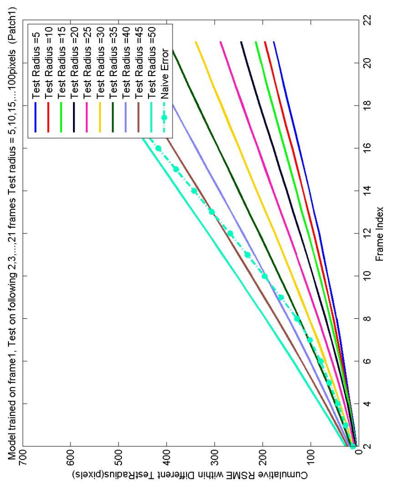
(g) David3 part 1



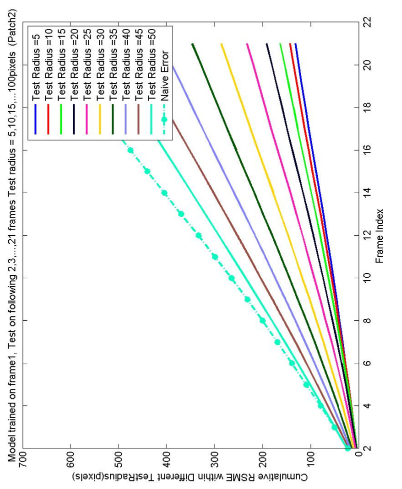
(h) David3 part 2



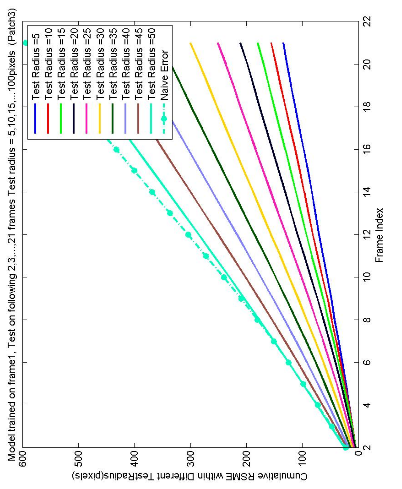
(i) David3 part 3



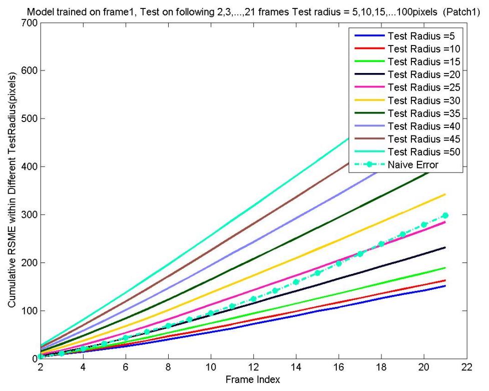
(j) Jogging part 1



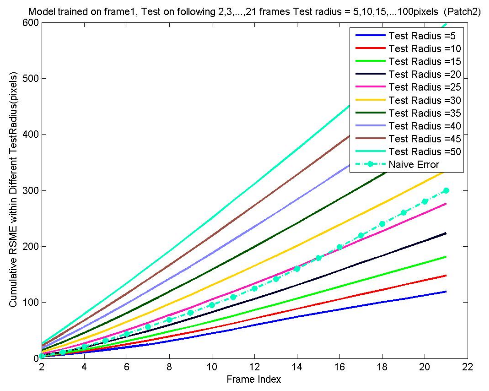
(k) Jogging part 2



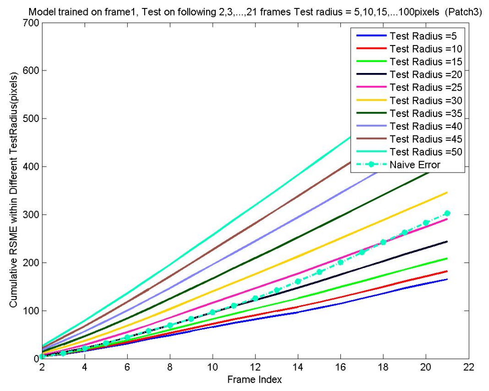
(l) Jogging part 3



(m) Woman part 1



(n) Woman part 2



(o) Woman part 3

FIGURE 3.22: The mean of cumulative RMSE result of different test radius averaged over 10 video segments. These results are of four videos (*Basketball*, *Crossing*, *David*3, *Jogging* and *Woman*), with training radius equal to 20 pixels and sample number equal to 100. In each video, the target has three parts.

3.3 A Simple Direct Displacement Prediction Tracker

In the previous two sections, preliminary experiments on the potential benefits and problems of part-based models as well as an exploration of the direct displacement prediction technique are introduced. After confirmation of the potential of the direct displacement prediction technique, this section further develops point-based initialisation and direct displacement prediction to form a basic tracker, which is named the simple direct displacement prediction (DDP) tracker.

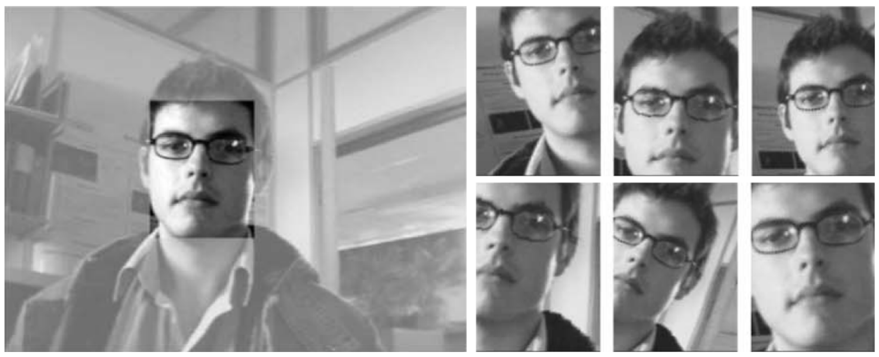
Adopting direct displacement prediction in tracking belongs to the area of regression-based tracking. A literature review on regression-based model-free tracking is introduced in Section 3.3.1. The simple DDP tracker is explained and evaluated in Section 3.3.2. Section 3.3.4 concludes by discussing the limitations of the simple DDP tracker.

3.3.1 Literature Review on Regression-based Model-free Tracking

Since Cootes et al. (Cootes, Edwards, & Taylor, 2001) proposed in the active appearance model (AAM) to learn the relationship between image intensity difference and model parameter displacements, which requires an offline learning stage to obtain a linear regressor numerically approximating the Jacobian, researchers have realised that discriminatively-trained regressors could be effectively used for object localisation.

Inspired by the work of Cootes et al. (Cootes et al., 2001), traditional template-based tracking, which minimises the difference between a reference template and a image region to locate the target, can be posed as a regression-based tracking problem, which is to find the set of parameters' values which can best describe the motion and deformation of the target through the sequence (Jurie & Dhome, 2002). The parameter variations can be written as a linear function of a difference image which describes the difference between the reference target image and the current image. Specifically, in (Jurie & Dhome, 2002) which is proposed to track rigid objects, the template is represented by a pyramid of sub-templates and each of them is tracked separately. Local motions provide a robust estimation of the target.

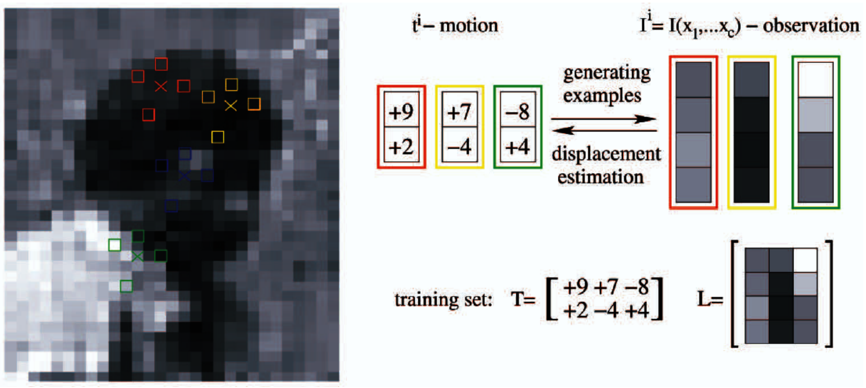
Several methods, such as (Williams et al., 2005), (Patras & Hancock, 2010) and (Ellis et al., 2011), proposed to exploit discriminatively-trained regressors to predict new object states, while a classifier is incorporated to validate the predictions. Williams et al. (Williams et al., 2005) propose a real-time tracker using a relevance vector machine (RVM) which extends the linear predictors to non-linear regression. Specifically, a training set is obtained by perturbation using Gaussian distribution and the probabilistic RVM is learned based on the training set to directly estimate displacement from the



(a) (b)

Figure 3.23: An example of a training set for a face tracking case in (Williams et al., 2005), which describes the process of generating examples from a single seed image. (a) A labelled seed image in which the clear part is the foreground and the blurred part is the background. (b) Some typical examples used to train the relevance vector machines including images after deformation and their corresponding displacements in translation, rotation and scaling (Williams et al., 2005).

target region. This paper demonstrated the real-time tracking of cars, faces and hands. As an example, Figure 3.23 shows the generation of training samples from a single image in (Williams et al., 2005).



t^i – motion
$I^i = I(x_1,...x_c)$ – observation
generating examples
displacement estimation
+9 +2
+7 −4
−8 +4
training set: $T = \begin{bmatrix} +9 & +7 & -8 \\ +2 & -4 & +4 \end{bmatrix}$
L=

Figure 3.24: The image is perturbed by the motion parameters, creating the set of synthesised examples of observed intensities I^i of perturbed support set and motions t^i (Zimmermann et al., 2009).

Zimmermann et al. (Zimmermann et al., 2009) propose to learn a sequence of linear regressors (referred to as predictors), each of increased precision but lower robustness. It has been shown that this step is essential for producing accurate results, as a single regressor can be trained to be either robust or precise, but not both simultaneously. The object is modelled by a collection of local motion predictors, which makes the estimation of object translation robust. Because of the simplicity of the linear predictor and the number of the predictors used, the algorithm is able to make very efficient motion estimations. The mapping between intensities of the perturbations of the support set and the corresponding motions is illustrated in Figure 3.24. How the estimation is generated

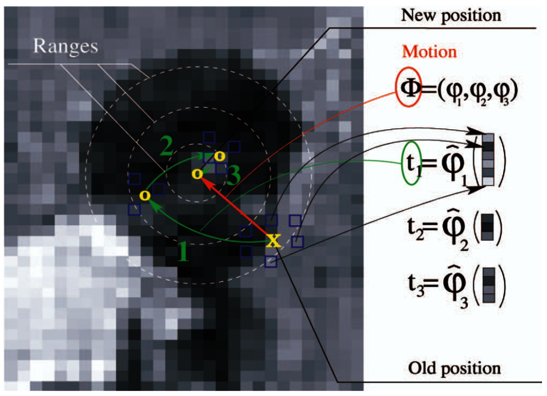


FIGURE 3.25: The working mechanism of sequential linear predictor in (Zimmermann et al., 2009). The sequential predictor $\Phi = (\varphi_1, \varphi_2, ..., \varphi_m)$ estimates a vector of motion parameters $\mathbf{t}$ (denoted by red arrow) in m steps by m different predictors $\varphi_1 ... \varphi_m$. Particular predictors and the number of steps are the subject of learning (Zimmermann et al., 2009).

through the sequential linear predictor is presented in Figure 3.25 The selection of an optimal sequence of predictors from a set of learned predictors is performed by searching for the cheapest path in a graph, which is determined by the complexity and ranges of the predictors.

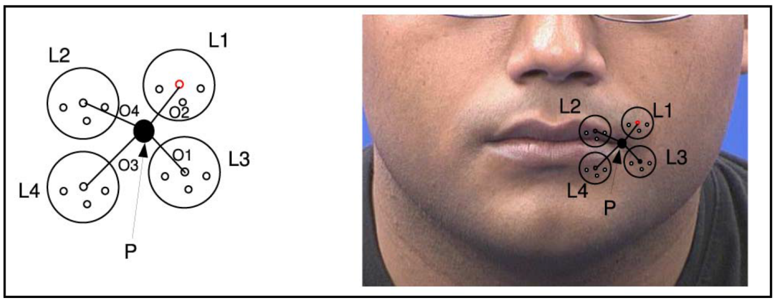


FIGURE 3.26: The configuration of a rigid flock of linear predictors in (Ong & Bowden, 2011). The position of the flock is defined by reference point P and the flock consists of separate linear predictors, denoted as $(L1, L2, L3, L4)$. Each of the linear predictor has a rigid offset from point P, which are $O1, O2, O3, O4$ separately.

Ong and Bowden (Ong & Bowden, 2011) use flocks of linear predictors (LP) for facial feature tracking, in which the related visual context for tracking any facial point needs an offline learning stage. Specifically, each linear predictor maps the template difference to the displacement vector of a tracked facial feature. The multiple LPs are then grouped into rigid flocks (as shown in Figure 3.26) to track a single facial point, which improves the accuracy and robustness of general facial feature tracking. Ong and Bowden (Ong & Bowden, 2011) also introduce a bias factor into the linear predictor to make it a full linear regressor. It is known from (Zimmermann et al., 2009) that increasing the range of training displacements brings more robustness but reduces the accuracy of estimation.

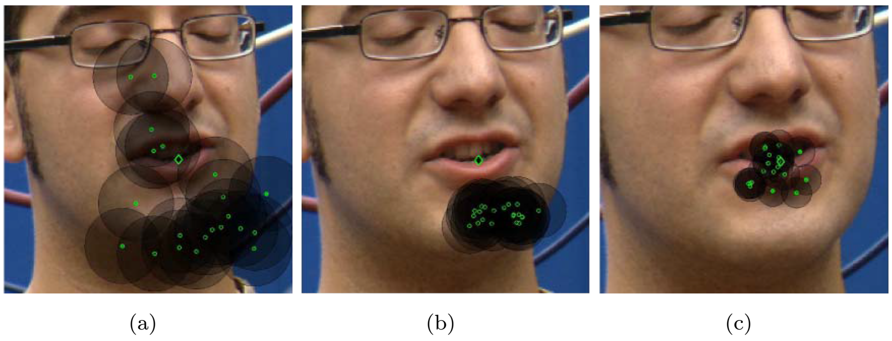


Figure 3.27: The linear predictor flocks of decreasing sizes in (Ong & Bowden, 2011). The feature displacement is predicted with a cascade of linear predictor flock from the largest (a) to smallest (c).

To solve this problem, Ong and Bowden (Ong & Bowden, 2011) employ a chain of LPs of decreasing sizes, as shown in Figure 3.27.

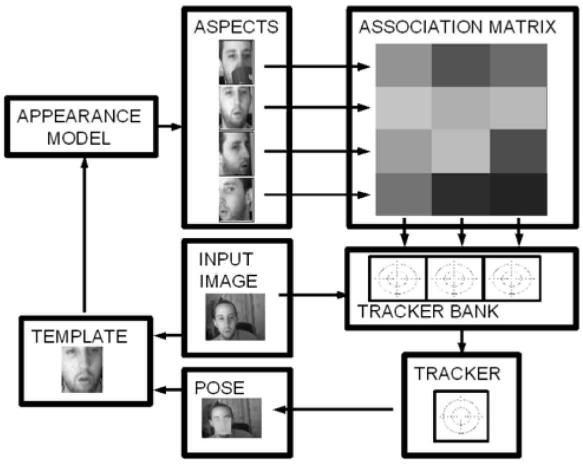
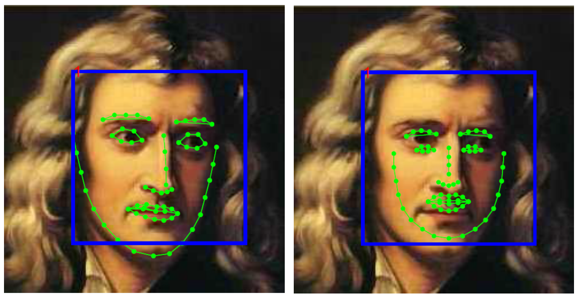


Figure 3.28: The generic algorithm framework of (Ellis et al., 2011). An appearance model stores all aspects of a target. A tracker bank includes all linear predictors, which are associated with the aspects of the target by an association matrix.

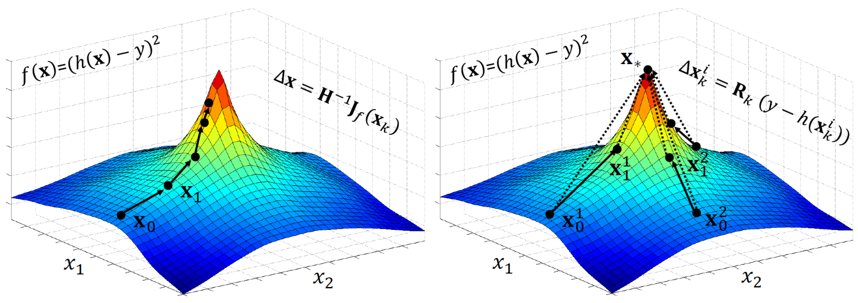
To address the problem that regression-based tracking tends to require an offline training stage, Ellis et al. (Ellis et al., 2011) propose an online linear predictor tracker. Identifying the target location in the first frame is the only supervision required in this paper. Specifically, the templates are stored and clustered incrementally to identify the modes or aspects of the target appearance, which means that the appearance model with multiple model templates is adaptively learned on-the-fly. Sets of spatially localised linear displacement predictors are employed to associate with various modes of the appearance model. Figure 3.28 illustrates the generic algorithm framework of (Ellis et al., 2011).

Furthermore, the successful cascade of linear regressors, popularised by Xiong and Torre (Xiong & De la Torre, 2013) for face alignment, can be traced back to the model-free tracking work of (Zimmermann et al., 2009). Instead of employing the linear regressor



(a) (b)

FIGURE 3.29: An image example used for training and an example of initialisation in (Xiong & De la Torre, 2013). (a) An image manually labelled with 66 landmarks, x_*. Blue outline indicates the face detector. (b) Mean landmarks, x_0, initialised using the face detector.


$f(\mathbf{x}) = (h(\mathbf{x}) - y)^2$
$\Delta\mathbf{x} = \mathbf{H}^{-1}\mathbf{J}_f(\mathbf{x}_k)$
$\mathbf{x}_1$
$\mathbf{x}_0$
x_1
x_2
$f(\mathbf{x}) = (h(\mathbf{x}) - y)^2$
$\mathbf{x}_*$
$\Delta\mathbf{x}_k^i = \mathbf{R}_k(y - h(\mathbf{x}_k^i))$
$\mathbf{x}_1^1$
$\mathbf{x}_1^2$
$\mathbf{x}_0^1$
$\mathbf{x}_0^2$
x_1
x_2

(a) (b)

FIGURE 3.30: The illustration of the comparison between Newton's method and the supervised descent method (SDM) in (Xiong & De la Torre, 2013), where the goal is to minimize a nonlinear least squares (NLS) function, $f(x) = (h(x) - y)^2$, where $h(x)$ is a nonlinear function , x is the vector of parameters to optimize, and y is a known scalar. The z-axis is reversed for visualisation purposes. (a) Newton's method to minimise $f(x)$. The traditional Newton update has to compute the Hessian and the Jacobian at each step. (b) SDM learns a sequence of generic descent maps R_k from the optimal optimisation trajectories (indicated by the dotted lines). Each parameter update Δx^i is the product of R_k and a sample-specific component $(y - h(x_k^i))$.

introduced above, Xiong and Torre (Xiong & De la Torre, 2013) propose a supervised descent method (SDM) to solve a non-linear optimisation problem for face alignment, which learns a sequence of descent directions mapping the motion parameters and appearance differences without calculating the Jacobian or the Hessian. In detail, during offline training, the SDM learns a sequence of descent directions sampled at different points. During testing, the predictions given by each level are used directly as the initial test locations for the next level, which is the cascaded regression. The prediction of last cascade level is determined as the final result of a frame. Figure 3.29 gives an example of the image and labels used for offline training of SDM in (Xiong & De la Torre, 2013) and the initialisation of the task of face alignment using cascaded regression. The difference between Newton's method and the SDM is illustrated in Figure 3.30.

Previous regression-based tracking methods tend to use models (regressors) of high complexity to address planar or rigid object tracking (Jurie & Dhome, 2002; Williams et al., 2005). For example, Williams et al. (Williams et al., 2005) adopted RVM to provide displacement predictions when tracking faces, hands and cars. Most previous regression-based tracking methods are based on the linear predictor (Jurie & Dhome, 2002; Zimmermann et al., 2009; Ong & Bowden, 2011; Ellis et al., 2011), which models a linear mapping from the intensity difference between the sparse base support pixels and those from the current target to a displacement vector space (the motion of the feature point). LP trackers require the selection of the reference point and the support piexels or support regions. They usually need an offline learning stage or a hard coded model to find out the optimal support sets. Ellis et al. (Ellis et al., 2011) avoid this problem by evaluating the performance of a predictor online and allowing poor performers to be replaced. However, the method has to maintain a tracker bank of linear predictors associated with various modes of target appearance. These requirements make tracking complicated and limit applications. While cascaded regression (Xiong & De la Torre, 2013) has been used for what is essentially structured object detection, it has never been applied to online general object tracking.

The direct displacement prediction strategy directly models the relationship between image features and displacements. There is no template required in the framework. Aggregation of multiple predictions of the target location from different test samples cancels out random outliers and aggregates correct predictions, which removes the need for selection of support regions. In addition, the DDP method has the potential of integration into the SDM framework, which can be expected to further improve accuracy of DDP.

3.3.2 Simple Direct Displacement Prediction Tracker

After research on the power of the direct displacement prediction technique of LEAR (Martinez et al., 2013) in tracking, we now consider a simple direct displacement prediction tracker (see Figure 3.31), in which point-based initialisation and support vector regression (SVR) techniques are the main components. As in the preliminary experiments of the DDP technique described in Section 3.2, the aggregation method in LEAR is used to aggregate all predictions given by a regressor to achieve a single robust prediction of target. The SVR is used as the regression method here. As stated in LEAR (Martinez et al., 2013), other regression algorithms are also applicable. As the direct displacement prediction directly predicts the locations of a number of parts of the object, the tracker is initialised by defining part locations instead of a set of bounding boxes. The tracked target object is initialised with the ground truth locations of three points

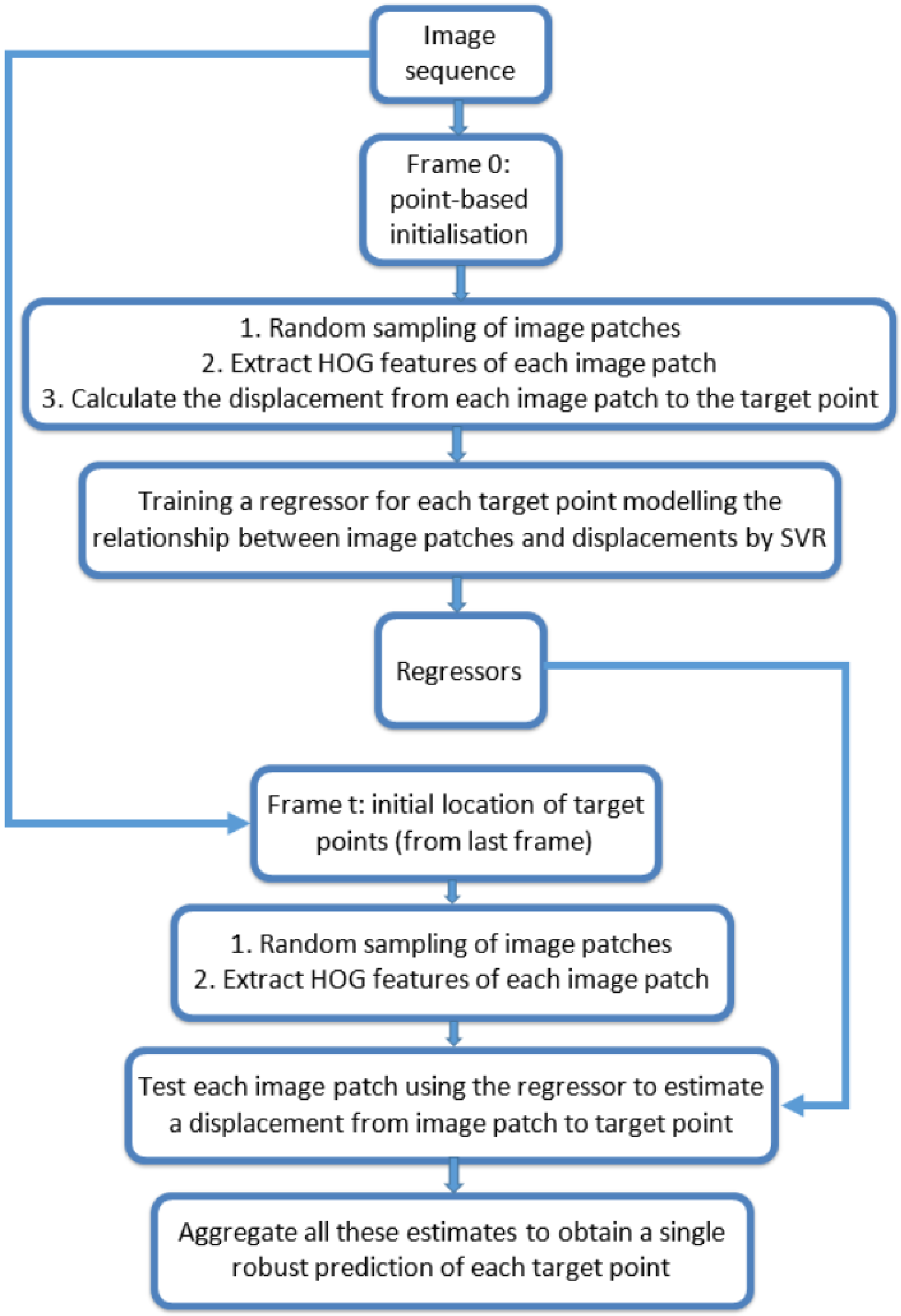


Image sequence
Frame 0: point-based initialisation
1. Random sampling of image patches
2. Extract HOG features of each image patch
3. Calculate the displacement from each image patch to the target point
Training a regressor for each target point modelling the relationship between image patches and displacements by SVR
Regressors
Frame t: initial location of target points (from last frame)
1. Random sampling of image patches
2. Extract HOG features of each image patch
Test each image patch using the regressor to estimate a displacement from image patch to target point
Aggregate all these estimates to obtain a single robust prediction of each target point

FIGURE 3.31: The framework of the simple Direct Displacement Prediction tracker.

centred at three target parts in the first frame, shown as red dots in Figure 3.32. The initialisation is the same as that used in preliminary experiments (Figure 3.8) exploring the potential ability of the direct displacement prediction. Each target point is tracked separately. To test the performance of the DDP, in real tracking, the tracker is trained in the first frame and tested through the whole video. This means that three regressors are trained for three target points separately in the first frame of a video sequence.

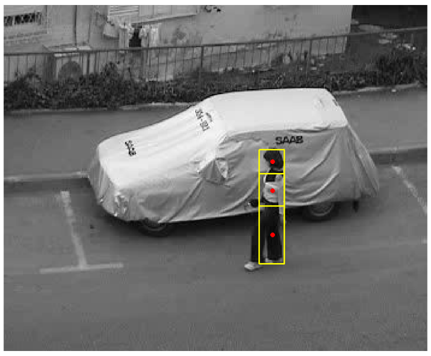


FIGURE 3.32: The part-based representation of a target. Yellow rectangles are different parts and red dots are their centres.

Specifically, the location of a target point is denoted as L^* and $*$ means ground truth. $L^* = (x^*, y^*)$ is the ground truth location of a part. Given the image I, the ground truth part location L^*, and the sample location $S = \{s_j, j \in [1, N_s]\}$ (which is obtained by randomly sampling around L^* in training and around initial target location L^{init} in testing), the direct displacement prediction problem is then to estimate the displacement vector $D^* = L^* - S$. Regressors r_x and r_y are responsible for the estimation of Δx and Δy respectively, and $\hat{D} = (\Delta x, \Delta y)$. Each sample is a local image descriptor $\Phi(I, s_j, p_s)$ extracted from a square image patch centred at a sample location s_j with size $p_s \times p_s$, $j \in [1, N_s]$ and N_s is the number of the samples. Both regressors, r_x and r_y use the image descriptor $\Phi(I, S, p_s)$ as their input. The output of the regressor is the estimate of D^* defined as:

$$\hat{D} = (r_x(\Phi(I, S, p_s)), r_y(\Phi(I, S, p_s))). \tag{3.3}$$

The estimated target location obtained by evaluation of the regressor is defined by $\hat{l} = S + \hat{D}$. The estimated target location obtained by the full simple DDP tracker is denoted by $\hat{L}$, obtained through the aggregation method in LEAR (Martinez et al., 2013).

The evaluation of every sample gives an estimate of the target location. Multiple estimates of the target location are gathered to form a single robust prediction of the target location, as shown in Figure 3.33. Specifically, the multiple estimates are denoted as $\{\hat{l}_j\}_{j=1:N_s}$. The estimates are combined and summarised to form an unnormalised mixture of Gaussian distribution A by adding a component to it with predefined covariance

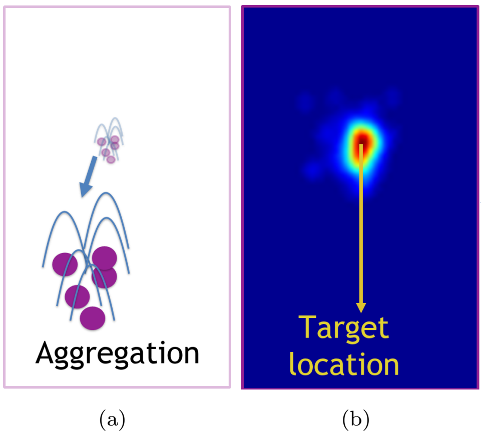


Figure 3.33: The illustration of how the confidence of the estimated target location is obtained. (a) Purple points represent different predictions of target location. At each of these points, a Gaussian unit is gathered. (b) After summarising the Gaussian units at different predicted locations, a heat map is obtained and the peak location is selected as the final robust estimation of target location. The peak value (after normalisation by the number of predictions gathered) is used as the confidence of the estimated target location.

as:

$$A(x) = \sum_{j=1}^{N_s} N(x; \hat{l}_j, \Sigma_{ev}), \tag{3.4}$$

where Σ_{ev} is set to be 3, the same as used in LEAR (Martinez et al., 2013). The target location of the full tracker is then obtained as:

$$\hat{L} = \arg\max_x A(x). \tag{3.5}$$

A measure of confidence on the prediction $\hat{L}$ can be computed as:

$$p(\hat{L}) = max(A)/N_s. \tag{3.6}$$

By taking the sum of multiple estimates of the target location in Eq. 3.4, it is unlikely that a single wrong estimate will have an impact on the peak location (the final predicted target location of the full tracker). This is in contrast to a multiplicative relation, where the effect of a wrong estimate can be dramatic (Martinez et al., 2013).

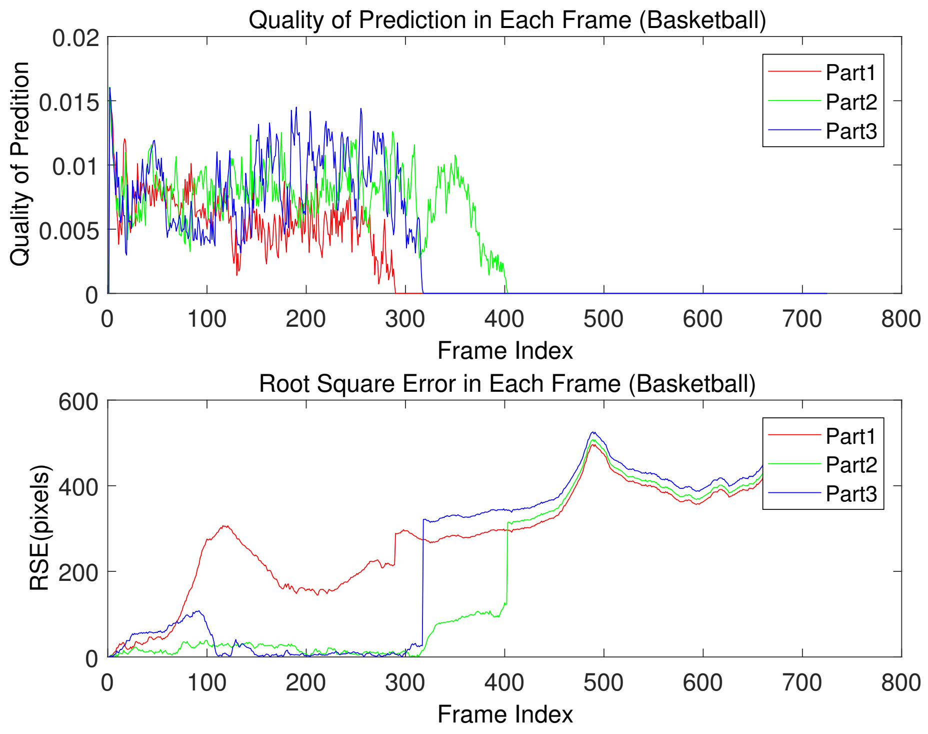
The target is defined by three target points, and the output of each regressor is a 1-dimensional vector. To estimate the displacement vector, two regressors have to be trained, for every target point. In total, $3 \times 2 = 6$ regressors are used to estimate target location by evaluating local appearance descriptors. The relation between the image descriptor and the target location is learned by SVR. Epsilon-SVRs with a histogram intersection kernel are employed, as in LEAR (Martinez et al., 2013). r_x and r_y are learned in the same way. Training is performed in the first frame of video sequence without updating in the following frames. The number of samples, N_s, is 50. The image sample size p_s is decided by the size of the target part, defined as $p_s = (H + W)/2$, where H and W are the height and width of a target part. The definition is to make sure that the image sample can cover most of the target part area. The sampling radius is 15 pixels. The initial target location, except for the first frame, is the predicted target location in previous frame. There is no motion model considered in the simple DDP tracker.

To increase the regressor's predictive power, the intensity histogram is replaced with HOG features (Dalal & Triggs, 2005) to represent the image samples. The advantages of HOG features include (1) more edge orientations beyond horizontal and/or vertical ones are considered, (2) HOG features pool over relatively small image regions, and (3) HOG features are robust to illumination variations of the target. These characteristics also make HOG features more sensitive to the spatial location of the target than, e.g. the intensity histogram, which is especially useful when tracking. This property is also important when updating in model-free tracking, because the predicted target location is used to update the appearance model or shape model of the target, in which small localisation error may propagate over time and accumulate, causing the tracker to drift.

