New Approaches for Object Tracking and Image-based Quality Control

DISSERTATION
zur Erlangung des Grades eines Doktors
der Ingenieurwissenschaften

vorgelegt von
Ahmad Delforouzi

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2020

Betreuer und erster Gutachter
Prof. Dr.-Ing. Hubert Roth
Universität Siegen

Zweiter Gutachter
Prof. Dr.-Ing. habil. Marcin Grzegorzek
Universität zu Lübeck

Tag der mündlichen Prüfung
22. Juli 2020

Acknowledgements

I would like to express my gratitude to all of these people. In particular, I would like to thank most sincerely my second supervisor Prof. Dr. Marcin Grzegorzek for assisting and motivating me throughout the entire period of my studies in his research group. I keep him in my mind his kind supervision.

I would like to thank Prof. Dr. Hubert Roth who kindly agreed to be my first supervisor and for his attempts regarding the oral examination and his guidance for the formal steps after defense. I also thank Prof. Dr. Reiner Brück and Prof. Dr. Markus Lohrey for being members of my oral exam commission.

I am particularly grateful to all of my colleagues from the Group of Medical Data Science, University of Lübeck as well as my colleagues in the Group of Pattern Recognition in the University of Siegen, especially Dr. Seyed Amir Hossein Tabatabaei, Prof. Dr. Kimiaki Shirahama and M.Sc. Muhammad Adeel Nisar for their great scientific contributions during my study in Siegen. Also, I would thank my great students M.Sc. Bhargav Pamarthi and David Holighaus for their kind collaboration.

I would like to thank my mother Nahid, my brothers Alireza and Mahmoud, my sister Nargess for their encouragement during my study and life. Especially, I would like to warmly thank my wife, my love Nasim, for her very kind support during my study and residence in Siegen. I would like to mention my son Adrian who warms our small family in Germany. Finally, I want to remember my father for encouraging and supporting me in the long way of education from the school to Master, I dedicate this thesis to him.

Abstract

The dissertation at hand is concerned with the scientific solutions for the problems of visual object tracking and image-based railway quality control. In the tracking scenario, an object of interest is followed automatically. The type, size and, general features of the object are not known and the system does not have any knowledge of it. The object tracking schemes are studied based on three different sensors namely rectangular, omnidirectional and, 360-degree cameras. For the rectangular videos, two object-detection-based methods are proposed. The first tracker uses a pre-trained object detector and does not update, whereas the second one is updated during the tracking. These systems compare five famous training based object detectors. The results demonstrate the trackers' efficiency and show the preference of offline tracker to the online one.

The polar cameras - namely omnidirectional and 360-degree sensors - provide videos with a wider field of view than the conventional normal rectangular ones. Replacing conventional security cameras with 360-degree ones allows a significant reduction of hardware costs as well as software license and maintenance costs. There are many trackers based on conventional rectangular videos in the literature, whereas the number of polar object following systems, in comparison, is very limited and they are not yet matured. Most of the projects which are going to be discussed in this work are processing the 360-degree videos. Two proposed methods unwrap the polar videos using image rectification; then a modified version of Tracking Learning Detection tracker and another state-of-the-art detector are applied. To increase the speed of the process, a trapezoidal tracker is proposed to eliminate the rectification part. In another proposed scheme, a SURF based algorithm is used to improve performance. This tracker uses two learning based modules for interesting points matching and challenges recognizing respectively. The other proposed method combines a polar candidates generation method and color binary features to improve its accuracy and speed. The experiments show that the last method has the best accuracy and speed among the proposed methods and it outperforms the state-of-the-art polar trackers.

In the second part of the dissertation, a vision-based quality control method of concrete railroad sleepers is presented. This system captures an image sequence by a high resolution, fast and moving camera from railway top view and applies a proposed image-based crack detector to control the railway sleepers quality. This scheme first locates the sleepers within the images and then, generates crack candidates on the sleeper images and finally, detects and classifies the cracks and by applying a supervised classifier on the candidates. The classifier uses geometrical features to detect and classify the cracks on the concrete sleepers. The experimental results show that the crack detector successfully finds the cracks.

Zusammenfassung

Die vorliegende Dissertation befasst sich mit der Problematik der visuellen Objektverfolgung und bildbasierte Bahnqualitätskontrolle. Im Objektverfolgungsszenario wird ein Objekt von Interesse automatisch verfolgt. Hierbei sind Art, Größe und im Allgemeinen die Eigenschaften ebendiesen Objekts nicht bekannt und der Tracker hat keinerlei Kenntnis über das Objekt. Die Objektverfolgung wird auf der Grundlage von drei verschiedenen Sensoren untersucht, nämlich rechteckigen, omnidirektionalen und 360-Grad-Sensoren. Für die rechteckigen Videos werden zwei auf Objekterkennung basierende Tracker vorgeschlagen. Der erste Tracker verwendet einen vortrainierten Objektdetektor und aktualisiert nicht, während der zweite während des Tracking aktualisiert wird. Anhand dessen werden fünf bekannte trainingsbasierte Objektdetektoren verglichen. Die Ergebnisse zeigen die Effizienz der Tracker und die Präferenz des Offline- gegenüber dem Online-Tracker.

Die Polarsensoren, d.h. omnidirektionale und 360- Grad-Sensoren, liefern polare Videos. Sie verfügen über ein breiteres Sichtfeld als die herkömmlichen rechteckigen Sensoren. Der Austausch herkömmlicher Sicherheitskameras durch 360-Grad Kameras reduziert die Kosten für Hardware, Softwarelizenzierung und Wartung erheblich. In der Literatur finden sich viele Tracker, die auf herkömmlichen rechteckigen Videos basieren, während die Anzahl der Polar-Tracker im Vergleich dazu sehr begrenzt ist und diese noch nicht ausgereift sind. Die meisten Projekte, die in dieser Dissertation behandelt werden, verarbeiten Videos von 360-Grad-Sensoren. Zwei vorgestellte Tracker wandeln die polaren Videos mittels Bild-Rektifizierung in ein rechteckiges Format um. Anschließend erfolgt die Anwendung einer modifizierten Version des Tracking- Learning-Detection Tracker für die erste Objektverfolgungsmethode. Die Tracking- Geschwindigkeit soll erhöht werden, indem ein trapezförmiger Tracker vorgeschlagen wird, um den Rektifikationsprozess zu vermeiden. In einem anderen Vorschlag wird ein SURF-basierter Tracker verwendet, um die Trackingleistung zu verbessern. Dieser verwendet zwei lernbasierte Module, um interessante Punkte abzugleichen und Herausforderungen zu erkennen. Schließlich werden ein Polar-Kandidaten-Erzeugungsverfahren und Farbbinärmerkmale verwendet, um deren Genauigkeit und Geschwindigkeit zu verbessern. Die Experimente zeigen, dass diese Methode die beste Genauigkeit und Geschwindigkeit unter den vorgeschlagenen polaren Trackern bietet und die modernen Polar-Tracker übertrifft.

Im zweiten Teil dieser Dissertation wird ein visionsbasiertes Qualitätskontrollverfahren für Eisenbahnschwellen aus Beton dargelegt. Hierbei wird eine Bildsequenz von einer hochauflösenden und sich schnell bewegenden Kamera von Draufsicht auf die Eisenbahn aufgenommen und der vorgeschlagene bildbasierte Rissdetektor überprüft den Zustand der Schwellen. Die vorgeschlagene Methode findet zunächst die Schwellen in den Bildern. Anschließend werden auf den Schwellenbildern Riss-Kandidaten erzeugt und schließlich werden Risse anhand geometrischer Merkmale erkannt und klassifiziert, indem ein überwachter Klassifizierer auf die Kandidaten angewendet wird. Die experimentellen Ergebnisse zeigen, dass der Rissdetektor die Risse erfolgreich findet.

Contents

Chapter 1

Introduction

Nowadays, cameras are everywhere and provide a huge amount of information. The number of smart-phone users in the world is forecast to grow from 2.1 billion in 2016 to around 2.5 billion in 2019[1]. The number of video surveillance cameras was globally more than 245 million in 2014 and is increasing very fast[2]. The visual data themselves do not provide directly the users with any added value and they need to be semantically interpreted in a particular application context to become useful.

This chapter is structured as follows: Section 1.1 presents the basic concept of this dissertation. In Section 1.2, the motivation behind this thesis is outlined. Afterwards, the contribution of this dissertation to the two main areas of this thesis, namely object tracking and railroad quality control, is given in Section 1.3. Section 1.4 presents an overall structure of the thesis.

1.1 Fundamental Concept

This dissertation presents novel scientific methods in the area of visual data understanding. Visual data understanding systems processes digital videos to add valuable information to the raw videos. The thesis is composed of two main following parts:

- Visual Object Tracking: By knowing the location of a certain object in the first frame of a video, the task of following this object is a fascinating task in the area of visual data understanding. This task becomes scientifically more interesting when certain real world challenges are involved in object tracking.

[1]https://www.statista.com/

[2]https://ihsmarkit.com/index.html

- Image-based Quality Control: Digital image processing extracts valuable knowledge from raw image-based data. This knowledge may be used in various subdisciplines namely quality control in industrial productions. The task of digital video processing is scientifically interesting when some challenges are included.

1.1.1 Fundamental Concept of Visual Object Tracking

Visual object tracking plays an important role in the real world. An object tracker follows an object in consecutive video frames. The human brain uses tracking in order to perform one of its most important tasks: detecting and recognizing an object. For instance, when a faraway object is approaching, it is tracked continuously in the brain. As the object reaches a maximum distance, it can be recognized. In the area of digital data processing, there are many applications which are benefiting from object tracking. For instance, medicine, military, security systems and, intelligent transport systems are using object tracking to find solutions for the problems in their applications. 360-degree cameras are new sensors containing excellent features such as wide field-of-view 360-180-180 degree, low-weight and easy moving, high-quality and high-resolution videos. These wonderful features dedicate new fields to improve the application of visual data understanding in the real world. This thesis presents scientific methods for object tracking in 360-degree videos and undertakes the object tracking task independently of type and size of the objects.

1.1.2 Fundamental Concept of Image-based Quality Control

A railroad sleeper is the underlying infrastructure of the railroad. Due to the weight of the railroad as well as the trains passing over it, some cracks appear on the sleepers over time. The aim of this thesis is to develop novel scientific methods to control the quality of the images of sleepers. A high-quality camera with high-resolution images is used for quality control of the railway sleepers. This dissertation performs the task of quality control of the railway sleeper not only for detection of cracks but also for classification of them based on a pre-defined damaging level.

1.2 Motivation

This thesis presents scientific algorithms in two major areas namely object tracking and a vision-based railway monitoring system. Generally, object tracking systems try to handle the following challenges:

- Occlusion: the object of interest becomes hidden by another object in the scene.
- Object rotation: the object of interest has out-of-plane rotation compared to the 2d-plane of the lens of the camera.
- Tracking speed limitation: the quantity of tracked frames in a time unit is limited by the speed of the tracker.
- Size variation: when the object of interest departs the camera, it becomes smaller and when it approaches the camera, it becomes bigger.
- Background clutter: the scene is very complicated and non-smooth.
- Similar objects: the scene has at least one object which is similar to the object of interest.
- Scene departure: object of interest leaves the scene and may appear later.

The motivation of this thesis is raised from the following limitations/shortages in science and technology:

(M_1) **Rectangular Tracking:** None of the available trackers can handle all the mentioned challenges simultaneously. Most trackers are able to handle a simplified scenario. New trackers based on new solutions are still required to improve the tracking task.

(M_2) **Polar-Rectangular Tracking:** The rectangular trackers cannot be applied to the 360-degree videos. The available trackers are not able to follow the object when the object is rotated and distorted. They are also slow in the case of high-resolution 360-degree videos. The available state-of-the-art trackers are not able to track the object of interest in the presence of challenges which are involved in the 360-degree videos. For instance, TLD [KMM12] loses the object of interest, in the case of polar videos.