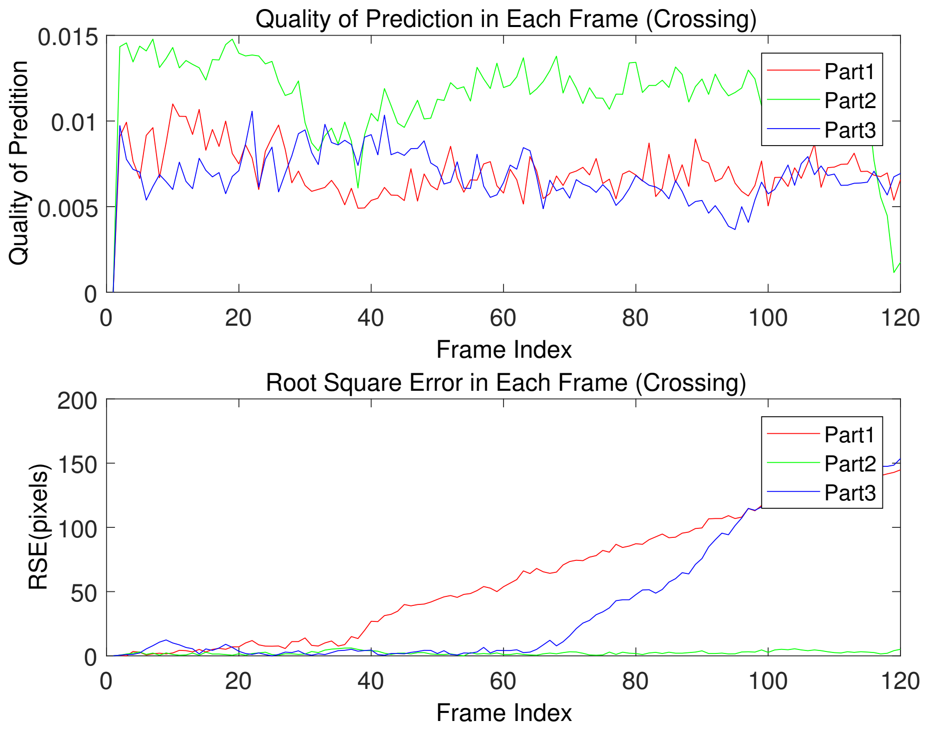
3.3.3 Evaluation of the Simple Direct Displacement Prediction Tracker

The simple DDP tracker is evaluated on five videos from CVPR2013 benchmark (Wu et al., 2013), including *Basketball*, *Crossing*, *David*3, *Jogging* and *Woman*. The measurements for the evaluation include the quality of prediction and the root square error (RSE) of the prediction. The quality of prediction is the confidence on the prediction calculated by Eq. 3.6, while the root square error is the Euclidean distance between the estimated target location and its ground truth. Experimental results of evaluation of the simple DDP tracker using HOG features are as shown in Figure 3.34. Figure 3.34(a) shows the result obtained from video *Basketball*. In the beginning of the video, the target player is occluded by another player. The regressor of Part 1 loses its target very quickly. The regressors of Part 2 and Part 3 are more robust to the occlusion. At around frame 300, the target player bypasses a player with the same basketball uniform. The regressor of Part 2 and Part 3 mistakenly treat that player as the target. Figure 3.34(b) shows the result obtained from video *Crossing*. It shows that the regressor for Part 2 is able to track the target through the whole video, while regressors of Part 2 and 3 lose the target because of background clutter and scale variation. Figure 3.34(c) shows the result of video *David*3. At around frame 20, the target is partially occluded by a road sign post, causing tracking of Part 1 to fail. The background of the target is significantly changed around frame 40, causing failures in tracking Part 2 and 3. Figure

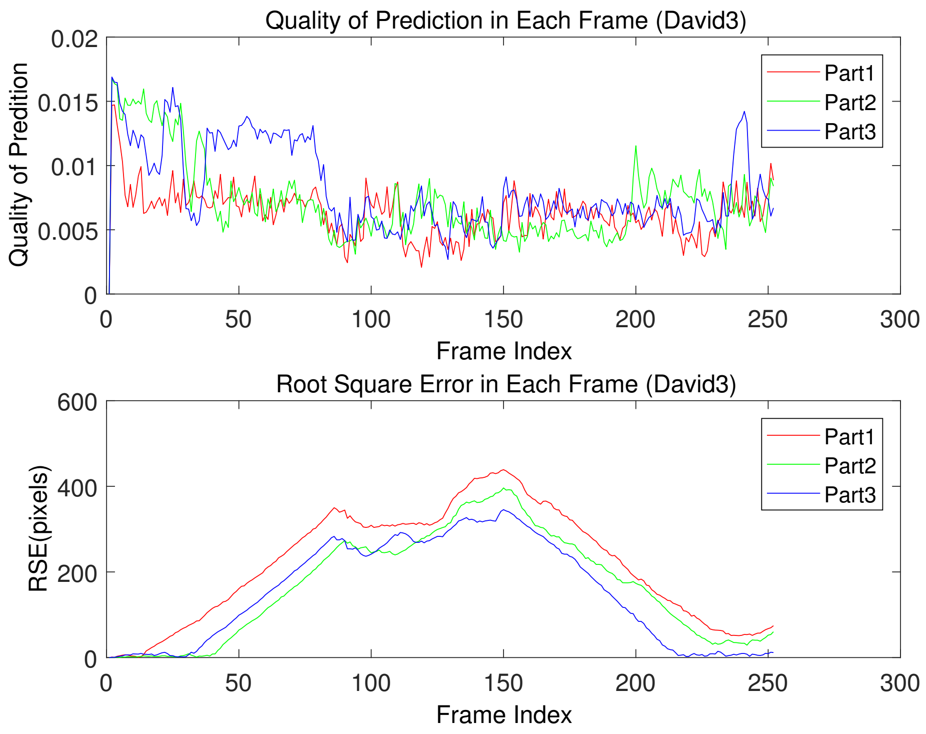
3.34(d) shows the result of video *Jogging*. Around frame 60, the target is fully occluded for several frames, which causes the failure. Before that, the target is correctly tracked. Figure 3.34(e) shows the result of video *Woman*. At around frame 30, the background of Part 2 changes significantly, causing tracking failure of Part 2, while other two parts are tracked correctly. At around frame 90, the target is half occluded, then only Part 1 is still correctly tracked for a number of frames. Experimental results show that, in a simple tracking situation, e.g., when the target is not occluded or has no fast movement, the tracker can track the target successfully through a number of frames without updating. When tracking failure of one part happens, it is often found that other parts are still correctly tracked. It is expected that explicit spatial constrains between parts would give more robust performance. Quality of prediction measures also show that the quality is positively related to the accuracy of the prediction. The experimental results have confirmed the ability of the direct displacement prediction tracker to handle easy tracking scenes, and have also shown proof of concept of the simple DDP tracker and its potential to be improved. The DDP tracker takes about 0.2s to process a frame and provide predictions. 25% of the time is used to calculate HOG features of image samples. 40% of the time is used to predict target location using SVR. The remaining time is used to do all other processing in the algorithm.



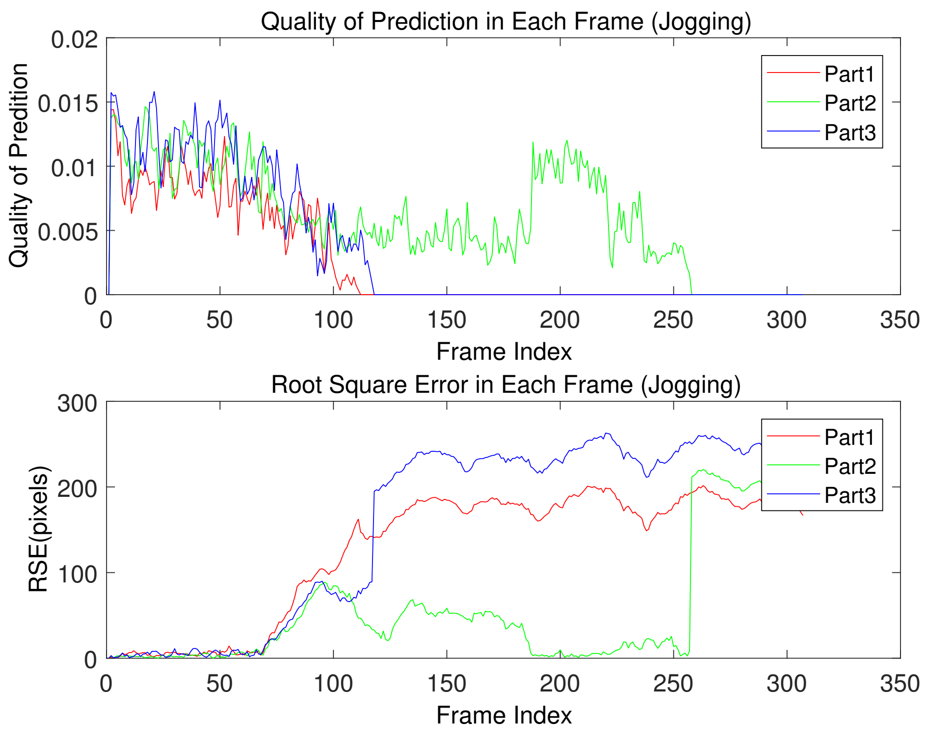
(a) Basketball



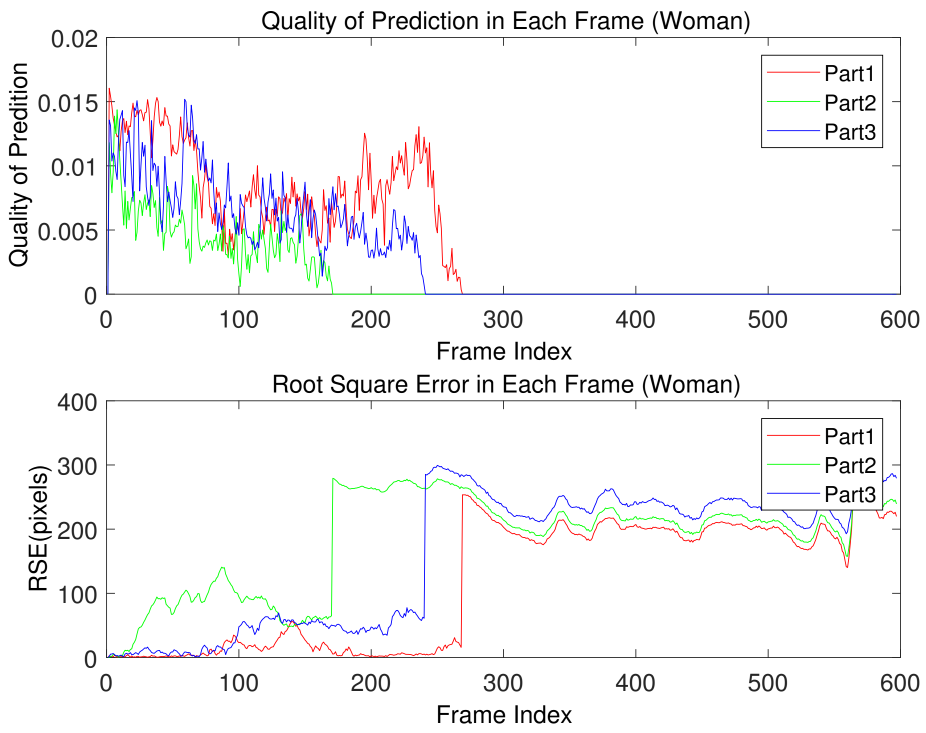
(b) Crossing



(c) David3



(d) Jogging



(e) Woman

FIGURE 3.34: Experimental results of evaluation of estimated three target parts locations from the simple DDP tracker on videos (*Basketball*, *Crossing*, *David3*, *Jogging* and *Woman*).

3.3.4 Limitations of the Simple DDP Tracker

A simple direct displacement prediction tracker, which is able to handle easy tracking scenes, has been built. However, there are a large number of challenging conditions in tracking, such as illumination changes, occlusion, non-rigid deformation, rigid deformation, etc.

To perform successful tracking in more complex conditions, the regressor trained in the first frame has to be updated to adapt to variations of the target over time. As explained in Chapter 2, updating is critical in visual tracking, updating with inaccurate image information will accumulate over time and cause a drift problem in the end. Before adopting a strategy to decide when to update, the more important issue is to improve the accuracy of the predicted target location. If the predicted target location is accurate, then it is less possible for the tracker to include incorrect target information during updating.

Until now, the proposed tracker tracks all three parts of the target separately. Experimental results in Figure 3.34 show that, in many cases, one part or two parts are traced correctly while the remaining part(s) has/have already drifted away. In visual tracking, the background variations are complex and the variations of background of different parts may not be consistent over time. It happens that the regressor for one part is reliable while the regressor for another part is not working due to the significant appearance variation of the part, which might be caused by illumination variation, occlusion, background clutter or other factors. Based on the literature review on the part-based tracker SPOT (L. Zhang & van der Maaten, 2013), it is known that spatial information is critical for tracking. The shape model of a target would provide spatial constraints to the prediction of target location, making the prediction more robust in tracking with challenging conditions.

In addition, currently, there is no motion model in the simple DDP tracker. It has been shown that the direct displacement prediction strategy works well when test locations are sampled relatively close to the target. A flexible and robust motion model would be able to provide good initial target location which can facilitate the direct displacement prediction. Also, a motion model will make the proposed a full tracker instead of 'tracking by detection'.

Based on experimental results and the above analysis, the advantages of the simple direct displacement prediction tracker can be concluded:

- The point-based initialisation avoids background in representing the target. It brings convenience in establishing shape model implicitly by considering the displacements between different points.
- Training a regressor modelling the relationship between image patches and displacements (from the image patches to target location) using SVR provides an opportunity of estimating target location directly rather than tracking by matching.
- The aggregation method used in the simple DDP tracker gives a single robust estimation of target location by aggregating correct predictions and cancelling out random predictions.
- The proposed simple DDP tracker is able to handle easy tracking scenes without illumination changes, occlusion, non-rigid deformation, rigid deformation, etc.

The disadvantages of the simple direct displacement prediction tracker can be summarised as well:

- The accuracy of prediction given by the simple direct displacement prediction tracker can be further improved to assist correct updating.
- An updating strategy, which is necessary for handling varied tracking scenarios with different challenging conditions, is missing in the simple DDP tracker.
- Spatial constraints are not utilised in the simple DDP tracker. It can be expected that spatial relations can help estimate the location of the whole target robustly when the target is partly occluded and the available part is tracked correctly.
- A proper motion model is not considered in the simple DDP tracker. A proper motion model can provide initial location which is closer to the real target location, which is beneficial for further estimation by regression.

Chapter 4

Tracking by Regression with Incrementally Learned Cascades

Based on the work on the simple DDP tracker presented in Chapter 3, this chapter will present the main contribution of this thesis, a part-based tracking method coined Tracking by Regression with Incrementally Learned Cascades (TRIC-track). TRIC-track (see Figure 4.1) combines cascaded regression, a novel implicit shape model, incremental learning and integrates a multiple temporal scale motion model on top of the simple DDP tracker. Finally, TRIC-track is evaluated using both hand-crafted and state-of-the-art deep-learned features.

4.1 TRIC-track: An Overview

Four main problems with current part-based trackers are identified in Chapter 1, and this thesis proposes to address all these issues with one method. The local fitness-based approach is replaced by direct displacement-based tracking in which the proposed tracker predicts the two dimensional displacement vector between the centre of a sampled image patch and the target (part) location using regressors (see Figure 4.2). In doing so, local patches contribute to the solution by directly 'voting' for the target (part) location. In addition, while template-based approaches need to model part appearance and shape fitness separately, the proposed direct displacement prediction by regression tracker implicitly learns the shape and possible deformations of an object. It does so by tracking each part using not only the local evidence for that part, but also evidence provided by neighbouring parts and the object as a whole.

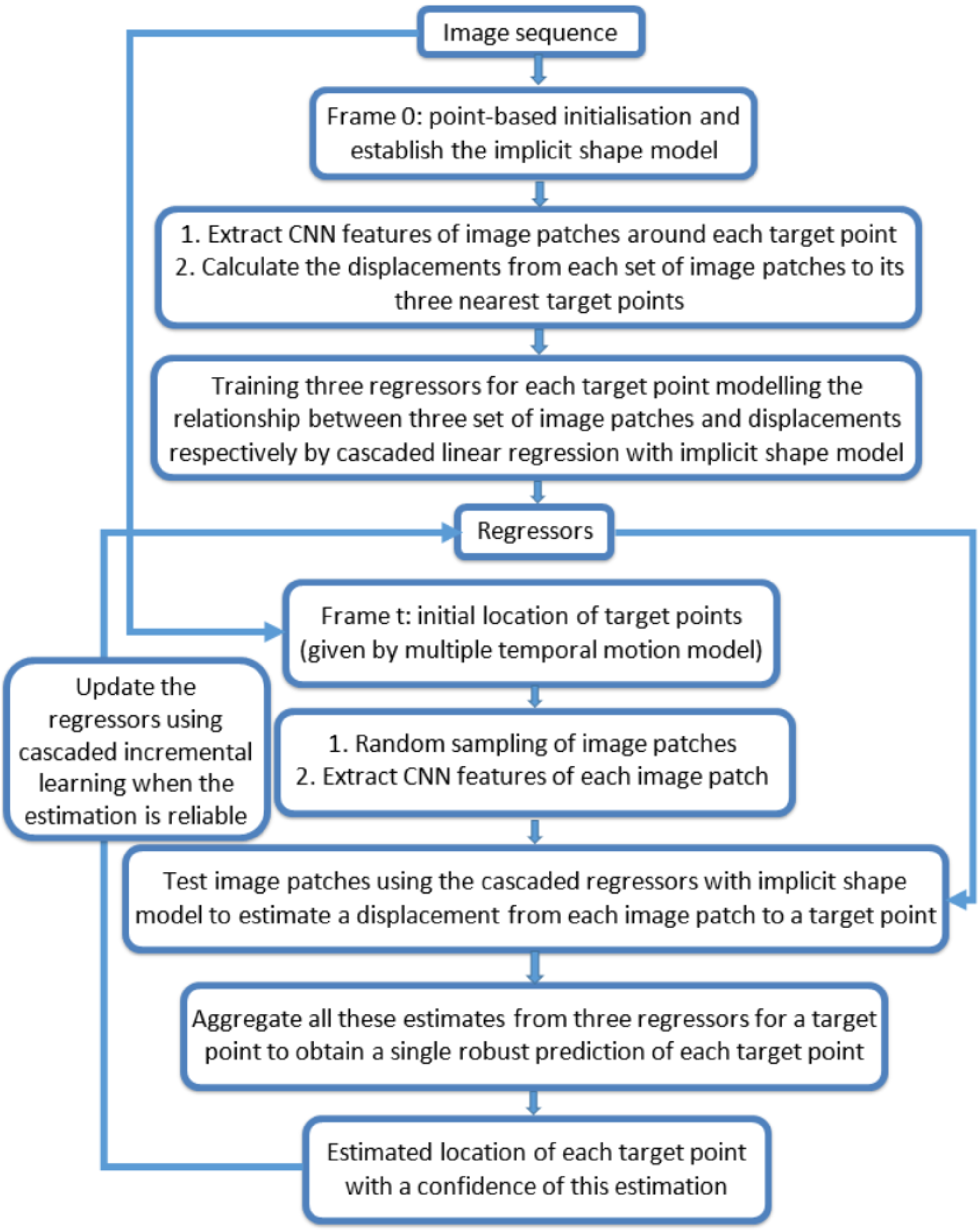


Image sequence
Frame 0: point-based initialisation and establish the implicit shape model
1. Extract CNN features of image patches around each target point
2. Calculate the displacements from each set of image patches to its three nearest target points
Training three regressors for each target point modelling the relationship between three set of image patches and displacements respectively by cascaded linear regression with implicit shape model
Regressors
Frame t: initial location of target points (given by multiple temporal motion model)
Update the regressors using cascaded incremental learning when the estimation is reliable
1. Random sampling of image patches
2. Extract CNN features of each image patch
Test image patches using the cascaded regressors with implicit shape model to estimate a displacement from each image patch to a target point
Aggregate all these estimates from three regressors for a target point to obtain a single robust prediction of each target point
Estimated location of each target point with a confidence of this estimation

FIGURE 4.1: The framework of the TRIC-track tracker.

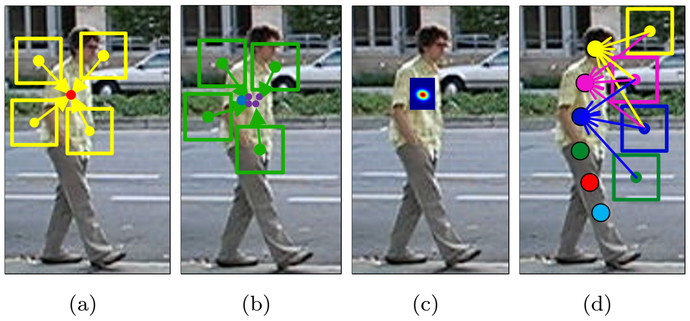


FIGURE 4.2: (a) Training a direct displacement regressor with four examples. (b) Testing a regressor. Four test patches sampled around the initial location (blue dot) provide predictions (purple dots). (c) Evidence aggregation map. (d) Location-based initialisation and implicit shape model.

The proposed TRIC-track adopts cascaded regression (Cootes et al., 2001; Xiong & De la Torre, 2013), specifically the Supervised Descent Method (SDM) (Xiong & De la Torre, 2013), which has been successfully applied to the localisation of facial points. Cascaded regression makes increasingly smaller steps towards the target location, each regressor in the cascade being trained on a smaller region around the ground truth and thus having a smaller expected error. While SDM has been used for what is essentially structured object detection, it has never been used for online model-free tracking. The key difference between detection of a known object and generic object tracking is that appearance and structure models of the former can be learned offline on potentially hundreds of thousands of images, while the models for the latter must be initialised on a single frame.

To learn the appearance changes of parts over time, this thesis takes inspiration from the incremental learning of cascaded regression proposed by Asthana et al. (Asthana, Zafeiriou, Cheng, & Pantic, 2014). That work personalises SDM for facial point localisation, initialising the SDM offline on a large database of faces, and using newly tracked faces to incrementally update it. In the TRIC-track approach the SDM is initialised on the first frame only, and it uses Local Evidence Aggregation (Martinez et al., 2013) of the separate regression contributions to provide a confidence level, which is used to decide whether the tracker can use the tracking result to update the trained regressors. Moreover, this thesis adopts deep learned features to replace hand crafted features to maximise the performance of the proposed TRIC-track tracker. A shape correction step is conducted when outlier predictions appear.

4.2 Limitations of Previous Methods

Previous regression-based tracking methods tend to use models (regressors) of high complexity to address planar or rigid object tracking. For example, Williams et al. (Williams et al., 2005) adopted Relevance Vector Machine (RVM) to provide displacement predictions when tracking faces, hands and cars. Zimmermann et al. (Zimmermann et al., 2009) trained a number of independent linear predictors from which the optimal sequential predictor for tracking planar objects is selected. As a single motion prediction is made by the linear predictor of each level, the selection of the support set which contains the image intensity is important as well. All of these requirements make the tracking complicated and limit its application.

Some regression-based approaches are similar to the proposed TRIC-track in some aspects. For example, Williams et al. (Williams et al., 2005) proposed to use perturbations of a seed image to train a RVM, which is a routine similar to TRIC-track for online learning of the regressors. Furthermore, Zimmermann et al. (Zimmermann et al., 2009) proposed the use of a linear cascade of predictors instead of a single regressor, which is a similar idea to SDM. The main differences are however that 1) TRIC-track does part-based structured object tracking, 2) TRIC-track uses the novel techniques developed for structured regression of (Xiong & De la Torre, 2013) instead of more cumbersome techniques such as structured SVM, 3) TRIC-track combines predictions in a robust manner using evidence aggregation (Martinez et al., 2013) rather than resorting to a classifier to validate the predictions, and 3) TRIC-track integrates a multiple temporal scale motion model to initialise the search in subsequent frames.

Furthermore, TRIC-track does not require an explicit shape model (Xiong & De la Torre, 2013) but rather imposes spatial consistency implicitly. It is interesting to note that, for face alignment (at its core a structured object alignment problem), the introduction of discriminatively-trained regression has revolutionised the state-of-the-art.

Finally, TRIC-track shows that CNN features can be efficiently and effectively used for direct-displacement based tracking. CNN features have been used in tracking before for classifying foreground versus background (Nam & Han, 2016) or to create saliency maps (Ma et al., 2015; L. Wang et al., 2015; Hong, You, Kwak, & Han, 2015). The proposed TRIC-track approach, however, does not need a direct prediction of target and background, but rather uses the features as a way to obtain very distinct patch descriptions. The discriminative power of these CNN patch descriptors over conventional descriptors like HOG and SIFT have recently been shown in several works (Fischer et al., 2014; Hou et al., 2015; Zagoruyko & Komodakis, 2015).

4.3 TRIC-Track

This section describes the proposed method of tracking by regression with incrementally learned cascades (TRIC-track). This section first explains direct displacement prediction by regression in 4.3.1. The implicit structure model with the global part is illustrated in 4.3.2. The section then introduces the adopted framework of cascaded linear regression (Xiong & De la Torre, 2013) in 4.3.3. The cascaded incremental learning and multiple temporal motion modelling are explained in 4.3.4 and 4.3.6. The feature adopted is introduced in 4.3.5.

In the simple DDP tracker described in Section 3.3, the target is represented with three parts, as shown in Figure 3.8, and the centres of parts are tracked separately in every frame. The parts used in the simple DDP tracker are meaningful, for example, the first part covers the head area, while the second part covers the upper body area. This representation is not flexible enough when handling general deformable object tracking, as it's hard to automatically detect meaningful parts representing an object. In TRIC-track the target is therefore described by N parts of equal size, actually N equally spaced points, instead of the previous non-general representation of a target.

In the simple DDP tracker, SVR is used as the regression method. As stated in LEAR (Martinez et al., 2013), other regression algorithms are also applicable. Updating the cascaded SVR incrementally needs re-training during each cascade level of regression, which is cumbersome. Thus, SVR is replaced with linear regression as used in SDM (Xiong & De la Torre, 2013), which is simpler, more efficient and allows easier implementation of incremental updating.

4.3.1 Direct Displacement-based Prediction

As the proposed tracker directly predicts the locations of a number of parts of the object by modelling the displacements from local image patches to those parts (targets), the tracker is initialised by defining part locations as 2D points instead of a set of bounding boxes (see Figure 4.4). N initial points representing corresponding part locations are used to model the target's structure in the first frame of an image sequence. The locations are denoted as $L^* = [l_1^*, ...l_i^*, ...l_N^*]$, where N is the number of parts and $l_i^* = (x_i^*, y_i^*)$ is the ground truth location of part i.

In the training stage, given an image I and part locations L^*, a set of training sample locations S is obtained by randomly sampling around L^*, where each l_i^* is sampled independently. Each training sample is a local image descriptor $\Phi(I, S, p_s)$ extracted from a square image patch centred at sample location S with size $p_s \times p_s$. $\Phi(I, S, p_s)$ and the displacement D^* ($D^* = L^* - S$) between S and the target's location L^* are then used to train the regressor R, which uses image information to predict the displacement to the part location.

Similarly, in the test stage, a number of test samples S_{test} are selected around an initial candidate target location L_{init}. The location L_{init} is determined by the multiple temporal scale motion model described in Section 4.3.6. The regressor R then predicts the displacement D from these samples' locations S_{test} to the target using local image feature $\Phi(I_{test}, S_{test}, p_s)$ by:

$$\hat{L} = S_{test} + R\Phi(I_{test}, S_{test}, p_s), \tag{4.1}$$

where $D = R\Phi(I_{test}, S_{test}, p_s)$ and $\hat{L}$ is the optimal estimation of target location ($\hat{L} = L^*$ in frame 0). As there is no means of determining the quality of a single prediction, a number of predictions are combined to determine where the target is, as explained in Section 3.3.2 and the last paragraph of Section 4.3.3. The rationale behind this is that correct predictions will aggregate around the same location, reinforcing each other, but erroneous predictions will be more or less random (Martinez et al., 2013).

4.3.2 Implicit Shape Model with Global Part

The shape models employed by previous part-based online tracking methods can be categorised as the independent model (Shahed Nejhum et al., 2008; Liu et al., 2015), the star model (Adam et al., 2006; M. Yang et al., 2007; Kwon & Lee, 2009; L. Zhang & van der Maaten, 2013), the tree model (L. Zhang & van der Maaten, 2013) and the hierarchical model (Cehovin et al., 2013), as shown in Figure 4.3. With the independent shape model (Figure 4.3(a)), all parts are tracked independently. The star model (Figure 4.3(b)) connects each part with the centre of the target. In a tree model (Figure 4.3(c)), each part's location is only dependent on the location of its parent. A hierarchical model (Figure 4.3(e)) contains a global layer describing the whole target and a local layer containing a geometrical constellation of parts. Each part is connected with its nearest neighbours and also guided by the global layer.

The direct displacement prediction method provides a unique opportunity to combine local information from multiple parts, something not possible with local fitness methods. Consider part i and its neighbours part $i-1$ and part $i+1$ (see Figure 4.4). When seeking to locate part i, three separate regressors are trained using image information from S_{i-1}, S_i and S_{i+1} respectively (conforming to Figure 4.3(d) and see Figure 4.4). Similarly, in the testing stage samples around initial candidate locations of part $i-1$, part i and

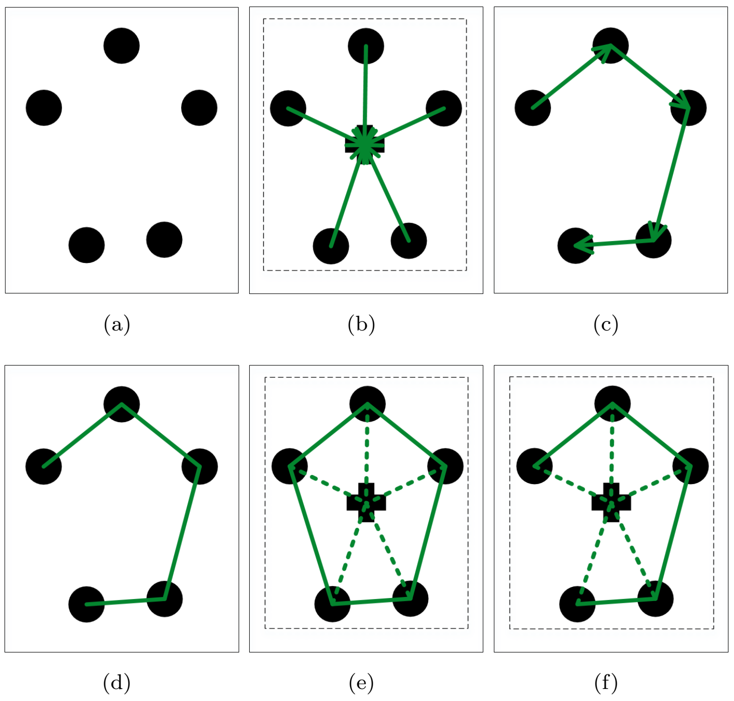


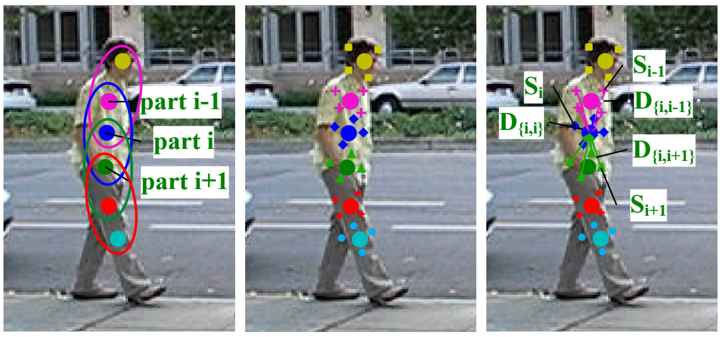
FIGURE 4.3: (a) The independent shape model. The black dots represent different parts. (b) The star shape model. The dashed rectangle represents the target area. The black cross represents the target centre. Green edges represent the spatial relation between parts and the target centre. (c) The tree shape model. (d) The proposed implicit shape model. (e) The hierarchical model. The dashed rectangle is the global layer. (f) The proposed implicit shape model with the global part.

part $i+1$ are used to predict the location of part i. In this way, the spatial relationships between neighbouring parts are implicitly learned and applied by TRIC-track.

Modelling groups of three parts in this way balances local and global shape. A given part's movements are closely correlated with its neighbours', while state changes in far away parts are less likely to affect the given part's. By combining target parts in overlapping groups of three (including a part and its two neighbouring parts), these local shape relations effectively create a global shape model (see Figure 4.4). Using image information from multiple parts also helps to avoid over-fitting. This embedding of an implicit shape model in a set of regressors is only possible because TRIC-track uses direct displacement prediction for tracking. Template matching approaches require target shape to be made explicit in the appearance model.

In preliminary experiments, a star shape model, with one part as the root and other parts directly connected with the root, was also tried , but the spatial constraints of the star shape model were found to be very loose, so that parts tend to drift easily. In the common shape definition in facial points localisation (Martinez et al., 2013), features sampled around different target points are concatenated to train one regressor to predict the whole shape of the face. Preliminary experiments found this shape model to be less robust than the proposed implicit shape model, as it loses the spatial constraints from neighbouring parts. Concatenating features together is not a strong shape model for general object tracking.

Although linking the parts together with the implicit shape model forms a global shape, global image information is not applied, which would be a lost opportunity. In the same way that the direct displacement prediction mechanism provides the opportunity to combine local information from multiple parts, it can also add global image information to the local image information. Similar to utilising the image information from neighbouring local parts, global features are made use of by adding a global part, the $(N+1)_{th}$ part (conforming to Figure 4.3(f) and see Figure 4.5). It is the third neighbour of each local part and its location is initialised at the centre of the bounding box given in frame 0. Thus, for part i, its neighbours are part $i-1$, part $i+1$ and part $N+1$, as shown in Figure 4.5(a). To locate part i with the global feature, an extra regressor is trained, modelling the relationship between image information from S_{N+1} and displacement from S_{N+1} to the location of part i. The difference is that the size of the sample patch extracted from S_{N+1}, noted with g_s, is much bigger than the sample size of local parts, noted with p_s, to include global target information. When the image information from local part neighbours is not able to predict the location of target part correctly, for example when three consecutive parts are occluded, the global image information can provide compensation. The neighbours of the global part are all the other parts, and the location of the global part is determined by prediction from all other parts.



(a) (b) (c)

FIGURE 4.4: (a) Parts initialisation. (b) Training samples obtained around each part. (c) Implicit shape model. Three regressors estimate one part's location, each trained using features from samples around a different part.

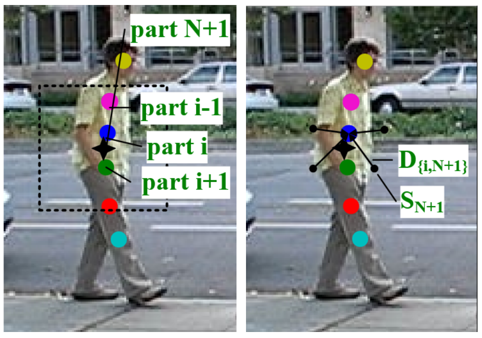


FIGURE 4.5: (a) Parts initialisation with the global part (the black star). The samples extracted around the global part will include global information, as shown with the black rectangle. (b) Predictions from the global part (black arrows). The black dots are sample locations around the global part.

4.3.3 Cascaded Linear Regression Tracker

In the implementation of displacement prediction by regression, this thesis adopts a cascaded linear regression framework, Supervised Descent Method (SDM) (Xiong & De la Torre, 2013), which was originally proposed to locate facial landmarks. This thesis adopts the SDM method to seek the optimal prediction of the target location.

Newton's method and its variants are accepted as the classical way of solving nonlinear optimisation problems. It calculates or estimates the Hessian matrix and Jacobian matrix to update the minimum. Newton's method requires the function being optimised to be twice differentiable, which is often not the case in computer vision problems. Thus, numerical estimation of the Hessian matrix would be computationally expensive. The Hessian matrix could be large and non-positive definite when applying Newton's method to computer vision applications. SDM was proposed to solve these drawbacks by minimising a Non-linear Least Squares (NLS) function without calculation of the Jacobian or the Hessian matrix. Specifically, a sequence of average descent directions (regressors) is learned by minimising the mean of NLS functions sampled at different points where the corresponding minimums are known. During testing, using the learned sequential average descent directions, SDM is able to minimise the NLS function. When an unseen test location appears, a corresponding minimum would be given by applying the learned generic directions (regressors) to that test location.

The idea of SDM is basically to use several levels of regression. The test samples are evaluated by the first cascade level of regressor, then the predictions obtained are considered as the test samples for the second cascade level of regressor. This process is conducted over all the following cascades. In SDM (Xiong & De la Torre, 2013), the R

is learned with nearly a thousand images offline (to be reliable) and only one prediction is performed to estimate the facial landmark location. See (Xiong & De la Torre, 2013) for more details.

Crucially, as offline learning is not available for model-free visual tracking, only ground-truth information in the first frame is given. The proposed method is not trained offline as in (Xiong & De la Torre, 2013), but is initialised based on the part locations automatically obtained in frame 0 (explained in Section 4.4) and learned and updated on the fly. In order to handle the larger variability of objects, the set of parts are split into overlapping sub-groups of three parts plus a global part, as stated in Section 4.3.2. Each part's location is predicted by local parts from its related sub-group and the global part. Then, all of these predictions are combined in a robust manner using aggregation. The cascaded linear regression framework used in the proposed tracker is presented as follows.

When training for a given image I and initial part locations L^* including the global part location l^*_{N+1}, $l^*_{N+1} = (x^*_{N+1}, y^*_{N+1})$, K samples are obtained for each part by randomly sampling from a local area around the part's location l^*_i, $i \in (1, N+1)$. With preliminary experiments presented in Chapter 3, it was found that image patches within a close area around the target, where image patches can cover some target (i.e. foreground) information, can reliably predict target location while those further away cannot, as there is no consistent relation between background data and the position of a moving object. Sampling locations around part i are denoted as $S_i = [s_{i1}, ...s_{ij}, ...s_{iK}]^T$ where s_{ij} is a 2D location in the image, $s_{ij} = (x_{ij}, y_{ij})$. The local features $\Phi(I, S_i, p_s)$ are extracted from square image patches centred at S_i.

The displacement vector (the learning goal) is defined to be the vector d^*_{ij} from s_{ij} to l^*_i with $d^*_{ij} = (x^*_i - x_{ij}, y^*_i - y_{ij})$. The set of all displacement vectors corresponding to K samples in S_i is $D^*_i = [d^*_{i1}, ..., d^*_{ij}, ...d^*_{iK}]^T$. D^*_i is the set of displacements from S_i to l^*_i. This notation is extended to $D^*_{i,i'}$ to allow samples S'_i of another point i' to predict displacements to point i, so that $D^*_{i,i+1}$ represents the displacements from S_{i+1} to l^*_i. This is how local shape is learned. To avoid confusion, this thesis uses D^*_i as shorthand for $D^*_{i,i}$.

As described in Section 4.3.2, four groups of samples are combined to predict part i's location, which means the features from S_{i-1}, S_i, S_{i+1} and S_{N+1} and their corresponding displacements $D^*_{i,i-1}$, $D^*_{i,i}$, $D^*_{i,i+1}$ and $D^*_{i,N+1}$ are used to train four regressors $R_{i,1}$, $R_{i,2}$, $R_{i,3}$ and $R_{i,4}$ (descent directions) for a local part i. The regressors for the global part are trained in the same way. The only difference is its neighbours are all local parts. For simplicity, this thesis uses the situation of a local part i as an example to explain the mechanism of training and testing of regressors. The trained regressors are able to map the samples' local features $\Phi(I, S_{i-1}, p_s)$, $\Phi(I, S_i, p_s)$, $\Phi(I, S_{i+1}, p_s)$ and $\Phi(I, S_{N+1}, g_s)$ to the ground truth displacement vectors $D^*_{i,i-1}$, $D^*_{i,i}$, $D^*_{i,i+1}$ and $D^*_{i,N+1}$, which means that the local evidence of all three sets of samples and the global evidence of the fourth set of samples can be used to predict a single target part's location.

The regressor is obtained by minimising the difference between the predictions and the ground truth. For example, the loss functions for $R_{i,1}$, $R_{i,2}$, $R_{i,3}$ and $R_{i,4}$ of part i are:

$$||D^*_{i,i-1} - R_{i,1}\Phi(I, S_{i-1}, p_s)||, \tag{4.2}$$

$$||D^*_{i,i} - R_{i,2}\Phi(I, S_i, p_s)||, \tag{4.3}$$

$$||D^*_{i,i+1} - R_{i,3}\Phi(I, S_{i+1}, p_s)||, \tag{4.4}$$

$$||D^*_{i,N+1} - R_{i,4}\Phi(I, S_{N+1}, g_s)||, \tag{4.5}$$

and the regressors are obtained by minimising these loss functions. In the following text, we refer to the group of $R_{i,1}$, $R_{i,2}$, $R_{i,3}$ and $R_{i,4}$ as R_i. It is hard to find the optimal prediction of the direct displacement in one step (Xiong & De la Torre, 2013). Therefore, the displacement is learned in a cascade of steps with decreasing distance to the target. The cascaded linear regression method (Xiong & De la Torre, 2013) first learns R^0_i, where 0 denotes the first level of the cascaded regression. Similarly, D^{0*}_i denotes the displacements used for training R^0_i. R^0 is obtained by:

$$\underset{R^0_{i,1}}{\arg\min} \, ||D^{0*}_{i,i-1} - R^0_{i,1}\Phi(I, S^0_{i-1}, p_s)|| + ||\omega||^2, \tag{4.6}$$

$$\underset{R^0_{i,2}}{\arg\min} \, ||D^{0*}_{i,i} - R^0_{i,2}\Phi(I, S^0_{i}, p_s)|| + ||\omega||^2, \tag{4.7}$$

$$\underset{R^0_{i,3}}{\arg\min} \, ||D^{0*}_{i,i+1} - R^0_{i,3}\Phi(I, S^0_{i+1}, p_s)|| + ||\omega||^2, \tag{4.8}$$

$$\underset{R^0_{i,4}}{\arg\min} \, ||D^{0*}_{i,N+1} - R^0_{i,4}\Phi(I, S^0_{N+1}, g_s)|| + ||\omega||^2, \tag{4.9}$$

where $||\omega||^2$ is the regularisation term. Please note that each regressor is trained with its corresponding samples and displacements. Eqs. (4.6)-(4.9) are minimised to obtain $R^0_{i,1}$, $R^0_{i,2}$, $R^0_{i,3}$ and $R^0_{i,4}$. For brevity X^0_{i-1} is used to denote $\Phi(I, S^0_{i-1}, p_s)$. $R^0_{i,1}$, $R^0_{i,2}$, $R^0_{i,3}$ and $R^0_{i,4}$ can be estimated by Ridge Regression (Hoerl & Kennard, 1970):

$$R^0_{i,1} = [(X^0_{i-1})^T(X^0_{i-1}) + \lambda E]^{-1}(X^0_{i-1})^T D^{0*}_{i,i-1}, \tag{4.10}$$

$$R^0_{i,2} = [(X^0_{i})^T(X^0_{i}) + \lambda E]^{-1}(X^0_{i})^T D^{0*}_{i,i}, \tag{4.11}$$

$$R^0_{i,3} = [(X^0_{i+1})^T(X^0_{i+1}) + \lambda E]^{-1}(X^0_{i+1})^T D^{0*}_{i,i+1}, \tag{4.12}$$

$$R^0_{i,4} = [(X^0_{N+1})^T(X^0_{N+1}) + \lambda E]^{-1}(X^0_{N+1})^T D^{0*}_{i,N+1}, \tag{4.13}$$

where E is the identity matrix and λE is added to make $X^T X$ numerically stable. After one cascade level, R^0 is obtained and first level predictions of part i's location can be obtained by uniting neighbouring parts' predictions as follows:

$$S_i^1 = (S_{i-1}^0 + R_{i,1}^0\Phi(I, S_{i-1}^0, p_s)) \cup (S_i^0 + R_{i,2}^0\Phi(I, S_i^0, p_s)) \\ \cup (S_{i+1}^0 + R_{i,3}^0\Phi(I, S_{i+1}^0, p_s)) \cup (S_{N+1}^0 + R_{i,4}^0\Phi(I, S_{N+1}^0, g_s)), \quad (4.14)$$

where S_i^1 is the first cascade level prediction of part i. First level predictions of all other parts are obtained similarly.

Like the first level of cascaded regression, $R_{i,1}^1$, $R_{i,2}^1$, $R_{i,3}^1$ and $R_{i,4}^1$ are obtained by minimising equations 4.6, 4.7, 4.8, 4.9 with S_{i-1}^1, S_i^1, S_{i+1}^1 and S_{N+1}^1 and their corresponding displacements $D_{i,i-1}^{1*}$, $D_{i,i}^{1*}$, $D_{i,i+1}^{1*}$ and $D_{i,N+1}^{1*}$. The R^2,R^3,... are learned in the same way.

In the test stage, the same number of cascaded regressors contribute to the final target location prediction S_{testi}^n of part i, in which n is the number of cascade levels and fixed to 4 in the experiments. Local Evidence Aggregation for Regression (LEAR) (Martinez et al., 2013) is used as a principled way of combining the individual evidences. In LEAR, each prediction contributes a unit two-dimensional Gaussian with a fixed standard deviation. LEAR then aggregates all these predictions into an un-normalised likelihood map of the target's location. The peak value of this likelihood distribution determines part i's position.

4.3.4 Cascaded Incremental Updating

One of the challenges of visual tracking is that ground truth templates, or in TRIC-track's case ground truth direct displacements, are available only for the first frame. The appearance of tracked parts and the shape of the whole object are, however, likely to change over time, especially for deformable objects. The problem is then to decide when and how to update the appearance model (i.e. regressors) without succumbing to the model drift problem.

To avoid drift, the proposed tracker should only update the regressors with a new prediction when it is confident of that prediction. For example, when a part is occluded, although its estimated location is correct because of the shape constraints, the proposed tracker should not update its regressor. Thus, the tracker only updates a part's regressors when the confidence of the part's predicted location is higher than an empirically determined threshold δ_v. As described in 4.3.3, the prediction of cascaded regression results in a summation of unit Gaussians, one for every prediction made. To estimate the confidence in the final prediction, the peak value of the likelihood map of each part is divided by the total number of predictions. The new peak value is used to evaluate the goodness of predicted target location. The theory behind this is that correct predictions tend to point to the real target location, so that the aggregation map is condensed. Wrong predictions tend to be random and the aggregated distribution is sparse. If the new peak value is greater than δ_v, the predictions are densely distributed. Thus, predictions are reliable and this tracking result can be used to update regressors. δ_v is decided by observing the aggregation maps of different videos in preliminary experiment (presented in Chapter 3), which shows the value of δ_v when the target is located correctly. As δ_v is a normalised peak value of aggregated Gaussian units (corresponding to predictions of target location), it is irrelevant to video content and the empirically determined value of δ_v is the same across all experiments.

The proposed tracker incrementally updates its regressor by adding new training data (Asthana et al., 2014), samples and corresponding displacements from sample location to target location, after estimating the target location in every frame. Turning the update of cascaded regression into cascaded incremental updating is non-trivial due to the inter-relationships between the cascades: if the top-level regressor improves, this would change the training of all subsequent steps in the cascade. Specifically, given the initial D^0 and X^0, first cascade level regressor R^0 is calculated. D^0 and X^0 are then propagated through R^0, the D^1 and X^1 are obtained and regressor R^1 is computed. Similarly, the tracker computes R^2 with D^2 and X^2 obtained by propagating D^1 and X^1 through R^1. The process is continued until the predefined number of cascade levels is reached. Please note that the tracker learns the cascade models on the fly and combines the method with the implicit shape model to update the image information and the shape model at the same time.

For face alignment, Asthana et al. proposed a Parallel Cascade of Linear Regression method to address the problem of sequential incremental updating described above (Asthana et al., 2014). The work reported in this thesis however found that, in tracking, using a sequential incremental update provides better results than the parallel incremental updating approach of (Asthana et al., 2014). This is because, in (Asthana et al., 2014), the statistics of shape parameters of the face are trained on an offline database. The perturbations for training the cascade of regression functions are covered by the learned statistics, without relying on the previous iteration. However, offline training is not available in visual tracking and the limited image information is not able to model the variations of regressors. Thus, the training of each cascade level must rely on the regression predictions made by the previous level.

Given the feature matrix $X(A)$ and displacement matrix $D(A)$, where A is the number of training samples, $R(A)$ is obtained by:

$$R(A) = V(A)X(A)^T D(A), \tag{4.15}$$

$$V(A) = [X(A)^T X(A) + \lambda E]^{-1}, \tag{4.16}$$

then with B new training samples $X(B)$ and corresponding displacement matrix $D(B)$ obtained from the local neighbours and a given part tracked with high confidence, as derived in (Asthana et al., 2014), the updated regressor $R(A+B)$ can be found as:

$$R(A+B) = R(A) - QR(A) + V(A+B)X(B)^T D(B), \tag{4.17}$$

where,

$$V(A+B) = V(A) - QV(A), \tag{4.18}$$

$$Q = V(A)X(B)^T UX(B), \tag{4.19}$$

$$U = [E + X(B)V(A)X(B)^T]^{-1}. \tag{4.20}$$

Note that the new samples added for the update of each part during incremental updating are collected around four parts (the part itself, its two neighbouring parts and the global part) at each cascade level. In this way the shape and appearance aspects of the regressors models are jointly updated, making the proposed tracker robust to the non-rigid deformation of articulated objects.

4.3.5 CNN Features

4.3.5.1 Tracking with Convolutional Neural Networks

Instead of hand-crafted features, Convolutional Neural Network (CNN) features are currently being adopted in tracking algorithms (Ma et al., 2015; L. Wang et al., 2015; Hong et al., 2015; Nam & Han, 2016), showing state-of-the-art performance (Kristan et al., 2015). Most algorithms make use of a network pre-trained on ImageNet from which they extract features (Ma et al., 2015; L. Wang et al., 2015; Hong et al., 2015). The features are extracted from the convolutional layers (Ma et al., 2015; L. Wang et al., 2015), the fully connected layers (Hong et al., 2015) or a combination of multiple layers (Ma et al., 2015; L. Wang et al., 2015; Nam & Han, 2016). The lower convolutional layers (closer to the input layer) give more spatial information and the higher convolutional and fully connected (FC) layers encode more semantic information (Ma et al., 2015).

For better tracking results the networks can be pre-trained or fine-tuned on the first frame and (partially) updated during tracking (Ma et al., 2015; Nam & Han, 2016), although this is not necessary for all networks (L. Wang et al., 2015; Hong et al., 2015).

Wang et al. (L. Wang et al., 2015) use the features of the 10th (4-3) and 13th (5-3) convolutional layers of the VGG-16 network (Simonyan & Zisserman, 2014) to produce two heatmaps of the target , capturing both higher level semantic and lower level selective features. The most discriminating heatmap is then used to localize the target.

Ma et al. (Ma et al., 2015) use the VGG-19 network (Simonyan & Zisserman, 2014) from which they extract feature maps from the 8th (3-4), 12th (4-4) and 16th (5-4) convolutional layers. The feature maps are then used for tracking using a correlation-filter-based approach with a coarse to fine search.

Hong et al. (Hong et al., 2015) use the pre-trained R-CNN network (Girshick, Donahue, Darrell, & Malik, 2014) to produce saliency maps. The saliency map is obtained by training a SVM on the output of the first convolutional layer in the first frame and backpropagating the output features, which are predicted to belong to the target, through the network. The target is then found in the saliency map using sequential Bayesian filtering, as in (Felzenszwalb et al., 2010).

MDNet (Nam & Han, 2016) makes use of a novel CNN architecture, including shared layers with video-specific branches, pre-trained on positive and negative patches from frames of all sequences in a tracking dataset to obtain a generic representation of the target. In the tracking phase the video-specific branches are replaced by a single fully connected layer, and positive and negative samples from the first frame are used to train this new network.

Wang et al. (L. Wang, Ouyang, Wang, & Lu, 2016) propose a sequential training method for convolutional neural networks (CNNs) to effectively transfer pre-trained deep features for online applications. A CNN is regarded as an ensemble with each channel of the convolutional feature map is treated as a base learner. Each base learner is updated using a different loss criterion to avoid over-fitting. Online fine-tuning of the CNN is then formulated as a sequential ensemble learning problem. To build the best ensemble, the base learners are sequentially selected by importance sampling. Online tracking is conducted as foreground/background separation by the sequentially learned ensemble.

Qi et al. (Qi et al., 2016) propose a CNN-based tracking framework which takes full advantage of features from different CNN layers and uses an adaptive Hedge method to hedge several CNN based trackers into a single stronger one. Man steps of the proposed algorithm include: 1) extracting CNN features from different convolutional layers using the pre-trained VGG-Net; 2) constructing weak trackers using correlation filters where each one is trained with CNN features from one layer; 3)hedging weak trackers into a stronger one using an improved Hedge algorithm.

Teng et al. (Teng et al., 2017) present a deep architecture which combines the temporal and spatial information to improve the tracking performance. The deep architecture contains three networks, a Feature Net, a Temporal Net, and a Spatial Net. The Feature Net extracts general feature representations of the target. With these feature representations, the Temporal Net encodes the trajectory of the target and directly learns temporal correspondences to estimate the object state from a global perspective. The Spatial Net further refines the object tracking state using local spatial object information.

Huang et al. (C. Huang, Lucey, & Ramanan, 2017) propose to improve the speed of deep trackers without losing accuracy. The idea is to take an adaptive approach, where easy frames are processed with cheap features (such as pixel values), while challenging frames are processed with invariant but expensive deep features. The adaptive tracking problem is formulated as a decision-making process, and an agent learnt is used to decide whether to locate objects with high confidence on an early layer, or continue processing subsequent layers of a network.

Yun et al. (Yun, Choi, Yoo, Yun, & Choi, 2017) propose to track the target by repetitive actions learned by the proposed action-decision network (ADNet). The ADNet learns to selects the optimal actions to track the target from its current state. During training, the input is an image patch cropped at the position of the previous state and the output is the probability distribution of actions including translation and scale changes.

Han et al. (Han, Sim, & Adam, 2017) propose a visual tracking algorithm which adopts a convolutional neural network (CNN) with multiple branches to represent the target. The target state is estimated by an ensemble of all branches while online model update is performed by the standard error back-propagation. The network has a different number of fully connected layers in individual branches and maintains multi-level target representations based on a CNN using the branches.

4.3.5.2 CNN Features in TRIC-track

In the earlier work of this thesis, HOG features are used as a descriptor of the image patch. In recent papers it is however shown that CNN features are more discriminative than conventional features such as HOG, SIFT and DAISY (Fischer et al., 2014; Hou et al., 2015; Zagoruyko & Komodakis, 2015). Therefore CNN features are adopted in the TRIC-track framework.

The image patch size in TRIC is much smaller than the image size in deep neural networks. A complex and deep structure is not suitable for the image with a limited size. Based on the literature review in Section 4.3.5.1, the VGG-16 network (Simonyan & Zisserman, 2014) has the simplest structure. As an initial setup the 16-layer deep convolutional neural network pre-trained on ImageNet is used (called VGG-16D in the paper). Since the image patch size of the samples from local parts is 24×24 pixels, determined by the re-scaled image size, which is relatively small and results in a shallower neural network, only the first four convolutional layers are chosen to use, reducing the network to the architecture as shown in Table 4.1. Earlier convolutional layers already show discriminative behaviour (Hou et al., 2015) without giving too much semantic meaning and thus being too specific (L. Wang et al., 2015). Furthermore, using only four convolutional layers is beneficial in terms of processing speed.

Features are generated by passing a patch through the network and extracting the output of the last convolutional layer, which results in a 128 dimensional feature vector for input patches of 14×14. For patches larger than 14×14 pixels, the 128 output feature maps are reduced to a 128 dimensional vector by applying max pooling to the feature maps, which keeps the most distinct values in the feature vector and guarantees the calculation efficiency.

Table 4.1: The CNN architecture

	Size	Stride	Padding
Input Image	24x24x3		0
Conv1-1	3x3x64	1	0
ReLU			
Conv1-2	3x3x64	1	0
ReLU			
Max Pool	2x2	2	0
Conv2-1	3x3x128	1	0
ReLU			
Conv2-2	3x3x128	1	0

4.3.6 Multiple Temporal Scale Motion Model

This thesis adds a multiple temporal scale (MTS) motion model, proposed by Khan et al. (Khan et al., 2015), to the proposed regression based tracker, which makes the method a full tracker instead of 'tracking by detection'. As shown in the results section, displacement based regression works well when test locations are sampled relatively close to the target. A good initial estimation of the target position is expected to significantly help provide accurate final predictions of target position.

To counter occlusions and abrupt motion variations, Khan et al. proposed a visual tracker operating over multiple temporal scales (MTS) (Khan et al., 2015). This framework learns motion models from different temporal scales of the target tracking history (i.e. already tracked frames), and applies those models at different temporal scales in the future (see Figure 4.6). Here, a temporal scale is a specific sequence of moments in time, e.g. $[t-4:t]$. The construction of motion models over different temporal scales provides a much richer description of the target's recent path across the image plane. When making predictions, the model set represents variations in target motion better than any single model of this set. The application of these models over multiple temporal scales in the future allows a tracker to overcome periods of occlusion of the object.

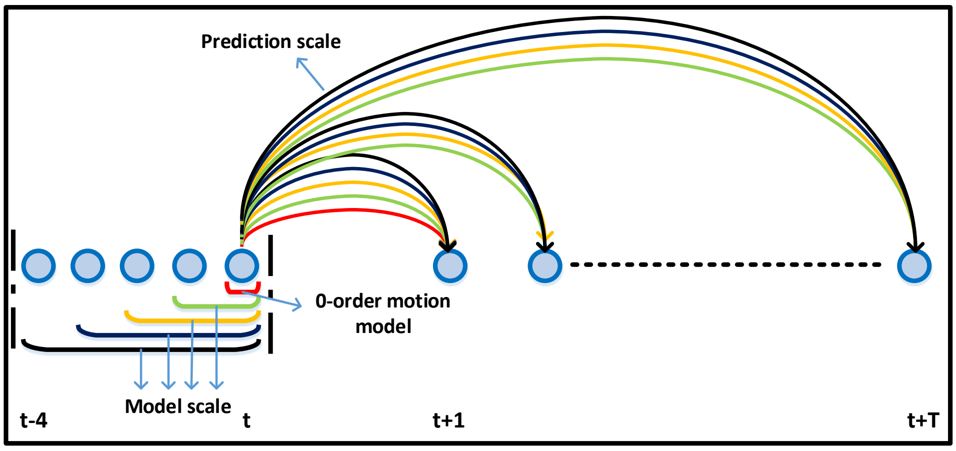


FIGURE 4.6: Multiple motion models are learned from the recent tracking history at different temporal scales, and each model is applied over multiple temporal scales in the future.

Simple motion models $\mathbf{M}$ are learned over multiple model-scales and are used to make state predictions over multiple prediction-scales, by fitting a first-order polynomial function. $\mathbf{M}$ is learned at a given model-scale separately for the x-location, and y-location of the target's state. For instance, an $\mathbf{M}$ learned at model-scale m, predicts a target's x-location at time t as:

$$\tilde{x}_t = \beta_o^m + \beta_1^m t, \tag{4.21}$$

where β_1 is the slope, and β_o the intercept. Model parameters can be learned inexpensively via weighted least squares.

A set of learnt motion models at time t is denoted as $[\mathbf{M}_t^2, ..., \mathbf{M}_t^{|\mathbf{M}_t|}]$, where $|.|$ is the number of model scales. Model scales of 2, 3, 4 and 5 are used, as in (Khan et al., 2015). Each model predicts target state $l(\tilde{x}, \tilde{y})$ at T prediction-scales. After applying the $|\mathbf{M}_t|$

models over T prediction-scales, T sets of motion predictions are available at time t. This thesis refers to (Khan et al., 2015) for more details.

In the proposed part-based tracking approach, a separate MTS motion model is learned independently for each part. The $T \times |\mathbf{M}_t|$ motion predictions together with the prediction of the 0-order motion model, i.e. the predicted location at time $t-1$, are available at time t. The top ranking motion prediction is used to initialise the regression search, where the ranking is determined by a Support Vector Machine (SVM) trained to distinguish between foreground and background patches. The tracker does not adopt regression method to determine the best motion prediction for two reasons: first, the motion predictions obtained with a multiple temporal scale motion model can cover a relatively large area meaning that regression needs a big training area which is time consuming and regression may not work well; second, the selected motion prediction is only used to initialise the search, which does not have a high accuracy requirement, and the simple classifier is sufficient.

4.4 Evaluation

This thesis evaluates the proposed method of part-based tracking by regression with incrementally learned cascades (TRIC) on the CVPR2013 benchmark with six experiments: four internal experiments designed to optimise the parameters for TRIC, one investigation experiment exploring the function of different algorithm components, and one external experiment comparing TRIC with the state-of-the-art trackers. More recent benchmarks, VOT2014 and VOT2015, evaluate the trackers by allowing re-initialisation; however, these benchmarks only provide bounding box ground truth for re-initialisation, part-based ground truth is not available to correctly re-initialise the tracker. Evaluation on the VOT2014/15 would therefore be an unfair comparison for the proposed tracker.

The dataset used for evaluation of the TRIC tracker in the external experiment is the full dataset of (Wu et al., 2013). It includes 50 video sequences (including 51 tracking instances) and 29 state-of-the-art trackers. This benchmark provides videos with challenging conditions such as scale variation, occlusion, deformation, fast motion, illumination variation, in-plane rotation, out-of-plane rotation, background clutter and so on. Thus, this tracker can avoid over-fitting to a small subset or one specific attribute. All videos are manually tagged with what the main challenges of the video are. For example, DEF is a subset containing all videos with the attribute ‘deformation - non-rigid object deformation’ (Wu et al., 2013). OCC is a subset including all videos with the attribute ‘occlusion’. SV is the subset in which the videos have ‘scale variation’. The internal studies are performed on 19 videos of the DEF set. The parameters are tuned with the subset and then tested on the full dataset to avoid over-fitting.

The performance of the various trackers is measured using precision (Babenko et al., 2011; J. a. F. Henriques et al., 2012; Wu et al., 2013) and success plots. The precision plot measures the percentage of frames whose estimated location is within the given threshold distance of the ground truth (Wu et al., 2013). The success plot measures the percentage of frames for which the overlap divided by the union of the predicted and ground truth bounding boxes exceeds a given threshold ratio which varies from 0 to 1. This thesis reports on one-pass evaluation (OPE), i.e. the tracker is run throughout the whole video initialised only with the ground truth in the first frame. To rank the performance, as (Wu et al., 2013), this thesis uses the precision obtained for a location error threshold of 20 pixels as the precision score for precision plot. For success plot, the area under curve (AUC) is used as the success score.

Because TRIC predicts the location of the target directly, and TRIC is a part-based tracker, for the comparison with the state-of-the-art trackers or TRICs with different parameters, the tracker retro-fits a bounding box. Specifically, this study retro-fits a bounding box by performing a linear transformation, found by the transformation of the part location, on the bounding box of the first frame.

Initialisation: In the absence of manually labelled part locations, the parts can be automatically initialised given a bounding box enclosing the global object, which is illustrated in Figure 4.7. Specifically, given the bounding box, an eroded bounding box with a margin $\triangle d$ is treated as foreground mask while the area outside is treated as background mask. The learning based digital matting (Zheng & Kambhamettu, 2009) segmentation method is adopted to obtain the contour of whole target with the masks. Then the skeleton of a whole target is found by morphological operation on the target contour and the tracker initialises the part locations with N equidistant points along the skeleton in the first frame. N is fixed to 6 for all experiments.

Please note that the masks for segmentation are hard to define automatically as many of the tracking targets are articulated and the given bounding box ground truth inevitably includes background, which leads to a poor foreground mask. An accurate target contour requires proper segmentation method which is out of the current research scope, and explicit initialisation which needs extra manual interaction of the expert. Thus, given only a bounding box ground truth and the totally automatic initialisation method, it is impossible for now to have perfect initial part locations for the proposed tracker, which deteriorates the tracker's performance accordingly.

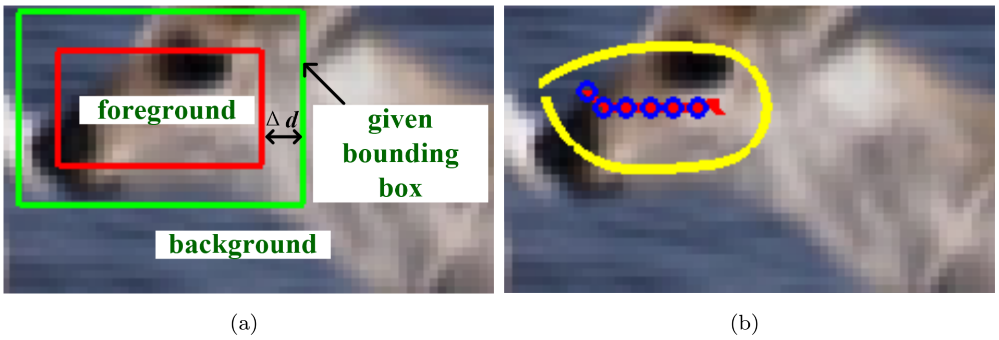


FIGURE 4.7: (a) The given bounding box is green while the eroded box with the margin $\triangle d$ is red. A mask, using the area within the red rectangle as the foreground and the area outside the green rectangle as the background, for segmentation could be obtained. (b) Applying the segmentation method (Zheng & Kambhamettu, 2009) to the mask generates the target contour (yellow). The skeleton found through morphological operation is the red line and the initialised part locations are the blue circles.

Parameters: For calculation efficiency, the tracker re-scales images using the initial frame as the reference to make every tracked whole target's scale approximately equal to 50×50 pixels. Specifically, the ratio is determined by $(50 \times 2)/(w+h)$, where w and h are the bounding box groundtruth of the target in the first frame. The tracker adopts CNN features extracted from location S_i with sample size $p_s \times p_s$ for local part or $g_s \times g_s$ for global part to represent image information of sample patch. Each sample's feature is represented by $\Phi(I, s_{ij}, p_s, b)$ or $\Phi(I, s_{ij}, g_s, b)$ in which b is the bias.

As the object is expected to move, the background is not correlated with the target movement. Image information acquired from the background will only give approximately correct predictions in the recent future, while image information derived from the object itself is expected to remain accurate over time. Thus, image patches should ideally always capture part of the foreground, which means the size of image patch should be approximately equal to the implicit target part's size. The image patch size p_s is therefore defined by: $p_s = (w_s + h_s)/(N-1)$, in which w_s and h_s are scaled w and h. This definition ensures that the patch size is directly related to the density of target parts. The sample patch size for global part is g_s defined by: $g_s = (w_s + h_s)/2$.

$N = 6$ parts are used, representing a balance between shape expressivity and computational complexity. All experiments use fixed $\delta_v = 0.01$, determined empirically, and $K = 200$. The sampling radius is set to 30 pixels and the number of cascade levels is 4. In my case, p_s is 24 pixels and g_s is 48 pixels. $\lambda = 0.001$ (see Figure 4.8(d)) is set in the Eq. 4.13 in Section 4.3.3.

Shape Correction: It is noticed that occasionally prediction of some of the parts are completely off target while others are still on target, which affects the performance for three reasons. Firstly, since the tracker retrofits a bounding box to the final prediction of parts, the bounding box would be much larger than it should be. Secondly, the predictions provided by the motion model are based on located part position in each frame, so that outliers would affect the accuracy of a motion model. Finally, the update step also uses information based on the predicted part location, making the regressor learn on faulty data when there are outliers.

To reduce the impact of outlier predictions, they are detected and removed using a shape correction process. This is built on the assumption that the arrangement of parts as initialised in the first frame does not change much during tracking, while allowing for changes in scale and rotation. A measure which is able to check these constraints is the relative length between parts defined by Eq. 4.23.

These relative lengths are calculated for the ground truth parts locations in the first frame and compared against the relative lengths calculated in the current frame. The deviation between the ground truth relative lengths of a part and the current frame's is then calculated as follows:

$$DEV_i = \frac{\sum_{j=1..N} \sum_{k=1..N} |Q_r^c(i,j,k) - Q_r^s(i,j,k)|}{N^2}, \tag{4.22}$$

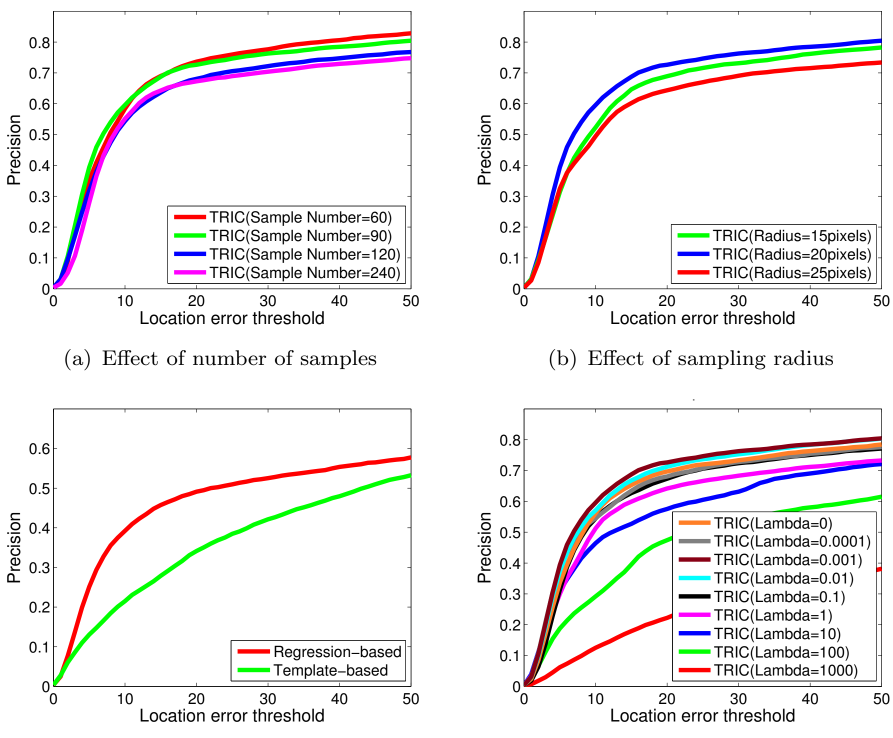
$$Q_r(i,j,k) = \frac{max(||l_i^* - l_j^*||, ||l_i^* - l_k^*||)}{min(||l_i^* - l_j^*||, ||l_i^* - l_k^*||}, \tag{4.23}$$

where DEV_i is the deviation of part i, N is the total number of local parts, $Q_r^c(i,j,k)$ is the relative length for part i among part i, part j and part k in the current frame and $Q_r^s(i,j,k)$ is the same but then for the initial frame.

The outlier parts can now easily be filtered from the set of located parts. This is done by calculating the median of the deviation of all parts and removing those parts which have a deviation more than twice the median deviation. The set of non removed parts can then be matched with their corresponding parts in the initial set and an affine transform is calculated between these corresponding parts. Finally the affine transform can be used to calculate the most likely true position of the removed parts.

4.4.1 Parameter Optimisation

Internal tests are performed to optimise the internal parameters of the cascaded regressors. The first internal study examines the influence of the sampling density used in the training and test stages. The second investigates the effect of the sampling radius. The third internal study compares the performance of direct-displacement prediction by regression with a template based tracker using the same features and motion model, to investigate the cascaded regression's advantage over the template based method. The fourth internal study evaluates the effect of lambda in the ridge regression. The first, second and fourth experiments are performed with the full TRIC tracker. For the third experiment, this thesis compares the TRIC tracker, not using the multi-temporal scale motion model and updating, with the template-based tracker which uses the same features and motion model (a 0-order motion model). In all internal experiments, each frame is rescaled to make the tracked target be approximately 30×30 using the same method explained in 'Initialisation' part.



(a) Effect of number of samples

(b) Effect of sampling radius

(c) Regression vs template

(d) Effect of regularisation parameter

FIGURE 4.8: (a) The effect of number of samples on the tracking accuracy of TRIC (sampling radius = 20 pixels). (b) Effect of sampling radius of TRIC. (c) Comparison between template-based tracker and regression-based tracker. (d) Effect of regularisation parameter in ridge regression. The metric in (a), (b), (c) and (d) is the precision plot.

First, this thesis investigates the effect of sampling density by setting the number of samples to 60, 90, 120, and 240 with the sampling radius fixed to 20 pixels. The results are shown in Figure 4.8(a). The general trend clearly shows that there is not a large difference in the performance of TRIC with sample number either 60 or 90. 90 is chosen as the sample number because it is more accurate within a smaller error threshold.

Second, the effect of the sampling radius is determined by varying its value in the range of 15, 20 and 25 pixels (see Figure 4.8(b)). The results clearly show that a large sampling area is more likely to include poor samples and small one is not robust enough. 20 pixels is selected to balance the accuracy and robustness of the tracker.

Third, for the comparison between the regression-based tracking method and the template-based method, this thesis allows the latter to perform a full search in the same area from which the former samples its test locations, i.e. a circular area of radius 20 pixels. Figure 4.8(c) clearly shows the gain achieved by adopting a regression-based approach.

Fourth, this thesis exploits the effect of λ in the Ridge Regression by setting the value of λ to 0 and 10^k, $k = -4 : 3$. Following examination of the results shown in Figure 4.8(d), $\lambda = 0.001$ is chosen as the optimal value.

4.4.2 Investigation of Algorithm Components

This thesis also investigates the impact of different algorithm components in the TRIC tracker. TRIC-S refers to the TRIC tracker without the implicit shape model. TRIC-I is the TRIC tracker without incremental learning, i.e. learning only from the first frame of the video while TRIC-M denotes the TRIC tracker without the multiple temporal scale motion model (i.e. assuming 0-order motion). As explained in Section 4.4, this thesis reports on one-pass evaluation (OPE), i.e. the tracker is run throughout the whole video initialised only with the ground truth in the first frame. To rank the performance, as (Wu et al., 2013), this thesis uses the precision obtained for a location error threshold of 20 pixels as the precision score for precision plot. For success plot, the area under curve (AUC) is used as the success score. The precision score is shown in the legend of the precision plot, while the success score is shown in the legend of the success plots. Considering the time consumption, the calculation of image features is most time consuming, as the calculation of HOG or CNN features of image samples is expensive and the integration of the implicit shape model makes the number of image samples increase by three times after every cascade of regression. The TRIC-S is more efficient than the TRIC as the number of image samples in TRIC-S is 1/3 of the number in TRIC because of the lack of the shape model. Similarly, TRIC-I runs faster than TRIC as the number of image samples nearly reduces by 50% without incremental learning. As the motion model is a linear model, the time consumed is limited. TRIC-M has nearly the same efficiency as the TRIC.

Table 4.2 shows the precision score, as well as the precision plot ranking obtained, for TRIC, TRIC-S, TRIC-I, and TRIC-M. This clearly shows the value of the implicit shape model. For DEF, without the implicit shape model, performance decreases by 58%. This shows that the implicit shape model is a key algorithm component in TRIC-track. For OCC, performance decreases by 22.9% without incremental learning, indicating that

Table 4.2: Performance of TRIC, without implicit shape model (TRIC-S), without incremental learning (TRIC-I), and without motion model (TRIC-M), on the DEF, OCC, SV and full dataset of the CVPR 2013 benchmark data. # is the rank obtained against the 29 trackers in the benchmark (Wu et al., 2013), while Score is the precision score.

Trackers	TRIC		TRIC-S		TRIC-I		TRIC-M	
Dataset	#	Score	#	Score	#	Score	#	Score
ALL	1	0.796	25	0.357	4	0.626	2	0.756
DEF	1	0.817	25	0.343	2	0.676	2	0.721
OCC	1	0.786	26	0.332	3	0.606	2	0.726
SV	1	0.747	25	0.347	7	0.553	2	0.738

the incremental learning is crucial for scenarios with occlusions, and 7.6% without the motion model, as the motion model employed by TRIC predicts multiple frames ahead, thus overcoming brief occlusions.

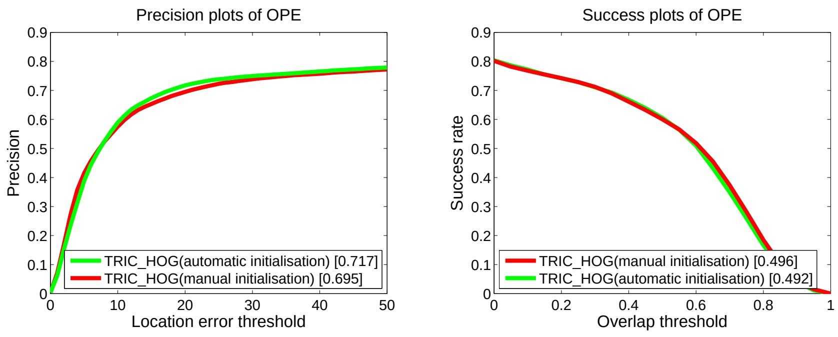


Figure 4.9: Precision plots and success plots of TRIC using HOG features with automatic initialisation and TRIC using HOG features with manual initialisation. OPE means one-pass evaluation. The values shown in the legend of precision plots are the precision scores, while the values shown in the legend of success plots are the success scores.

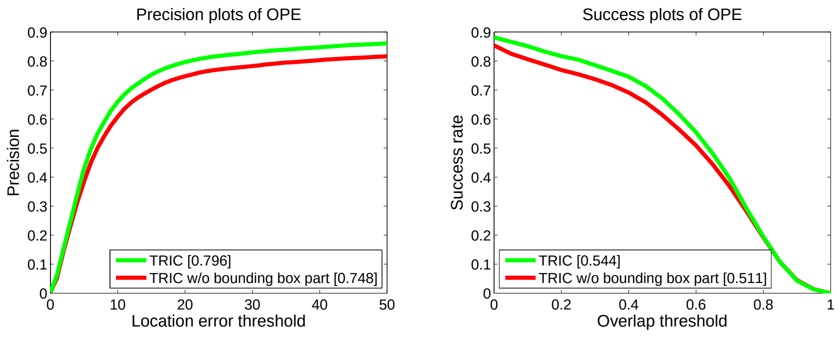


Figure 4.10: Precision plots and success plots of TRIC and TRIC without the global part.