(M_3) **Polar Tracking:** The image rectification process is very slow especially in high-resolution 360-degree videos. Slow tracking, in turn, may lead to an object loss. Some fast trackers for the 360-videos must be developed. If the tracker can bypass the rectification process, the processing load is drastically saved.

(M_4) **Challenge-Detecting Tracking:** Each challenge in the 360-degree videos has its own solution. If the challenges can be detected separately, a distinct solution can be applied to a certain challenging situation. For instance, challenges like in-plane and out-of-plane rotation should be detected and handled by using rotation-resistance features.

(M_5) **Fast Polar Tracking:** The available polar trackers for high-resolution 360-degree videos work slowly. These videos require fast algorithms for object tracking to decrease the probability of the object lost. In other words, faster tracking leads to more stable tracking.

(M_6) **Railway Monitoring:** Since heavy trains are passing over the railroad, the sleepers are destroyed gradually. This process starts with appearing short and thin cracks on the sleepers. If the tiny cracks on the sleepers can be detected early, the probability of railway damage and consequent railroad accidents is decreased. The shortage of staff for manual monitoring of the long railroads in different countries shows the necessity of an automatic system for railway monitoring. The quantity of available scientific methods for automatic sleeper monitoring is very limited and focused on crack detection on wooden sleepers. There is a gap in the area of automatic crack detection and classification on concrete sleepers.

(M_7) **Training-based Railway Monitoring:** The vision-based sleeper detection using a rule-based method is very sensitive to the variable environmental conditions. Novel training-based methods for the sleeper location and crack detection are required. These methods have more generality than rule-based methods.

1.3 Contribution

This thesis contributes to science and technology by proposing the following scientific algorithms to overcome the problems stated in the section of motivation. The scientific contributions of this thesis have been published in internationally visible journals and conferences.

(C_1) **Rectangular Tracker:** The author's contributions to rectangular object tracking have been published in MDPI Sensors journal [DPG18]. This method proposes scientific multi-modal trackers for rectangular videos which handle the tracking challenges very well. First, synthetic data from the object of interest are generated and then a deep-learning based object detector is trained using the data. The object detector finds the object of interest in the following frames. The detector is updated by using the recently detected object at certain intervals. The proposed online tracker based on train-based object detectors is proposed for the first time in this thesis.

(C_2) **Modified-TLD Tracker:** The contribution of this thesis to object tracking in 360-degree videos as a modified version of a famous tracker (TLD) [KMM12] has been adapted to 360-degree videos and published in International Conference on Pattern Recognition (ICPR2016) [Del+16b]. MTLD restricts the searching area of the object of interest to decrease the computation load. By proposing a FIFO data structure for the training data to adapt the tracker to the last appearances, the stability of the proposed tracker is increased. This method can track objects even when they have the out-of-plane rotation and varying environment.

(C_3) **Trapezoid-shape Tracker:** The author's contribution to object tracking in 360-degree videos by using a polar trapezoid-shape method of candidate generation and color-classifiers in HSV domain has been published in the proceedings of the International Symposium on Multimedia in (ISM 2016) [Del+16a]. This method first generates trapezoid-shape candidates on the polar images and then using three classifiers namely variance classifier, color-max classifier and, color number classifier finds the object of interest. The color-based classifiers and the polar trapezoid-shape candidate generation method are the contributions of this paper [Del+16a].

(C_4) **SURF Tracker:** The author's contribution to object tracking in 360-degree videos by using SURF (Speeded Up Robust Features) extraction and matching has been published in the proceedings of the International Symposium on Multimedia in (ISM 2017) [DG15]. This method first finds interesting points on the object of interest and the scene. Then, it matches the interesting points in the object and the scene by using a proposed classifier. Another classifier is proposed for detecting the challenging situation (i.e., occlusion, out-of-plane-rotation, departing or approaching the camera). The proposed challenge detector, as well as the matching classifier, are the contributions of this paper [DG15].

(C_5) **Polar Color-binary Tracker:** The contributions of the thesis to the polar object tracking namely new circular polar object selection and region of interest selection as well as the proposed color binary classifiers have been published in the Journal of Multimedia Tools and Application [Del+18] in 2018. This tracker first generates overlapped polar candidates in the polar area of interest, then uses a hierarchical structure of classifiers in which each classifier rejects irrelevant candidates and passes the rest to the next classifier. The hierarchical structure is composed of variance and color-binary classifiers. This method is able to handle the challenges involved in 360-degree videos namely occlusion, object rotation, tracking speed, size variation, background clutter, similar objects, and scene departure.

(C_6) **Railway Monitoring:** In this paper [Del+17] first, the sleeper within a given image is located. Then, some candidates for cracks on the image of the sleeper are generated and two methods identify the cracks among the candidates and finally, the cracks are classified. The contributions to the sleeper monitoring system have been published in the proceedings of the 10th International Conference on Computer Recognition Systems, 2017 [Del+17]. This publication won the Best Paper Award. The scientific methods for automatic detection and classification of tiny cracks on concrete railway sleepers are proposed, tested and evaluated for the first time in this thesis.

(C_7) **Training-based Railway Monitoring:** The proposed system in [Tab+18] also uses the two-steps of sleeper location and crack detection similar to (C_6), but it generates some candidates for the sleeper; then, by using a supervised classifier, the sleeper is detected. This contribution to the sleeper monitoring system has been published in the International Journal of Pattern Recognition and Artificial Intelligence, 2018 [Tab+18].

1.4 Overview

The dissertation is divided into two main domains: The first one presents a deep scientific work contributing to the area of object tracking using different cameras, while the second one is a scientific work for an application of video understanding in the railway industry. The thesis is organized into five chapters. After the general information given in this chapter, Chapters 2 and 3 present a scientific view to the object tracking problem and the fourth chapter deals with scientific methods for the application of video understanding in the industry. The fifth chapter concludes this dissertation.

Chapter 2 shows the problem of object tracking in rectangular videos. First, a literature review of object tracking methods including the state of the art methods is given. In the next Section, object detection is investigated. Despite object detection and tracking having very similar terms, they are different in concept and application levels when reviewed in detail. As a solution, object detection can be used as a basis for object tracking. Well-known object detection methods then will be presented and compared in detail and two frameworks for tracking are proposed to use the mentioned object detectors for the tracking purpose. In fact, the frameworks are a tool to convert the detectors to trackers.

The mentioned detectors then are compared in the context of tracking in terms of tracker stability and speed.

Chapter 3 presents object tracking in the polar videos captured from 360-degree and omnidirectional cameras. A literature review of object tracking methods in available polar videos including the state of the art methods is given. 360-degree cameras versus omnidirectional cameras are compared and their features are explained. Then, the proposed modified method on the TLD (Tracking-learning-Detection) tracker is presented. In the next step, the proposed object tracking method based on deep learning is presented and then, the proposed polar object tracking method in 360-degree videos using the trapezoid-shape object definition is presented. In the next section, the proposed object tracking method using SURF feature descriptor and matching is presented and finally, the proposed polar model for fast object tracking in the 360-degree videos is explained.

Chapter 4 proposes scientific algorithms for an industrial system for railway sleeper quality monitoring using video processing technology. A literature review of available methods in crack detection methods in roads, bridges and wooden sleepers is given. Then, the proposed methods for sleeper location in the images are explained and in the next section, the proposed methods for crack detection on the sleepers and finally, the experiments and conclusion are given.

In Chapter 5, first a summary of the proposed tracking methods in Chapters 2 and 3 as well as the proposed sleeper monitoring system in Chapter 4 is presented. Then, the scientific methods proposed in this dissertation are concluded and the possible future trends are proposed.

Chapter 2

Object Tracking in Rectangular Video

Extracting and analyzing object trajectories from videos are basic problems in computer vision and have important applications in the event understanding, robot localization, video surveillance, etc. Trajectories of objects represent high-level semantic features, which can be used to automatically understand object activities in different kinds of videos.

This chapter proposes scientific approaches for rectangular object tracking, including author's contributions and is based on the author's journal paper (C1) [DPG18] recently published in MDPI Sensors journal. The organization of this chapter is as follows: Section 2.1 states the problem of object tracking and motivates its practical importance. A literature overview of object tracking in rectangular images is given in Section 2.2. The investigated training-based object detectors are presented and discussed in Section 2.3. In Section 2.4, two proposed algorithms for object tracking based on object detection are introduced. Experiments and results are presented in Section 2.6. Finally, conclusions are drawn and further possible research directions are listed in Section 2.7.

2.1 Problem Statement

By knowing the location of a desired object in the first frame, tracking this object becomes a fascinating topic for video processing from both scientific and industrial viewpoints. This task becomes scientifically more interesting when there is complexity involved in video sequences. This complexity can include a moving camera, object uncertainty, background clutter, small size and low resolution of the object, size variation, appearance

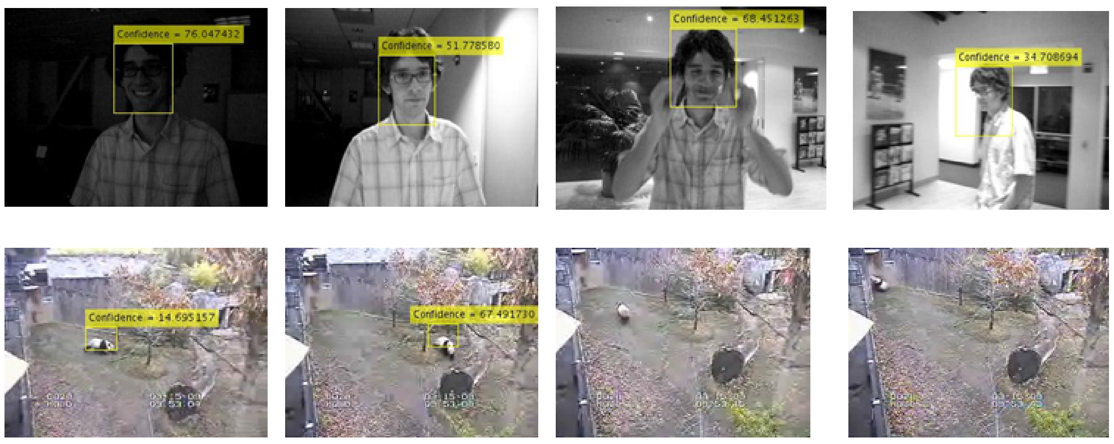

Figure 2.1: Challenges in two videos: First line (David) and second line (Panda) show illumination change, out-of-plane rotation, appearance change and background clutter

changes, occlusion, articular objects, illumination change, and out-of-plane rotation. Some of the challenges are shown in Figure 2.1.

2.2 Related Work

In this section, an overview of the available object tracking methods in rectangular videos is given. The trackers are categorized into the following four groups:

1. Feature-based Trackers: The methods identify objects by extracting features from some interesting points and track the object of interest by matching the interesting points in the successive frames.

2. Filter-based Trackers: These trackers follow the object of interest in the coming frames by finding the maximum of the output of a filter function like correlation or convolution between the image of the interesting object and the next frame of the video.

3. Learning-based Trackers: The trackers use a training-based structure for example to find the object of interest within the coming frames.

4. Hybrid Trackers: There are some trackers which combine the mentioned techniques to handle the tracking problem.

In the following, the tracking methods including the state of the art based on the aforementioned classes are explained.

2.2.1 Feature-based Trackers

Recently, many methods have been developed to overcome the challenging object tracking problem in videos. Some trackers focus on human tracking with various techniques such as common energy minimizing [MSR15; MRS14] and data association [WYX14], and other trackers focus on object tracking using various methods like point tracking [Bou00] and feature descriptors [Sak+15]. Multiple cues such as color, shape, and position are selected as human tracking features. In the case of the fixed cameras, the background is still. Thus, background detection and subtraction allow foreground object detection. In other words, looking for only moving objects in a limited area allows finding objects of interest. There are many other methods for human detection and tracking in the literature [Lin+07; WZM12; LSS05; WYX14; MSR15; MRS14; WN07; WY10]. These methods often use very simple features to detect humans. The human body is usually described by some simple shapes such as a circle for the top part and a cylindrical shape for other body parts. Thus, a very commonly used method is modeling the human appearance to 2d shapes [WZM12] or 3d shapes [LSS05]. Some other methods use offline-trained objects for the human tracking problem [WYX14]. In [WN07], the authors use Edgelet-base detectors for human tracking.