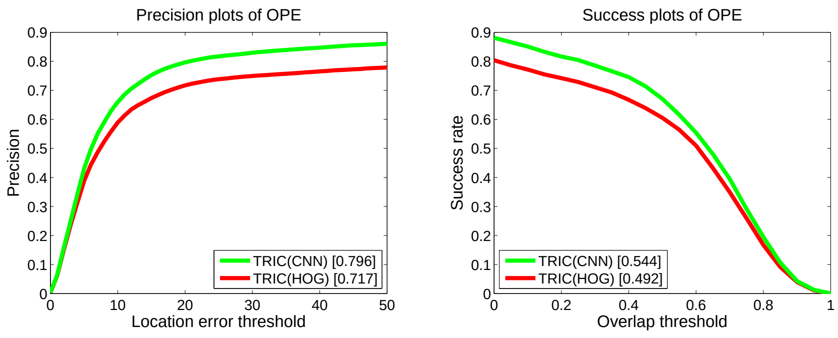


FIGURE 4.11: Comparison of CNN vs HOG features.

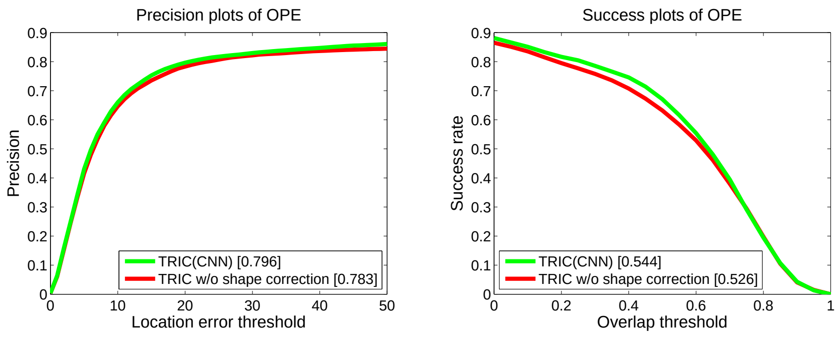


FIGURE 4.12: Precision plots and success plots of TRIC and TRIC without shape correction.

In addition to the experiments on the main algorithm components, this thesis also performs experiments to compare TRIC with manual initialisation and TRIC with automatic initialisation, see Figure 4.9, which shows that similar results with automatic initialisation and with manual initialisation are obtained. The segmentation method provides relatively reliable initialisation of parts' locations. The global part is added to extend the proposed implicit shape model to be implicit shape model with the global part. A corresponding experiment is executed to verify the advantage of bringing global information, see Figure 4.10. Experimental results show that adding the global part increases the precision score by 6.4% and the success score by 6.5%. When the image information from local parts is not able to give a confident prediction, the image information from the global part can make a confident prediction of target location, because the spatial relationship between global part and target part is less flexible. Comparison between HOG features and CNN features adopted in TRIC-track is also conducted, see Figure 4.11, which clearly shows that with the more discriminative power of CNN features, the precision score increases by 11% and the success score increases by 10.6%. A shape correction model is also added to remove the outliers of predicted parts locations, see Figure 4.12, which shows that the shape correction model improves the performance.

4.4.3 Comparison to the State of the Art

To evaluate the TRIC tracker's performance, the TRIC tracker is tested on the whole dataset and compared with the results of all the 29 state-of-the-art trackers included in the benchmark (Wu et al., 2013).

Results are shown in Figure 4.13 (precision plot) and Figure 4.14 (success plot). For clarity, only the top ten trackers are displayed. The values of the precision score and the success score for the corresponding state-of-the-art methods are included in the legends of Figure 4.13 and Figure 4.14.

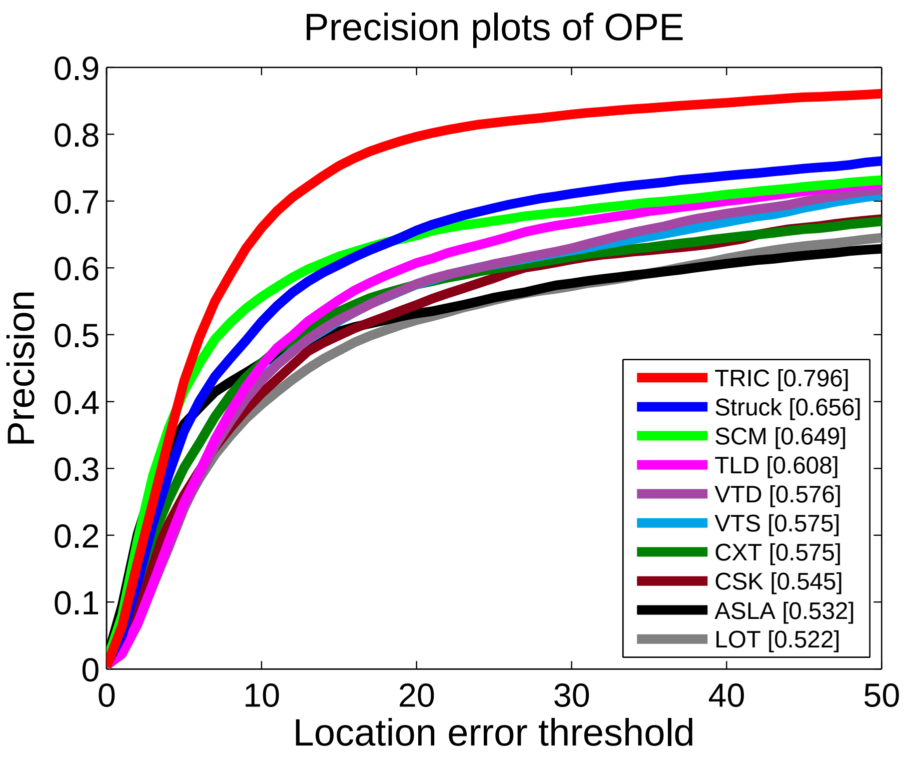


FIGURE 4.13: Precision plot of OPE for all 50 sequences.

The precision plot of TRIC is approximately equal to the third-highest ranked tracker (SCM (Zhong, 2012)) for a threshold up to 5 pixels, and for higher thresholds it significantly outperforms all other trackers. Struck (Hare et al., 2011) comes second. Similarly, TRIC clearly outperforms all other trackers for overlap thresholds up to 0.61, and is approximately equal to SCM after that. One interpretation of these results is that TRIC is a robust tracker, never making very large errors, but not particularly precise. I present another interpretation, and this is that the retro-fitting of a bounding box onto the tracked parts of an articulated deformable object is unsuitable for direct comparison with bounding box trackers, as deformations of extremities (e.g. someone extending

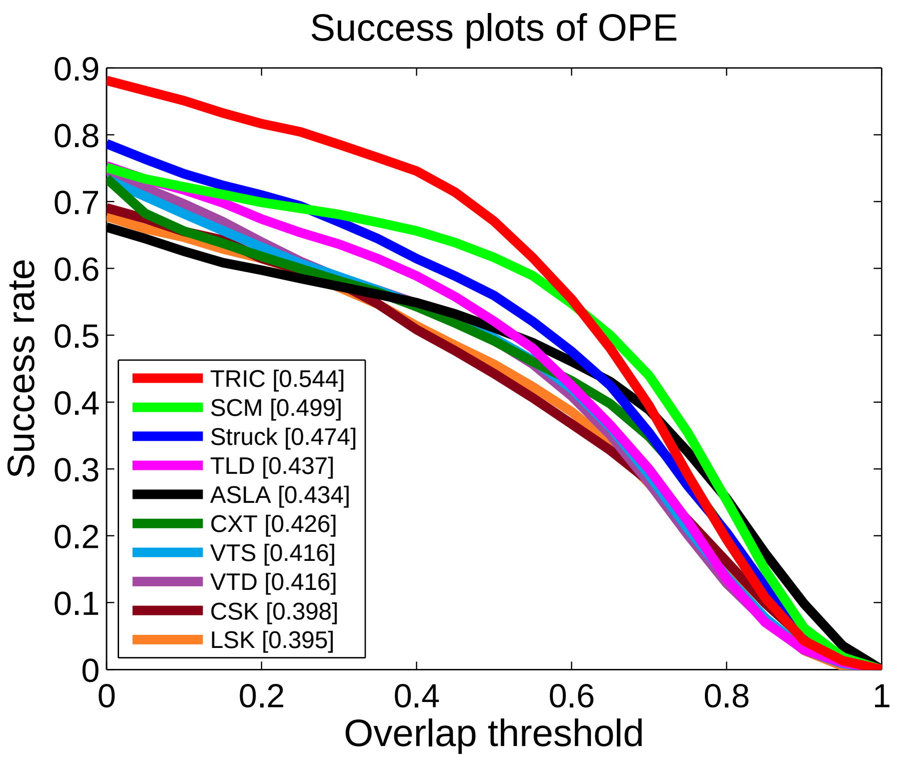


FIGURE 4.14: Success plot of OPE for all 50 sequences.

their arm) causes the mean location of the parts to move away from where the bounding box ground truth would be.

4.4.4 Conclusion

This thesis proposes a part-based tracker employing direct displacement prediction rather than the traditional local matching of an appearance model. The method employs cascaded regression to directly predict parts' locations from local image information described by deep learned features, learning the inference models on-the-fly. Spatial relationships between parts and spatial relationships between local parts and the global part are captured implicitly by a set of regressors. This thesis integrates a multiple temporal scale motion model to initialise the cascaded regression search close to the target and to cope with occlusions. Automatic initialisation and shape correction are added to complete the proposed tracker. Experimental results clearly demonstrate the value of the method's component parts, and comparison with the state-of-the-art techniques in the CVPR 2013 Visual Tracker Benchmark shows that TRIC ranks first on the full dataset.

Chapter 5

Tracking by Locally Continuous Regression with Incrementally Learned Cascades

Chapter 4 has shown that the framework of SDM (Xiong & De la Torre, 2013) can be successfully adapted to the problem of online tracking of generic objects and that cascaded regression (SDM) improves the accuracy of direct displacement prediction compared to ordinary linear regression. It also showed how the incremental updating of cascaded regression proposed by (Asthana et al., 2014), which uses newly tracked faces to incrementally update the offline trained regressor for facial point localisation, can be adapted to the incremental update of the regressor in general object tracking by adding new training data.

SDM has avoided calculation of the Hessian matrix or Jacobian matrix, when solving nonlinear optimisation problems in the context of computer vision, by learning a sequence of descent directions. However, there are three major drawbacks of sampling-based regression (i.e. SDM) within the TRIC-track framework, which are as follows:

- The samples used for training SDM are obtained through random sampling of starting points for cascaded prediction from a circular area centred at the ground truth target location. The sampling process is computationally expensive as it needs to be conducted at every cascade level for each target point.

- Similarly to training, the samples used for updating SDM are obtained through random sampling. The sampling process needs to be conducted at every cascade level for each target point and thus is time consuming.

- Each regressor is learned by minimising the least squares error on a limited set of training image patches and their corresponding displacements. The regressor is updated with a limited set of samples as well. Not all available image information and displacements within a sample space can be utilised to train and update a regressor.

An alternative to sampling-based linear regression is Continuous Regression (Sánchez-Lozano et al., 2016), which can cope with the above three problems. With Continuous Regression, the shape displacement is regarded as a continuous variable, and the feature space is approximated by its first-order Taylor expansion. Only the feature at the ground truth target location needs to be sampled, and all displacements are handled in a very efficient manner.

In this chapter, Continuous Regression is introduced to the task of model-free tracking. However, it is then observed that the Taylor expansion in itself is not a good feature approximation when dealing with scenarios with such a high variance in target appearance. To alleviate this problem, this chapter proposes Locally Continuous Regression, which unifies sampling-based regression with continuous regression in an efficient manner. The proposed strategy is then applied to training and incremental updating in the main framework of TRIC-track, which shows a six-fold speed improvement without sacrificing performance of the tracker.

5.1 Continuous Regression

5.1.1 Motivation of Continuous Regression

In both facial landmark tracking and general object tracking problems, SDM is used in the same way. Specifically, both problems require one to learn a mapping matrix R between image features, extracted at the sample locations, and the displacements δs, from ground truth target location s^* to the sample locations $(s^* + \delta s)$. The samples can be obtained through a random sampling around the ground truth target location. The mapping can be described as: $\delta s = R\Phi(I, s^* + \delta s)$. The difference between these two problems is that, the learning of R is performed only in the first frame of a video sequence in general object tracking, while the learning is performed offline over thousands of images in facial landmark tracking. Moreover, the kind of targets can be various in visual tracking, while the target in facial landmark tracking is the face only. Thus, variations in appearance and shape of target in visual tracking tend to be more complex. This difference limits direct application of Continuous Regression to general object tracking, which will be explained in Section 5.1.4.

The least squares problem in a general form in visual tracking can be formulated as[1]:

$$\arg\min_{R} \sum_{k=1}^{K} ||\delta s_k - R\Phi(I, s^* + \delta s_k)||_2^2, \tag{5.1}$$

where s^* is the ground truth target location, i.e. the ground truth shape. K is the number of perturbations, i.e. the number of patches sampled for the top-level cascade.

As shown above, sampling-based regression has the following three limitations (Sánchez-Lozano et al., 2016):

- Training an SDM model is computationally expensive as it requires the data to be sampled per cascade level in training regressor for each target point (part), with both the memory and the time needed for training increasing dramatically with the number of samples.
- Similarly to training, updating an SDM model is time consuming because of the need of sampling per cascade level for each target point, which limits the capacity of SDM and its extensions for learning on-line and in real-time.
- SDM uses a limited set of samples to train and update a regressor instead of using all available image information and the corresponding displacements.

5.1.2 Overview of Continuous Regression

To overcome these limitations, Sanchez-Lozano et al. (Sánchez-Lozano et al., 2016) proposed Continuous Regression, and incorporated it into facial landmark tracking. The proposed Continuous Regression is a solution to the least squares problem, generating the first face tracking method with real-time incremental learning capabilities.

With Continuous Regression, the shape displacement is regarded as a continuous variable. The feature space is approximated by its first-order Taylor expansion, so that the feature space becomes linear with respect to shape displacement, yielding a closed-form solution for the continuous domain of displacements.

With this approximation, Continuous Regression only needs the data to be sampled at the ground-truth locations, which means that no sampling at perturbations of the ground truth location is required. Thus, one can sample and store the ground-truth data only once, and then train each cascade level, or even a new model under a different configuration extremely quickly.

[1]The parameter p_s (size of sampled image patch) is removed from Φ for the sake of simplicity.

Sanchez-lozano et al. have incorporated Continuous Regression into the cascaded regression framework (Sánchez-Lozano et al., 2016) and demonstrated that Cascaded Continuous Regression (CCR) has notable computational advantage over the standard SDM, without sacrificing performance. A graphical overview of the Continuous Regression is shown in Figure 5.1.

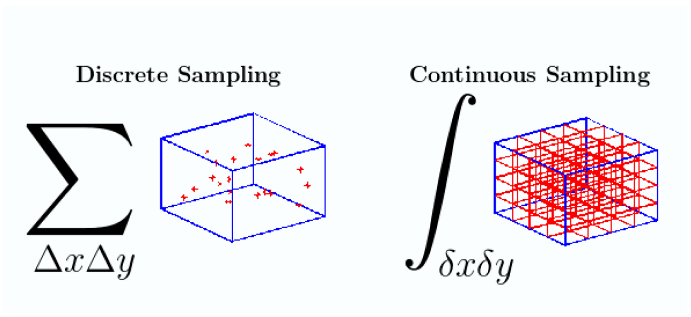


Figure 5.1: Difference between sampling-based regression and continuous regression. The continuous regression accounts for all the samples within a neighbourhood, whereas sampling-based needs to sample the data from a given distribution. The figure is from (Sánchez-Lozano et al., 2016).

5.1.3 Formulation and Solution of Continuous Regression

The complete formulation and solution of Continuous Regression is briefly presented as follows. Regarding the shape displacement as a continuous variable, the least squares problem is extended to the continuous domain of the shape displacement. The extension of the least squares problem is then defined as (Sánchez-Lozano et al., 2016):

$$\arg\min_{R} \int_{\delta s} ||\delta s - R\Phi(I, s^* + \delta s)||_2^2 p(\delta s) d\delta s, \tag{5.2}$$

where $p(\delta s)$ describes the pdf of the sampling distribution, which is parameterised by its mean μ and covariance Σ. Instead of using the discrete samples of a sample space only, $p(\delta s)$ models a whole sample space around a target point. The relation between discrete samples, used in the sampling-based regression, and the sample distribution, used in Continuous Regression, is shown in Figure 5.2.

To allow us to replace a set of discrete samples with a continuous sample distribution, Sanchez-Lozano et al. propose to approximate the feature space by its first-order Taylor expansion:

$$\Phi(I, s^* + \delta s) \approx \Phi(I, s^*) + J^* \delta s, \tag{5.3}$$

where $J^* = \frac{\partial \Phi(I,s)}{\partial s}|_{(s=s^*)}$, evaluated at $s = s^*$, is the Jacobian of the feature representation of image I, with respect to shape coordinates (target location) s, at s^*. Utilising

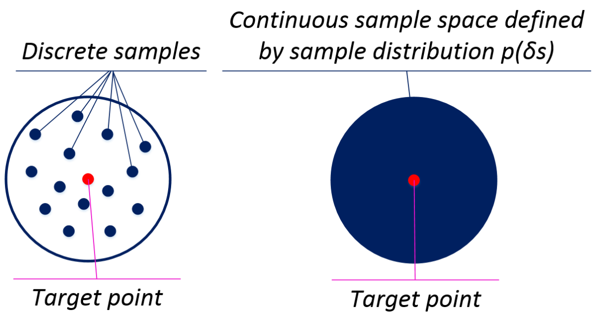

Discrete samples
Continuous sample space defined by sample distribution p(δs)
Target point
Target point

FIGURE 5.2: The relation between discrete samples and the continuous sample space.

Eq. 5.3, the least squares problem can be described as:

$$\arg\min_R \int_{\delta s} p(\delta s)||\delta s - R\Phi(I, s^* + \delta s)||_2^2 d\delta s \approx \tag{5.4}$$

$$\arg\min_R \int_{\delta s} p(\delta s)||\delta s - R\big(\Phi(I, s^*) + J^*\delta s\big)||_2^2 d\delta s = \tag{5.5}$$

$$\begin{aligned}\arg\min_R \int_{\delta s} p(\delta s)[\delta s^T\delta s - 2\delta s^T R(X^* + J^*\delta s) +\\ (X^* + J^*\delta s)^T R^T R(X^* + J^*\delta s)]d\delta s,\end{aligned} \tag{5.6}$$

where $X^* = \Phi(I, s^*)$ is the feature vector extracted at the ground-truth point s^*. The solution for R is obtained by minimising Eq. 5.6 and R is then calculated by:

$$R = \Big(\mu X^{*T} + (\Sigma + \mu\mu^T)J^{*T}\Big)\Big(X^*X^{*T} + 2X^*\mu^T J^{*T} + J^*(\Sigma + \mu\mu^T)J^{*T}\Big)^{-1}. \tag{5.7}$$

The reader is referred to (Sánchez-Lozano et al., 2016) for more details of the derivation of R. Note that the solution of R is not determined by the actual sampling pdf, but rather by its mean and covariance.

Although there is no analytical form solution for the Jacobian of image features, their gradients can be approximated. In particular, the derivatives of the image features with respect to the x coordinate are obtained by:

$$\nabla\Phi_x = \frac{\partial\Phi(I, x)}{\partial x} \approx \frac{\Phi(I, x + \Delta x) - \Phi(I, x - \Delta x)}{2\Delta x}. \tag{5.8}$$

Similarly, the empirical derivative of the image features with respect to the y coordinate can be obtained. The Δx is set to the minimum displacement, 1 pixel, in (Sánchez-Lozano et al., 2016), and $J^* = [\nabla\Phi_x, \nabla\Phi_y]$.

To help calculate Eq. 5.7, a matrix form is adopted in (Sánchez-Lozano et al., 2016). The shorthand notations are: $M = [\mu, \Sigma + \mu\mu^T]$, $B = \begin{pmatrix} 1 & \mu^T \\ \mu & \Sigma + \mu\mu^T \end{pmatrix}$, $D^* = [X^*, J^*]$.

Eq. 5.7 then can be defined by:

$$R = MD^{*T}(D^{*}BD^{*T})^{-1}. \tag{5.9}$$

In addition to the theoretical contribution, Continuous Regression formulation has important computational benefits compared to sampling-based methods (such as SDM). Eq. 5.7 clearly shows that the calculation of R only needs the ground truth feature X^{*}, its Jacobian J^{*} and the sampling statistics. That is to say, within the cascaded regression framework, an image frame only needs to be sampled once in the first cascade level. For the following cascaded levels, there is no need to repeat the sampling process, and the only need is to change the sampling statistics with respect to the ground truth target location. Thus, it is only needed to sample the image once for training regressors under different configurations in all cascade levels, which is time efficient compared with the sampling-based SDM.

5.1.4 Limitation of Continuous Regression

The main limitation of Continuous Regression is that the smoothness of features must be satisfied when approximating image features using the first-order Taylor expansion. The smoothness of features means that the image features must have similar appearance. In the face alignment case, in which local information on a face is used to infer a facial point location, the smoothness of features has been shown to be sufficiently satisfied. However, it is not the case in visual tracking, and especially within the framework of TRIC-track, as explained as follows:

- First, most objects in visual tracking have more complex appearance. The variation in target appearance between two adjacent frames in visual tracking is usually higher than that in facial landmark tracking. To increase the robustness of the appearance model (regressor) in TRIC-track, the samples used in training are not obtained from a local area close to the target point. The samples should cover a relatively large area, compared with the sampling area in facial point localisation, centred at the target location. These samples are at most 30 pixels away from the target in TRIC-track. For example, to estimate the centre location of a human head, the samples from the whole head are included in the search region, and must thus be approximated by the Taylor expansion. In contrast, for face alignment only the local samples around the nose are used to detect a facial point on the nose.

- Second, in the implicit shape model proposed in TRIC-track, a target part location is predicted not only by the image features around the current part, but also the image features around its two neighbouring parts. In this case, the ground truth location is the current part location, and the image features can come from its neighbouring part and are relatively far away from the ground truth location. In this case, Taylor expansion is not able to correctly approximate the image features, around the neighbouring parts, using just the image feature and its Jacobian at the current part location.

Despite the impressive results shown in face tracking (Sánchez-Lozano et al., 2016), the high variance of model-free tracking makes Continuous Regression much less accurate than sampling-based regression. Continuous Regression is limited by the Taylor expansion. This problem is explored by an experiment testing Continuous Regression on visual tracking scenarios.

Specifically, in a video frame, given a ground truth target location, shown as the cyan dot in Figure 5.3, two kinds of regression methods, Continuous Regression (CR) and sampling-based regression (SR), were used to train a regressor to predict the target location, respectively. The same sample space with zero mean and a certain covariance Σ is used for both methods during training and testing. More precisely, sample statistics are used to train a regressor by Continuous Regression, while the samples acquired from the same sample statistics are used to train a regressor by sampling-based regression. The test samples from the same sample statistics are then evaluated by the above two regressors. The image feature used is the HOG feature. The Euclidean distances between the ground truth target location and test samples, predictions from CR, predictions from SR are measured respectively. The lower Euclidean distances between the ground truth target location and the predictions, the higher accuracy of the predictions. This experiment is performed on every frame of five videos, which are $Shaking$, $David3$, $Woman$, $Skating1$ and $MotorRolling$. They are of different levels of difficulty for tracking and selected from the CVPR2013 benchmark (Wu et al., 2013) as the test videos in this evaluation.

In every frame, both of these methods are applied to the sample space with a range of covariances Σ, from 10 to 60, to explore how accurate the predictions from Continuous Regression can be, i.e. how accurately Taylor expansion can approximate image features, in visual tracking scenarios. Figure 5.3 shows the experiment with a lower covariance Σ_1 and a higher covariance Σ_2. The test sample space in Figure 5.3(a) and Figure 5.3(c) are obtained by random sampling from a bivariate normal distribution with zero mean and a specified covariance matrix Σ_1. The test sample space in Figure 5.3(b) and Figure 5.3(d) are obtained in the same way with zero mean and covariance Σ_2. $\Sigma_1 = \begin{pmatrix} 10 & 0 \\ 0 & 10 \end{pmatrix}$ and $\Sigma_2 = \begin{pmatrix} 30 & 0 \\ 0 & 30 \end{pmatrix}$.

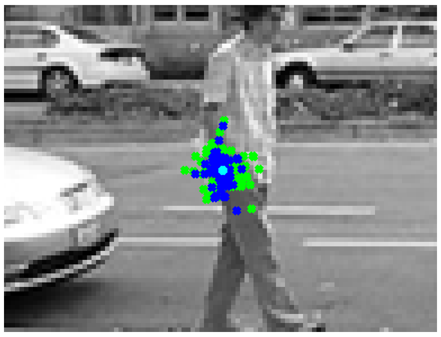
Figure 5.3(a) shows the visualised result of the regressor R_a trained by Continuous Regression with lower variance Σ_1. Figure 5.3(b) shows the visualised result of the regressor R_b trained by Continuous Regression with higher variance Σ_2. The test locations are shown as green crosses. The target locations estimated by the regressor are shown as blue crosses. The number of test locations is 100.

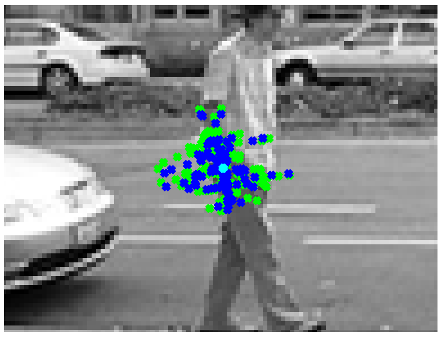
Figure 5.3(c) shows the visualised result of the regressor R_c trained by sampling-based regression with lower variance Σ_1. Figure 5.3(d) shows the visualised result of the regressor R_d trained by sampling-based regression with higher variance Σ_2. The test locations are shown as green crosses. The target locations estimated by the regressor are shown as magenta crosses.

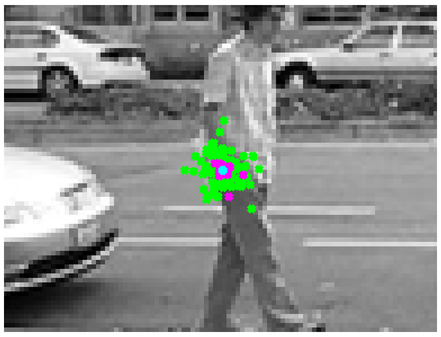
Figure 5.3(e) and 5.3(f) show the proportion of samples whose Euclidean distances to the target location is lower than a certain value. The Euclidean distance from a sample location to the target location is normalised by the maximum distance from the initial test location to the target location.

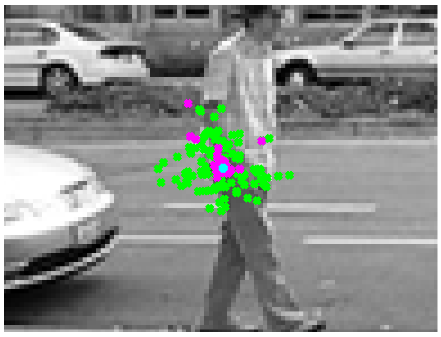
The distributions of locations predicted by Continuous Regression (blue crosses) in Figure 5.3(a) and Figure 5.3(b) are less dense than those of sampling-based regression (magenta crosses) in Figure 5.3(c) and Figure 5.3(d). More importantly, the difference in distribution density of estimated target location between Figure 5.3(b) and 5.3(d) is much bigger than that between Figure 5.3(a) and 5.3(c), which is quantitatively illustrated in Figure 5.3(e) and 5.3(f). The green curve represents the sample locations (test locations), and the sample locations are the same for Continuous Regression and the sampling-based regression. The image features extracted at sample locations are used as input of a regressor. The regressor then predicts one target location based on each of these sample locations.

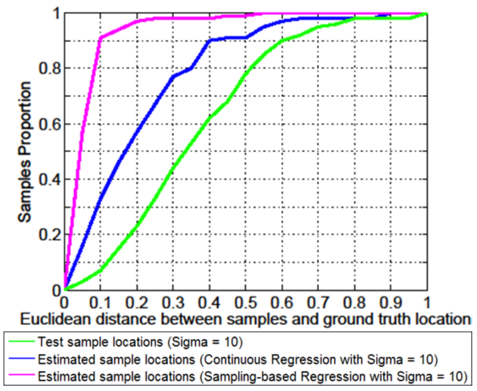
Figure 5.4 shows the sample proportion vs Euclidean distance result of two regressors (CR and SR) with the covariance which ranges from 10 to 60. It shows the result of all frames (in total 252) of video *David*3. The results of other videos are very similar to the result of this video. With the increase of the covariance, the number of test samples, which is also the number of training samples used in the sampling-based regression method, increases accordingly. Specifically, it is empirically calculated by Σ^2. Figure 5.4 shows that, with the increase of the covariance, the proportion of SR predictions whose scaled Euclidean distance to the ground truth target location is lower than 0.1 decreases


(a) Continuous Regression with lower variance Σ_1

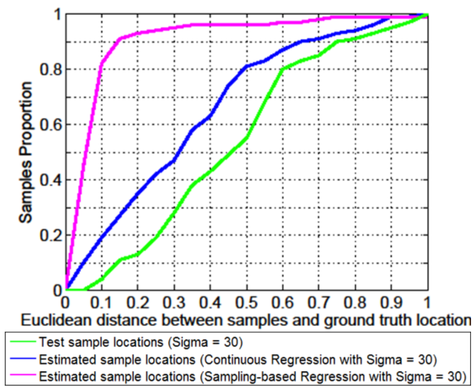

(b) Continuous Regression with higher variance Σ_2


(c) Sampling-based regression with lower variance Σ_1


(d) Sampling-based regression with higher variance Σ_2

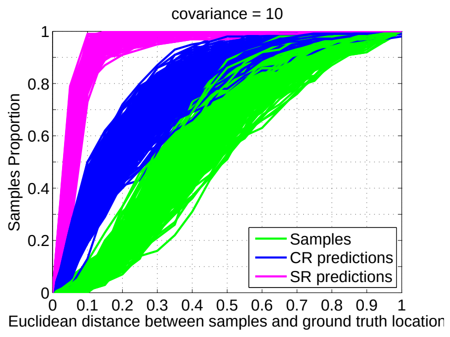


(e) Result of the lower variance Σ_1

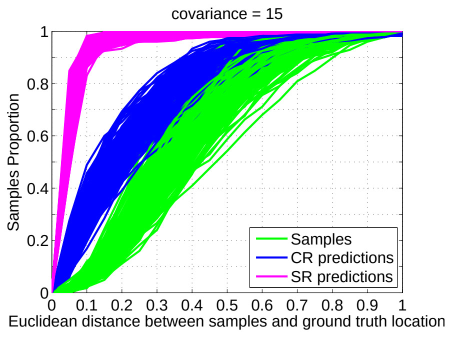


(f) Result of the higher variance Σ_2

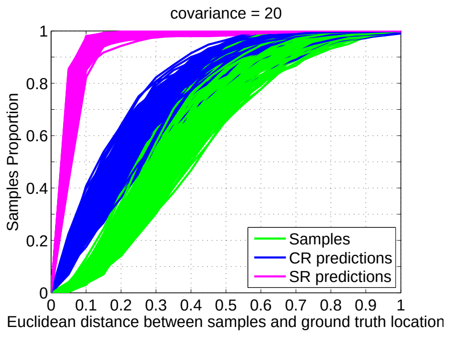
FIGURE 5.3: (a), (c) and (e) show the result of the lower variance Σ_1. (b), (d) and (f) show the result of the higher variance Σ_2. (a) and (b) show the result of Continuous Regression. (c) and (d) show the result of the sampling-based regression. The cyan dot is the ground truth target location. Green crosses are the initial test locations, while blue and magenta crosses are the predicted locations by the regressor. The x-axis of (e) and (f) is the Euclidean distance between the sample location and the ground truth target location. The x-coordinate of (e) and (f) are normalised by the maximum distance between the target and the test sample location, for clear comparison. The y-axis of (e) and (f) is the proportion of the samples whose Euclidean distances to the target location is lower than a certain value.


covariance = 10
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

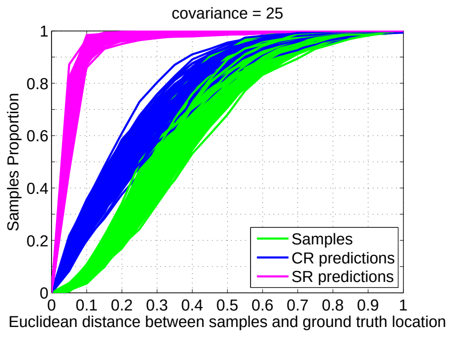
(a)


covariance = 15
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

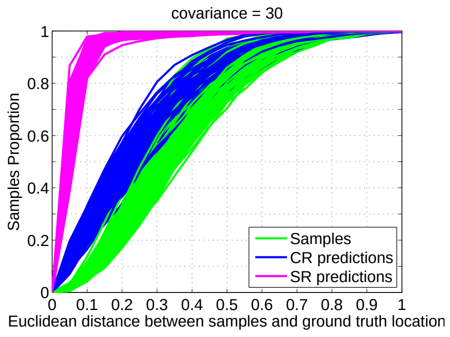
(b)


covariance = 20
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

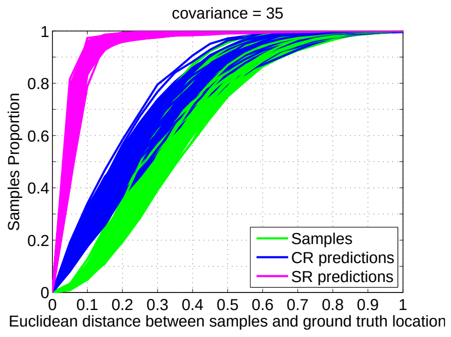
(c)


covariance = 25
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

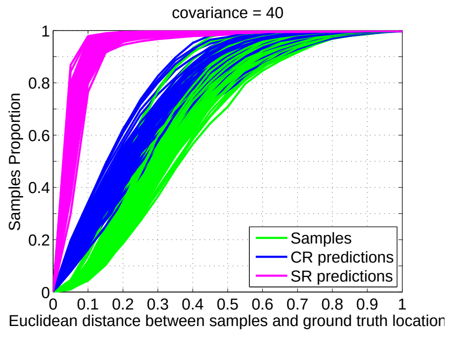
(d)


covariance = 30
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

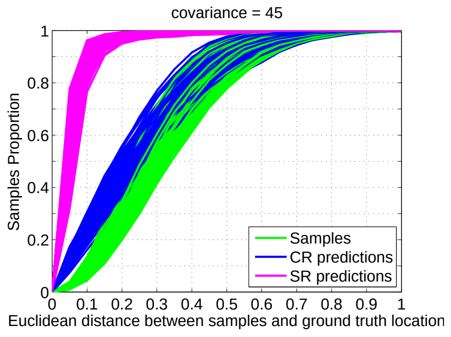
(e)


covariance = 35
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

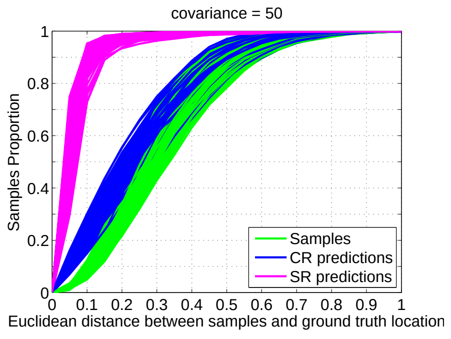
(f)


covariance = 40
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

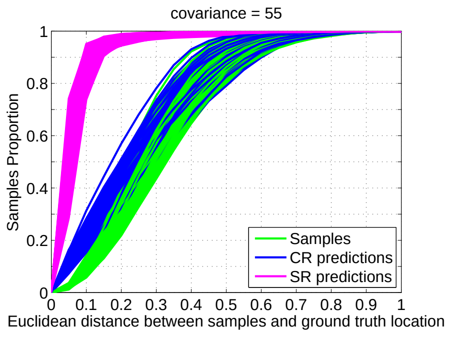
(g)


covariance = 45
Samples Proportion
Euclidean distance between samples and ground truth location
Samples
CR predictions
SR predictions

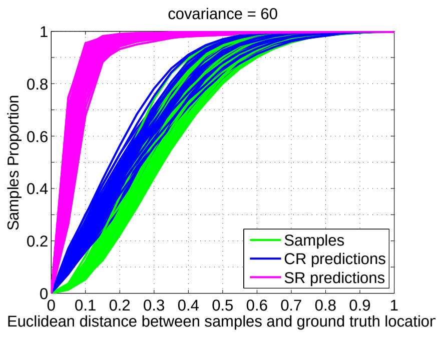
(h)



(i)



(j)



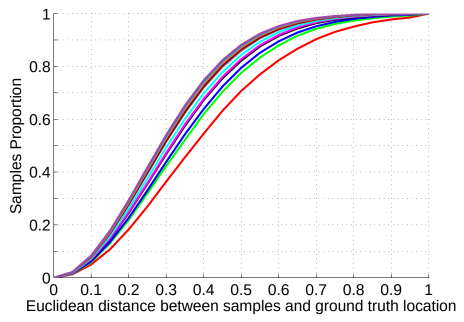
(k)

FIGURE 5.4: Results of Euclidean distances from the ground truth target location to the test samples, CR predictions and SR predictions in every frame in video *David*3. X axis shows the scaled Euclidean distance. Y axis shows the proportion of samples whose Euclidean distance to the ground truth target location is lower than a certain value. Every curve is the result of a frame in video *David*3.