Object tracking can be formulated in terms of object detection. For example, SURF features [Bay+08], which were originally proposed for object detection, can be used for object tracking. In [Sak+15], the authors employ and compare SURF and SIFT features for object tracking and show that SURF features show better results. SURF-based object tracking methods are presented in [SNH12; Mia+11]. These methods are unable to handle occlusion and are fragile to background clutter. The SURF-tracker is usually combined with other methods to improve the performance of tracking. In [Li+11], a combination of the SURF features and camshift methods was used for object tracking in an indoor environment. In [ZH13] the authors combine mean-shift, SURF and a two-stage matching for tracking. In [Gup+13], a dynamic object model and the surf features are used for human tracking.

2.2.2 Filter-based Trackers

In [Wan+18], the authors propose a kernel cross-correlator to improve the robustness of linear cross-correlator based trackers. The method can handle affine transformations. Multiple-object tracking is an important task in automated video surveillance systems. In [WYX14], the authors present a multiple-human-tracking approach that takes the single-frame human detection results as input and associates them to form trajectories while improving the original detection results by making use of reliable temporal information in a closed-loop manner. First, it forms tracklets from which reliable temporal information is extracted and then refines the detection responses inside the tracklets, which also improves the accuracy of the tracklets. After this, local conservative tracklet association is performed and reliable temporal information is propagated across tracklets so that more detection responses can be refined. The global tracklet association is done to resolve association ambiguities.

Although correlation filters are not commonly used, they can track complex objects through rotations, occlusions, and other distractions at over 20 times more than the rate of current state-of-the-art techniques. The oldest and simplest correlation filters use simple templates and generally fail when applied to the tracking procedure. More modern approaches such as ASEF and UMACE perform better, but their training needs are poorly suited to the tracking process. A visual tracker requires robust filters to be trained from a single frame and dynamically adapted as the appearance of the target object changes. The author in [Bol+10] presents a new type of correlation filter, a minimum output sum of squared error (MOSSE) filter, which produces stable correlation filters when initialized using a single frame. A tracker based upon MOSSE filters is robust to variations in lighting, scale, pose, and nonrigid deformations while operating at 669 frames per second. Occlusion is detected based upon the peak-to-side-lobe ratio, which enables the tracker to pause and resume where it left off when the object reappears.

Detection and tracking of humans in video streams are important for many applications. In [WN07], the authors present an approach to automatically detect and track multiple, possibly partially occluded humans in a walking or standing pose from a single camera, which may be stationary or moving. A human body is represented as an assembly of body parts. Part detectors are learned by boosting a number of weak classifiers which are based on edge-let features. Responses of part detectors are combined to form a joint likelihood model that includes an analysis of possible occlusions. The combined detection responses and the part detection responses provide the observations used for tracking. Trajectory initialization and termination are both automatic and rely on the confidences computed from the detection responses. An object is tracked by data association and mean-shift methods. This system can track humans with both inter-object and scene occlusions with static or non-static backgrounds.

The core component of most modern trackers is a discriminative classifier, tasked with distinguishing between the target and the surrounding environment. To cope with natural image changes, this classifier is typically trained with translated and scaled sample patches. Such sets of samples are riddled with redundancies any overlapping pixels are constrained to be the same. Based on this simple observation, an analytic model for data sets of thousands of translated patches proposed in [Hen+15]. By showing that the resulting data matrix is circulant, it can be diagonalized with the discrete Fourier transform (DFT), reducing both storage and computation by several orders of magnitude. Interestingly, for linear regression, the formulation is equivalent to a correlation filter, used by some of the fastest competitive trackers. For kernel regression, however, a new kernelized correlation filter (KCF), unlike other kernel algorithms, has the exact same complexity as its linear counterpart. Building on it also, a fast multi-channel extension of linear correlation filters, via a linear kernel is proposed.

The kernelized correlation filter (KCF) is one of the state-of-the-art object trackers. However, it does not reasonably model the distribution of the correlation response during tracking, which might cause the drifting problem, especially when targets undergo significant appearance changes due to the occlusion, camera shaking, and/or deformation. In [Zha+17], the authors propose an output constraint transfer (OCT) method that by modeling the distribution of the correlation response in a Bayesian optimization framework is able to mitigate the drifting problem. OCT builds upon the reasonable assumption that the correlation response to the target image follows a Gaussian distribution, which selects training samples and reduces model uncertainty. OCT is rooted in a new theory which transfers the data distribution to a constraint of the optimized variable, leading to an efficient framework to calculate correlation filters.

Correlation filter-based trackers have recently achieved excellent performance, showing great robustness to challenging situations exhibiting motion blur and illumination changes. However, since the model that they learn depends strongly on the spatial layout of the tracked object, they are notoriously sensitive to deformation. Models based on color statistics have complementary traits: they cope well with variation in shape, but suffer when illumination is not consistent throughout a sequence. Moreover, color distributions lonely cannot be discriminative. In [Zha+17], the authors show a simple tracker combining complementary cues in a ridge regression framework can operate fast.

Learning a large-scale regression model has been used for visual tracking as in recent correlation filter (CF)-based trackers [CT18]. Different from the conventional CF-based algorithms in which the regression model is solved based on circulant training samples, authors in [CT18] propose learning linear regression models via a single-convolutional layer and the gradient descent (GD) technique. In this convolution-based approach, the samples are cropped from an image in a sliding-window manner rather than being circularly shifted from one base sample. As a result, the abundant background context in the images can be fully exploited to learn a robust tracker. The proposed tracker is based on two independent regression models, namely a holistic regression model and a texture regression model. The holistic regression model is trained based on the entire object patch to predict the object location, whereas the texture regression model is trained based on the local object textures. The foreground map outputted by the texture regression model is not only helpful to boost the location prediction in the case of large variations, but it is also an important clue for estimating the object size. With the foreground map outputted by the texture regression model, this method is able to estimate the object size by optimizing a novel objective function based on the object-background contrast.

The extraction of moving objects from their background is a challenging task in visual surveillance systems. Object detectors with a single threshold often fail to resolve ambiguities and correctly segment the object. In [WY10], the authors propose a method that uses three thresholds to accurately classify pixels as the foreground or background. These thresholds are adaptively determined by considering the distributions of differences between the input and background images and are used to generate three boundary sets. These boundary sets are then merged to produce a final boundary set that represents the boundaries of the moving objects. The merging step proceeds by first identifying boundary segment pairs that are significantly inconsistent. Then, for each inconsistent boundary segment pair, its associated curvature, edge response, and shadow index are used as criteria to evaluate the probable location of the true boundary. The resulting boundary is finally refined by estimating the width of the halo-like boundary and referring to the foreground edge map. This method consistently performs well under different illumination conditions, including indoor, outdoor, moderate, sunny, rainy, and dim cases.

2.2.3 Learning-based Trackers

In [WY13], the authors propose an object tracking method based on transfer learning. They train an autoencoder by using auxiliary natural images as feature extractor offline and then use an additional classification layer online. In [NH16], the authors use transfer learning for object tracking. Some layers in an offline-trained CNN are transferred to an online classifier with an updating layer of a binary classifier. This classifier produces some candidates around the previous target which are further evaluated to output the target. In [OP16], the authors use recurrent neural networks for the task of object tracking in 2D laser data for robotics applications. In [Zha+17], the authors use oblique random forests for object tracking. This method uses HOG features and deep neural network based models as the features. Incremental steps update the tracker.

Online multi-object tracking aims at estimating the tracks of multiple objects instantly with each incoming frame and the information provided up to the moment. It still remains a difficult problem in complex scenes, because of the large ambiguity in associating multiple objects in consecutive frames and the low discriminability between objects' appearances. In [BY18], the authors propose a robust online multi-object tracking method that can handle these difficulties effectively. First, the tracklet confidence is defined using the detectability and continuity of a tracklet. In this way, a multi-object tracking problem decomposed into small sub-problems based on the tracklet's confidence. Then, the online multi-object tracking problem is solved by associating tracklets and detections in different ways according to their confidence values. Based on this strategy, tracklets sequentially grow with online-provided detections, and fragmented tracklets are linked up with others without any iterative and expensive association steps. For the more reliable association between tracklets and detections, a deep appearance learning method learns a discriminative appearance model using large training data sets, since the conventional appearance learning methods cannot distinguish multiple objects with large appearance variations. In addition, online transfer learning for improving appearance discriminability is combined by adapting the pre-trained deep model during online tracking.

An efficient visual tracker is proposed in [YC18], which directly captures a bounding box containing the target object in a video by means of sequential actions learned using deep neural networks. The deep neural network to control tracking actions is pre-trained using various training video sequences and fine-tuned during actual tracking for online adaptation to a change of target and background. Pre-training is done by utilizing deep reinforcement learning (RL) as well as supervised learning. The use of RL enables even partially labeled data to be successfully utilized for semi-supervised learning.

An attention network for object tracking is proposed in [KP18]. To construct the attention network for sequential data, long-short-term memory (LSTM) and a residual framework into a residual LSTM (RLSTM) are combined. The LSTM, which learns temporal correlation, is used for temporal learning of object tracking. In the RLSTM method, the residual framework, which achieves the highest accuracy in ImageNet large scale visual recognition competition (ILSVRC) 2016, learns the variations of spatial inputs and thus achieves the spatio-temporal attention of the target object. Also, rule-based RLSTM learning is used for robust attention.

An effective combination of discriminative and generative tracking approaches is proposed in [AG17] in order to take the benefits from both. This algorithm exploits the discriminative properties of Faster R-CNN which helps to generate target specific region proposals. A new proposal distribution is formulated to incorporate information from the dynamic model of moving objects and the detection hypotheses generated by deep learning. The generative appearance model from the region proposals and perform tracking through sequential Bayesian filtering by variable rate color particle filtering (VRCPF) is constructed.

A novel event-triggered tracking framework is proposed in [GW18] for fast and robust visual tracking in the presence of model drift and occlusion. The resulting tracker not only operates in real-time but also is resilient to tracking failures caused by factors such as fast motion and heavy occlusion. Specifically, the tracker consists of an event-triggered decision model as the core module that coordinates other functional modules, including a short-term tracker, occlusion and drift identification, target re-detection, short-term tracker updating, and online discriminative learning for a detector. Each functional module is associated with a defined event that is triggered when the conditions are met. The occlusion and drift identification module is intended to perform an online evaluation of short-term tracking. When a model drift event occurs, the target re-detection module is activated by the event-triggered decision model to relocate the target and reinitialize the short-term tracker. The short-term tracker updating is carried out at each frame with a variable learning rate depending on the degree of occlusion. A sampling pool is constructed to store discriminative samples that are used to update the detector model. This tracker can effectively detect model drift and restore the tracking process.

2.2.4 Hybrid Trackers

In [ZS18], the authors combine 6 different trackers in a winner-take-all framework to improve the strength of the overall tracker against various challenges compared to the individual trackers. The selection method of the trackers is based on a performance prediction model. The authors propose a long-term motion tracker for intelligent vehicles. A set of independent classifiers were trained sequentially on different small data sets.

In [LSS05], the authors address the problem of detecting pedestrians in crowded real-world scenes with severe overlaps. This problem is too difficult for any type of model or feature alone. Instead, an algorithm that integrates evidence in multiple iterations and from different sources is presented. The core part of the method is the combination of local and global cues via probabilistic top-down segmentation. Altogether, this approach allows examining and comparing object hypotheses with high precision down to the pixel level. This method is able to reliably detect pedestrians in crowded scenes, even when they overlap and partially occlude each other. In addition, the flexible nature of the approach allows it to operate on very small training sets.

Local part-based human detectors are capable of handling partial occlusions efficiently and modeling shape articulations flexibly, while global shape template-based human detectors are capable of detecting and segmenting human shapes simultaneously. In [Lin+07], the authors describe a Bayesian approach to human detection and segmentation combining local part-based and global template-based schemes. The approach relies on the key ideas of matching a part-template tree to images hierarchically to generate a reliable set of detection hypotheses and optimizing it under a Bayesian MAP framework through global likelihood re-evaluation and fine occlusion analysis. In addition to detection, the approach is able to obtain human shapes and poses simultaneously.

In [YL18], the authors propose a semantics-aware visual object tracking method, which introduces semantics into the tracking procedure and extends the model of the object with explicit semantics prior to enhance the robustness of three key aspects of the tracking framework, i.e., appearance model, search scheme, and scale adaptation. First, a semantic object proposal generation method for the input video sequence to generate high-quality category-oriented object proposals is presented. Then a hybrid semantics-aware tracking algorithm with semantic compatibility is proposed. This algorithm takes full advantages of globally sparse semantic object proposal prediction and locally dense prediction with a template model and semantic distractor-aware color appearance model. Further, the tracker exploits semantics to localize the object accurately via an energy minimization framework based scale adaptation method, which jointly integrates dense location prior, instance-specific color and category-specific semantic information.

An implicit assumption in many generic object trackers is that the videos are blur free. However, motion blur is very common in real videos. The performance of a generic object tracker may drop significantly when it is applied to videos with severe motion blur. In [DH16], the authors propose a new Tracking-Learning-Data approach to transfer a generic object tracker to a blur-invariant object tracker without de-blurring image sequences. Before object tracking, a large set of unlabeled images is used to learn the objects' visual prior knowledge, which is then transferred to the appearance model of a specific target. During object tracking, online training samples are collected from the tracking results and the context information. Different blur kernels are involved with the training samples to increase the robustness of the appearance model to severe blur, and the motion parameters of the object are estimated in the particle filter framework.

There is a progress in the area of object detection by using learning-based techniques like deep learning. However, this progress was not extended to trackers. In this chapter, five famous training based object detectors i.e., ACF [Dol+14], R-CNN [Gir+14], fast R-CNN [Gir15], faster R-CNN [Sha+17] and YOLO [RF17] are considered for object tracking

Table 2.1: Distribution of object tracking properties over related methods.

	Multiple Target Tracking	Outdoor Application	Real-time Tracking	Moving Camera	Occlusion Handling
Ding et al. [DH16]		X		X	
Yao et al. [YL18]		X		X	X
Leibe et al. [LSS05]	X	X			X
Zheng et al. [ZS18]		X		X	X
Guan et al. [GW18]		X	X	X	X
Milan et al. [MSR15]	X	X		X	X
Milan et al. [MRS14]	X	X		X	X
Akok et al. [AG17]		X		X	X
Zhang et al. [KP18]		X		X	X
Yun et al. [YC18]		X		X	X
Bae et al. [BY18]	X	X	X	X	X
Zhang et al. [Zha+17]		X	X	X	
Ondruska et al. [OP16]	X			X	X
Nam et al. [NH16]	X	X		X	
Wang et al. [WY13]		X		X	
Chen et al. [CT18]	X	X		X	
Henriques et al. [Hen+15]	X	X	X	X	
Wu et al. [WN07]	X	X		X	X
Bolme et al. [Bol+10]		X	X	X	X
Wang et al. [Wan+18]		X		X	
Wang et al. [WYX14]	X	X		X	
Sakai et al. [Sak+15]		X		X	

and a comparative study among the detectors is done in this context. Two methods for offline tracking (training before tracking) and online tracking (training while tracking) are proposed. The former uses a pre-trained model for object detection in the space dimension (i.e., still images) and another offline trained classifier for the association of the objects in the time dimension. The latter is a short-term tracker with an online training procedure which updates the detector over time. In other words, the offline tracker divides the tracking task into two separate tasks: detection of objects in frames and finding the object of interest among the objects of each frame. The object detector performs the first part and the second part is a time series analyzer for tracking. The online tracker trains a detector with the positive and negative data generated from the first frame and then the detector is applied to the next frames of a certain part of the

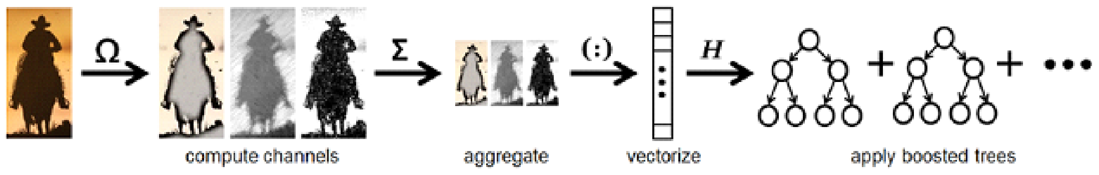

Figure 2.2: Block diagram of the steps of the ACF detector [Dol+14]

video; the detector is re-trained with the recently detected objects within the video part. To the best knowledge of the authors, the five mentioned detectors have not previously been compared in online and offline trackers. The online tracker is far different from two-step trackers [YC18; AG17; KP18; BY18] which first detect objects in images and then associate the detected objects using another classifier. In contrast to [WY13; NH16], the proposed online tracker does not have any offline phase.

2.3 Training-based Object Detection

This section highlights some state-of-the-art object detectors namely aggregated channel features (ACF) [Dol+14], region-based convolutional neural network (R-CNN) [Gir+14], fast R-CNN [Gir15], faster R-CNN [Sha+17], and YOLO [RF17] which are exploited for object tracking in the author's research in this chapter.

For ACF, a channel refers to a certain component that defines pixel values in the image [Dol+14]. For example, a color image is an aggregation of three channels (red, green and blue). The color data of an image are stored in three arrays of values, known as channels. By extracting specific information from the original image or by manipulating the input image, a desired channel is generated. After obtaining the channels from an input image, some features from the channels are extracted. These features are called channel features [Dol+14]. ACF uses decision trees for object detection and classification. The sum of every single channel is calculated to form the channel features. The procedure steps for ACF are shown in Figure 2.2. Given an input image, the procedure of ACF detection is as follows:

1. Several channels are computed by using the input image.
2. Sum of every block of pixels in the channels is calculated.
3. The resulting lower resolution channels are vectorized.
4. Boosting is used to learn decision trees over these features to distinguish the object from the background. Each parent node in the tree stores a binary function. The

binary function in the root node is evaluated on the data for each data point. The function value determines which child node is visited next. This continues until reaching a leaf node. The leaf node contains the response.

5. A multistage sliding-window approach is employed to localize the exact position of the object in the image. A sliding window determines whether it has the desired object or not. Therefore, the desired object is detected.

ACF [Dol+14] uses positive instances of objects in images that are used for training and it also automatically collects negative instances from the images during training.

Region-based convolutional neural network (R-CNN) is an object detector based on the convolutional neural network (CNN) and performs convolution products on small patches of its input and extracts local patterns [Li+18]. The input can be the image pixels or the output of the last layer. A typical CNN has the following layers [ON15]:

1. Input layer: The input layer is the input image given to the network or the output of the last layer.

2. Convolutional layer: Convolutional layers apply a convolution operation.

3. Pooling layer: Pooling reduces the dimensions of each feature map but retains the most important information. The output from the convolutional and pooling layers represent high-level features of the input image.

4. Fully connected layer: The purpose of the fully connected layer is to use these features for classifying the input image into various classes based on the training data set.

5. ReLU layer: ReLU stands for Rectified Linear Unit. The purpose of ReLU is to introduce non-linearity to CNN. Since most of the real-world data is non-linear, non-linearity in the network is necessary.

6. Softmax layer: It takes a vector of arbitrary real-valued scores and maps it to a vector of values between zero and one and the sum of the values in the vector is equal to one.

R-CNN combines region proposals with features computed by a CNN. First, R-CNN computes the region proposal using a selective search technique [Uij+13]. Selective search generates around 2000 region proposals that have the highest probability of containing an object. After coming up with a set of region proposals, these proposals are then warped into an image size that can be fed into a trained CNN that extracts a feature vector for each region. The image region which is located within the bounding box (selected by a user) is a positive example and the background is a region which does not have the object of interest, is a negative example. In order to decide whether to label a region as a positive or negative example, an IoU (Intersection over Union) threshold is used. The region proposals with greater than or equal to 0.5 IoU overlap with a ground-truth are positives and the rest are negatives. CNN of R-CNN has 3 convolutional layers and 2 fully connected layers. The procedure steps for R-CNN are shown in Figure 2.3 (left). R-CNN is slow because it performs a CNN for each object proposal, without sharing the computation.

Fast R-CNN improves the original R-CNN to increase the processing speed. To find out the reason for the slow implementation of R-CNN, suppose there are N region proposals, then R-CNN runs CNN N times that takes a lot of time. Fast R-CNN feeds the input image directly to CNN [Gir15]. It is able to solve the problem of the speed by sharing the computation of the convolution layers between different proposals, swapping the order of generating region proposals and running CNN. Since the convolutional layer does not change the spatial relationship between the adjacent pixels, it projects the coordinates in the raw image to the corresponding neuron in the convolution network. Therefore, the image can be computed through the convolution network once and the processing time is saved [Gir15]. The fast R-CNN is up to ten times faster than the R-CNN but it is not real-time. The procedure steps for fast R-CNN are shown in Figure 2.3 (middle) and described as follows:

1. The input image is fed to a convolution layer.
2. From each candidate, some features are extracted and then they are fed to an RoI pooling layer to decrease the dimension of the feature map.
3. Fully connected layers are applied for classification.
4. A softmax layer detects the object and a regression layer determines the object location.

Faster R-CNN [Sha+17] brings object detection toward the real-time application. It uses a region proposal network (RPN) after the last convolutional layer as shown in Figure 2.3 (right). RPN takes an image feature map as input and outputs a set of rectangular object proposals, each with an objectness score. Faster R-CNN models this process with a fully convolutional network. RPN detects whether the current region, which is generated from a sliding window and different anchors (for each location, some proposals with different aspect ration are parameterized relative to their reference boxes

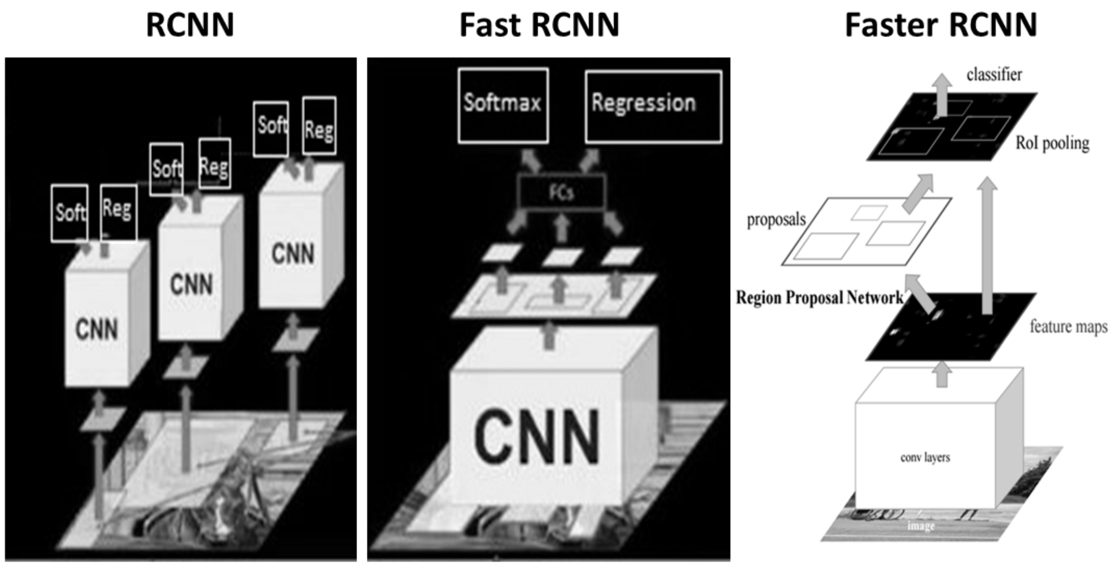

Figure 2.3: From left to right: The steps of R-CNN [Uij+13], fast R-CNN [Gir15] and faster R-CNN [Sha+17]

.

called anchors), is the desired object. When Faster R-CNN finds an object, the bounding box regression (regression determines how should a bounding box change to become more similar to the ground truth box) determines the location of the object. Faster R-CNN is up to ten times faster than fast R-CNN and it is real-time.