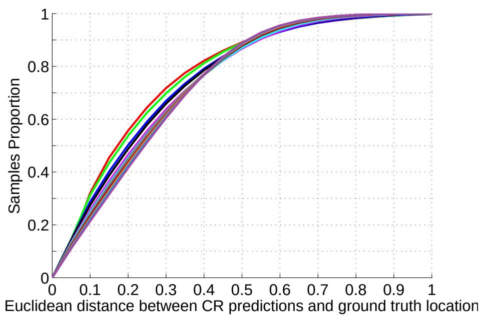
slightly, shown as magenta curves in Figure 5.4. The proportion of predictions from CR, whose scaled Euclidean distance to the ground truth target location is lower than 0.2 or 0.3, significantly decreases with the increase of the covariance of the sample space, shown as blue curves in Figure 5.4. The difference between the initial test locations and the predictions from CR becomes increasingly small with the increase of the variance of the sample space. It clearly illustrates that Continuous Regression is not able to accurately predict target location, i.e. the Taylor expansion is not able to accurately approximate image features of a sample space with a high variance.

Figure 5.5 shows average results of Figure 5.4 over all frames of video *David*3. Especially, Figure 5.5(b) shows that the performance of CR becomes worse as the covariance increases. Mean and variation of curves, similar to the curves in Figure 5.4, of five videos are shown in Figure 5.6.

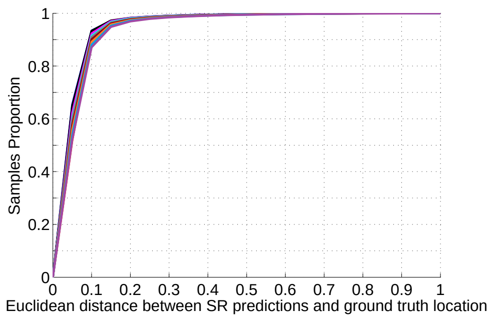
It can be seen from the experimental results that the regressor trained by Continuous Regression with a higher variance tends to give much less accurate predictions, compared with its corresponding sampling-based regression. It means that the first order Taylor



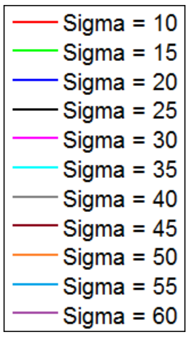
(a) Test samples



(b) CR predictions

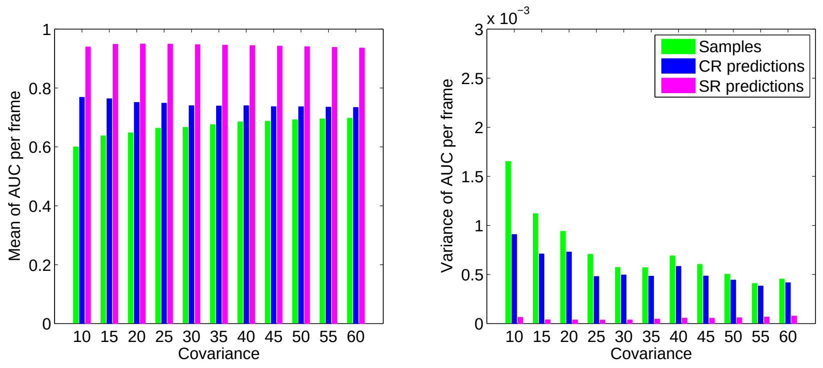


(c) SR predictions

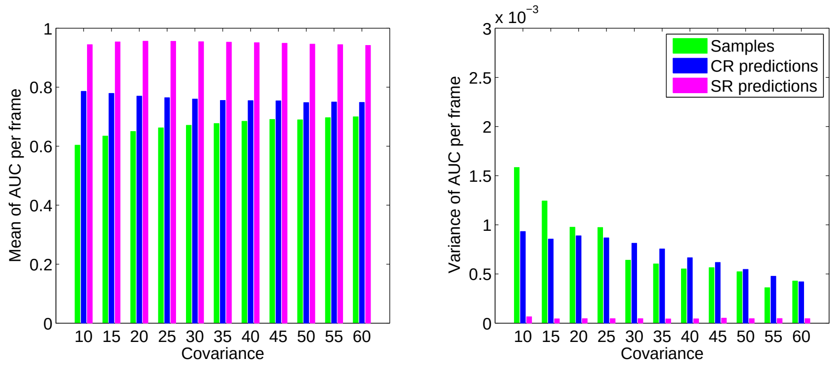


(d) The legend

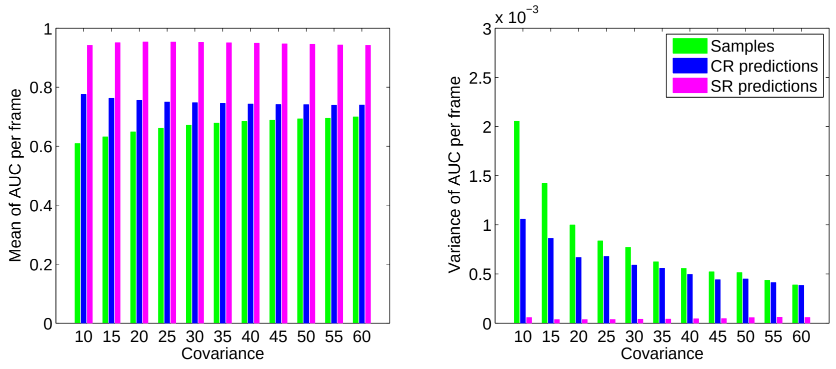
FIGURE 5.5: Results of average Euclidean distances from the ground truth target location to the test samples, CR predictions and SR predictions averaged over all frames in video *David3*. X axis shows the scaled Euclidean distance. Y axis shows the proportion of samples whose Euclidean distance to the ground truth target location is lower than a certain value. (a) The average Euclidean distances from test samples to the ground truth target location. (b) The average Euclidean distances from CR predictions to the ground truth target location. (c) The average Euclidean distances from SR predictions to the ground truth target location. (d) The legend of (a),(b) and (c).



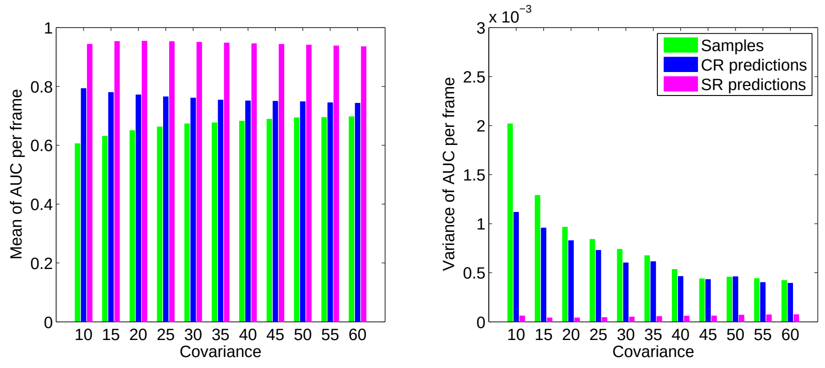
(a) David3



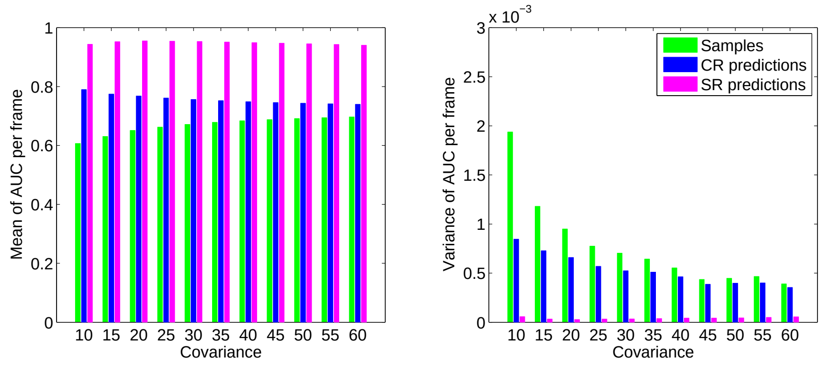
(b) MotorRolling



(c) Shaking



(d) Skating1



(e) Woman

FIGURE 5.6: Mean and variation of curves, another form of descriptions of the curves in Figure 5.4, of five videos.

expansion is not able to accurately approximate image features with a higher variance, as the smoothness of image features is not satisfied in the situation where the variation of image features is high. As a result, the first-order Taylor expansion of the original Continuous Regression can not be directly used in TRIC-track to approximate image feature space.

5.2 Locally Continuous Regression

We can alleviate the problem of Taylor expansion by applying a two-step displacement. An intermediate sampling can be applied first and the first-order Taylor expansion can be applied locally at each sample.

In particular, $s + \delta s$ can be split into $s + \delta s = s + \delta s_D + \delta s_C$, where $s + \delta s_D$ will be the linearisation point, and δs_C will be the continuous variable to integrate over. We can generate N perturbations (linearisation points), and then integrate over all the subspaces centred around them. In particular, the relations between these variables are illustrated in Figure 5.7. Then Eq. 5.3 is extended to:

$$\Phi(I, s^* + \delta s_{D_l} + \delta s_C) \approx \Phi(I, s^* + \delta s_{D_l}) + \Phi'(I, s^* + \delta s_{D_l})\delta s_C, \tag{5.10}$$

where $l \in [1, N]$. Eq. 5.10 can be written as:

$$\Phi(I, s^* + \delta s_{D_l} + \delta s_C) \approx X_l^* + J_l^* \delta s_C, \tag{5.11}$$

where $J_l^* = \frac{\partial \Phi(I, s^* + \delta s_{D_l})}{\partial s_C}$, and $X_l^* = \Phi(I, s^* + \delta s_{D_l})$. Note that the cardinality of the subspace, N, depends on the size of the whole sample space which is intended to be modelled. For a smaller sample space, the cardinality of the subspace can be less.

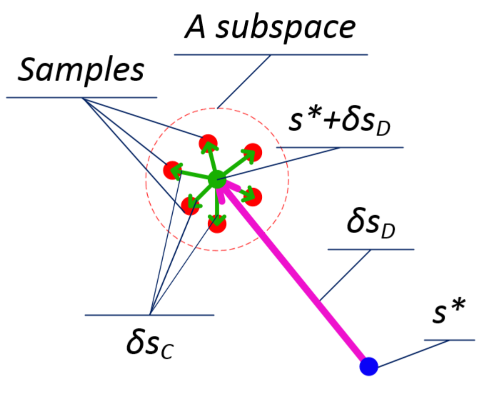


FIGURE 5.7: The illustration of the relation between a ground truth part location s^* and a subspace.

With this strategy, Eq. 5.2 is then extended to be:

$$\arg\min_R \sum_{l=1}^{N} \int_{\delta s_C} ||\delta s_{D_l} + \delta s_C - R\Phi(I, s^* + \delta s_{D_l} + \delta s_C)||_2^2 p(\delta s_C) d\delta s_C. \tag{5.12}$$

Applying the approximation in Eq. 5.11 to Eq. 5.12, then the least squares problem can be described by:

$$\begin{aligned}
&\arg\min_R \sum_{l=1}^{N} \int_{\delta s_C} p(\delta s_C)||\delta s_{D_l} + \delta s_C - R\Phi(I, s^* + \delta s_{D_l} + \delta s_C)||_2^2 d\delta s_C \approx \\
&\arg\min_R \sum_{l=1}^{N} \int_{\delta s_C} p(\delta s_C)||\delta s_{D_l} + \delta s_C - R\big(X_l^* + J_l^* \delta s_C\big)||_2^2 d\delta s_C = \\
&\arg\min_R \sum_{l=1}^{N} \int_{\delta s_C} p(\delta s_C)[\delta s_{D_l}^T \delta s_{D_l} + \delta s_C^T \delta s_C + (X_l^* + J_l^* \delta s_C)^T R^T R (X_l^* + J_l^* \delta s_C) \\
&\quad + 2\delta s_{D_l}^T \delta s_C - 2\delta s_C^T R(X_l^* + J_l^* \delta s_C) - 2\delta s_{D_l}^T R(X_l^* + J_l^* \delta s_C)] d\delta s_C.
\end{aligned} \tag{5.13}$$

After grouping independent, linear and quadratic terms, with respect to δs_C, Eq 5.12 is represented by:

$$\begin{aligned}
&\arg\min_R \sum_{l=1}^{N} \int_{\delta s_C} p(\delta s_C)||\delta s_{D_l} + \delta s_C - R\Phi(I, s^* + \delta s_{D_l} + \delta s_C)||_2^2 d\delta s_C \approx \\
&\arg\min_R \sum_{l=1}^{N} \int_{\delta s_C} p(\delta s_C)[\delta s_C^T A \delta s_C + 2\delta s_C^T b + 2\delta s_{D_l}^T (1 - R J_l^*)\delta s_C + \\
&\quad X_l^{*T} R^T R X_l^* + \delta s_{D_l}^T \delta s_{D_l} - 2\delta s_{D_l}^T R X_l^*] d\delta s_C,
\end{aligned} \tag{5.14}$$

where $X_l^* = \Phi(I, s^* + \delta s_{D_l})$, $A = (E - RJ_l^*)^T(E - RJ_l^*)$ (in which E is the identity matrix) and $b = J_l^{*T} R^T R X_l^* - R X_l^*$. The pdf $p(\delta s_C)$ is described by its mean μ_l and covariance Σ_l. By minimising Eq. 5.14, the closed-form solution of R is obtained as follows:

$$\begin{aligned}
R = &\left(\sum_{l=1}^{N} \Big(\mu_l X_l^{*T} + (\Sigma_l + \mu_l \mu_l^T) J_l^{*T} + \delta s_{D_l}(X_l^{*T} + \mu_l^T J_l^{*T})\Big)\right) \cdot \\
&\left(\sum_{l=1}^{N} \Big(X_l^* X_l^{*T} + 2X_l^* \mu_l^T J_l^{*T} + J_l^* (\Sigma_l + \mu_l \mu_l^T) J_l^{*T}\Big)\right)^{-1}.
\end{aligned} \tag{5.15}$$

Comparing Eq. 5.15 with Eq. 5.7, it is straightforward to see that Eq. 5.15 holds all combinations between sampling-based (only) regression and continuous regression. When $N = 1$, with $\delta s_{D_l} = 0$, the solution of R in Eq. 5.15 is exactly Eq. 5.7, the continuous regression. Moreover, this equation holds for the sampling-based regression as well. When $\mu_l = 0$ and $\Sigma_l = 0$, then $\delta s_C = 0$, the continuous space will then be the vacuum. Eq. 5.15 will be transferred to:

$$R = \left(\sum_{l=1}^{N} \delta s_{D_l} X_l^{*T} \right) \cdot \left(\sum_{l=1}^{N} X_l^{*} X_l^{*T} \right)^{-1}, \tag{5.16}$$

which is exactly the discrete linear regression explained in Chapter 4.

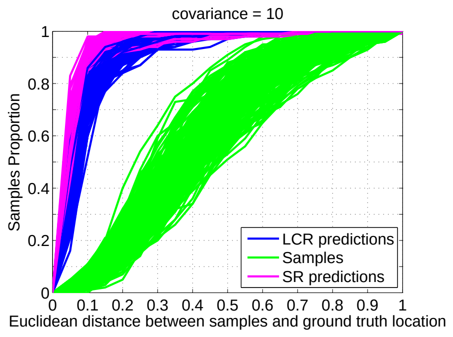
As previously, the matrix form can be used to help the computation of R: $M_l = [\mu_l, \Sigma_l + \mu_l \mu_l^T]$, $B_l = \begin{pmatrix} 1 & \mu_l^T \\ \mu_l & \Sigma_l + \mu_l \mu_l^T \end{pmatrix}$, $D_l^* = [X_l^*, J_l^*]$. Eq. 5.15 then can be calculated by:

$$R = \left(\sum_{l=1}^{N} \left(M_l D_l^{*T} + Y_l \right) \right) \left(\sum_{l=1}^{N} \left(D_l^* B_l D_l^{*T} \right) \right)^{-1}, \tag{5.17}$$

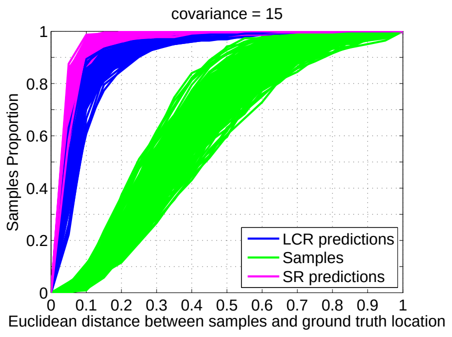
where $Y_l = \delta s_{D_l} (X_l^{*T} + \mu_l^T J_l^{*T})$.

The main idea of the proposed approach will then be to partition the space into regions of similar appearance, that can be further integrated. The region of similar appearance, a subspace, is a local area, which can be found empirically. A subspace can be defined by the sampling distribution (parametrised by its mean and covariance). Then the first-order Taylor expansion can be applied to a subspace to approximate the image features of the subspace. Image features and sample distribution of multiple subspaces can be aggregated respectively to obtain the image information and the sample distribution of the whole sample space.

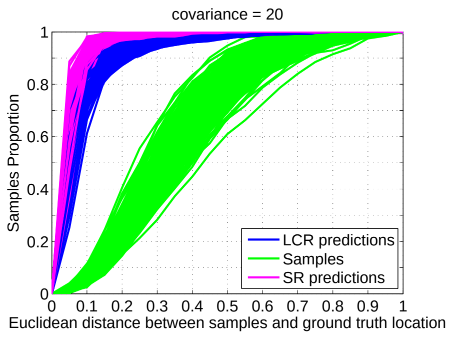
It is expected that, the higher the variance of image features, the more regions will need sampling. In the CCR/SDM framework, this means that the upper cascade levels will need higher-partitioning N, whereas lower levels will need less. This will be empirically studied and shown in Section 5.4. Figure 5.8 shows results similar to Figure 5.4. The only difference is the Continuous Regression (CR) is replaced with Locally Continuous Regression (LCR) and the number of clusters used is 16. It can be seen from the comparison between Figure 5.4 and 5.8, LCR has alleviated the problem of Taylor expansion and provided much more accurate predictions compared to original CR method. LCR has shown similar performance to the sampling-based regression.


covariance = 10
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

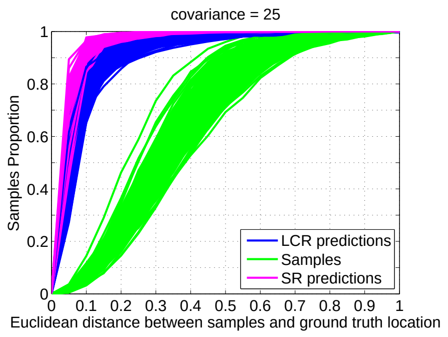
(a)


covariance = 15
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

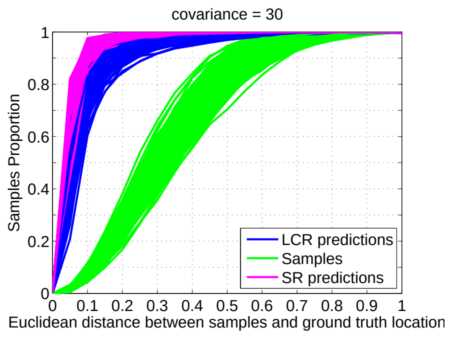
(b)


covariance = 20
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

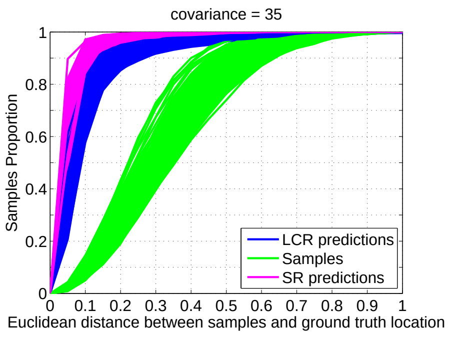
(c)


covariance = 25
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

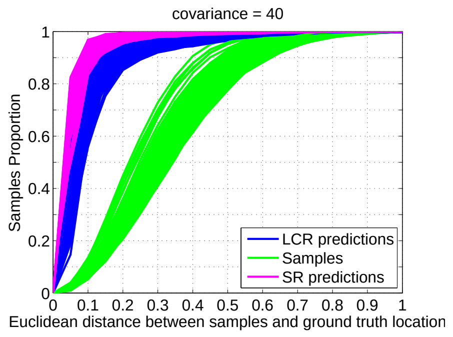
(d)


covariance = 30
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

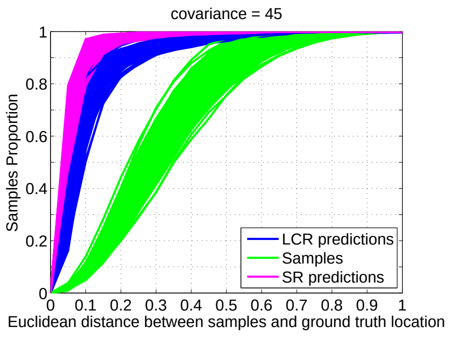
(e)


covariance = 35
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

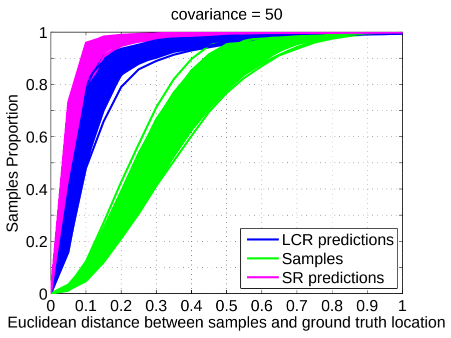
(f)


covariance = 40
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

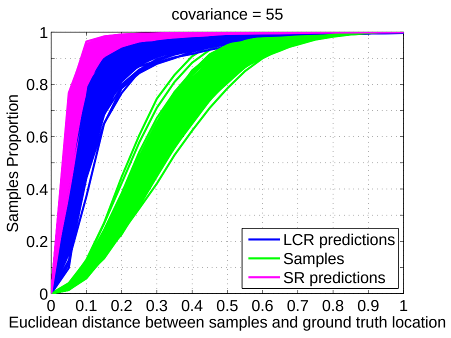
(g)


covariance = 45
Samples Proportion
Euclidean distance between samples and ground truth location
LCR predictions
Samples
SR predictions

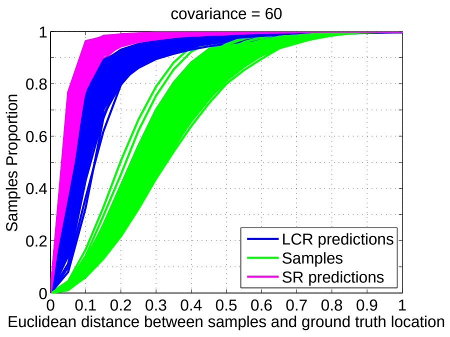
(h)



(i)



(j)



(k)

FIGURE 5.8: Results of Euclidean distances from the ground truth target location to the test samples, LCR predictions and SR predictions in every frame in video *David*3. X axis shows the scaled Euclidean distance. Y axis shows the proportion of samples whose Euclidean distance to the ground truth target location is lower than a certain value. Every curve is the result of a frame in video *David*3.

5.3 Integration of Locally Continuous Regression into TRIC-track

The methodology which integrates the proposed Locally Continuous Regression into the main framework of TRIC-track is explained in this section. This main framework of TRIC-track consists of cascaded regression, the implicit shape model and cascaded incremental updating. This methodology is named LC-TRIC (Tracking by Locally Continuous Regression with Incrementally Learned Cascades). It contains Cascaded Locally Continuous Regression, the implicit shape model and Cascaded Locally Continuous Incremental Updating (see Figure 5.9). The Cascaded Locally Continuous Regression tracker is explained in Section 5.3.1, and the Cascaded Locally Continuous Incremental Updating is explained in Section 5.3.2.

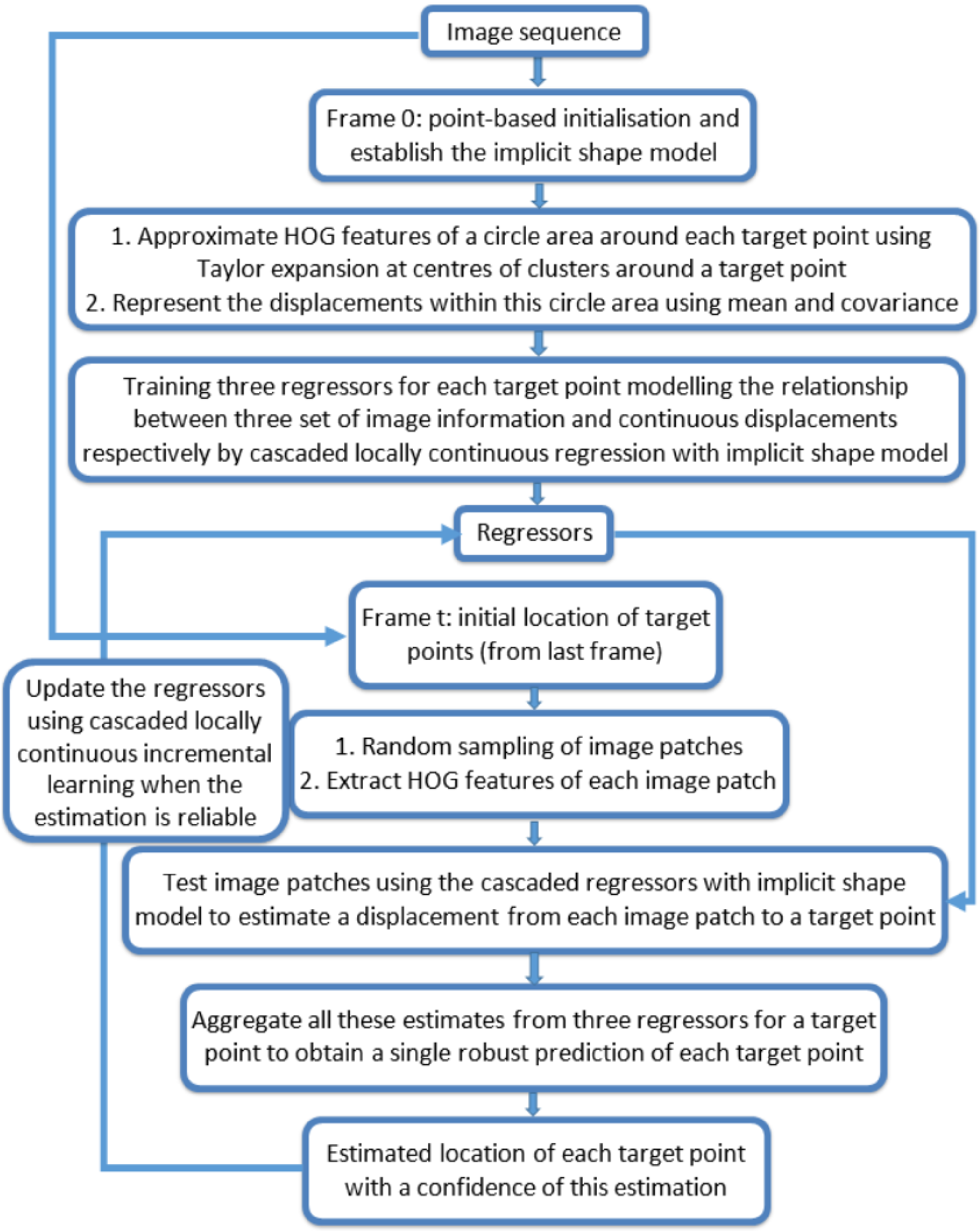


Image sequence
Frame 0: point-based initialisation and establish the implicit shape model
1. Approximate HOG features of a circle area around each target point using Taylor expansion at centres of clusters around a target point
2. Represent the displacements within this circle area using mean and covariance
Training three regressors for each target point modelling the relationship between three set of image information and continuous displacements respectively by cascaded locally continuous regression with implicit shape model
Regressors
Frame t: initial location of target points (from last frame)
Update the regressors using cascaded locally continuous incremental learning when the estimation is reliable
1. Random sampling of image patches
2. Extract HOG features of each image patch
Test image patches using the cascaded regressors with implicit shape model to estimate a displacement from each image patch to a target point
Aggregate all these estimates from three regressors for a target point to obtain a single robust prediction of each target point
Estimated location of each target point with a confidence of this estimation

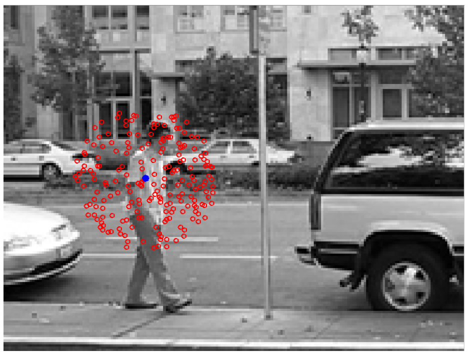
FIGURE 5.9: The framework of the LC-TRIC tracker.

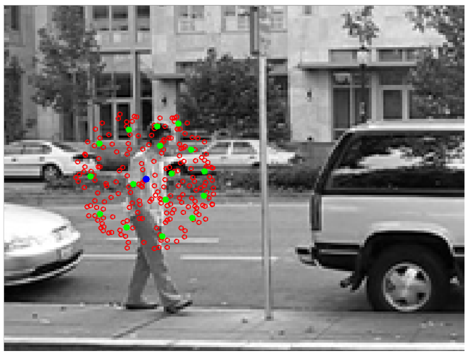
5.3.1 Cascaded Locally Continuous Regression with the Implicit Shape Model

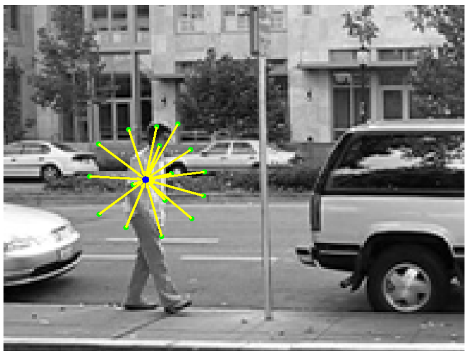
In the context of TRIC-track, the methodology centres on generating perturbations according to the method described in Chapter 4. However, instead of extracting features over the generated perturbations, the initial positions will be assigned to different partitions, and only the centre of each will require the feature extraction (δs_D will be the distance between the ground-truth and the partition centre). The way to assign the perturbed locations to partitions will be through clustering (e.g. using k-means) on the locations of the perturbations, as shown in Figure 5.10, where the number of clusters N will be a parameter to tune. Figure 5.10 also shows that the sample space becomes increasingly small, resulting in fewer clusters in lower cascade levels. Once the perturbed locations have been assigned to a cluster, the inner statistics for each of them (μ_l, Σ_l) can be computed by measuring the distance (δs_C) between the points assigned to the cluster and their corresponding centre. Each of the clusters will have a different set of statistics, which are used to integrate over each of the regions.

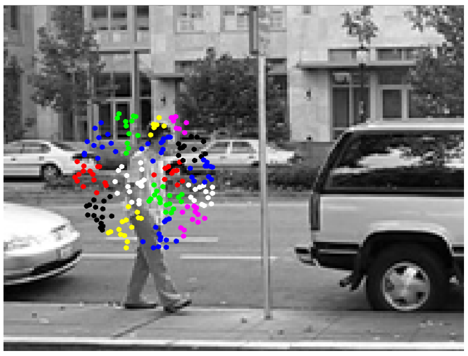
Specifically, given image I (usually the first frame of a video sequence) and the initial part location s_i^*, $i \in [1, N_p]$, K samples are obtained for each part by randomly sampling from a circular area around the part's location, as in TRIC-track. N_p is the total number of the target parts. Sample locations around part i are denoted as $P_i = [p_{i1}, ...p_{ij}, ...p_{iK}]^T$ where p_{ij} is a 2D location in the image, $p_{ij} = (x_{ij}, y_{ij})$. Note that there is no need to calculate the image features of all these sample locations. The partitions around each part location s_i^* are obtained by clustering the K samples around the part based on their locations. Each cluster has a centre. The displacement from the target location s_i^* to the centre of l_{th} cluster is denoted as $\delta s_{D_{il}}$. Only the features and their Jacobian at the centres of clusters need to be computed. The image feature and its Jacobian at the l_{th} cluster centre around part i is denoted as X_{il}^* and J_{il}^* separately. The inner statistics of the cluster can be modelled by its mean μ_{il} and covariance Σ_{il}, which are obtained by measuring the displacement, δs_C, from the centre of the cluster to the samples assigned to the cluster.

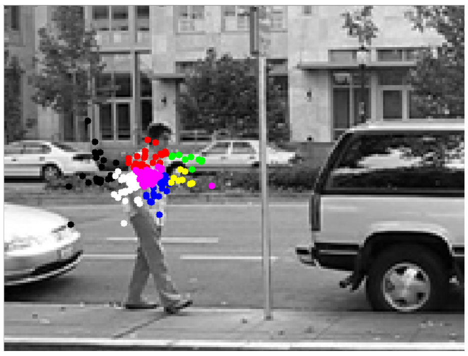
The implicit shape configuration of the target is shown in Figure 5.11. Because of the implicit shape model structure of TRIC-track, three regressors are trained for each part using image features and displacement distributions from three parts separately. Specifically, for part i, its neighbours are part $i-1$ and part $i+1$, as shown in Figure 5.11. When seeking to locate part i, three separate regressors $(R_{i,i-1}$ $R_{i,i}$ and $R_{i,i+1})$ are trained using image features, from part $i-1$, part i and part $i+1$, and the statistics of displacements, from s_i^* to the samples from sample space around part $i-1$, part i and part $i+1$, respectively.

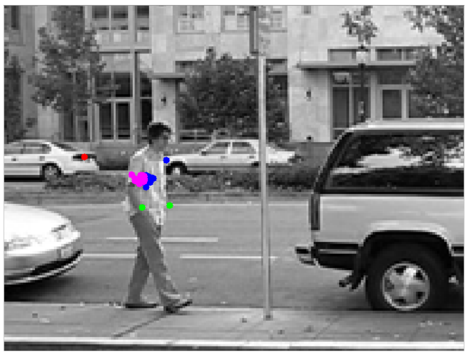

(a)

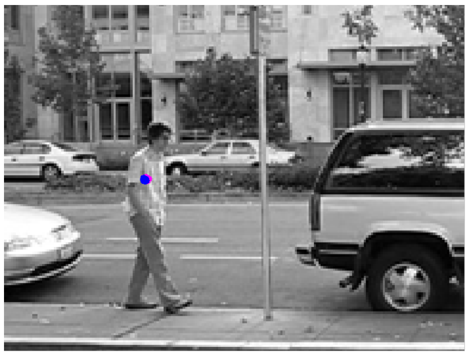

(b)


(c)


(d)


(e)


(f)



(g)

FIGURE 5.10: The illustration of modelling more sample regions with Locally Continuous Regression by clustering. Continuous Regression are applied to these subspaces. (a) The red dots are sample locations around a ground truth part location (the blue dot). (b) The green dots are means of different subspaces which are obtained through clustering on sample locations. (c) The yellow arrows represent the displacements from the ground truth part location to the local subspaces centres. (d) Dots in separate colourful areas represent 16 subspaces modelling the sample space in the first cascade level. (e) 8 subspaces model the sample space in the second cascade level. (f) 4 subspaces model the sample space in the third cascade level. (g) 2 subspaces model the sample space in the fourth cascade level.

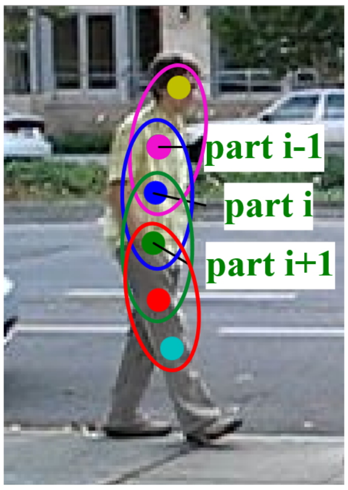


FIGURE 5.11: The implicit shape structure in TRIC-track. Part i's location is constrained by the information from its two neighbouring parts, part $i+1$ and part $i-1$, and itself.

The regressor $R_{i,i+1}$, which models the relationship between image features around part $i+1$ and the distribution of displacements from part i to samples around part $i+1$, then can be calculated by:

$$R_{i,i+1} = \left(\sum_{l=1}^{N}\left(M_{i+1,l}D_{i+1,l}^{*T} + Y_{i+1,l}\right)\right)\left(\sum_{l=1}^{N}\left(D_{i+1,l}^{*}B_{i+1,l}D_{i+1,l}^{*T}\right)\right)^{-1}, \tag{5.18}$$

where $M_{i+1,l} = [\mu_{i+1,l}, \Sigma_{i+1,l} + \mu_{i+1,l}\mu_{i+1,l}^{T}]$, $B_{i+1,l} = \begin{pmatrix} 1 & \mu_{i+1,l}^{T} \\ \mu_{i+1,l} & \Sigma_{i+1,l} + \mu_{i+1,l}\mu_{i+1,l}^{T} \end{pmatrix}$, $D_{i+1,l}^{*} = [X_{i+1,l}^{*}, J_{i+1,l}^{*}]$, and $Y_{i+1,l} = (s_{i+1}^{*} + \delta s_{D_{i+1,l}} - s_i^{*})(X_{i+1,l}^{*T} + \mu_{i+1,l}^{T}J_{i+1,l}^{*T})$.

The regressor $R_{i,i-1}$ and $R_{i,i}$ are trained following the same method as training $R_{i,i+1}$. $R_{i,i-1}$ models the relationship between image features around part $i-1$ and the distribution of displacements from part i to samples around part $i-1$. $R_{i,i}$ models the relationship between image features around part i and the distribution of displacements from part i to samples around part i. The training process of the three regressors for part i is then finished. The regressors for N_p parts are all trained using Locally Continuous Regression similarly.