YOLO is state-of-the-art in object detection and based on deep networks [RF17], where a hierarchical structure was used for the sake of object labeling and classification. It stands for You Only Look Once, meaning that it only looks at the image one time and simultaneously processes the whole image. It performs detection and classification in real time. Instead of a large softmax in the Imagenet (another deep-learning-based object detector [Rus+15]), YOLO uses a group of softmax as a hierarchical tree of visual concepts, each softmax is deciding for a similar group of objects. In this way, YOLO classifies objects more accurately than Imagenet. It operates very fast due to its using much fewer floating point operations compared to VGG-16 [SZ14]. It applies a single CNN to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. Like Faster R-CNN, it adjusts priors on bounding boxes instead of predicting the width and height outright. However, it still predicts the x and y coordinates directly. Best results are achieved when the image size (input layer) is equal to 544×544; as the size decreases, so does the accuracy; however, the speed increases. YOLO is faster and more precise than faster R-CNN.

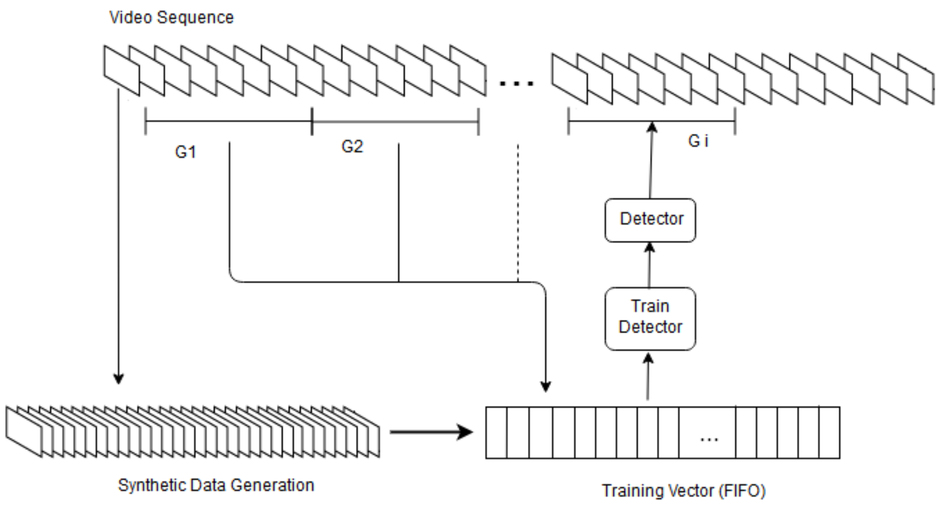

Figure 2.4: Block diagram of the proposed online tracker. In the first frame, from this selected object, synthetic data is generated. Then a detector (i.e., R-CNN, fast R-CNN, faster R-CNN or ACF) is trained using the generated data and applied to the first segment of frames G_1 to detect the objects of interest in them. The detected objects and the synthetic data added to the training vector T_v which is then used to update the detector. This process is continued until the end of the video.

2.4 Online Tracker

The steps of the proposed online object tracking method are shown in Figure 2.4. In the first frame, a user selects an object of interest. From this object, the online tracker generates synthetic data and trains a detector using synthetic data. It then segments the video frames into unequal-length groups of frames $\{\mathbf{G}_1, \mathbf{G}_2, ..., \mathbf{G}_n\}$. Each group is composed of several frames i.e., $\mathbf{G}_1 = \{\mathbf{F}_2, ..., \mathbf{F}_{m1}\}, \mathbf{G}_2 = \{\mathbf{F}_{m1+1}, \mathbf{F}_{m1+2}, ..., \mathbf{F}_{m2}\}, ..., \mathbf{G}_n = \{\mathbf{F}_{m(n-1)+1}, \mathbf{F}_{m(n-1)+2}, ..., \mathbf{F}_{mn}\}$.The trained detector tracks the first group $\mathbf{G}_1$ by detecting the object of interest within $\mathbf{G}_1$. The list of the objects of interest for the videos of this research is given in Table 2.4 in Section 2.6. At the end of the first iteration, the online tracker copies the detected objects within $\mathbf{G}_1$ as well as synthetic data to a training vector $\mathbf{T}_v$ (see Figure 2.4). The tracker uses the training vector $\mathbf{T}_v$ to train the detector for the second time and then applies the detector to the second segment $\mathbf{G}_2$ and then concatenates the detected objects within this segment to the training vector $\mathbf{T}_v$. The tracker steadily updates the training vector prior to each training iteration until the last frame of the video. It updates the detector by using a first input first output (FIFO) procedure. Thus, the tracker follows recent appearances of the object. Section of the experiments and results investigates the length of the training vector in terms of accuracy.

Let $\mathbf{I}_t^{x,y}$ denote the pixels of a single frame t, the following equation expresses the output $\mathbf{T}_t^n$ of the online tracker for this frame:

$$\mathbf{T}_t^n = D(\mathbf{I}_t^{x,y}, W^n(t)) \tag{2.1}$$

In this equation, $D(.)$ is the detector response which is a function of the image pixels and a trained network with the weights of $W^n(t)$. The tracker calculates initial weights for detection of the object of interest in the second frame using N synthetic positive and negative samples generated from the first frame (i.e., $W^n(1) = F_1(\mathbf{I}_1^1, \mathbf{I}_1^2, ..., \mathbf{I}_1^N)$). While updating the detector D in a certain interval Δt (shown in Figure 2.4), the weights $W^n(t)$ are updated by using synthetic data as well as the detected objects from the first frame to the frame before the current frame $(t-1)$.

$$W^n(t) = F_2(\mathbf{I}_1^1, \mathbf{I}_1^2, ..., \mathbf{I}_1^N, \mathbf{I}_2^d, \mathbf{I}_3^d, ..., \mathbf{I}_{t-1}^d) \tag{2.2}$$

Where $\mathbf{I}_i^d$ is the object of interest for the frame i. By using the online tracker and the four detectors of ACF, R-CNN, fast R-CNN and faster R-CNN, four online trackers are made. These trackers are compared in Section 2.6.

To have an effective tracker, the detectors must be trained using enough number of training data. The variety of data is also an important factor in the training process. To have such diversity, the following process is proposed:

1. Rotated copies of the object of interest using the rotating angles {-10, -9.9, -9.8, ... , 0 , ... , 9.9, 10}.
2. Additive salt and pepper noise, with noise densities {0.001, 0.002, ... , 0.008 }.
3. The enhanced version of the rotated images in item 1 using the contrast adjustment.
4. The enhanced version of the rotated images in item 1 using the histogram equalization.
5. Increasing and decreasing of the image brightness with brightness factors {0.81, 0.82, 0.83, ... ,1.29 , 1.3}.
6. Resized version of the object of interest using the resizing factor {0.555, 0.560, 0.565, ... , 1.55}.

The tracker combines the mentioned items to generate various data. Different total numbers of synthetic data in the experiments are used (i.e., 500, 800, 1000, 2000, 4000

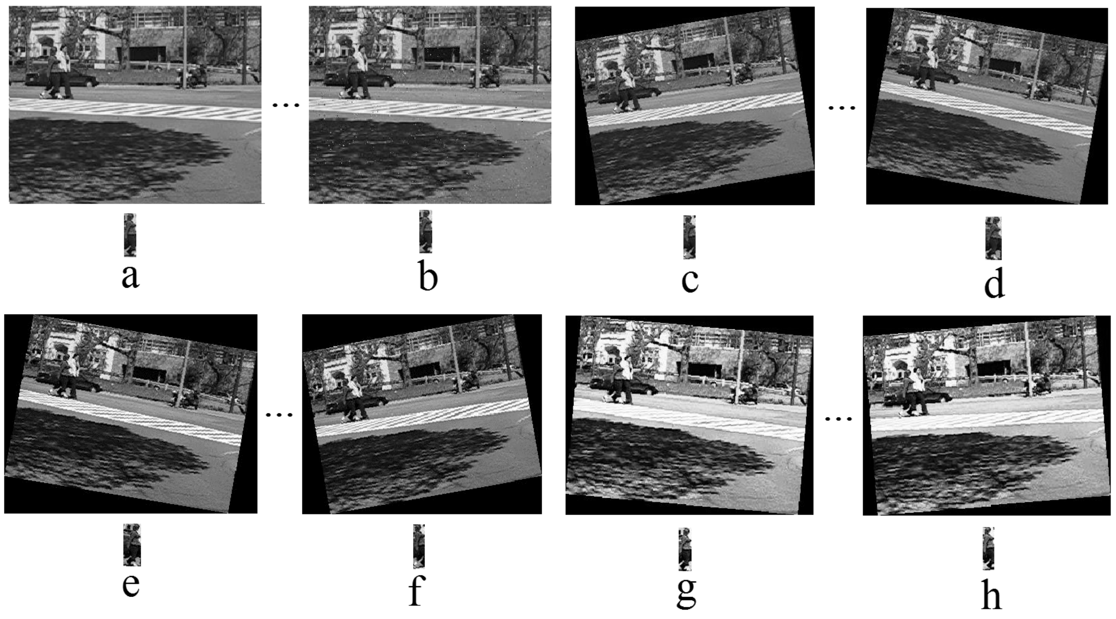

Figure 2.5: Synthetic data for the training of the detectors with their objects of interest; the first frame of "Pedestrain1"(a), this frame with additive salt and pepper noise, with noise density 0.008(b), rotated version of (a) with -10 and 9 degrees (c,d), rotated version of the enhanced image (a) using histogram equalization (e,f) and using contrast adjustment (g,h)

and 10000) to investigate the effect of this parameter on the tracker's accuracy. The number and types of synthetic data are given in Table 2.2. The parameters in this table are selected in an optimized way to preserve the tracker's accuracy and speed. Initially, a small training set was chosen and poor results were obtained. Then, the parameters were gradually optimized to maximize the tracker's accuracy and speed. Therefore, they are independent of the type of objects and videos.

Figure 2.5 shows examples of synthetic data for "Pedestrain1". Figure 2.5(a) shows the first frame and the selected object (object of interest), this frame is added by the salt and pepper noise, with a noise density 0.008 (Figure 2.5(b)) and rotated with -10 and 9 degrees (shown in Figure 2.5(c,d)), enhanced using histogram equalization and then rotated with 9 and -10 degrees (shown in Figure 2.5(e,f)) and enhanced using contrast adjustment and then rotated with 5 and -5 degrees (shown in Figure 2.5(g,h)). The object of interest for the selected frames is shown below each image.

The detected objects within frames of the first group $\mathbf{G}_1$ (i.e., $\{\mathbf{I}_2^d, \mathbf{I}_3^d, ..., \mathbf{I}_{m1}^d\}$) are concatenated to the training vector $\mathbf{T}_v = \{syntheticdata, \mathbf{I}_2^d, \mathbf{I}_3^d, ..., \mathbf{I}_{m1}^d\}$. By using the updated vector $\mathbf{T}_v$, the tracker retrains the detector. The updated detector is then used for tracking

Table 2.2: The number and types of synthetic data are given. The first row shows the total number of synthetic data and each column shows the number and types of synthetic data for a certain number of synthetic data.

Total number of synthetic data	**500**	**800**	**1000**	**2000**	**4000**	**10000**
First Frame	1	1	1	1	1	1
Additive Noise	2-9	2-9	2-9	2-9	2-9	2-9
Rotation	10-200	10-200	10-200	10-200	10-200	10-200
Rotation + Intensity Adjustment	201-400	201-599	201-400	201-400	201-400	201-400
Rotation + Contrast Enhancing	401-500	600-800	401-601	401-601	401-601	401-601
Brightness Change	-	-	602-1000	602-1000	602-1000	602-1000
Brightness Change + Resizing	-	-	-	1001-2000	1001-2000	1001-2000
Rotation + Brightness Change	-	-	-	-	2001-4000	2001-4000
Last Row + Intensity Adjustment	-	-	-	-	-	4001-10000

Table 2.3: Parameter sets for the online trackers

	t_1	t_2	t_3	t_4	t_5	t_6	l_1	l_2	l_3	l_4	l_5	l_6	l_7
$SET1$	200	300	500	1000	2000	5000	10	20	50	100	500	1000	-
$SET2$	200	410	450	500	1000	2000	20	30	40	50	100	500	1000

of frames (detection of the object of interest) in the second group $\mathbf{G}_2$. The detected objects within frames of $\mathbf{G}_2$ are again concatenated to the training vector $\mathbf{T}_v$ and updated vector $\mathbf{T}_v$ is used for tracking in the third group $\mathbf{G}_3$. The length of the groups $\{\mathbf{G}_1, \mathbf{G}_2, ..., \mathbf{G}_n\}$ is obtained from Equation 2.3. During training, when $\mathbf{T}_v$ becomes full, the data at the beginning of $\mathbf{T}_v$ is replaced by data from the current image (FIFO). This process continues until the end of the video. The length of $\mathbf{T}_v$ is investigated in terms of the tracker's accuracy in Section 2.6.

The length of each group l_s is a crucial parameter for this algorithm. Long segments decrease the tracker's stability, and on the other hand, short segments decrease the tracker's speed. After extensive experiments, the optimum lengths of segments were obtained. They are calculated using the following equation:

$$l_s = \begin{cases} l_1 & : k < t_1 \\ l_2 & : t_2 > k \geq t_1 \\ l_3 & : t_3 > k \geq t_2 \\ l_4 & : t_4 > k \geq t_3 \\ l_5 & : t_5 > k \geq t_4 \\ l_6 & : t_6 > k \geq t_5 \\ NoUpdate & : k \geq t_6 \end{cases} \tag{2.3}$$

Where $l_1,...,l_6$ are the lengths of the group of frames, k is the frame index, and $t_1,...,t_6$ are the thresholds of the segments' lengths according to Table 2.3. In the experiments for ACF, 2 sets of parameter $SET1$ and $SET1$ (shown in Table 2.3) are chosen and called ACF1 and ACF2 respectively. For R-CNN, fast R-CNN and faster R-CNN, $SET2$ is used.

2.5 Offline Tracker

The steps of the offline tracker are shown in Figure 2.6. This tracker feeds the video frames to YOLO-detector and the Kalman filter. The output of the offline tracker ($\mathbf{T}(.)$) is the YOLO response with the maximum IoU of the estimated pose by the Kalman filter. If there is no intersection between the two responses, the offline tracker selects the YOLO response with the lowest Euclidean distance to the Kalman filter response. The response of the Kalman filter is a point with the maximum probability of the object presence in the frame. A bounding box with the same size of the object in the last frame is drawn around the point to represent the object region. The output of the offline tracker $\mathbf{T}_t^f$ can be expressed using the following equation:

$$\mathbf{T}_t^f = \mathbf{T}(\mathbf{D}(\mathbf{I}_t^{x,y}, W^d), K(\mathbf{I}_t^{x,y})) \tag{2.4}$$

Where $\mathbf{D}(.)$ is the output of YOLO and $K(.)$ is the response of the Kalman filter. YOLO is a function of the pixels ($\mathbf{I}_t^{x,y}$) and weights of a deep network. Contrary to the online tracker, the weight of the offline detector W^d is not updated during tracking.

Detailed information about the Kalman filter is available in [Zor17b; Zor17a]. To estimate the position of the object of interest in the next frame P_{k-1}, the Kalman filter uses the object position $P(p_x, p_y)$ (center of mass) and its velocity $V(v_x, v_y)$ (see the following equations).

$$v_k = p_k - p_{k-1} \tag{2.5}$$

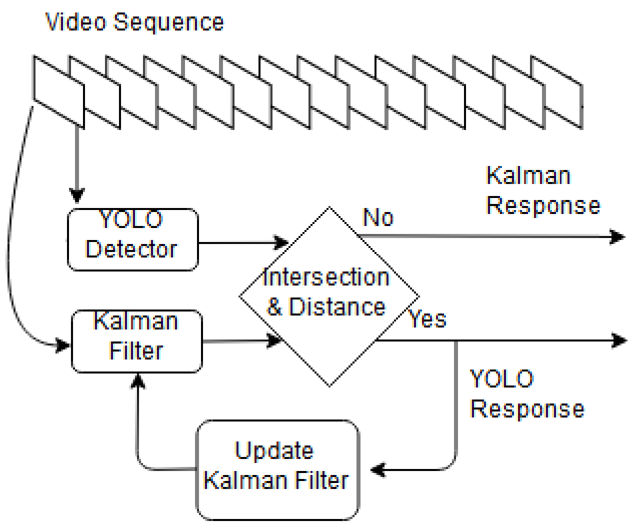

Figure 2.6: Block diagram of the proposed offline tracker. All frames are fed to YOLO and the Kalman filter. The offline tracker outputs the YOLO response which has the maximum IoU with the estimated pose of the Kalman filter. The YOLO response also updates the Kalman filter.

Where P_k and P_{k-1} are the position of the object of interest in the current and last frames respectively.

If YOLO does not detect any object in a frame or if the Euclidean distance between the response of the Kalman filter and the nearest YOLO response is more than a predefined threshold (100 pixels), the object is considered to be occluded with other objects or to have left the scene. YOLO threshold value is set to 0.15 to minimize the probable object missing. Lower and higher values lead to the high false alarm rate and the object missing respectively. Since YOLO has a strong pre-trained model for the object detection[1] and due to its slow training, YOLO is used in the offline tracker. The initial experiments showed that the detection rate of YOLO is high for the data set, but its classification accuracy is not precise. Therefore, the output labels of YOLO are ignored. Intermediate results of the offline tracker are shown in Figure 2.7. First, YOLO detects all the objects in the scene. Then, the Kalman filter selects the object of interest within each frame and associates them. Figure 2.7(a) shows three successive frames of "Volkswagen". The results of YOLO on the frames are shown in Figure 2.7(b). Figure 2.7(c) shows the Kalman filter's results on the detected objects of Figure 2.7(b) and Figure 2.7(d) shows the final results of the offline tracker.

[1]https://pjreddie.com/darknet/yolo/

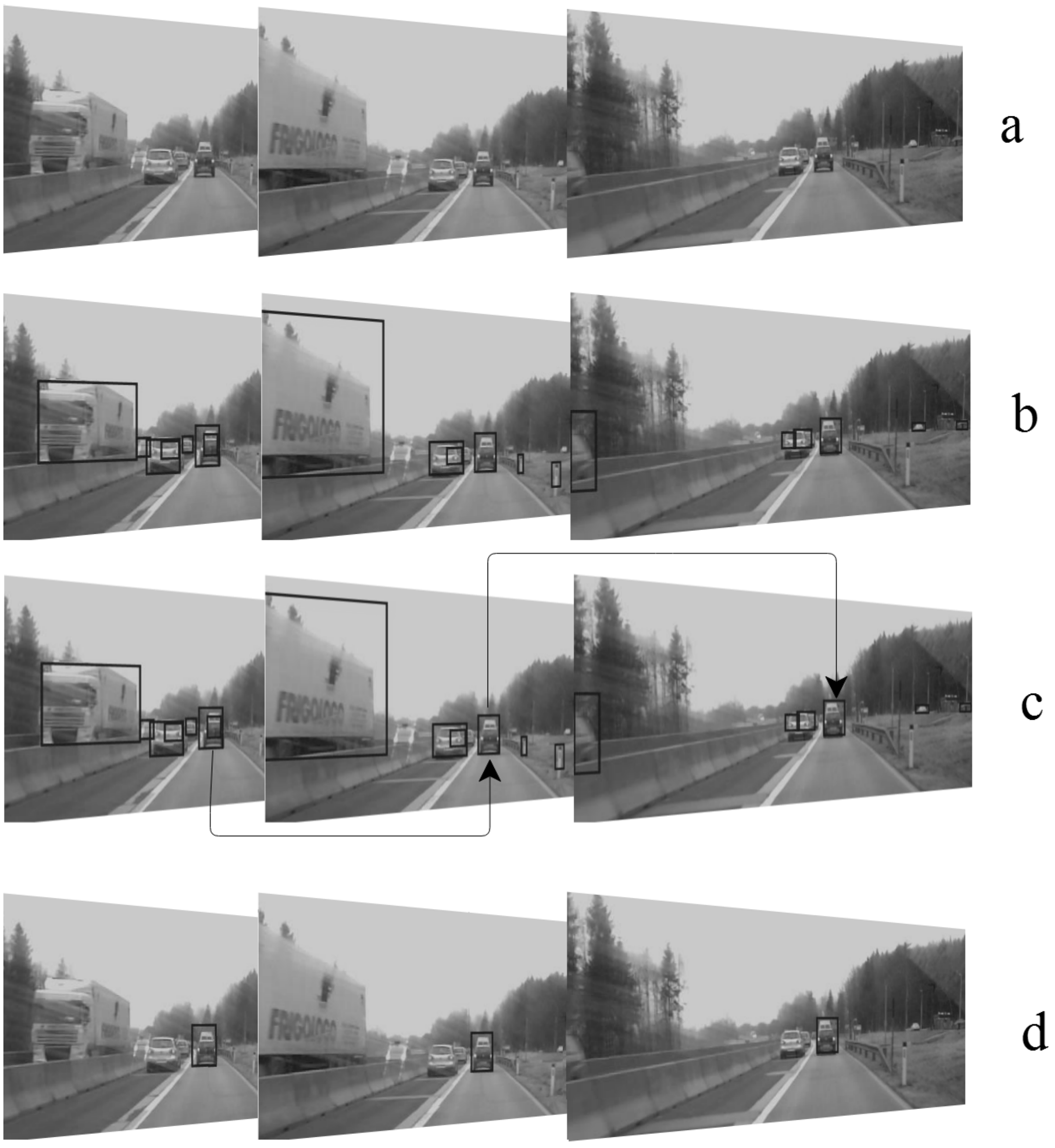

Figure 2.7: Intermediate results of the proposed offline tracker: (a) Three frames of "Volkswagen" (b) The results of YOLO detector are shown on the frames. YOLO detects all the cars in the scene. (c) The Kalman filter estimates the location of the object of interest in the next frame. The offline tracker selects the nearest object to the output of the Kalman filter. (d) The final results of the offline tracker. The Kalman filter converts YOLO from a detector to a tracker.

2.6 Experiments and Results

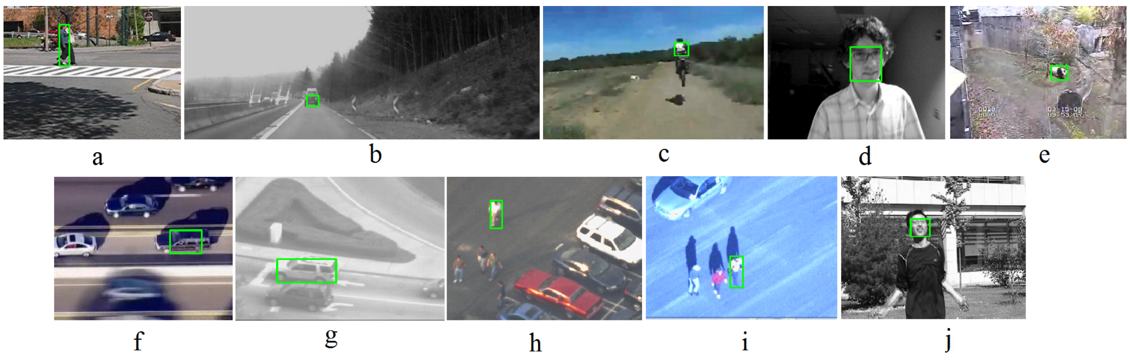

Figure 2.8: Snapshots of videos in the experiments. The figures from (a) to (j) show "David", "Jump", "Pedestrain1", "Pedestrain2", "Pedestrain3", "Car", "Motocross", "VW", "Car chase" and "Panda" respectively. The ground truth is shown on each image with a green rectangle.