In TRIC-track, a cascade of regressors is learned to make increasingly smaller steps towards the target. Similarly, the Cascaded Locally Continuous Regression method first learns R_i^0, where 0 denotes the first level of the cascaded regression. $R_{i,i-1}$, $R_{i,i}$, and $R_{i,i+1}$ are then noted as $R_{i,i-1}^0$, $R_{i,i}^0$, and $R_{i,i+1}^0$ respectively.

After training the first cascade level, the statistics of the sample space can be updated by testing the regressor. When testing, the use of the regressor is the same as SDM in TRIC-track. Utilising the first-order Taylor expansion to approximate the image features of the training samples does not work during testing, as the regressors are overfit to these features. K samples around each part i are obtained by random sampling in the beginning of the test stage. Similarly to Eq. 4.14, the sample locations around part i can be updated by applying the first cascaded regression and uniting neighbouring parts' predictions as follows:

$$P_i^1 = (P_{i-1}^0 - R_{i,i-1}^0\Phi(I, P_{i-1}^0, p_s)) \cup (P_i^0 - R_{i,i}^0\Phi(I, P_i^0, p_s)) \cup (P_{i+1}^0 - R_{i,i+1}^0\Phi(I, P_{i+1}^0, p_s)), \tag{5.19}$$

where P_i^1 is the updated sample locations around part i, which is the first cascade level prediction of part i actually. As the P_i^1 contains predictions from samples around three parts, the number of instances within P_i^1 is three times as much as P_i^0. A weighted random sampling without replacement strategy is used to reduce the number of predictions (updated samples) within the P_i^1. Specifically, the distances between each sample with all other samples are calculated and the reciprocal of the sum of the distances is used as the weight of this sample. Samples with larger weights are given higher probability of being sampled. After this sampling process, the number of samples retained around each part is still K. First level predictions of all other parts are obtained similarly.

The statistics of the sample space around each part can be updated by clustering the P^1 around each part. As the samples around each part become closer to the part than those in the previous cascade level, the whole sample space around each part can be modelled by fewer partitions again obtained by clustering on the sample locations. New partitions of this cascade level are found by clustering and capture the new statistics of the updated sample space. The image features and their Jacobians at the centres of the updated partitions are calculated. The statistics of δs_C of each subspace is updated as well. Like the first level of cascaded continuous regression, $R^1_{i,i-1}$, $R^1_{i,i}$ and $R^1_{i,i+1}$ are obtained by equation 5.18. The R^2, R^3,... are learned in the same way.

As with TRIC-track, in the test stage, the same number of cascaded regressors contribute to the final prediction P_i^{n-1}, in which n is the number of cascade levels and fixed to 4 in the experiments. The aggregation technique, Local Evidence Aggregation for Regression (LEAR) (Martinez et al., 2013), is again used to determine the part i's position from the last cascade level prediction P_i^{n-1}.

5.3.2 Cascaded Locally Continuous Incremental Updating

In TRIC-track, the incremental updating, inspired by Asthana et al. (Asthana et al., 2014), is used to update the regressor, trained in the first frame of a video sequence, by adding new data (image features and corresponding displacements) in the current frame. With the incremental updating, there is no need to retrain the regressor in every frame. However, the cascaded incremental update used in TRIC-track, as explained in 4.3.4, is time consuming, because the updating of the cascaded regression model requires the data to be sampled per cascade level for each target part. Another drawback is that the updating in TRIC-track is performed with a limited set of samples. The incremental Cascaded Continuous Regression (iCCR) in (Sánchez-Lozano et al., 2016) can be introduced to TRIC-track to cope with the above two problems. Nevertheless, similarly to Cascaded Continuous Regression (CCR), iCCR has its limitation because of the Taylor expansion. The Cascaded Locally Continuous Incremental Updating is proposed and introduced into TRIC-track in Section 5.3.2.2.

5.3.2.1 Incremental Cascaded Continuous Regression

To update the Cascaded Continuous Regression (CCR) model using a new set of images and estimated shapes, Sanchez-Lozano et al. (Sánchez-Lozano et al., 2016) propose incremental CCR (iCCR) achieving real-time updating in the cascaded regression framework. An overview of the incremental learning process is illustrated in Figure 5.12.

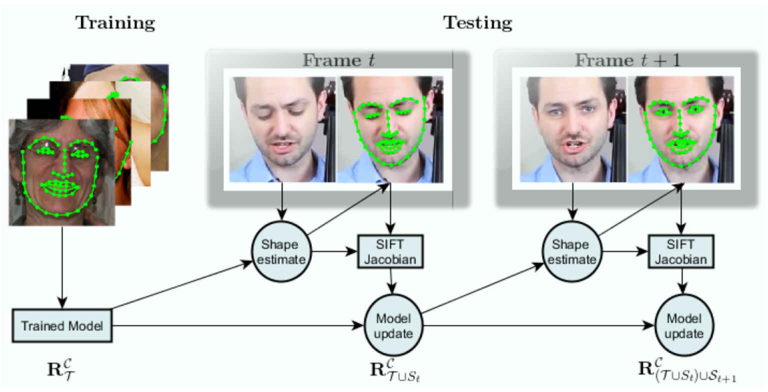

Training
Testing
Frame t
Frame $t+1$
Shape estimate
SIFT Jacobian
Model update
Shape estimate
SIFT Jacobian
Model update
Trained Model
$\mathbf{R}_{\mathcal{T}}^{\mathcal{C}}$
$\mathbf{R}_{\mathcal{T}\cup\mathcal{S}_t}^{\mathcal{C}}$
$\mathbf{R}_{(\mathcal{T}\cup\mathcal{S}_t)\cup\mathcal{S}_{t+1}}^{\mathcal{C}}$

FIGURE 5.12: Overview of the incremental cascaded continuous regression algorithm (iCCR). The originally model $R_{\mathcal{T}}$ learned offline is updated with each new frame, thus sequentially adapting to the target face (Sánchez-Lozano et al., 2016).

Specifically, given a regressor $R_{\mathcal{T}}$ trained using Eq. 5.9, on a training set $\mathcal{T}$. The covariance matrix for the training data is denoted as $V_{\mathcal{T}} := Cov(X, X) = D_{\mathcal{T}}^{*} B (D_{\mathcal{T}}^{*})^{T}$. The incremental online learning is to update $R_{\mathcal{T}}$ with a set of $\mathcal{S}$ new images, from which the ground truth image features and Jacobians are extracted. Let $D_{\mathcal{S}}^{*}$ be the data (image features and Jacobians) corresponding to the new added samples for updating. The new regressor would be computed as:

$$R_{\mathcal{T}\cup\mathcal{S}} = M \left(D^{*} + D_{\mathcal{S}}^{*} \right)^{T} (V_{\mathcal{T}\cup\mathcal{S}})^{-1}, \tag{5.20}$$

where

$$(V_{\mathcal{T}\cup\mathcal{S}})^{-1} = (V_{\mathcal{T}} + D_{\mathcal{S}}^{*} B D_{\mathcal{S}}^{*T})^{-1}. \tag{5.21}$$

Applying the Woodbury identity (Brookes, 2011) to Eq. 5.21:

$$(V_{\mathcal{T}\cup\mathcal{S}})^{-1} = V_{\mathcal{T}}^{-1} - V_{\mathcal{T}}^{-1} D_{\mathcal{S}}^{*} (B^{-1} + D_{\mathcal{S}}^{*T} V_{\mathcal{T}}^{-1} D_{\mathcal{S}}^{*})^{-1} D_{\mathcal{S}}^{*T} V_{\mathcal{T}}^{-1}. \tag{5.22}$$

The updating with iCCR does not require the sampling process in each cascade level for each part. Only the image features and their Jacobians at ground truth locations are needed, which is time efficient.

5.3.2.2 Cascaded Locally Continuous Incremental Updating

Similarly to Continuous Regression, incremental Cascaded Continuous Regression (iCCR) can not be directly applied to the general visual tracking problem because the smoothness requirement of the image feature space is usually not satisfied in general object tracking, especially within the implicit shape model of TRIC-track. The iCCR is then extended and the Cascaded Locally Continuous Incremental Updating is proposed and integrated into TRIC-track's framework in this section.

As in TRIC-track, a part's regressors in LC-TRIC are updated only when the confidence of the part's predicted location is higher than an empirically determined threshold δ_v. After obtaining the prediction of the target part location in every frame, the regressor is incrementally updated by adding new training data, image features and corresponding displacement distributions, using Cascaded Locally Continuous Incremental Updating strategy.

Similarly to the Locally Continuous Regression in training, given the prediction of the target part location in current frame, instead of generating perturbations around each part, a few initial centres of different partitions are generated. A partition can be called a cluster as well. The centres of the clusters around a part are assumed to have the same displacements to the target part location in the current frame as those in the first frame. The statistics of each partition is also kept the same as that in the first frame. Only the centres of clusters require the calculation of the image features and the Jacobians. The inner statistics, the image feature and its Jacobian of the centre of a new partition are used to update the regressor trained in the first frame.

Specifically, given the regressor trained using Eq. 5.18 in the first frame of a video sequence, $R_{\mathcal{T}}$, $\mathcal{T}$ is the training set, and $R_{\mathcal{T}}$ is obtained by:

$$R_{\mathcal{T}} = U_{\mathcal{T}}(V_{\mathcal{T}})^{-1}, \tag{5.23}$$

$$U_{\mathcal{T}} = \sum_{l=1}^{N} \left(M_l D_l^{*T} + Y_l \right), \tag{5.24}$$

$$V_{\mathcal{T}} = \sum_{l=1}^{N} \left(D_l^{*} B_l D_l^{*T} \right), \tag{5.25}$$

then with new training data from a cluster C_a, $R_{\mathcal{T}\cup C_a}$ is calculated by:

$$R_{\mathcal{T}\cup C_a} = (U_{\mathcal{T}} + M_a D_a^{*T} + Y_a)(V_{\mathcal{T}} + D_a^* B_a D_a^{*T})^{-1}, \tag{5.26}$$

$$(V_{\mathcal{T}} + D_a^* B_a D_a^{*T})^{-1} = V_{\mathcal{T}}^{-1} - V_{\mathcal{T}}^{-1} D_a^* (B_a^{-1} + D_a^{*T} V_{\mathcal{T}}^{-1} D_a^*)^{-1} D_a^{*T} V_{\mathcal{T}}^{-1}, \tag{5.27}$$

$$Y_a = \delta s_{D_a} (X_a^{*T} + \mu_a^T J_a^{*T}), \tag{5.28}$$

where $M_a = [\mu_a, \Sigma_a + \mu_a \mu_a^T]$, $D_a^* = [X_a^*, J_a^*]$ and $B_a = \begin{pmatrix} 1 & \mu_a^T \\ \mu_a & \Sigma_a + \mu_a \mu_a^T \end{pmatrix}$. The predicted target part location in frame t is denoted as $s(t)^*$. The centre of the cluster $a(t)$ in current frame t can be calculated by: $a(t) = s(t)^* + a(1) - s(1)^*$, where $a(1)$ and $s(1)^*$ are the corresponding cluster centre and the ground truth target location in the first frame respectively. With Continuous Regression, only the image features X_a^* and the Jacobian J_a^* at the updated cluster centre need to be calculated. The mean μ_a and covariance Σ_a are the same as the mean and the covariance of the corresponding cluster in the first frame. There are the same number N clusters around a target part as the training stage. The $R_{\mathcal{T}\cup C_1 \cup C_2 \cup \ldots \cup C_N}$ can be calculated by:

$$R_{\mathcal{T}\cup C_1 \cup C_2 \cup \ldots \cup C_N} = \left(U_{\mathcal{T}} + \sum_{j=1}^{N} (M_j D_j^{*T} + Y_j)\right) \left(V_{\mathcal{T}} + \sum_{j=1}^{N} D_j^* B_j D_j^{*T}\right)^{-1} \tag{5.29}$$

After adding all training data from N clusters for each regressor using Eq. 5.29, the incremental updating of the regressor is finished. It is worth highlighting that the updating of different cascade levels can be performed in parallel. The mean and covariance of the sample distribution are obtained in training and available for incremental updating, making parallel updating of different cascade levels possible.

5.4 Evaluation

The proposed Tracking by Locally Continuous Regression with Incrementally Learned Cascades (LC-TRIC) is evaluated in this section. First, three studies are performed to optimise three parameters of LC-TRIC in Section 5.4.1 using the CVPR2013 dataset. Second, LC-TRIC is assessed with the recently published VOT2015 benchmark, and compared with its corresponding sampling-based regression in Section 5.4.2. Experimental results show that LC-TRIC improves the speed by six times compared to the sampling-based regression, without sacrificing the performance.

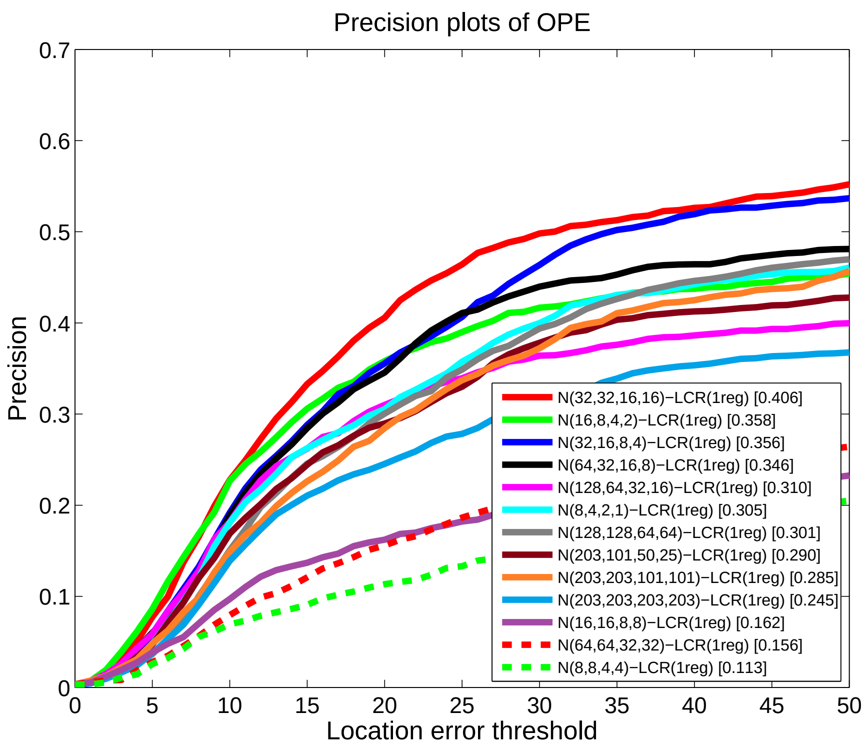
5.4.1 Internal Evaluation

The internal evaluation is performed to optimise three internal parameters of the proposed LC-TRIC tracker. The internal parameters include: 1) the number of clusters, N, in each cascade level, 2) the regularisation term in regression, lambda λ, and 3) the updating threshold δ_v. The image feature used is the HOG feature. Other features are also applicable and this can be explored in future work. Other parameters are set the same values as those in TRIC-track. The first internal study investigates the effect of the number of clusters in each cascade level on LC-TRIC without and with incremental updating, respectively. This experiment concludes the optimal value of the number of the clusters. The second examines the influence of the regularisation term λ in regression. The third studies the impact of the updating threshold. The first experiment is performed on the LC-TRIC tracker with and without incremental learning separately. The second and third experiments are performed on the full LC-TRIC tracker. Five videos of different levels of difficulty for tracking are selected from CVPR2013 benchmark (Wu et al., 2013) as the test videos in the internal evaluation. The videos are $Shaking$, $David3$, $Woman$, $Skating1$ and $MotorRolling$, which are in order of increasing difficulty. The measurements used in the internal evaluation are the same as those adopted in CVPR2013 benchmark, explained in Section 4.4, which are precision and success plots.

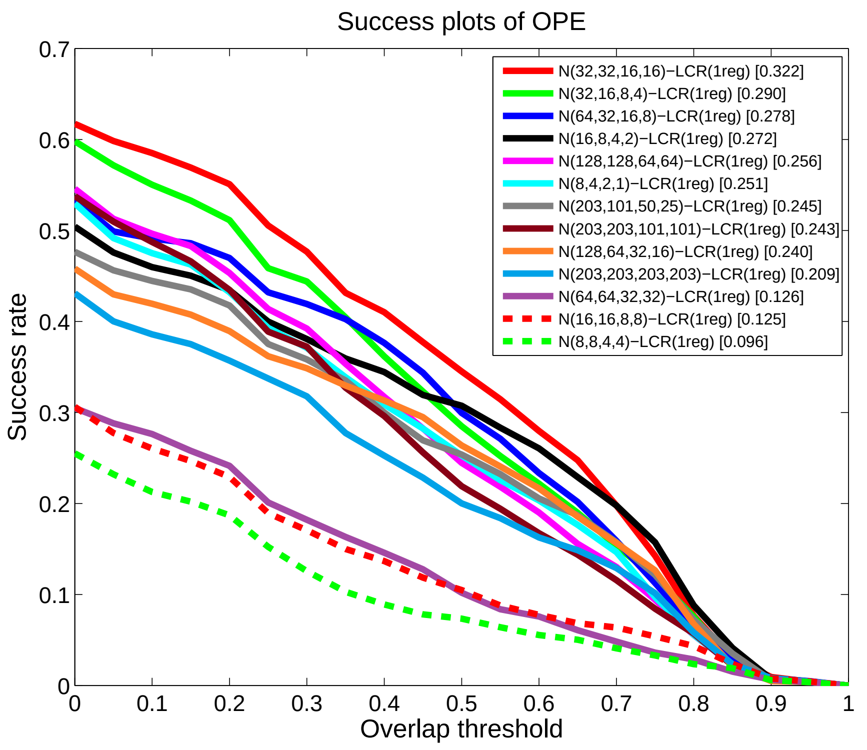
First, the number of clusters in each cascade level is investigated. There are in total four cascade levels used, as TRIC-track explained in Chapter 4. The number of perturbations in TRIC-track is 203 when sampling radius is equal to 20 pixels, which is determined by the optimal sample density - 90 samples with sampling radius equal to 15 pixels (see Figure 4.8(a)). Cascaded Locally Continuous Regression with $N = [203, 203, 203, 203]$ is equal to sampling-based regression. $N = [203, 203, 203, 203]$ is the highest value of the number of clusters for four cascade levels. It is expected that the number of clusters depends on the range of a sample space. Fewer clusters are expected to be needed in further cascade levels because of the increasingly small sample space. Specifically, in total there are 13 combinations of cluster numbers N for four cascade levels evaluated in this study, which are $[203, 203, 203, 203]$, $[203, 203, 101, 101]$, $[203, 101, 50, 25]$, $[128, 128, 64, 64]$, $[128, 64, 32, 16]$, $[64, 64, 32, 32]$, $[64, 32, 16, 8]$, $[32, 32, 16, 16]$, $[32, 16, 8, 4]$, $[16, 16, 8, 8]$, $[16, 8, 4, 2]$, $[8, 8, 4, 4]$ and $[8, 4, 2, 1]$. Framework 1 is Cascaded Locally Continuous Regression with concatenating features, which means only training one regressor for a part by concatenating the features from three parts. Framework 2 is Cascaded Locally Continuous Regression with the implicit shape model, which is to train three regressors for a part using image features and corresponding displacements from three parts. The 13 combinations are tested on Framework 1 and Framework 2, and the results are shown in Figure 5.13 and Figure 5.14 respectively. Both Framework 1 and Framework 2 are run on the five test videos given the initial target location. The experimental results show that the performance of Framework 2 is much better than that of Framework 1, as shown in Figure 5.13, Figure 5.14 and Figure 5.15, which further confirms the advantage of the implicit shape model. It can be seen from the result of Framework 2, the combinations of N which achieve the top two performance are $[32, 16, 8, 4]$ and $[16, 8, 4, 2]$. This proves that further cascade levels need less clusters to represent the whole sample space, as the sample variation becomes less and less with the increase of the cascade level. To further find the optimal combination of cluster number N, the two combinations, $[32, 16, 8, 4]$ and $[16, 8, 4, 2]$, are tested on the proposed LC-TRIC tracker, in which the Cascaded Locally Continuous Incremental Updating is added, the experimental result is shown in Figure 5.16. It is shown in Figure 5.14 that the two combinations have similar performance when tested on Locally Continuous Regression. When testing the two combinations with LC-TRIC tracker, which includes the incremental updating, $[16, 8, 4, 2]$ shows better performance than $[32, 16, 8, 4]$, as shown in Figure 5.16. It shows that the optimal combination of cluster number is $N = [16, 8, 4, 2]$ for four cascade levels in the LC-TRIC tracker.

Second, the regularisation parameter lambda λ is determined by varying its value in the range of 100, 1, 0.01, 0.001, 0.0001, with the number of clusters fixed to $N = [16, 8, 4, 2]$. This experiment is tested with LC-TRIC tracker on five test videos for internal evaluation. The results are shown in Figure 5.17. The results show that $\lambda = 0.01$ is the optimal value for LC-TRIC.

Third, the updating threshold is exploited by setting its value to 0.0065, 0.0075, 0.0085, 0.0095, 0.0100, 0.0105, 0.0115, 0.0125, 0.0135, 0.0145. The experiment is performed on the LC-TRIC tracker with the regularisation term set to 0.01 and cluster number $N = [16, 8, 4, 2]$. This study is performed on the full dataset of CVPR2013 benchmark. The experimental results are shown in Figure 5.18 and 5.19. The precision and the success plots in Figure 5.18 show that updating threshold of 0.0115, 0.0065 and 0.0100 display the top three performance and their performance are very close to each other. Area under the curve (AUC) of precision plot and success plot are shown in Figure 5.19. It shows that the update thresholds (ranging from 0.0100 to 0.0115) show steady higher performance than other values of update threshold. As in the success plots in Figure 5.18, updating threshold 0.0100 has higher performance than the other two values in the range with larger overlap threshold from 0.3 to 0.7, 0.0100 is selected as the optimal value of updating threshold for LC-TRIC.

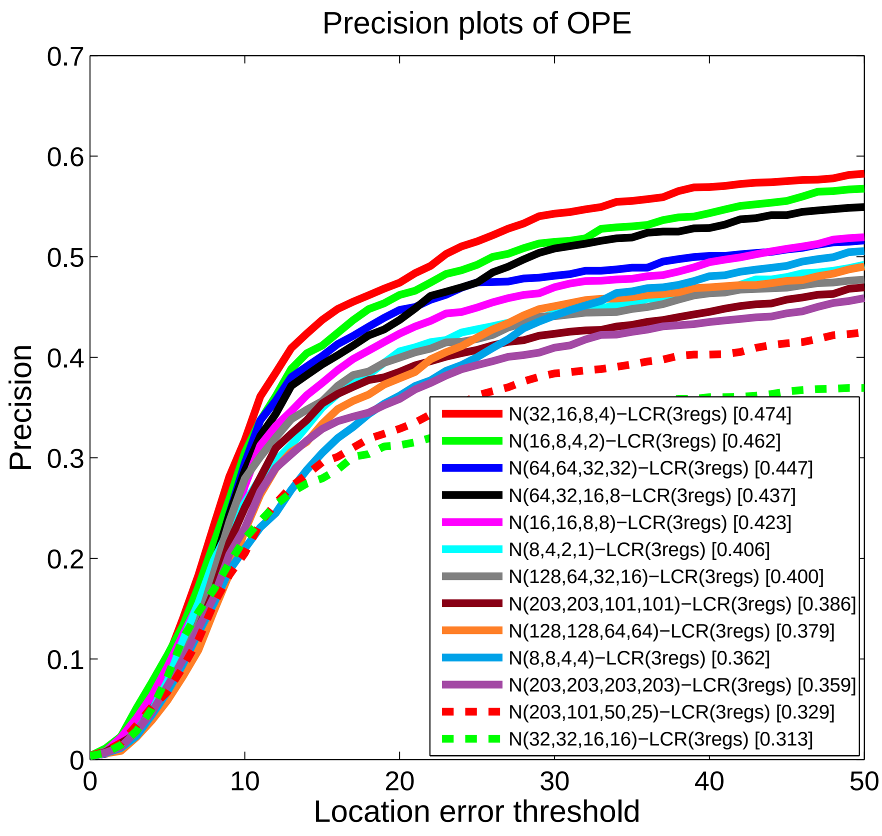


(a)

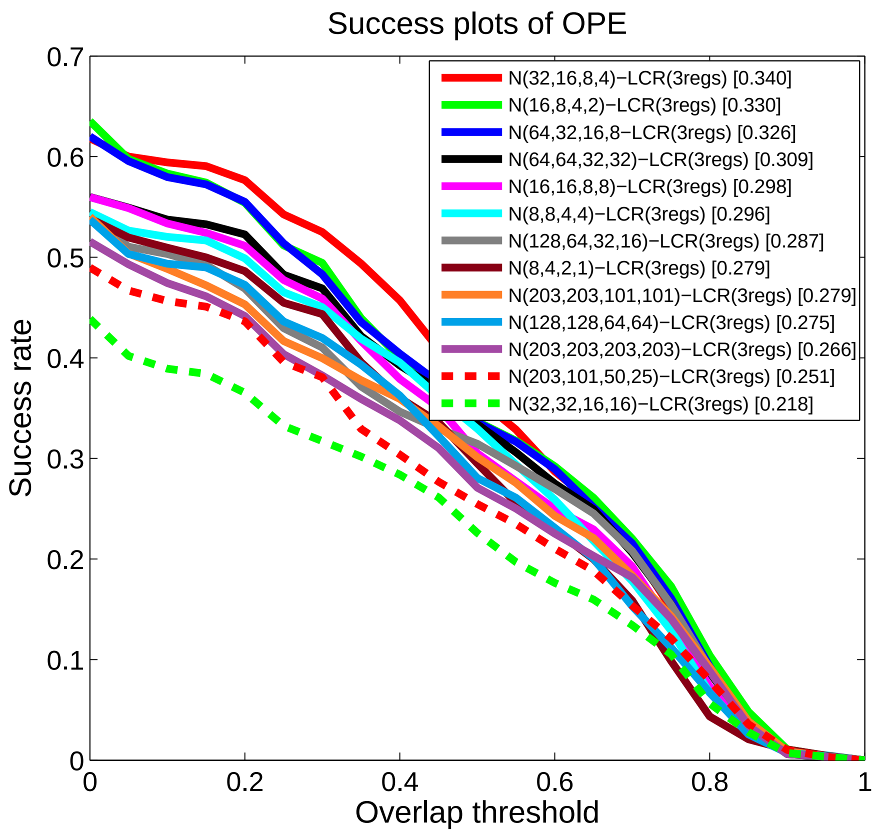


(b)

FIGURE 5.13: Precision and Success Plots of evaluation of different combinations of cluster numbers on Framework 1. Framework 1 is to use the Cascaded Locally Continuous Regression to train one regressor for a part by concatenating features of three parts.



(a)



(b)

FIGURE 5.14: Precision and Success Plots of evaluation of different combinations of cluster numbers on Framework 2. Framework 2 is the Cascaded Locally Continuous Regression with the implicit shape model, generating three regressors for a part.

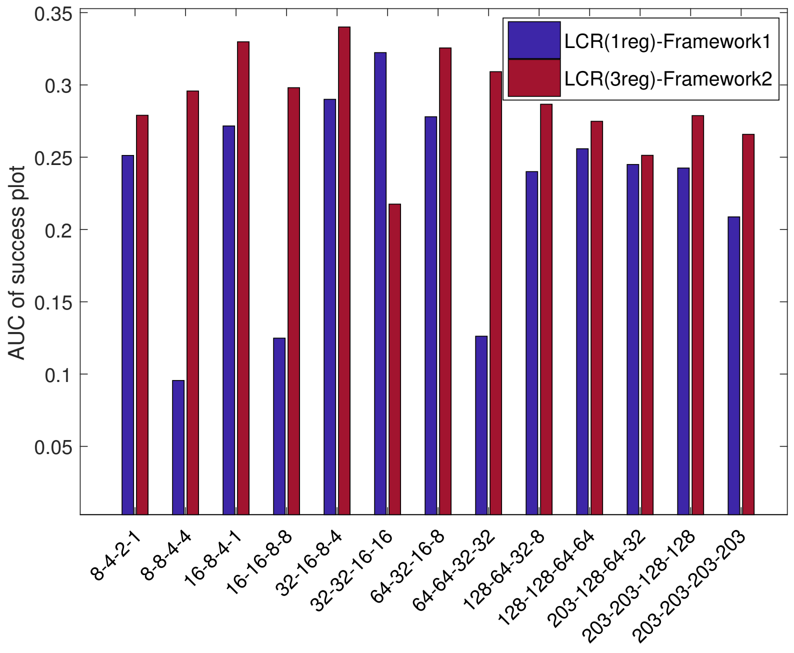
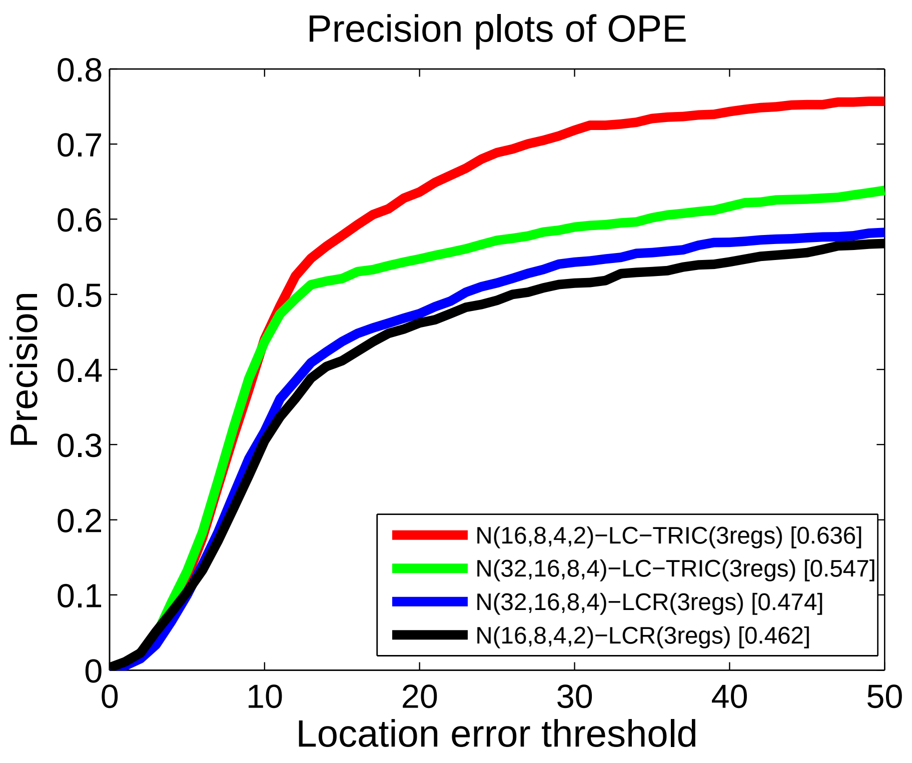
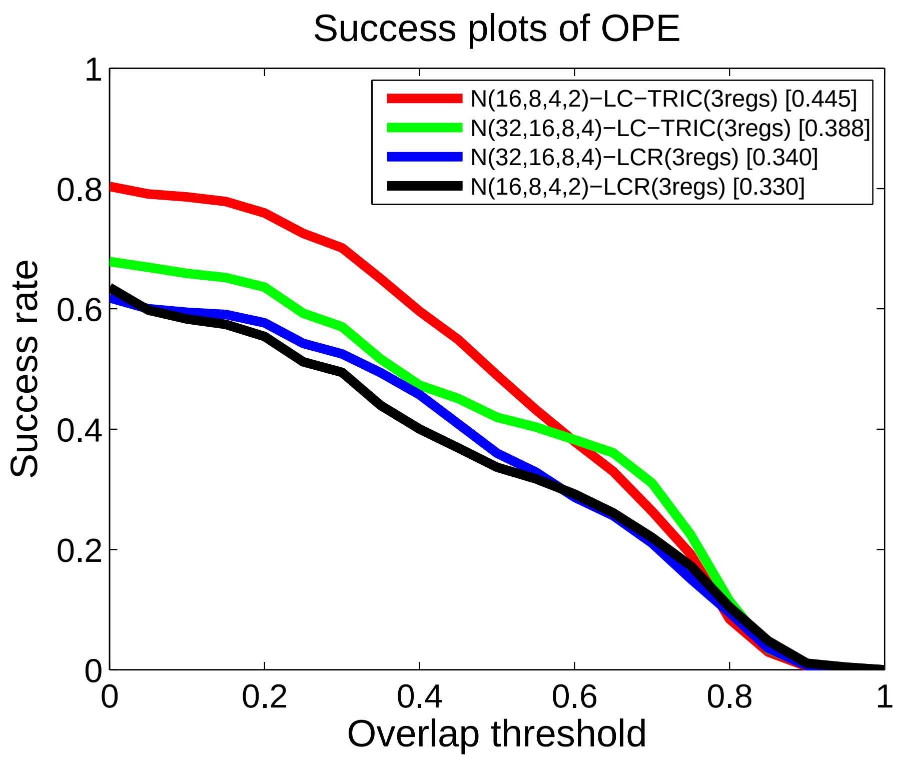


FIGURE 5.15: The comparison between Locally Continuous Regression with one regressor and Locally Continuous Regression with three regressors using different cluster number combinations. The y axis shows the AUC of the success plot of a tracker with a specific cluster number combination.

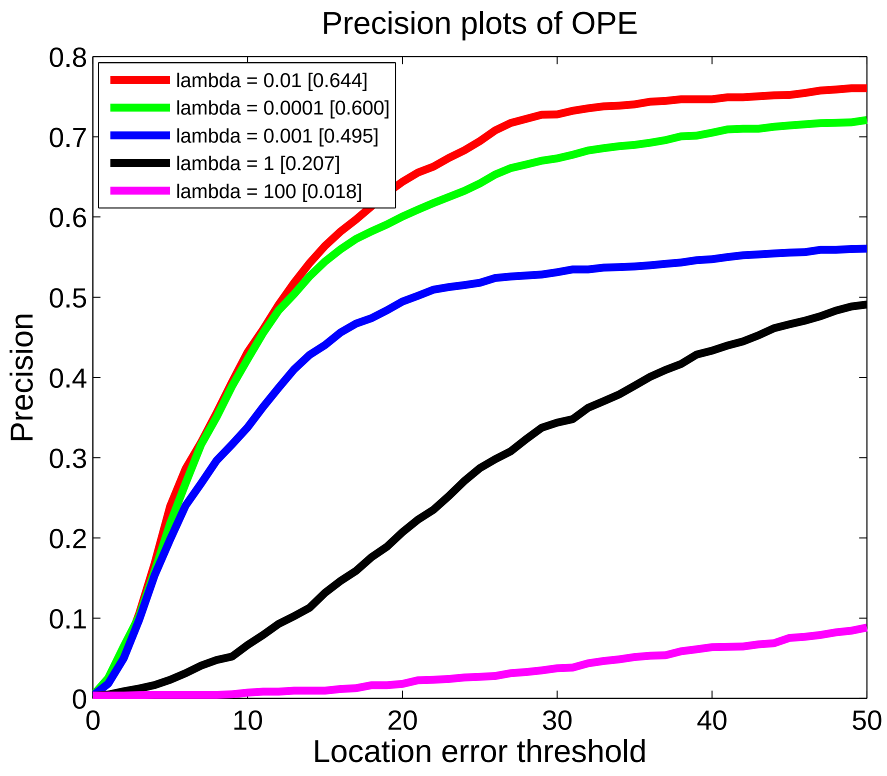


(a)

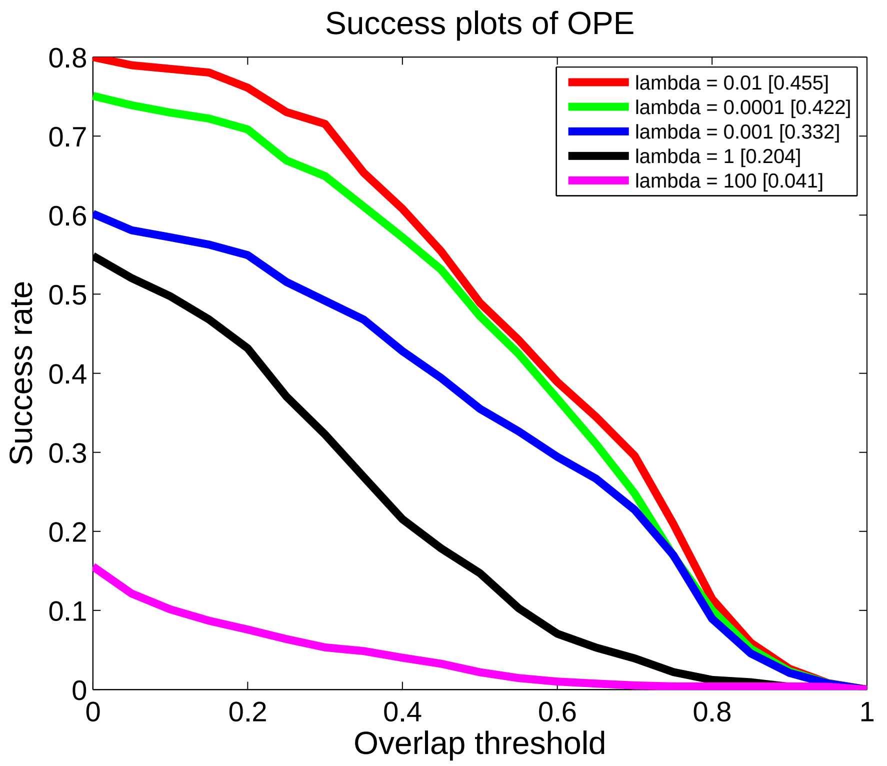


(b)

FIGURE 5.16: Precision and Success Plots of $N = [32, 16, 8, 4]$ and $N = [16, 8, 4, 2]$ tested on Locally Continuous Regression and the LC-TRIC tracker, respectively.

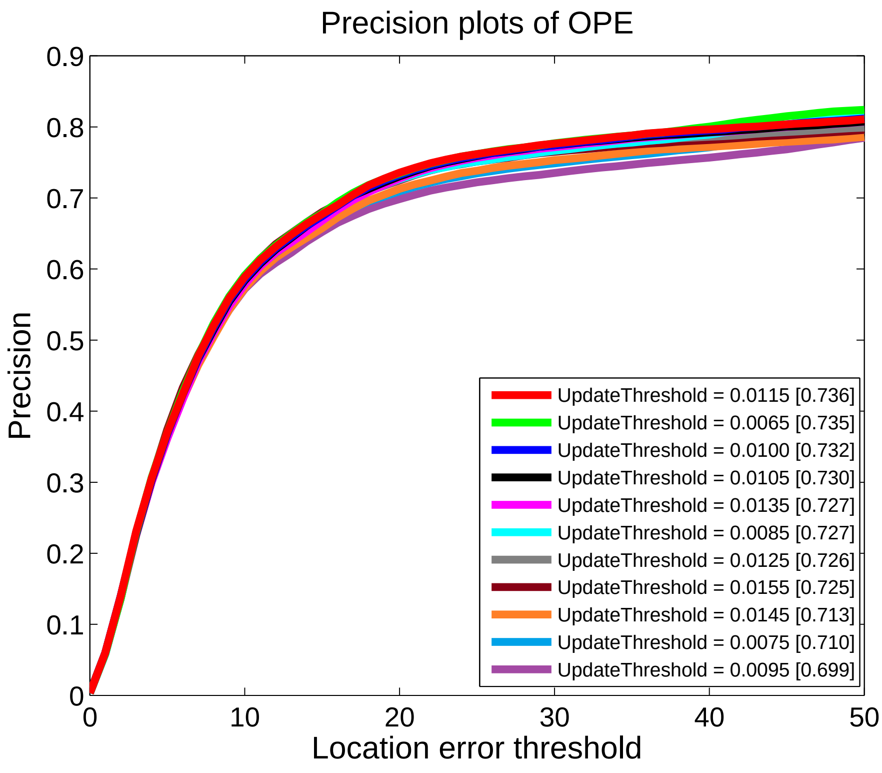


(a)

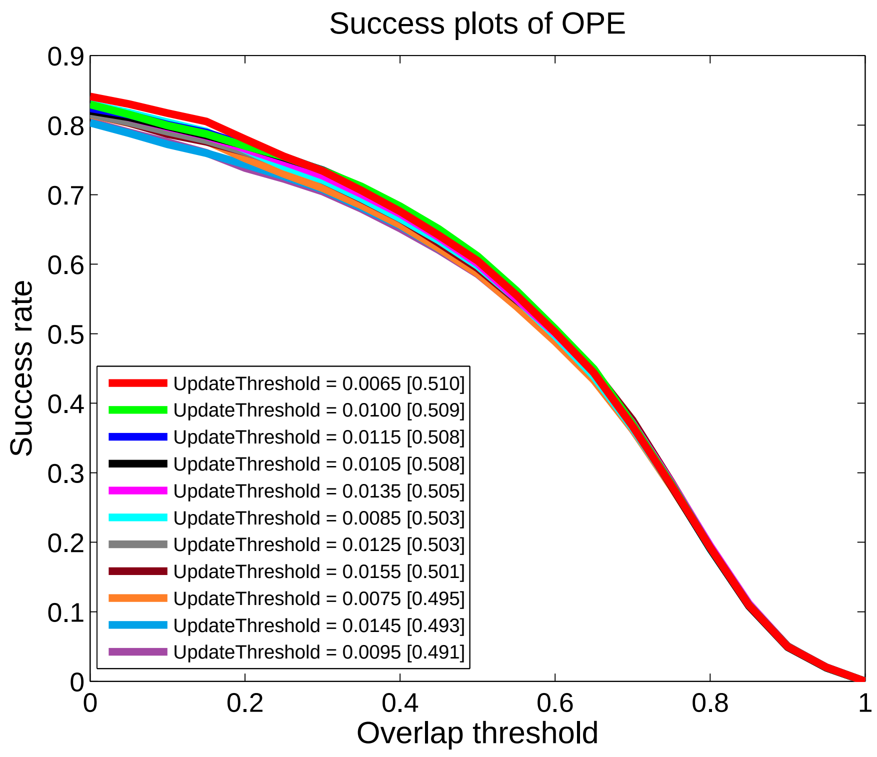


(b)

FIGURE 5.17: The effect of regularisation parameter in regression on LC-TRIC.

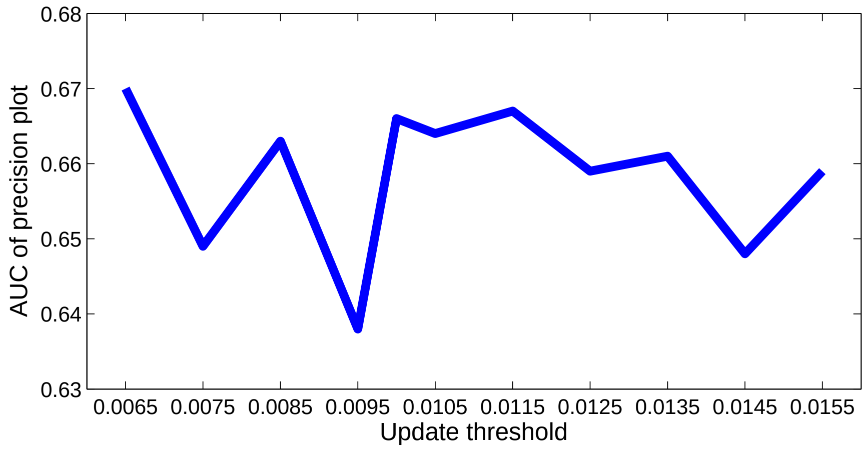


(a)

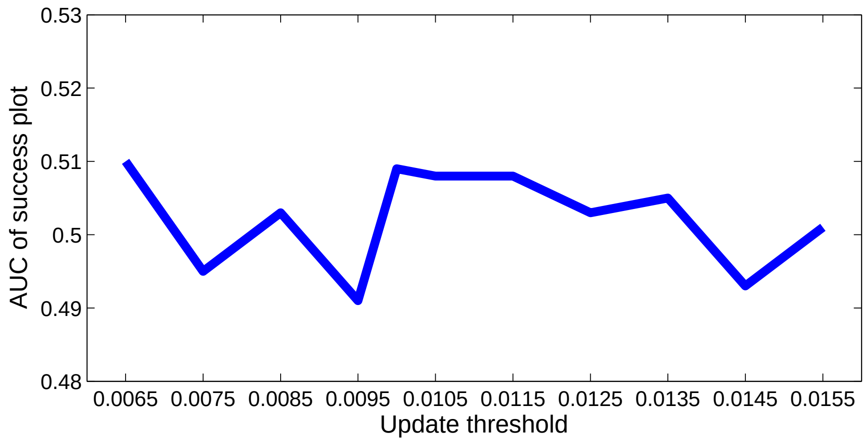


(b)

FIGURE 5.18: Effect of updating threshold on LC-TRIC.



(a)



(b)

FIGURE 5.19: AUC of precision plot and AUC of success plot of LC-TRIC tracker vs its update threshold, respectively.

5.4.2 External Evaluation

LC-TRIC, which integrates the proposed Locally Continuous Regression into the main framework of TRIC-track and thus contains Cascaded Locally Continuous Regression, the implicit shape model and Cascaded Locally Continuous Incremental Updating, is evaluated on the recently published VOT2015 dataset (Kristan et al., 2015). LC-TRIC is compared with its sampling-based counterpart. The sampling-based counterpart of LC-TRIC is the same framework of LC-TRIC, as explained in the last fourth paragraph from the end of Section 5.2, with zero mean and zero covariance of clusters of sample space, i.e. $N = [203, 203, 203, 203]$. It is the same as the main framework of TRIC-track, which includes Cascaded Regression, the implicit shape model and Cascaded Incremental Updating.

The dataset consists of 60 sequences which are selected from a large pool of sequences combined from existing datasets CVPR2013 benchmark (Wu et al., 2013) (50 sequences) and ALOV (Smeulders et al., 2014) (315 sequences), PTR (Vojir et al., 2014) and other sources, which makes sure that the VOT2015 dataset is a representative set of challenging sequences. The ground truth provided by the dataset is slightly more flexible; rotated bounding boxes described by the coordinates of their four corner points. However, the bounding boxes are still not able to fully exclude background pixels. The dataset tries to make the bounding box contain at most about 30% background pixels. Each frame of the dataset is labelled with five visual attributes, which are occlusion, illumination change, motion change, size change and camera motion. Any frame which doesn't show any of these five attributes is labelled as unassigned.

The main difference between the evaluation mechanism of the VOT2015 and that of the CVPR2013 benchmark is that VOT2015 allows for re-initialisation. Specifically, in CVPR2013 benchmark, given the initial target position in the first frame, the tracker must run through the whole video without re-initialisation. It may cause bias in performance when a tracker loses the target at an early stage of the video. In this case, even though the tracker is able to track the target in the late frames of a video, because of the loss of the target in an earlier frame, the tracker usually does not have a chance to get back to the target. In VOT2015, to avoid this kind of bias in performance, re-initialisation happens when the overlap between the estimated bounding box and the ground truth bounding box of the target reduces to zero. In practice, to reduce the bias in the robustness measure, the tracker is re-initialised five frames after the overlap becomes zero. Ten frames after re-initialisation are ignored in the computation of accuracy scores to further reduce the bias in accuracy measure (Kristan et al., 2015).

Performance on the VOT2015 dataset is measured by two weakly correlated measurements, *accuracy* and *robustness* (Kristan et al., 2015). The accuracy represents how well the predicted bounding box overlaps with the groundtruth bounding box. While robustness means the number of tracking failures when the tracker loses the target. Each tracker evaluated on the VOT2015 dataset is performed on each sequence 15 times to guarantee good statistics. A per-frame accuracy is acquired as an average over these runs. A per-sequence accuracy is obtained by averaging the per-frame accuracy. A per-sequence robustness is acquired by averaging the failure rates over different runs. The failure rate is obtained by dividing the length of video frames into the number of failures in a video sequence.

The VOT2015 benchmark (Kristan et al., 2015) also introduces the *expected average overlap* as a new metric to rank tracking algorithms; it combines the raw values of per-frame accuracies and failures in a principled manner and it provides a clear interpretation of the accuracy and the robustness. The expected average overlap estimates how accurate the estimated bounding box is after a certain number of frames are processed since initialisation. Specifically, consider a short-term tracking example on a video with N_s frames. A tracker is initialised in the beginning of the sequence and performs tracking till the end of the sequence. If a tracker drifts off the target, it will remain off the target till the end of the sequence. In this situation, the tracker's performance can be measured by calculating the average of per-frame overlaps, O_i, which includes the zero overlaps after the failure, i.e.:

$$O_{N_s} = \frac{1}{N_s} \sum_{i=1:N_s} O_i, O_i \in [0, 1]. \tag{5.30}$$

By averaging the average overlaps on a very large set of N_s frames long sequences, the expected average overlap $\hat{O}_{N_s} = \langle O_{N_s} \rangle$ is obtained (Kristan et al., 2015). This measure is evaluated for various sequence lengths ($N_s = 1 : N_{max}$), which results in the *expected average overlap curve*. In practice, as the VOT protocol re-initialises a tracker after each failure, a set of tracking segments are generated per sequence. The segments of all sequences are used to estimate the $\hat{O}_{N_s}$. All segments which have less than N_s frames and do not finish with a failure are removed, and the remaining segments are then converted into N_s-frame-long tracking instances by trimming or padding with zero overlaps to the size N_s (Kristan et al., 2015). An average overlap on each segment is computed and the average over all segments is the estimate of $\hat{O}_{N_s}$. Performing this process for different values of N_s gives an estimate of the *expected average overlap curve*. The *expected average overlap measure*, $\hat{O}$, is calculated by averaging the *expected average overlap curve* values over an interval $[N_{lo}, N_{hi}]$ of typical short-term sequence lengths in the VOT2015 dataset. Specifically, it is computed by:

$$\hat{O} = \frac{1}{N_{hi} - N_{lo}} \sum_{N_s = N_{lo}:N_{hi}} \hat{O}_{N_s}. \tag{5.31}$$

A batch kernel density estimate (KDE) (Kristan, 2013) is applied to estimate the sequence length pdf from the lengths of sixty sequences of the VOT2015 dataset, generating the range values $[N_{lo} = 108, N_{hi} = 371]$ (Kristan et al., 2015).

For the VOT2015 benchmark, 62 state-of-the-art trackers are measured. The results are shown in Table 5.1. In the table, the raw values of weighted mean accuracy A, the weighted mean robustness R and the expected average overlap $\hat{O}$ are displayed. The weighted mean accuracy is calculated by:

$$A = \frac{\sum_{i=1}^{60}(A_i * L_i)}{\sum_{i=1}^{60} L_i}, \tag{5.32}$$

where A_i is per-sequence accuracy and L_i is the length of the i_{th} video. The weighted mean robustness is computed following the same method.

The expected average overlap curves of the proposed LC-TRIC tracker and its corresponding sampling-based tracker are shown in Figure 5.20(a). The expected average overlap $\hat{O}$ is shown in Figure 5.20(b). The results show that the performance of both trackers are very close to each other. The AR raw plots of LC-TRIC tracker and its sampling-based counterpart are shown in Figure 5.21. The value in 'Accuracy' dimension is the weighted mean accuracy A, and the value in 'Robustness' dimension is the weighted mean of per-sequence failure rate scaled by 100 as in VOT2015 benchmark (Kristan et al., 2015). Specifically, the value in 'Robustness' dimension is calculated by:

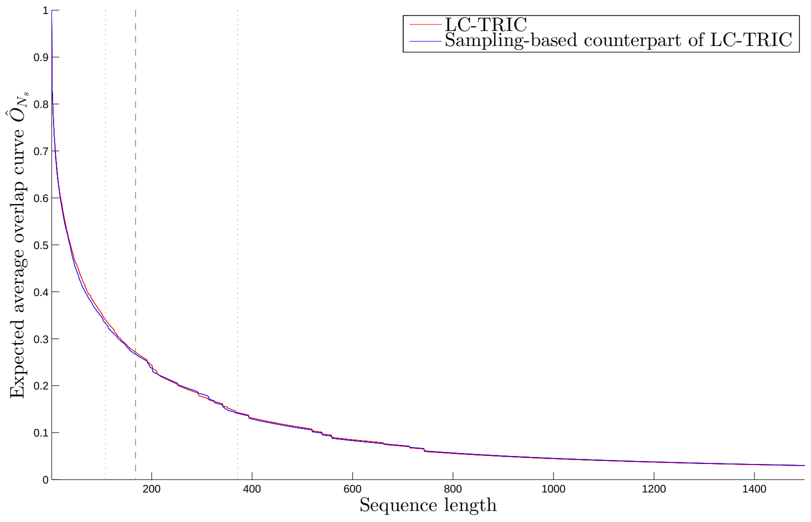
$$exp(-100 * (\sum_{i=1}^{60} R_i / \sum_{i=1}^{60} L_i)), \tag{5.33}$$

where R_i is the per-sequence robustness value and L_i is the length of the i_{th} video. The results in AR plots show that both the robustness and the accuracy of the LC-TRIC are very close to those of its sampling-based counterpart.

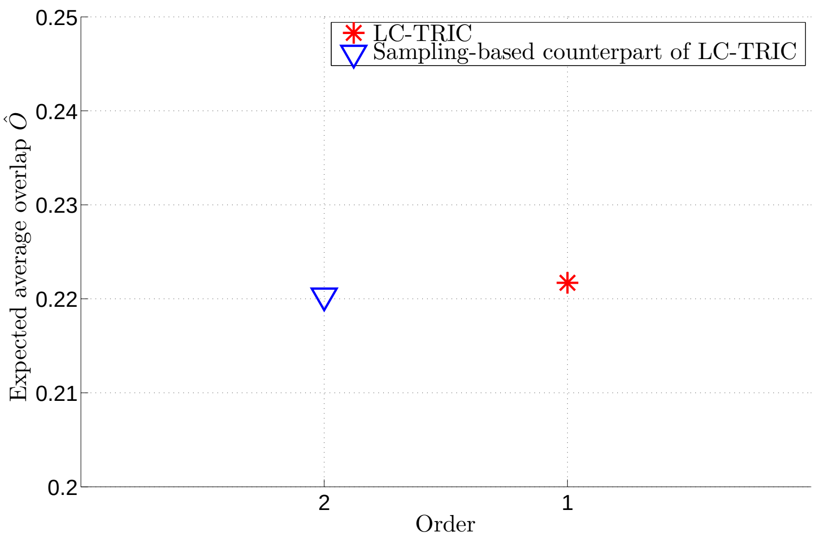
In addition to the accuracy, robustness and expected average overlap measurements for tracking, the tracking speed is also measured in VOT2015 benchmark (Kristan et al., 2015). The tracking speed is measured in *equivalent filter operations* (EFO) units. Specifically, the tracking speed is reported by dividing the measured tracking time with the time required for a filtering operation. The filter operation is to perform a maximum pixel value filter on a grayscale image of size 600×600 with a 30×30 pixel window. This operation can reduce the influence of hardware on tracking speed measurement.

Table 5.1: The table shows raw weighted mean accuracy (A), weighted mean robustness (R), expected average overlap (Φ), tracking speed (in EFO) and implementation details (M is Matlab, C is C or C++, G is GPU). 'LC-TRIC' is the proposed tracker. 'Sampling' is the sampling-based counterpart of LC-TRIC.

Trackers	A	R	Φ	Speed	Impl.
MDNet*	0.6	0.69	0.38	0.87	M C G
DeepSRDCF*	0.56	1.05	0.32	0.38	M C
EBT	0.47	1.02	0.31	1.76	M C
SRDCF*	0.56	1.24	0.29	1.99	M C
LDP*	0.51	1.84	0.28	4.36	M C
sPST*	0.55	1.48	0.28	1.01	M C
SC-EBT	0.55	1.86	0.25	0.8	M C
NSAMF*	0.53	1.29	0.25	5.47	M
Struck*	0.47	1.61	0.25	2.44	C
RAJSSC	0.57	1.63	0.24	2.12	M
S3Tracker	0.52	1.77	0.24	14.27	C
SumShift	0.52	1.68	0.23	16.78	C
SODLT	0.56	1.78	0.23	0.83	M C G
DAT	0.49	2.26	0.22	9.61	M
LC-TRIC	0.48	2.29	0.22	0.44	M C
MEEM*	0.5	1.85	0.22	2.7	M
RobStruck	0.48	1.47	0.22	1.89	C
OACF	0.58	1.81	0.22	2	M C
MCT	0.47	1.76	0.22	2.77	C
HMMTxD*	0.53	2.48	0.22	1.57	C
Sampling	0.48	2.46	0.22	0.07	M C
ASMS*	0.51	1.85	0.21	115.09	C
MKCF+	0.52	1.83	0.21	1.23	MC
AOG	0.51	1.67	0.21	0.97	binary
SME	0.55	1.98	0.21	4.09	M C
MvCFT	0.52	1.72	0.21	2.24	binary
SRAT	0.47	2.13	0.2	15.23	MC
Dtracker	0.5	2.08	0.2	10.43	C
SAMF*	0.53	1.94	0.2	2.25	M
G2T	0.45	2.13	0.2	0.43	M C
MUSTer	0.52	2	0.19	0.52	M C
TGPR*	0.48	2.31	0.19	0.35	M C
HRP	0.48	2.39	0.19	1.01	M C
KCFv2	0.48	1.95	0.19	10.9	M
CMIL	0.43	2.47	0.19	5.14	C
ACT*	0.46	2.05	0.19	9.84	M
MTSA-KCF	0.49	2.29	0.18	2.83	M
LGT*	0.42	2.21	0.17	4.12	M C
DSST*	0.54	2.56	0.17	3.29	M C
MIL*	0.42	3.11	0.17	5.99	C
KCF2*	0.48	2.17	0.17	4.6	M
sKCF	0.48	2.68	0.16	66.22	C
BDF	0.4	3.11	0.15	200.24	C
KCFDP	0.49	2.34	0.15	4.8	M
PKLTF	0.45	2.72	0.15	29.93	C
HoughTrack*	0.42	3.61	0.15	0.87	C
FCT	0.43	3.34	0.15	83.37	C
MatFlow	0.42	3.12	0.15	81.34	C
SCBT	0.43	2.56	0.15	2.68	C
DFT*	0.46	4.32	0.14	3.33	M
FoT*	0.43	4.36	0.14	143.62	C
LT-FLO	0.44	4.44	0.13	1.83	M C
L1APG*	0.47	4.65	0.13	1.51	M C
OAB*	0.45	4.19	0.13	8	C
IVT*	0.44	4.33	0.12	8.38	M
STC*	0.4	3.75	0.12	16	M
CMT*	0.4	4.09	0.12	6.72	C
CT*	0.39	4.09	0.11	12.9	M
FragTrack*	0.43	4.85	0.11	2.08	C
ZHANG	0.33	3.59	0.1	0.21	M
LOFT-Lite	0.34	6.35	0.08	0.75	M
NCC*	0.5	11.34	0.08	154.98	C
PTZ-MOSSE	0.2	7.27	0.03	18.73	C



(a)



(b)

Figure 5.20: (a) Expected average overlap curves. (b) Expected average overlap graph with trackers ranked from right to left.

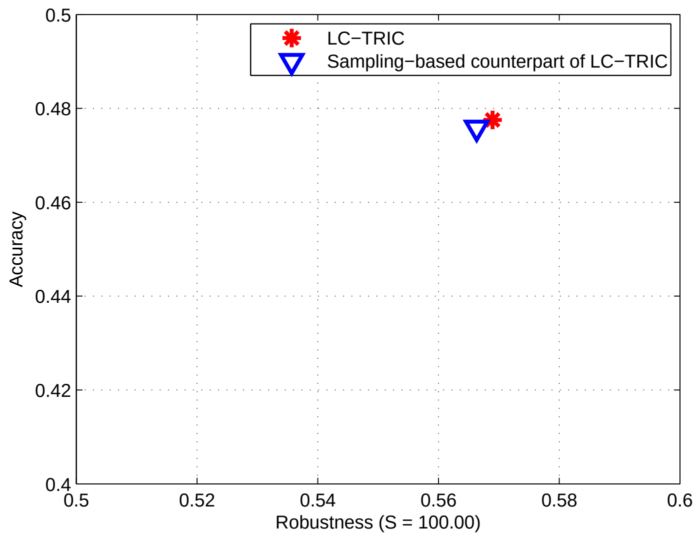


FIGURE 5.21: AR raw plots of LC-TRIC tracker and its sampling-based counterpart.

The tracking speed in EFO units of LC-TRIC and its sampling-based counterpart are shown in Figure 5.22. It shows that the tracking speed of LC-TRIC is six times higher than that of sampling-based counterpart of LC-TRIC.

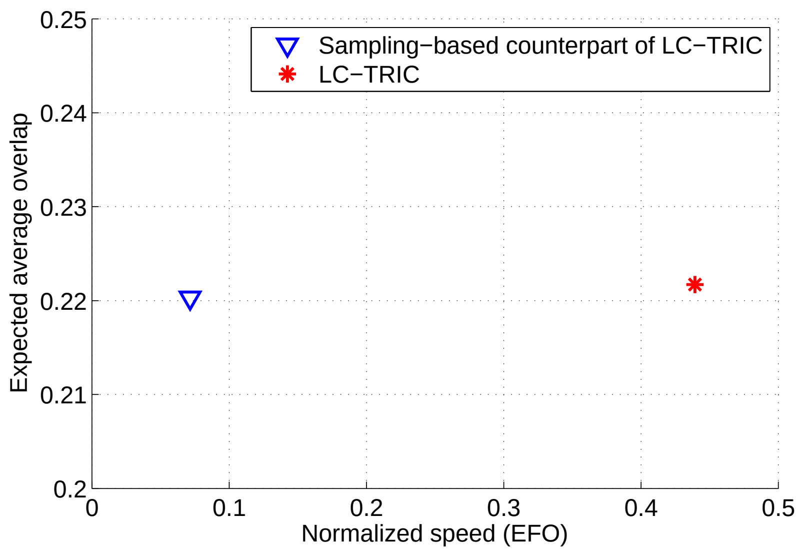


FIGURE 5.22: Expected average overlap scores vs the tracking speed in EFO units.

5.4.3 Conclusion

To address the problems of SDM in TRIC-track, this Chapter has introduced Continuous Regression to visual tracking. However, it is found that the Taylor expansion is not able to accurately approximate image features of sample space with a high variance in visual tracking. This problem is alleviated by Locally Continuous Regression strategy, proposed in this Chapter. It unifies sampling-based regression with Continuous Regression in an efficient manner. The Locally Continuous Regression is integrated into the main framework of TRIC-track (explained in Chapter 4) and shows six-fold speed improvement without sacrificing the performance, compared to its sampling-based counterpart.

Chapter 6

Conclusion

This thesis proposed an online part-based visual tracking system, based on direct displacement prediction technique rather than the traditional local matching of an appearance model. The method employs cascaded regression to directly predict parts' locations from local image features, learning the inference models on-the-fly. Spatial relationships between parts are implicitly captured by linking regressors in groups of parts. The proposed tracker is further improved by replacing the sampling-based regression with proposed Locally Continuous Regression, which brings about a computational efficiency without sacrificing the tracker's performance.

This chapter is organised as follows: Section 6.1 summarises contributions made in this thesis; Section 6.2 gives a complete summary of the work presented in this thesis; Section 6.3 presents some limitations of the proposed tracker and suggests future work to improve the tracker's performance.

6.1 Contributions

This thesis makes the following contributions:

- The thesis proposes a visual tracking method in which the local fitness-based approach is replaced by direct displacement-based tracking. Specifically, the tracker predicts the two-dimensional displacement vector between the centre of a sampled image patch and the target (part) location using regressors (see Figure 1.5). In doing so, local patches around a target contribute to the solution by directly 'voting' for the target (part) location.

- This thesis implicitly models the shape of a target using local evidence from multiple neighbouring parts and global information from a bounding box part. While template-based approaches need to model part appearance and shape fitness separately, this direct displacement prediction by regression tracker implicitly learns the shape and possible deformations of an object. It does so by tracking each part using not only the local evidence for that part, but also evidence provided by neighbouring parts and the object as a whole.

- The thesis adapts the framework of the supervised descent method (SDM) (Xiong & De la Torre, 2013) to the problem of online tracking of generic objects. While SDM has been used for what is essentially structured object detection, it has never been used for online model-free tracking. The key difference between detection of a known object and generic object tracking is that the appearance and structure models of the former can be learned offline on potentially hundreds of thousands of images, while the models for the latter must be initialised on a single frame. It is shown that it is possible to learn the cascade models on-the-fly without strong supervision.

- This thesis introduces Continuous Regression (Sánchez-Lozano et al., 2016) to replace sampling-based regression (SDM) in model-free tracking. With Continuous Regression, the shape displacement is regarded as a continuous variable, the feature space is approximated by its first-order Taylor expansion. Only the feature in the ground truth target location needs to be sampled. However, it is then observed that the Taylor expansion is only a good feature approximation in a relatively small region around the target. This region is too small to enable tracking. Therefore, this thesis proposes Locally Continuous Regression, which unifies sampling-based regression with continuous regression by repeating Continuous Regression in a few regions around the target in an efficient manner. It shows six times speed improvement without sacrificing performance of the tracker.

6.2 Summary

The goal of this thesis is to develop a robust part-based visual tracking method, which can address four main problems of current part-based trackers. The four problems addressed are as follows:

- Current part-based trackers rely on a response map estimating the likelihood that any given location in an image represents the target (part) (Adam et al., 2006; Shahed Nejhum et al., 2008; Kwon & Lee, 2009; L. Zhang & van der Maaten, 2013; T. Zhang et al., 2014; Liu et al., 2015). A template likelihood strategy's view of the image as a set of independent, identically distributed target locations introduces a number of inherent drawbacks (explained in Chapter 1).

- The spatial information utilised with the part-based models is limited and inflexible. Thus, they still cannot handle non-rigid object tracking situations which have severe or complex movement. An extra component, shape, is necessary for general object tracking.
- Another major issue is that there is no way of jointly learning shape and appearance for current part-based trackers. Template likelihood approaches to part-based tracking cannot directly use the appearance of one part to determine the location of another.
- Parts can have separate motions over time. This situation is not considered in current part-based trackers. Thus, more complex motion models are required, with parts' motion having separate factors.

In Chapter 2, the related concepts and main challenges of visual tracking were introduced. It explained three main components of tracking, including appearance model, motion model and search strategy. Especially, the state-of-the-art part-based model-free tracking methods were presented in Section 2.5, which gave a theoretical analysis of the algorithm components, advantages and drawbacks of current part-based methods.

After theoretical analysis of the advantages and problems of current part-based models in tracking, practical experiments on part-based models were performed in Chapter 3. Specifically, two experiments were introduced in Section 3.1. First, three different initialisation methods, including bounding-box, Hessian matrix and the stick figure initialisation methods, were combined with a standard MCMC method and evaluated. This experiment confirmed the benefit of the stick figure initialisation which defined the parts used in tracking. The second experiment was performed by examining the stick figure initialisation and bounding box initialisation with MCMC and A-BHMC (part-based tracker) respectively. The similar performances of B-MCMC and SF-MCMC had shown that stick figure initialisation had competitive performance. However, from the experimental results, some problems which needed to be addressed were identified as well. First, the parts were not initialised to be a proper size. Second, the parts lacked an explicit relationship between each other. Thirdly, the updating of the appearance model sometimes introduced error to the template. It was concluded that seeking an appearance model, which is robust to the variation of target's appearance and accurate to describe the target, is critical to correct tracking.

In Section 3.2, inspired by LEAR, the direct displacement prediction (DDP) method was explored by qualitative and quantitative experiments. This technique was further evaluated in real tracking scenes (Section 3.3), which showed proof of concept of the simple DDP tracker and its potential to be improved. First, the accuracy of the DDP tracker needed to be improved. If the predicted target location is more accurate, it will be less possible for the tracker to include incorrect target information during updating. Second, the DDP tracker tracked all three parts of the target separately. It happened frequently that the regressor for one part was reliable while the regressor for another part was not working. The shape model of a target would provide spatial constraints to the prediction of target location, making the prediction more robust in tracking with challenging conditions. Third, there was no motion model in the simple DDP tracker.

Based on the work of the simple DDP tracker, Chapter 4 gave a detailed introduction of the proposed part-based tracking method, Tracking by Regression with Incrementally Learned Cascades (TRIC-track). It is a part-based tracker employing direct displacement prediction rather than the traditional local matching of an appearance model. The method employed cascaded regression (SDM) to directly predict parts' locations from local image information described by deep learned features, learning the inference models on-the-fly. Spatial relationships between parts and spatial relationships between local parts and the global part were captured implicitly by a set of regressors. This thesis integrated a multiple temporal scale motion model to initialise the cascaded regression search close to the target and to cope with occlusions. Automatic initialisation and shape correction were added to complete the proposed tracker. Experimental results clearly demonstrated the value of the method's components, and comparison with the state-of-the-art techniques in the CVPR 2013 Visual Tracker Benchmark showed that TRIC ranks first on the full dataset.

TRIC has successfully adapted the framework of SDM (Xiong & De la Torre, 2013) to the problem of online tracking of generic objects and the cascaded regression (SDM) has optimised the accuracy of the direct displacement prediction compared to ordinary linear regression avoiding calculation of the Hessian matrix or Jacobian matrix. However, there are three major drawbacks of the sampling-based regression (i.e. SDM). First, the samples used for training SDM were obtained through random sampling, whichs is computationally expensive as it needs to be conducted at every cascade level for each target point. Second, similarly to training, the updating of SDM is time consuming. Third, each regressor is learned and updated with a limited set of training image patches and their corresponding displacements. Not all available image information and displacements within a sample space can be utilised to train and update a regressor. To address the problems of SDM in TRIC-track, Chapter 5 introduced Continuous Regression to visual tracking. It found that the Taylor expansion was not able to accurately approximate image features of sample space with a high variance in visual tracking. This problem was alleviated by Locally Continuous Regression strategy, proposed in this Chapter. It unified sampling-based regression with Continuous Regression in an efficient manner. The Locally Continuous Regression was integrated into the main framework of TRIC-track (explained in Chapter 4), and showed six-fold speed improvement without sacrificing performance, compared to its sampling-based counterpart.

6.3 Limitations and Future Work

This thesis has proposed a robust part-based visual tracking method, which addresses four main problems of current part-based trackers and yields state-of-the-art results. However, the current proposed tracker still has some limitations within its different algorithm components. This section discusses these limitations and potential work that could be done to improve the tracker's performance.

- Currently target parts are automatically initialised by a segmentation method given a bounding box enclosing the whole target object. Imperfect segmentation result of target may introduce unnecessary branches to the target skeleton, which further leads to uneven parts. In this case, the initialised parts are not able to completely cover the whole object. This problem might be solved by the following approaches: 1) improve the segmentation result by removing unnecessary small target areas and retaining the main target shape only; 2) instead of locating equally distant points along the skeleton as the target parts' locations, a feature detection technique can be utilised in the selection of target parts. Both these methods intend to provide a better representation of target parts of a whole object, which is robust for tracking throughout a video.

- In this thesis, target parts are represented by six target points forming an implicit shape model. The number six is used to balance flexibility of target shape and tracking efficiency. However, six points are limited when representing a target shape with a high degree of freedom in its movement. For example, when tracking human, six points usually just represent the main body's movement rather than cover the movements of all arms and legs. The obvious method to improve the flexibility of the shape model is to increase the number of target points and use the original implicit shape model. However, the kind of targets in visual tracking can be diverse and their structures can be various accordingly. The original implicit shape model, in which one part is constrained by its two adjacent parts, may not be flexible enough to describe different degrees of freedom of more different parts.

In this case, a hierarchical implicit shape model, which determines the constraint relationship of a target part based on the level of the target part located at in the target skeleton, may provide more practical spatial constrains than the proposed implicit shape model.

- During updating in LC-TRIC, the centres of the clusters around a part are assumed to have the same displacements to the target part location in the current frame as those in the first frame. The statistics of each partition are also kept the same as that in the first frame. It can be expected that target appearance keeps changing over time, and the statistics of each partition might change as well. However, the target variation between two adjacent frames is not given. Updating with a changeable statistics of each partition needs to be very careful, as it may introduce incorrect updating which would deteriorate the regressor rather than improve the regressor based on current frame information. This problem can be explored in future work.

- The top ranking motion prediction is used to initialise the regression search, where the ranking is determined by a Support Vector Machine (SVM) trained to distinguish between foreground and background patches currently. As SVM is a template-based approach, it has the drawbacks of a template-based method. It can happen that a local optima is found instead of a global optima. The multiple temporal motion model provides a rich description of target motion, which usually includes a good initial position (the closest position to the target). However, this optimal position is not selected sometimes. We have shown that displacement-based regression works well when test locations are sampled relatively close to the target. If the optimal initial position would always be selected, the tracker's performance will be improved accordingly. Given the confidence map of the predicted target location in the previous frame, the problem of motion model selection in current frame might be improved by considering the likelihood of motion predictions. A high likelihood means the motion prediction is likely to be close to the target.

- There is no failure detection system in the proposed tracker. Occlusion is a major challenge in visual tracking. A target can be partially or fully occluded for a period of time in a video. Moreover, a target may be out of view and re-appear over time. These two problems can be addressed by a failure detection mechanism of a target. When the target is out of view or fully occluded, this situation can be detected, and the regressors should stop updating. Although the target is out of view, the target motion should be predicted or the search of the target should be maintained. When the target re-appears, it can then be re-captured.

Appendix A

Preliminary Experimental Results

Table A.1: Average centre error over a whole video sequence of B-MCMC tracker for each video each run. The figure in bold is the best tracking result of the corresponding method for each video. μ is the mean of the average centre error of different runs for each video. σ is the standard deviation of the average centre error of different runs for each video.