Table 2.4: The characteristic of the TLD data set [KMM12]

Video	Frames #	Moving	Object	part. Occlusion	full Occlusion	Appea. change	Illum. change	Scale change	Similar objects
David	761	Moving	Human	Yes	No	Yes	Yes	Yes	No
Jumping	313	Moving	Human	No	No	No	No	No	No
Pedestrian1	140	Moving	Human	No	No	No	No	No	No
Pedestrian2	338	Moving	Human(top view)	Yes	Yes	No	No	No	Yes
Pedestrian3	184	Moving	Human(top view)	Yes	Yes	No	No	No	Yes
Car	945	Moving	Car	Yes	Yes	No	No	No	Yes
Motocross	2665	Moving	Motorcycle(rear view)	Yes	Yes	Yes	Yes	Yes	Yes
Volkswagen	8576	Moving	Car(rear view)	Yes	Yes	Yes	Yes	Yes	Yes
Car Chase	9928	Moving	Car(top view)	Yes	Yes	Yes	Yes	Yes	Yes
Panda	3000	Moving	Panda	Yes	Yes	Yes	Yes	Yes	No

To validate the proposed trackers, a set composed of 10 videos from TLD [KMM12] and VOT[2] data sets including more than 26800 frames and various objects of interest are selected. The data sets contain various types of tracking challenges like moving camera, long videos, object partial and full occlusion and appearance, illumination, scale change and similar objects which are listed in Table 2.4. Some videos in TLD data set are also available in other data sets like "Car chase" and "Pedestrain1" in VOT2018. Some videos are similar to others in the VOT2018. For instance, "Volkswagen" is similar to "Liver run", "Car1". "Yamaha" and "Traffic" are similar to "Motocross" and so on. They include various objects of interest for tracking like a car, a motorcycle, a car, a pedestrian, the human face,

[2]http://www.votchallenge.net/challenges.html

the human body and a panda as shown in Figure 2.8 and Table 2.4. Green rectangles in Figure 2.8 show the objects of interest. Among the objects of interest, "Car" is rigid and the other objects are articular. The data sets include short and long videos. To evaluate the proposed trackers, the sequences were manually annotated as ground truth [KMM12]. The plane rotation more than 50% was annotated as not visible [KMM12].

Table 2.5: Parameter set 1 regarding the online trackers.

Methods	**Length of T_v**	**Length of synthetic data**	**Stage**	**Epoch**
$ACF1$	614	210	5	-
$ACF2$	1000	500	3	-
$RCNN1, fastRCNN \& fasterRCNN$	1000	500	-	3
$RCNN2$	4000	4000	-	10

The parameters concerning online trackers are set according to Table 2.5. These parameters were selected using initial experiments to preserve the accuracy and speed of the trackers. Since the data sets contain various types of objects with different sizes and shapes, in different scenes, backgrounds, illumination conditions and different degrees of occlusion, the generality of the selected parameters is guaranteed. To generate synthetic data, the first frame of the video is exposed to a salt and pepper noise, rotation, intensity adjustment, histogram equalization, brightness change, resizing and contrast enhancement. The total number and the types of synthetic data are shown in Section 2.4. The experiments in this section use the following evaluation procedure. The trackers are initialized in the first frame of the video and track the object of interest (shown in Figure 2.8) up to the end. The produced trajectory is then compared to ground truth using the recall r, the precision p and the F-measure f. The F-measure is calculated using $f = 2 \times p \times r/(p + r)$. For each frame with a detected object, the object is considered to be correctly detected if the common area between the detected object and the ground truth is more than 50 percent [KMM12]. The number of the videos in this experiment is equal to the number of the videos used in other outstanding tracking methods like [KMM12]. The detailed results of the proposed trackers on the data sets in terms of Recall, Precision and F-measure are shown in Table 3.8.

The online R-CNN1 and online R-CNN2 trackers are trained with $SET1$ and $SET2$ (see Section 2.4) respectively. R-CNN2 is better than R-CNN1 for most videos. This experiment shows the preference of $SET2$ to $SET1$. $ACF2$ shows a better performance than $ACF1$ for most videos. It shows that the minimum segment size of frame groups should be set to 20 (discussed in Section 2.4). The parameter $SET2$ shows a better result

Table 2.6: Comparison of the trackers in terms of Recall (left number), Precision (left number) and F-measure (right number), the abbreviations of the videos are p_i: pedestrian i, Moto: motocross, VW: Volkswagen and Carc: Car chase

Method	**David**	**Jump**	P_1	P_2	P_3	**Car**	**Moto**	**VW**	**Carc**	**Panda**
ACF1	1/0.98/0.99	0.96/1/0.98	0.96/0.88/0.92	0.90/1/0.95	0.94/0.85/0.89	0.97/0.95/0.96	0.87/0.35/0.50	0.14/0.13/0.13	0.4/0.74/0.52	0.83/0.69/0.75
ACF2	0.99/0.99/0.99	0.87/0.99/0.93	0.96/0.82/0.88	0.84/1/0.91	0.93/0.96/0.94	0.93/0.98/0.95	0.97/0.56/0.71	0.78/0.65/0.71	0.78/0.64/0.70	0.84/0.85/0.85
R-CNN1	0.69/0.27/0.39	0.96/0.22/0.36	0.95/0.15/0.26	0.61/0.25/0.35	0.99/0.69/0.81	0.97/0.35/0.51	1/0.42/0.59	1/0.35/0.52	0.95/0.37/0.53	0.99/0.11/0.20
R-CNN2	0.97/0.48/0.64	0.96/0.36/0.52	0.74/0.15/0.25	0.75/0.56/0.64	0.99/0.67/0.80	0.99/0.90/0.94	0.98/0.40/0.57	0.96/0.57/0.72	0.94/0.45/0.61	0.99/0.37/0.54
fast R-CNN	0.90/0.19/0.31	1/0.01/0.02	1/0.29/0.45	0.80/0.21/0.33	1/0.73/0.84	0/0/0	1/0.31/0.47	1/0.13/0.23	0.99/0.29/0.45	0.40/0.07/0.12
Faster R-CNN	0.36/0.17/0.23	0.95/0.41/0.57	1/0.33/0.5	0.90/0.81/0.85	1/0.62/0.77	1/0.86/0.92	0.95/0.16/0.27	0.75/0.09/0.16	0.92/0.52/0.66	1/0.07/0.13
Offline ACF	0.72/0.49/0.58	0.30/1/0.46	0.40/0.49/0.44	0.20/1/0.33	0.89/0.99/0.93	0.59/0.81/0.68	1/0.39/0.56	0.43/0.66/0.52	0.45/0.68/0.54	0.08/0.83/0.15
YOLO	1/0.99/0.99	1/1/1	1/0.77/0.87	1/0.43/0.60	1/0.59/0.74	0.99/0.0.93/0.96	0.89/0.82/0.85	0.99/0.86/0.92	0.97/0.92/0.94	0.99/0.78/0.88

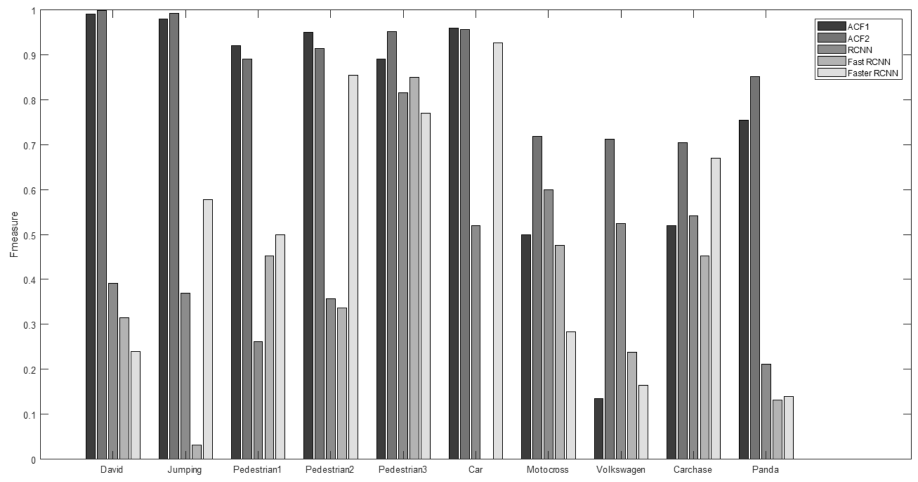

Figure 2.9: Tracker comparison in terms of F-measure

than $SET1$ and the smaller change in the step size (i.e., 20, 30, 40) shows better results. At the beginning of the video, the online tracker is not well-trained. Therefore, it should be updated in shorter intervals. The online tracker gradually adapts itself to the object of interest and the scene and therefore a longer interval for updating is enough. $SET2$ was chosen for the other online trackers i.e., fast R-CNN and faster R-CNN. Fast R-CNN and faster R-CNN show worse results than R-CNN and ACF for most of the videos. For the videos "Car chase", "Jump", "Pedestrain1" and "Pedestrain2", faster R-CNN shows a better result than R-CNN and fast R-CNN, but for 6 other videos, R-CNN is better. Faster R-CNN shows a better performance in the presence of similar objects. Among the online trackers, ACF2 has the best overall robustness, but YOLO is even more stable than ACF2. Figure 2.9 compares the performance of the trackers in terms of the F-measure.

In another experiment, the training vector length T_v, the number of training iterations

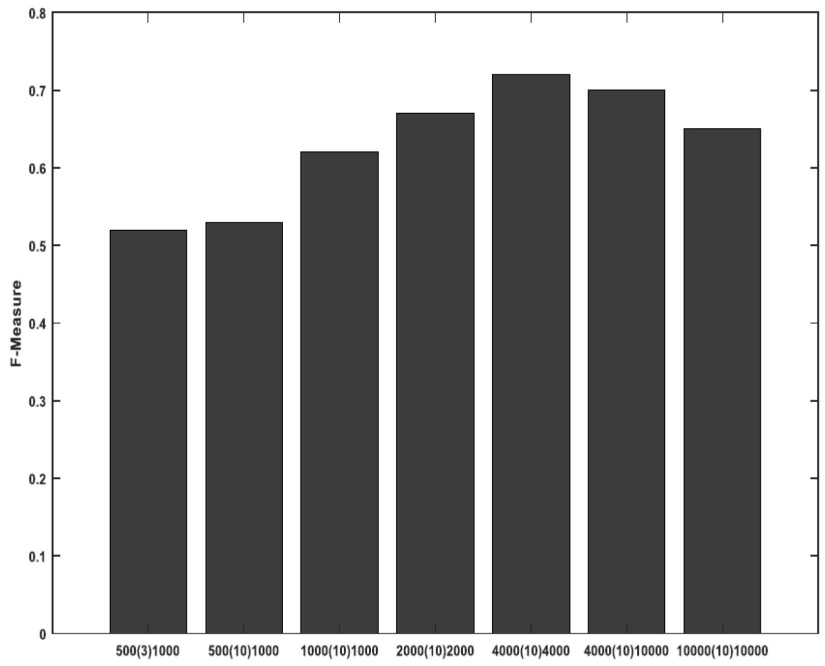

Figure 2.10: The tracker's performance versus the synthetic data length (left number), the training iteration number (middle number) and the training vector T_v length (right number).

and the total number of synthetic data are changed. To find the best accuracy of the trackers, different lengths of the synthetic data i.e., 500, 1000, 2000, 4000, 10000, and different training iterations i.e., 3 and 10 are tried and F-measure is calculated. The trackers' stability using average F-measure is shown in Figure 2.10. When the length of synthetic data is increased from 500 to 4000, the performance increases. But for the numbers longer than 4000, the performance decreases. Thus, the optimum length of the synthetic data is equal to 4000. With a training iteration increasing from 3 to 10, the trackers' stability increases. With a further increase, the tracker's speed decreases but its accuracy improves slightly. The training vector length T_v is increased from 1000 to 10000. In this case, when T_v exceeds 4000 (other parameters do not change) the accuracy falls down. Thus, the optimum T_v length is equal to 4000.