B-MCMC							
	1st	**2nd**	**3rd**	**4th**	**5th**	μ	σ
Basketball	18.4846	60.6795	81.7335	**15.967**	53.2302	46.01896	56.60144
Bolt	27.2266	101.5059	61.4204	**15.0166**	111.4539	63.32468	86.10805
Boy	**34.9404**	57.961	44.4395	48.9357	42.0318	45.66168	17.07654
CarDark	**22.3263**	24.7789	28.2403	24.0145	29.6763	25.80726	6.105551
CarScale	**34.7037**	38.7559	40.8635	41.0105	35.2129	38.1093	6.033897
Coke	62.4238	54.9384	**51.4838**	91.2753	99.4141	71.90708	43.89468
Couple	111.6141	135.5933	106.2408	**19.3326**	21.0535	78.76686	109.2074
Crossing	15.1708	14.4431	14.26	13.7706	**13.4282**	14.21454	1.335722
David	16.3985	16.6949	**15.5634**	16.0205	16.2418	16.18382	0.849851
David3	**29.4512**	136.9427	98.4663	96.7945	90.615	90.45394	77.37548
Deer	93.1164	82.1981	108.1672	**76.7905**	96.0482	91.26408	24.57053
Doll	30.9306	29.5039	35.3735	29.9704	**28.9586**	30.9474	5.155986
FaceOcc1	19.1253	19.1404	19.0874	19.0907	**18.9106**	19.07088	0.184797
Football1	25.022	34.9512	26.0263	**16.7196**	23.352	25.21422	13.08173
Girl	29.7838	30.6349	25.5819	30.3749	**22.8906**	27.85322	6.900018
Ironman	**126.8892**	168.9442	136.0867	162.9487	153.7576	149.7253	35.60875
JoggingLeft	21.2818	84.606	**20.4446**	22.0449	22.2676	34.12898	56.45316
JoggingRight	129.6905	128.1224	**106.639**	109.3512	115.9544	117.9515	21.14657
Lemming	121.3249	**106.4191**	132.2415	133.0117	134.6712	125.5337	23.82207
Liquor	96.9188	85.8126	95.3124	**80.0362**	84.741	88.5642	14.50009
Matrix	58.2867	**54.4839**	90.1259	56.9744	75.6391	67.102	30.69978
MotorRolling	137.2827	**135.9162**	138.1016	141.3464	139.1714	138.3637	4.095899
MountainBike	15.6451	15.7463	**15.4108**	15.4308	19.6608	16.37876	3.680438
Shaking	119.7842	62.6358	76.816	79.9077	**46.1249**	77.05372	54.75914
Singer1	**71.6088**	75.4469	74.8296	81.7254	78.5853	76.4392	7.710894
Singer2	191.9331	**36.4459**	182.4991	110.2354	36.8293	111.5886	150.734
Skating1	84.4211	212.7842	**83.6129**	146.6898	103.2833	126.1583	109.5489
Skiing	210.3809	239.3093	232.7703	273.8673	**187.9734**	228.8602	64.52384
Soccer	46.7028	50.3568	46.8675	**38.9799**	89.1747	54.41634	39.74155
Subway	139.8743	**139.064**	139.617	139.2792	139.728	139.5125	0.666214
Tiger1	59.8304	62.5591	60.2633	**57.2751**	59.2214	59.82986	3.812302
Tiger2	154.9796	76.547	**71.3427**	77.7805	74.9583	91.12162	71.5591
Trellis	50.0117	47.726	52.4292	52.1405	**44.666**	49.39468	6.501804
Walking	**60.5319**	62.8038	78.2871	83.9751	60.9007	69.29972	22.03985
Walking2	**45.0868**	45.2324	45.55	46.9965	45.4249	45.65812	1.537851
Woman	**21.169**	112.1144	135.0117	134.3214	131.3234	106.788	97.54812

Table A.2: Average centre error over a whole video sequence of SF-MCMC tracker for each video each run. The figure in bold is the best tracking result of the corresponding method for each video. μ is the mean of the average centre error of different runs for each video. σ is the standard deviation of the average centre error of different runs for each video.

	SF-MCMC						
	1st	**2nd**	**3rd**	**4th**	**5th**	μ	σ
Basketball	84.8935	166.3467	68.2097	78.9165	**63.8932**	92.45192	84.2856
Bolt	**13.013**	115.7899	136.7623	38.0169	21.3796	64.99234	114.2931
Boy	88.857	**52.5968**	54.6677	66.0338	72.588	66.94866	29.4858
CarDark	39.7756	**17.8251**	24.1061	20.8609	24.226	25.35874	16.96126
CarScale	**34.0069**	34.1215	34.0551	39.4509	38.7322	36.07332	5.534491
Coke	58.4316	57.5099	**49.4714**	52.3069	55.4454	54.63304	7.444378
Couple	112.6306	25.0979	**24.1675**	140.0794	139.5617	88.30742	118.3568
Crossing	18.7313	32.2215	**16.3689**	18.2451	18.2796	20.76928	12.93213
David	19.8659	**17.582**	19.3333	18.0245	19.7191	18.90496	2.072471
David3	30.5921	75.4512	48.5581	**25.9338**	85.9985	53.30674	53.35645
Deer	**73.4214**	74.8745	75.0062	76.3677	69.3692	73.8078	5.383082
Doll	76.0404	73.5744	53.5193	56.6152	**51.2235**	62.19456	23.40858
FaceOcc1	30.8877	26.2668	**22.6731**	25.7221	30.2861	27.16716	6.831224
Football1	41.4841	45.7104	32.1659	35.0626	**29.6254**	36.80968	13.31555
Girl	36.8572	36.4226	32.2576	**28.8745**	32.7083	33.42404	6.584006
Ironman	133.9112	**102.5302**	118.3444	122.6917	158.3687	127.1692	41.5195
JoggingLeft	64.2378	85.9324	**22.3909**	24.0203	52.7429	49.86486	54.20786
JoggingRight	125.9981	**16.2301**	132.1738	78.8589	100.8631	90.8248	93.56933
Lemming	**173.423**	241.47	178.6136	253.4667	267.0308	222.8008	87.38367
Liquor	**139.8184**	107.1641	98.4066	153.4594	127.901	125.3499	45.4165
Matrix	85.3967	64.0553	71.0317	**63.8054**	64.2215	69.70212	18.56833
MotorRolling	**63.6356**	82.6643	86.1868	87.3711	89.7842	81.9284	21.08646
MountainBike	**15.3911**	225.8138	17.6017	15.6871	15.9328	58.0853	187.534
Shaking	48.7236	57.8763	**34.4329**	66.8908	36.5658	48.89788	27.67135
Singer1	29.3255	34.2897	31.6106	**29.0599**	32.0934	31.27582	4.310225
Singer2	**48.7547**	70.4801	66.2133	51.3812	74.4683	62.25952	23.08642
Skating1	117.9823	67.6742	49.2576	**27.5364**	57.194	63.9289	67.23702
Skiing	**91.4363**	95.6357	166.5596	135.9009	146.8982	127.2861	65.48445
Soccer	39.1201	**33.915**	47.0522	75.8335	59.9074	51.16564	33.83078
Subway	125.8455	132.5902	136.4878	**12.4142**	119.8322	105.434	104.7779
Tiger1	58.6392	**58.1118**	61.8734	90.5025	60.9387	66.01312	27.55762
Tiger2	154.8936	**85.0203**	155.0346	157.6439	164.3499	143.3885	65.70725
Trellis	61.4361	73.2032	61.1932	**52.4055**	55.2917	60.70594	15.97383
Walking	**37.7149**	106.1706	101.3048	84.1239	100.2198	85.9068	56.3757
Walking2	42.8824	45.5097	74.9279	48.3863	**41.3009**	50.60144	27.72531
Woman	**21.2191**	110.0145	29.3871	21.6879	21.9043	40.84258	77.63133

Table A.3: Average centre error over a whole video sequence of B-A-BHMC tracker for each video each run. The figure in bold is the best tracking result of the corresponding method for each video. μ is the mean of the average centre error of different runs for each video. σ is the standard deviation of the average centre error of different runs for each video.

	B-A-BHMC						
	1st	**2nd**	**3rd**	**4th**	**5th**	μ	σ
Basketball	**47.2744**	116.3922	137.1747	121.6937	157.0405	115.9151	82.99659
Bolt	386.9082	380.9612	381.5772	**374.5496**	376.6567	380.1306	9.588219
Boy	228.6845	404.6215	**89.3345**	119.3385	108.3595	190.0677	263.3451
CarDark	107.6977	114.1419	112.6988	**103.0551**	113.7218	110.2631	9.564354
CarScale	62.5438	89.8404	91.5702	88.4637	**61.5865**	78.80092	30.64187
Coke	401.2142	127.3548	**121.386**	410.8237	394.7506	291.1059	304.6599
Couple	**159.6823**	200.4391	193.5125	196.4922	211.2073	192.2667	38.81603
Crossing	49.9284	46.0147	**26.8428**	30.5913	41.2061	38.91666	19.79828
David	58.7542	**56.8149**	60.775	59.5937	59.7454	59.13664	2.966556
David3	155.0191	195.5001	154.8645	**55.8486**	162.5313	144.7527	104.895
Deer	503.5028	500.1	**485.0268**	505.4001	506.3112	500.0682	17.477
Doll	63.8934	65.4454	**54.3458**	55.4511	57.0211	59.23136	10.16825
FaceOcc1	122.8012	135.7861	136.849	**122.4742**	139.4441	131.4709	16.3469
Football1	65.1079	104.6995	35.834	**35.7239**	65.6911	61.41128	56.74408
Girl	**28.6603**	81.4054	81.5621	81.6394	81.2896	70.91136	47.2389
Ironman	488.8336	489.8076	486.2054	481.7316	**406.6089**	470.6374	71.85883
JoggingLeft	214.171	211.6945	212.0694	**209.8648**	211.4854	211.857	3.087566
JoggingRight	180.6425	182.2148	**158.1556**	182.754	180.5372	176.8608	21.0023
Lemming	289.589	277.4688	263.493	297.0474	**226.433**	270.8062	55.7517
Liquor	**129.4573**	138.4843	147.4829	149.3731	193.2855	151.6166	49.2129
Matrix	231.7625	201.1329	207.7641	211.596	**113.9352**	193.2381	91.56104
MotorRolling	319.0109	330.7999	**311.3682**	321.8526	335.6286	323.732	19.23977
MountainBike	44.5147	**42.1508**	45.8017	44.3148	43.4538	44.04716	2.704976
Shaking	402.5567	**399.5986**	400.0365	401.2248	403.2857	401.3405	3.164011
Singer1	303.5629	**162.8304**	323.2581	301.9572	303.6145	279.0446	131.1121
Singer2	296.0887	298.3344	298.528	297.0243	**295.3361**	297.0623	2.774098
Skating1	**109.5981**	120.8071	116.9329	110.1522	127.483	116.9947	15.0357
Skiing	406.2431	410.6545	**288.2744**	404.4536	424.2314	386.7714	111.2076
Soccer	**312.5294**	345.4795	345.331	333.0382	342.0731	335.6902	27.80238
Subway	165.4007	165.7532	**162.8838**	165.2913	169.5176	165.7693	4.768697
Tiger1	453.1909	455.145	455.0304	**421.3588**	444.719	445.8888	28.73423
Tiger2	546.9368	544.191	543.1877	**496.3426**	543.0496	534.7415	43.04482
Trellis	181.5162	170.8577	**169.4938**	176.831	188.6214	177.464	15.76312
Walking	**30.69**	216.5838	225.3359	39.1251	40.7379	110.4945	201.9203
Walking2	46.464	56.9481	40.9571	43.5991	**40.2433**	45.64232	13.55623
Woman	149.5206	185.0621	141.26	**124.1749**	179.1081	155.8251	51.48324

Table A.4: Average centre error over a whole video sequence of SF-A-BHMC tracker for each video each run. The figure in bold is the best tracking result of the corresponding method for each video. μ is the mean of the average centre error of different runs for each video. σ is the standard deviation of the average centre error of different runs for each video.

	SF-A-BHMC						
	1st	**2nd**	**3rd**	**4th**	**5th**	μ	σ
Basketball	149.8325	195.3529	171.4725	330.8565	**128.3531**	195.1735	159.6635
Bolt	**32.6316**	169.1653	276.9514	172.0363	164.9751	163.1519	173.5059
Boy	411.4192	404.1092	**385.5584**	415.0583	416.5147	406.532	25.33769
CarDark	225.474	90.1548	80.3433	**74.1866**	156.4107	125.3139	129.8906
CarScale	37.3593	63.5929	**37.0882**	167.3969	181.264	97.34026	142.5414
Coke	406.3599	**396.9097**	402.4861	404.1892	401.9851	402.386	7.014557
Couple	180.4316	156.3013	**141.702**	185.1192	173.315	167.3738	36.08801
Crossing	**16.7707**	18.1739	18.2924	18.6338	94.9966	33.37348	68.91143
David	58.4518	70.4043	**55.11**	58.3463	59.1723	60.29694	11.72764
David3	210.2453	262.8496	264.0817	**154.5532**	262.3541	230.8168	96.78109
Deer	129.2965	122.8748	**103.5247**	132.3766	133.937	124.4019	24.83265
Doll	65.286	56.7438	58.4684	66.5776	**55.2826**	60.4717	10.26142
FaceOcc1	**69.8404**	70.0733	76.6306	70.8803	82.0963	73.90418	10.71812
Football1	144.5192	**21.0013**	140.5107	45.8859	142.0186	98.78714	120.6251
Girl	73.442	80.2422	**38.4857**	80.6707	82.0218	70.97248	36.92641
Ironman	**409.6376**	443.7865	493.2135	475.3752	502.8597	464.9745	76.49842
JoggingLeft	**209.8278**	220.0015	217.4724	264.3414	214.68	225.2646	44.33671
JoggingRight	174.6023	173.7487	174.125	**143.4866**	173.3894	167.8704	27.27677
Lemming	263.8233	**187.952**	293.4402	252.8347	262.5945	252.1289	77.9236
Liquor	183.4837	193.3745	151.2871	195.0397	**147.3667**	174.1103	46.1858
Matrix	575.478	574.3926	576.1731	**571.9042**	572.2366	574.0369	3.81525
MotorRolling	333.6467	322.1649	**316.2831**	318.3486	342.8908	326.6668	22.56586
MountainBike	34.967	30.1088	**27.0054**	29.69	34.5597	31.26618	6.820664
Shaking	397.573	**396.7803**	403.9986	399.6514	399.3354	399.4677	5.602884
Singer1	57.5251	56.9495	**56.8678**	57.0715	57.2086	57.1245	0.516606
Singer2	296.3571	294.763	295.6037	198.4253	**95.8147**	236.1928	178.0794
Skating1	106.3341	84.3954	105.5664	**76.4398**	91.23	92.79314	26.20899
Skiing	403.3579	436.2899	429.7671	439.8668	**192.9714**	380.4506	211.5505
Soccer	321.3901	343.5014	333.5409	**307.9258**	310.8809	323.4478	30.11829
Subway	158.5811	149.2787	160.1787	**142.6239**	151.9939	152.5313	14.28662
Tiger1	456.115	455.5111	**446.3521**	451.7226	482.41	458.4222	27.92462
Tiger2	544.9616	**540.2377**	544.5434	544.8145	547.6543	544.4423	5.330481
Trellis	154.5482	168.3249	**113.8191**	135.621	165.4487	147.5524	45.61808
Walking	30.0867	26.0456	**21.8904**	31.3223	23.9137	26.65174	8.009226
Walking2	30.7862	**27.1731**	28.5749	28.6241	27.8029	28.59224	2.729797
Woman	198.6427	193.1436	**40.1001**	196.996	198.3648	165.4494	140.2135

References

Adam, A., Rivlin, E., & Shimshoni, I. (2006, June). Robust fragments-based tracking using the integral histogram. In *2006 ieee computer society conference on computer vision and pattern recognition* (Vol. 1, p. 798-805).

Agarwal, S., Awan, A., & Roth, D. (2004, Nov). Learning to detect objects in images via a sparse, part-based representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *26*(11), 1475-1490.

Amit, Y., & Trouvé, A. (2007). POP: patchwork of parts models for object recognition. *International Journal of Computer Vision*, *75*(2), 267–282.

Andriluka, M., Roth, S., & Schiele, B. (2008, June). People-tracking-by-detection and people-detection-by-tracking. In *Computer vision and pattern recognition, 2008. cvpr 2008. ieee conference on* (p. 1-8).

Anjum, N., & Cavallaro, A. (2008, Nov). Multifeature object trajectory clustering for video analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, *18*(11), 1555-1564.

Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002, Feb). A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, *50*(2), 174-188.

Asthana, A., Zafeiriou, S., Cheng, S., & Pantic, M. (2014, June). Incremental face alignment in the wild. In *2014 ieee conference on computer vision and pattern recognition* (p. 1859-1866).

Avidan, S. (2004, Aug). Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(8), 1064-1072.

Avidan, S. (2007, February). Ensemble tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, *29*(2), 261–271.

Babenko, B., Yang, M. H., & Belongie, S. (2009, June). Visual tracking with online multiple instance learning. In *2009 ieee conference on computer vision and pattern recognition* (p. 983-990).

Babenko, B., Yang, M.-H., & Belongie, S. (2011, Aug). Robust object tracking with online multiple instance learning. *Pattern Analysis and Machine Intelligence, IEEE*

Transactions on, *33*(8), 1619-1632.

Balan, A. O., & Black, M. J. (2006, June). An adaptive appearance model approach for model-based articulated object tracking. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 1, p. 758-765).

Ballard, D. H., & Brown, C. M. (1982). *Computer vision* (1st ed.). Prentice Hall Professional Technical Reference.

Bar-Shalom, Y., Kirubarajan, T., & Li, X.-R. (2002). *Estimation with applications to tracking and navigation.* New York, NY, USA: John Wiley & Sons, Inc.

Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008, June). Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, *110*(3), 346–359.

Bengio, Y., Courville, A., & Vincent, P. (2013, Aug). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798-1828. doi: 10.1109/TPAMI.2013.50

Black, M. J., & Jepson, A. D. (1998). Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, *26*(1), 63–84.

Blom, H. A. P., & Bloem, E. A. (2007, January). Exact bayesian and particle filtering of stochastic hybrid systems. *IEEE Transactions on Aerospace and Electronic Systems*, *43*(1), 55-70.

Bolme, D. S., Beveridge, J. R., Draper, B. A., & Lui, Y. M. (2010, June). Visual object tracking using adaptive correlation filters. In *Computer vision and pattern recognition (cvpr), 2010 ieee conference on* (p. 2544-2550).

Boykov, Y., & Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *Computer vision, 2001. iccv 2001. proceedings. eighth ieee international conference on* (Vol. 1, p. 105-112 vol.1).

Brookes, M. (2011). The matrix reference manual [Computer software manual].

Cehovin, L., Kristan, M., & Leonardis, A. (2011, Nov). An adaptive coupled-layer visual model for robust visual tracking. In *2011 international conference on computer vision* (p. 1363-1370).

Cehovin, L., Kristan, M., & Leonardis, A. (2013, April). Robust visual tracking using an adaptive coupled-layer visual model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(4), 941-953.

Chang, C., & Ansari, R. (2005, March). Kernel particle filter for visual tracking. *IEEE Signal Processing Letters*, *12*(3), 242-245.

Chen, C. C., Lin, H. H., & Chen, O. T. C. (2011, May). Tracking and counting people in visual surveillance systems. In *2011 ieee international conference on acoustics, speech and signal processing (icassp)* (p. 1425-1428).

Collins, R. T., Liu, Y., & Leordeanu, M. (2005, Oct). Online selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machine*

Intelligence, *27*(10), 1631-1643.

Comaniciu, D., Ramesh, V., & Meer, P. (2000). Real-time tracking of non-rigid objects using mean shift. In *Computer vision and pattern recognition, 2000. proceedings. ieee conference on* (Vol. 2, p. 142-149 vol.2).

Comaniciu, D., Ramesh, V., & Meer, P. (2003, May). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(5), 564-577.

Cootes, T. F., Edwards, G. J., & Taylor, C. J. (2001, Jun). Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(6), 681-685.

Cristinacce, D., & Cootes, T. (2008, October). Automatic feature localisation with constrained local models. *Pattern Recogn.*, *41*(10), 3054–3067.

Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)* (Vol. 1, p. 886-893 vol. 1).

Danelljan, M., Khan, F. S., Felsberg, M., & v. d. Weijer, J. (2014, June). Adaptive color attributes for real-time visual tracking. In *2014 ieee conference on computer vision and pattern recognition* (p. 1090-1097).

Datar, M., Immorlica, N., Indyk, P., & Mirrokni, V. S. (2004). Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on computational geometry* (pp. 253–262).

Davison, A. J., Reid, I. D., Molton, N. D., & Stasse, O. (2007, June). Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(6), 1052-1067.

Donoho, D. L. (2006, April). Compressed sensing. *IEEE Transactions on Information Theory*, *52*(4), 1289-1306.

Ellis, L., Dowson, N., Matas, J., & Bowden, R. (2011, Nov 01). Linear regression and adaptive appearance models forfastsimultaneous modelling and tracking. *International Journal of Computer Vision*, *95*(2), 154–179.

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

Fergus, R., Perona, P., & Zisserman, A. (2005, June). A sparse object category model for efficient learning and exhaustive recognition. In *Computer vision and pattern recognition, 2005. cvpr 2005. ieee computer society conference on* (Vol. 1, p. 380-387 vol. 1).

Fischer, P., Dosovitskiy, A., & Brox, T. (2014). Descriptor matching with convolutional neural networks: a comparison to sift. *arXiv preprint arXiv:1405.5769*.

Fukunaga, K., & Hostetler, L. (1975, Jan). The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, *21*(1), 32-40.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 580–587).

Godec, M., Roth, P. M., & Bischof, H. (2011, Nov). Hough-based tracking of non-rigid objects. In *2011 international conference on computer vision* (p. 81-88).

Grabner, H., & Bischof, H. (2006). On-line boosting and vision. In *Proceedings of the 2006 ieee computer society conference on computer vision and pattern recognition - volume 1* (pp. 260–267). Washington, DC, USA: IEEE Computer Society.

Grabner, H., Leistner, C., & Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. In *Computer vision - ECCV 2008, 10th european conference on computer vision, marseille, france, october 12-18, 2008, proceedings, part I* (pp. 234–247).

Guo, Y., Xu, G., & Tsuji, S. (1994). Tracking human body motion based on a stick figure model. *Journal of Visual Communication and Image Representation*, *5*(1), 1 - 9.

Hampapur, A., Brown, L., Connell, J., Ekin, A., Haas, N., Lu, M., … Pankanti, S. (2005, March). Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking. *IEEE Signal Processing Magazine*, *22*(2), 38-51.

Han, B., Sim, J., & Adam, H. (2017, July). Branchout: Regularization for online ensemble tracking with convolutional neural networks. In *2017 ieee conference on computer vision and pattern recognition (cvpr)* (p. 521-530). doi: 10.1109/CVPR.2017.63

Hare, S., Saffari, A., & Torr, P. H. S. (2011, Nov). Struck: Structured output tracking with kernels. In *2011 international conference on computer vision* (p. 263-270).

Hayashi, T., Agamanolis, S., & Karau, M. (2008). Mutsugoto: a body-drawing communicator for distant partners. In *International conference on computer graphics and interactive techniques, SIGGRAPH 2008, poster proceedings* (p. 91).

Henriques, J. a. F., Caseiro, R., Martins, P., & Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. In *Proceedings of the 12th european conference on computer vision - volume part iv* (pp. 702–715).

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

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*, 55–67.

Hong, S., You, T., Kwak, S., & Han, B. (2015). Online tracking by learning discriminative saliency map with convolutional neural network. *arXiv preprint arXiv:1502.06796*.

Hou, Y., Zhang, H., & Zhou, S. (2015). Convolutional neural network-based image representation for visual loop closure detection. In *Information and automation, 2015 ieee international conference on* (pp. 2238–2245).

Huang, C., Lucey, S., & Ramanan, D. (2017, Oct). Learning policies for adaptive tracking with deep feature cascades. In *2017 ieee international conference on computer vision (iccv)* (p. 105-114). doi: 10.1109/ICCV.2017.21

Huang, K. S., & Trivedi, M. M. (2003). Video arrays for real-time tracking of person, head, and face in an intelligent room. *Mach. Vis. Appl.*, *14*(2), 103–111.

Jurie, F., & Dhome, M. (2002). Real time robust template matching. In *Proceedings of the british machine vision conference* (p. 10.1-10.10). BMVA Press.

Kalal, Z., Matas, J., & Mikolajczyk, K. (2010, June). P-n learning: Bootstrapping binary classifiers by structural constraints. In *Computer vision and pattern recognition (cvpr), 2010 ieee conference on* (p. 49-56).

Kalal, Z., Mikolajczyk, K., & Matas, J. (2012, July). Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(7), 1409-1422.

Kale, A., & Jaynes, C. (2006, June). A joint illumination and shape model for visual tracking. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 1, p. 602-609).

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, *82*(Series D), 35–45.

Khan, M. H. (2015). *Visual tracking over multiple temporal scales* (Unpublished doctoral dissertation). University of Nottingham, UK.

Khan, M. H., Valstar, M. F., & Pridmore, T. P. (2015). Mts: A multiple temporal scale tracker handling occlusion and abrupt motion variation. In *Asian conf. on computer vision.*

Kristan, M. (2013). *Fast kernel density estimator.* Matlab Central.

Kristan, M., Kovacic, S., Leonardis, A., & Pers, J. (2010, Dec). A two-stage dynamic model for visual tracking. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *40*(6), 1505-1520.

Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernandez, G., ... Pflugfelder, R. (2015). The visual object tracking vot2015 challenge results. In *Proceedings of the ieee international conference on computer vision workshops* (pp. 1–23).

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

Kristan, M., Matas, J., Leonardis, A., Vojir, T., Pflugfelder, R., Fernandez, G., ... Čehovin, L. (2016b). The visual object tracking vot2016 challenge results. In

Computer vision – eccv 2016 workshops proceedings, part ii (pp. 777–823). Cham: Springer International Publishing.

Kristan, M., Pflugfelder, R., Leonardis, A., Matas, J., Porikli, F., Čehovin, L., ... Niu, Z. (2013). The visual object tracking vot2013 challenge results. In *2013 ieee international conference on computer vision workshops* (p. 98-111).

Kristan, M., Pflugfelder, R., Leonardis, A., Matas, J., Čehovin, L., Nebehay, G., ... Niu, Z. (2014). The visual object tracking vot2014 challenge results. In *2014 european conference on computer vision workshops.*

Kwon, J., & Lee, K. M. (2009, June). Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling. In *Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on* (p. 1208-1215).

Kwon, J., & Lee, K.-M. (2010, June). Visual tracking decomposition. In *Computer vision and pattern recognition (cvpr), 2010 ieee conference on* (p. 1269-1276).

Kwon, J., & Lee, K. M. (2011, Nov). Tracking by sampling trackers. In *2011 international conference on computer vision* (p. 1195-1202).

Leibe, B., Schindler, K., Cornelis, N., & Van Gool, L. (2008, Oct). Coupled object detection and tracking from static cameras and moving vehicles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *30*(10), 1683-1698.

Li, H., Shen, C., & Shi, Q. (2011, June). Real-time visual tracking using compressive sensing. In *Computer vision and pattern recognition (cvpr), 2011 ieee conference on* (p. 1305-1312).

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

Lin, Z., Hua, G., & Davis, L. (2009, June). Multiple instance ffeature for robust part-based object detection. In *Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on* (p. 405-412).

Liu, T., Wang, G., & Yang, Q. (2015, June). Real-time part-based visual tracking via adaptive correlation filters. In *2015 ieee conference on computer vision and pattern recognition (cvpr)* (p. 4902-4912).

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

Lucas, B. D., & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on artificial intelligence - volume 2* (pp. 674–679).

Ma, C., Huang, J.-B., Yang, X., & Yang, M.-H. (2015). Hierarchical convolutional features for visual tracking. In *Proceedings of the ieee international conference on computer vision* (pp. 3074–3082).

Maggio, D. E., & Cavallaro, D. A. (2011). *Video tracking: Theory and practice* (1st ed.). Wiley Publishing.

Manjunath, B. S., & Ma, W. Y. (1996, Aug). Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(8), 837-842.

Martinez, B., Valstar, M., Binefa, X., & Pantic, M. (2013). Local evidence aggregation for regression based facial point detection. *Transactions on Pattern Analysis and Machine Intelligence*, *35*(5), 1149–1163.

Matthews, L., Ishikawa, T., & Baker, S. (2004, June). The template update problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(6), 810-815.

McGinnity, S., & Irwin, G. W. (2000, Jul). Multiple model bootstrap filter for maneuvering target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, *36*(3), 1006-1012.

Mei, X., & Ling, H. (2009, Sept). Robust visual tracking using l1 minimization. In *2009 ieee 12th international conference on computer vision* (p. 1436-1443).

Mikolajczyk, K., Schmid, C., & Zisserman, A. (2004). Human detection based on a probabilistic assembly of robust part detectors. In *European conference on computer vision* (Vol. I, pp. 69–81).

Minka, T. (2005, October). *Discriminative models, not discriminative training* (Tech. Rep.).

Mohan, A., Papageorgiou, C., & Poggio, T. (2001, Apr). Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *23*(4), 349-361.

Monti, F., & Regazzoni, C. S. (2010, Sept). Human action recognition using the motion of interest points. In *2010 ieee international conference on image processing* (p. 709-712).

Morris, B. T., & Trivedi, M. M. (2008, Sept). Learning, modeling, and classification of vehicle track patterns from live video. *IEEE Transactions on Intelligent Transportation Systems*, *9*(3), 425-437.

Moschini, D., & Fusiello, A. (2008). Tracking stick figures with hierarchical articulated icp. *Proceedings THEMIS*, 61–68.

Mountney, P., & Yang, G.-Z. (2008). Soft tissue tracking for minimally invasive surgery: Learning local deformation online. In *International conference on medical image computing and computer-assisted intervention* (pp. 364–372).

Nam, H., & Han, B. (2016, June). Learning multi-domain convolutional neural networks for visual tracking. In *2016 ieee conference on computer vision and pattern recognition (cvpr)* (p. 4293-4302). doi: 10.1109/CVPR.2016.465

Notomi, Y., Shiota, T., Popovi, Z. B., Weaver, J. A., Oryszak, S. J., Greenberg, N. L.,

. . . Martin-Miklovic, M. G. (2005). Measurement of ventricular torsion by two-dimensional ultrasound speckle tracking imaging. *Journal of the American College of Cardiology*, *45*(12), 2034 - 2041.

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

Okuma, K., Taleghani, A., De Freitas, N., Little, J. J., & Lowe, D. G. (2004). A boosted particle filter: Multitarget detection and tracking. In *European conference on computer vision* (pp. 28–39).

Ong, E.-J., & Bowden, R. (2011). Robust facial feature tracking using shape-constrained multiresolution-selected linear predictors. *IEEE Trans. Pattern Anal. Mach. Intell.*, *33*(9), 1844-1859.

Palmer, S. E. (1999). *Vision science: Photons to phenomenology.* MIT press.

Patras, I., & Hancock, E. R. (2010). Coupled prediction classification for robust visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, *32*(9), 1553–1567.

Pedersoli, M., Vedaldi, A., & Gonzlez, J. (2011, June). A coarse-to-fine approach for fast deformable object detection. In *Computer vision and pattern recognition (cvpr), 2011 ieee conference on* (p. 1353-1360).

Perez, P., Vermaak, J., & Blake, A. (2004, Mar). Data fusion for visual tracking with particles. *Proceedings of the IEEE*, *92*(3), 495-513.

Pernkopf, F. (2008). Tracking of multiple targets using online learning for reference model adaptation. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, *38*(6), 1465-1475.

Poole, A., & Ball, L. J. (2005). Eye tracking in human-computer interaction and usability research: Current status and future. In *Prospects, chapter in c. ghaoui (ed.): Encyclopedia of human-computer interaction. pennsylvania: Idea group, inc.*

Qi, Y., Zhang, S., Qin, L., Yao, H., Huang, Q., Lim, J., & Yang, M. H. (2016, June). Hedged deep tracking. In *2016 ieee conference on computer vision and pattern recognition (cvpr)* (p. 4303-4311). doi: 10.1109/CVPR.2016.466

Ramanan, D., Forsyth, D. A., & Zisserman, A. (2007, Jan). Tracking people by learning their appearance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(1), 65-81.

Rodriguez, A., Boddeti, V. N., Kumar, B. V. K. V., & Mahalanobis, A. (2013, Feb). Maximum margin correlation filter: A new approach for localization and classification. *IEEE Transactions on Image Processing*, *22*(2), 631-643.

Ross, D. A., Lim, J., Lin, R.-S., & Yang, M.-H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, *77*(1), 125–141.

Saleemi, I., Shafique, K., & Shah, M. (2009, Aug). Probabilistic modeling of scene dynamics for applications in visual surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(8), 1472-1485.

Sánchez-Lozano, E., Tzimiropoulos, G., Martinez, B., De la Torre, F., & Valstar, M. (2016). A functional regression approach to facial landmark tracking. *arXiv preprint arXiv:1612.02203*.

Schmid, C., & Mohr, R. (1997, May). Local grayvalue invariants for image retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *19*(5), 530-535.

Schnitzspan, P., Roth, S., & Schiele, B. (2010, June). Automatic discovery of meaningful object parts with latent crfs. In *Computer vision and pattern recognition (cvpr), 2010 ieee conference on* (p. 121-128).

Serby, D., Meier, E. K., & Gool, L. V. (2004, Aug). Probabilistic object tracking using multiple features. In *Pattern recognition, 2004. icpr 2004. proceedings of the 17th international conference on* (Vol. 2, p. 184-187 Vol.2).

Shahed Nejhum, S., Ho, J., & Yang, M.-H. (2008, June). Visual tracking with histograms and articulating blocks. In *Computer vision and pattern recognition, 2008. cvpr 2008. ieee conference on* (p. 1-8).

Shan, C., Tan, T., & Wei, Y. (2007, July). Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recogn.*, *40*(7), 1958–1970.

Sigal, L., Bhatia, S., Roth, S., Black, M., & Isard, M. (2004, June). Tracking loose-limbed people. In *Computer vision and pattern recognition, 2004. cvpr 2004. proceedings of the 2004 ieee computer society conference on* (Vol. 1, p. I-421-I-428 Vol.1).

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

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

Smith, K. C. (2007). *Bayesian methods for visual multi-object tracking with applications to human activity recognition* (Unpublished doctoral dissertation). Ãcole Polytechnique Fédérale de Lausanne, Lausanne , Switzerland.

Teng, Z., Xing, J., Wang, Q., Lang, C., Feng, S., & Jin, Y. (2017, Oct). Robust object tracking based on temporal and spatial deep networks. In *2017 ieee international conference on computer vision (iccv)* (p. 1153-1162). doi: 10.1109/ICCV.2017.130

Veenman, C. J., Reinders, M. J. T., & Backer, E. (2001, Jan). Resolving motion correspondence for densely moving points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(1), 54-72.

Vojir, T., Noskova, J., & Matas, J. (2014, November). Robust scale-adaptive mean-shift for tracking. *Pattern Recogn. Lett.*, *49*(C), 250–258.

Wang, L., Ouyang, W., Wang, X., & Lu, H. (2015). Visual tracking with fully convolutional networks. In *Proceedings of the ieee international conference on computer vision* (pp. 3119–3127).

Wang, L., Ouyang, W., Wang, X., & Lu, H. (2016, June). Stct: Sequentially training convolutional networks for visual tracking. In *2016 ieee conference on computer vision and pattern recognition (cvpr)* (p. 1373-1381). doi: 10.1109/CVPR.2016.153

Wang, S., Lu, H., Yang, F., & Yang, M. H. (2011, Nov). Superpixel tracking. In *2011 international conference on computer vision* (p. 1323-1330).

Williams, O., Blake, A., & Cipolla, R. (2005). Sparse bayesian learning for efficient visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(8), 1292–1304.

Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., & Ma, Y. (2009, Feb). Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(2), 210-227.

Wu, Y., Lim, J., & Yang, M.-H. (2013). Online object tracking: A benchmark. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2411–2418).

Xiong, X., & De la Torre, F. (2013). Supervised descent method and its applications to face alignment. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 532–539).

Xu, T., Peng, P., Fang, X., Su, C., Wang, Y., Tian, Y., ... Huang, T. (2012, Sept). Single and multiple view detection, tracking and video analysis in crowded environments. In *2012 ieee ninth international conference on advanced video and signal-based surveillance* (p. 494-499).

Yang, H., Shao, L., Zheng, F., Wang, L., & Song, Z. (2011, November). Recent advances and trends in visual tracking: A review. *Neurocomput.*, *74*(18), 3823–3831.

Yang, M., Yuan, J., & Wu, Y. (2007, June). Spatial selection for attentional visual tracking. In *2007 ieee conference on computer vision and pattern recognition* (p. 1-8).

Yao, R., Shi, Q., Shen, C., Zhang, Y., & van den Hengel, A. (2013, June). Part-based visual tracking with online latent structural learning. In *Computer vision and pattern recognition (cvpr), 2013 ieee conference on* (p. 2363-2370).

Yilmaz, A., Javed, O., & Shah, M. (2006, December). Object tracking: A survey. *ACM Comput. Surv.*, *38*(4).

Yilmaz, A., Li, X., & Shah, M. (2004, Nov). Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1531-1536.

Yun, S., Choi, J., Yoo, Y., Yun, K., & Choi, J. Y. (2017, July). Action-decision networks for visual tracking with deep reinforcement learning. In *2017 ieee conference on computer vision and pattern recognition (cvpr)* (p. 1349-1358). doi: 10.1109/CVPR.2017.148

Zagoruyko, S., & Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4353–4361).

Zhang, L., & van der Maaten, L. (2013, June). Structure preserving object tracking. In *2013 ieee conference on computer vision and pattern recognition* (p. 1838-1845).

Zhang, T., Jia, K., Xu, C., Ma, Y., & Ahuja, N. (2014, June). Partial occlusion handling for visual tracking via robust part matching. In *2014 ieee conference on computer vision and pattern recognition* (p. 1258-1265).

Zheng, Y., & Kambhamettu, C. (2009, Sept). Learning based digital matting. In *2009 ieee 12th international conference on computer vision* (p. 889-896).

Zhong, W. (2012). Robust object tracking via sparsity-based collaborative model. In *Proceedings of the 2012 ieee conference on computer vision and pattern recognition (cvpr)* (pp. 1838–1845). Washington, DC, USA: IEEE Computer Society.

Zhou, J., Gao, D., & Zhang, D. (2007, Jan). Moving vehicle detection for automatic traffic monitoring. *IEEE Transactions on Vehicular Technology*, *56*(1), 51-59.

Zhou, S. K., Chellappa, R., & Moghaddam, B. (2004, Nov). Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, *13*(11), 1491-1506.

Zhu, L., Chen, Y., Yuille, A., & Freeman, W. (2010, June). Latent hierarchical structural learning for object detection. In *Computer vision and pattern recognition (cvpr), 2010 ieee conference on* (p. 1062-1069).

Zhu, Q., Yeh, M.-C., Cheng, K.-T., & Avidan, S. (2006). Fast human detection using a cascade of histograms of oriented gradients. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 2, p. 1491-1498).

Zhu, X., & Ramanan, D. (2012, June). Face detection, pose estimation, and landmark localization in the wild. In *Computer vision and pattern recognition (cvpr), 2012 ieee conference on* (p. 2879-2886).

Zimmermann, K., Matas, J., & Svoboda, T. (2009). Tracking by an optimal sequence of linear predictors. , *31*(4), 677-692.