In an ablative study on the online ACF tracker, the updating/training process is removed. The results of this study are shown and compared in Figure 2.11. According to this figure, the tracker updating process increases accuracy. In another experiment, the synthetic data generation was removed. In this case, trackers cannot follow the object of interest.

The visualization comparison of the trackers for selected frames of the 10 videos and

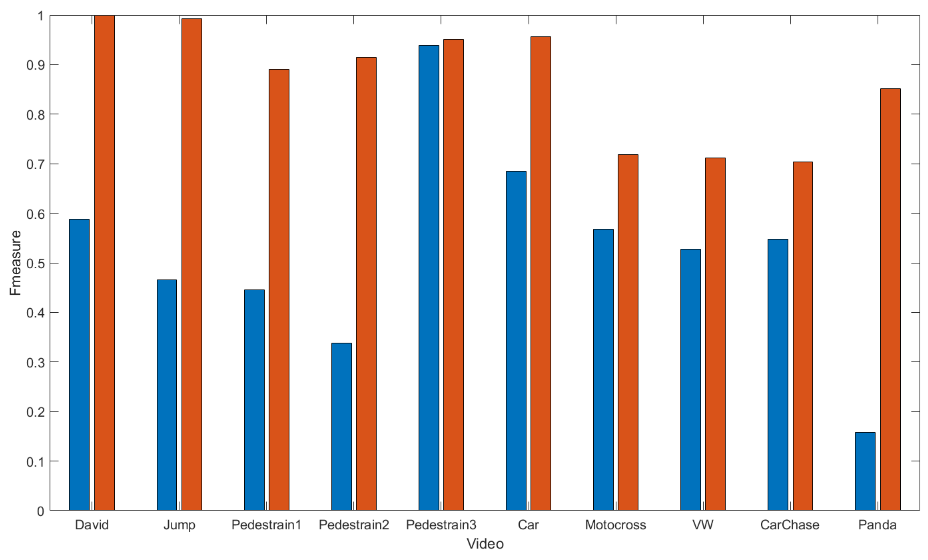

Figure 2.11: For the ACF tracker, the updating process was removed. The results of the online tracker and the ablative study are compared.

the ground truth is shown in Figure 2.12. From each video, two frames were randomly selected and shown in different rows.

The sequence in Figure 2.12(a) "Pedestrain1" has similar and articular objects and pose change. YOLO and ACF can follow the pedestrian very well but, R-CNN, fast R-CNN, and faster R-CNN miss it. The video in Figure 2.12(b) "Volkswagen" is a long video which contains similar objects (cars), occlusion and illumination change. R-CNN, ACF and YOLO show better results than faster R-CNN, and fast R-CNN is tracking the background. In the first frame except for fast R-CNN, the rest of the trackers can follow the car (the object of interest), but in the second frame only YOLO and ACF keep tracking the car. YOLO is more stable than ACF in this case. In this video, there are many frames without the desired car, but R-CNN and fast R-CNN trackers mistakenly follow another car in the scene. The sequence in Figure 2.12(c) "Motocross" includes appearance, illumination and scale change. In this example, except for R-CNN, the rest of the trackers show good results. The video in Figure 2.12(d) "David" contains partial occlusion, scale, and strong illumination change. In this case, R-CNN, fast R-CNN and faster R-CNN have poor results but ACF and YOLO show precise results. The sequence in Figure 2.12(e) "Panda" has occlusion, out-of-plane rotation, appearance and scale change. Fast R-CNN follows an incorrect object for both frames and R-CNN tracks one frame correctly and the other one incorrectly. The others i.e., YOLO, ACF, and fast R-CNN work well. The video in Figure 2.12(f) "Car chase" is a long video

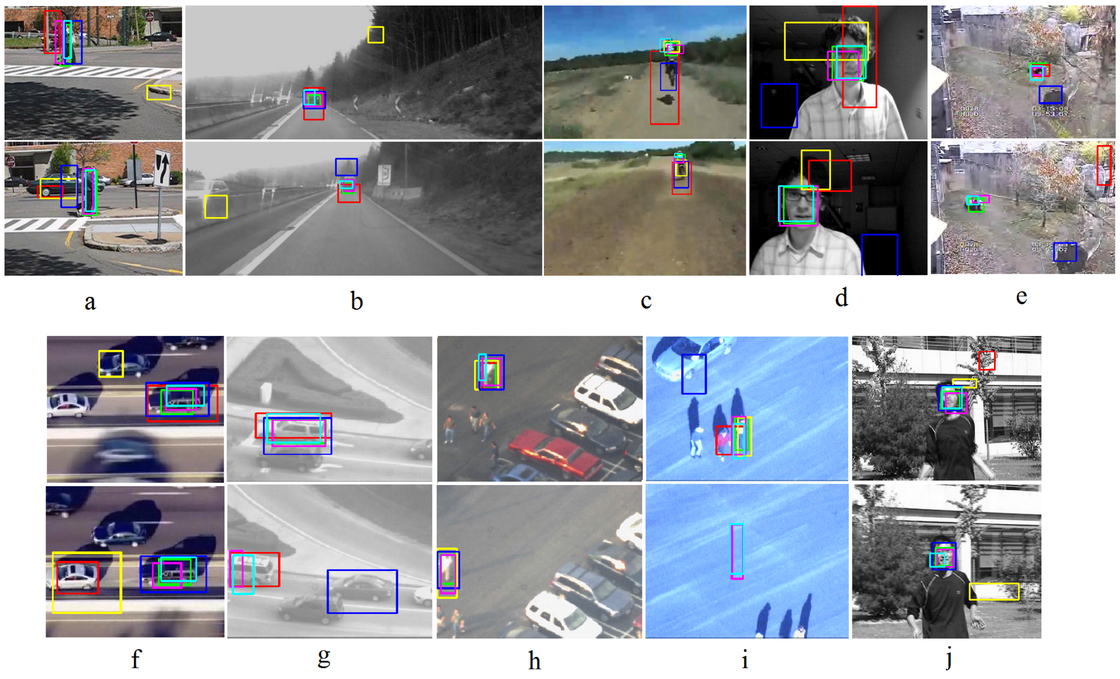

Figure 2.12: The visualization results of the proposed trackers are shown and compared. The results of R-CNN, fast R-CNN, faster R-CNN, ACF, YOLO and Ground truth are shown with red, yellow, blue, magenta, cyan and green frames respectively.

which includes occlusion, similar objects, scale and illumination change. R-CNN and fast R-CNN mistakenly follow another object whereas faster R-CNN, YOLO and ACF track the correct object. Figure 2.12(g) shows a sequence "Car" which has occlusion and similar objects. In this case, since the object (the white car) is leaving the scene, there is no ground truth for this frame. But R-CNN, YOLO, and ACF track the object of interest. Thus, this is considered as a false positive. Faster R-CNN tracker follows a similar object and fast R-CNN Tracker misses the object in both frames. The sequence in Figure 2.12(h) "Pedestrain3" includes similar objects and object occlusion. All the trackers except for YOLO can track the object of interest. In the first selected frame, YOLO partially detects the human, but it misses the object of interest in the second frame. In this case, YOLO cannot detect the human from the top view. Figure 2.12(i) shows a video "Pedestrain2" which has occlusion and similar objects. For the first frame, fast R-CNN, YOLO, and ACF can track the object correctly but R-CNN follows another human and faster R-CNN tracks a part of a car instead of the human. In the second frame, there is no human in the scene, but faster R-CNN tracks a point in the background. The video in Figure 2.12(j) "Jumping" contains a strong movement and blurring. In this case, ACF and YOLO track

Table 2.7: The specification of the hardware system for the experiments

	Brand	**Specification**
CPU	Intel	Core i7-7700K CPU @ 4.20GHz
GPU	GeForce	GTX 1080 Ti/PCIe/SSE2
Ram	Kingston	15.6 GB

correctly. Generally, among the tested trackers, ACF and YOLO trackers show better results.

The proposed trackers have been implemented on a hardware system with a specification as shown in Table 2.7. ACF, R-CNN, fast R-CNN and faster R-CNN were implemented using MATLAB. Except for ACF, the other trackers use GPU. The detection part of YOLO was implemented using C++ on a GPU machine, and the Kalman filter was implemented using C++ on CPU.

For each tracker, the average run time of all 10 videos has been measured and shown in Figure 2.13. The average fps (frame per second) for R-CNN, fast R-CNN, faster R-CNN, ACF and YOLO trackers are 0.2, 0.5, 0.24, 4 and 9 respectively. As shown in Figure 2.13, YOLO tracker has the fastest implementation because it does not use training while tracking. Among the online trackers, ACF-tracker (implemented in CPU) shows an effective implementation than R-CNN, fast R-CNN and faster R-CNN (implemented in GPU). Thus, ACF-tracker is faster than the rest of the online trackers. The main difference among the online trackers' speed happens in the training phase and therefore ACF is even faster than faster R-CNN in this phase.

2.7 Conclusions and Future Trends

A comprehensive comparative study in the context of object tracking using five famous recently proposed detectors was done in this chapter. Two trackers based on online and offline tracking were proposed. The online tracker first generates positive and negative synthetic data from the first frame and then trains the detectors. The detectors detect online the object of interest in the next frames and put the object in a training vector. The detector is updated at certain intervals using the training vector. The detector detects the objects of interest in the next frames. This procedure continues until the last frame. The ACF tracker showed the best results among the examined methods for online tracking from the speed and accuracy perspectives. In the offline scenario, YOLO generates some

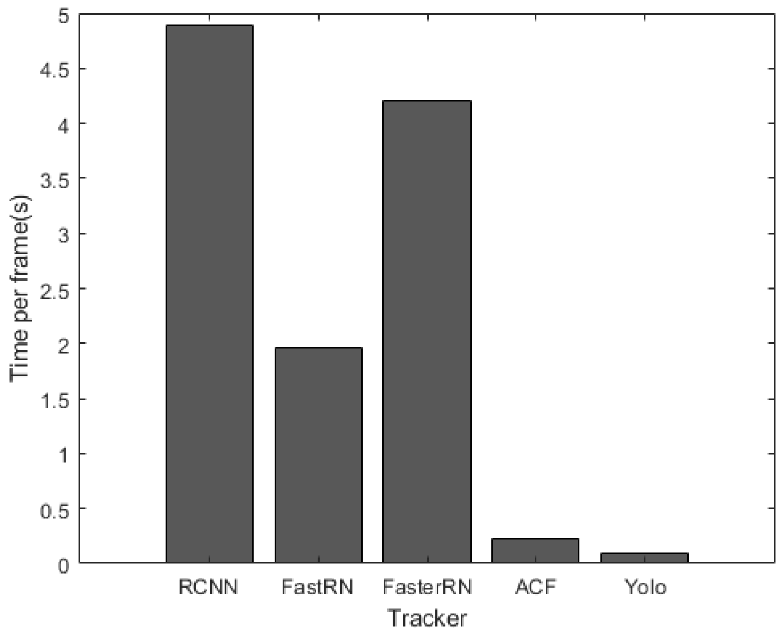

Figure 2.13: Comparison of Trackers in terms of run time is shown. Yolo tracker is the fastest tracker and ACF is the fastest one among the online trackers.

candidates, then the Kalman filter tracks the object of interest. Extensive experiments showed that YOLO outperforms the rest of the trackers.

This chapter is concluded as follows:

- The ACF tracker has the best results among the online trackers from both accuracy and speed viewpoints.
- Among the online trackers, ACF is the fastest tracker even though it has been implemented in CPU whereas the other trackers run on the CPU machine.
- Among the R-CNN based trackers (i.e., R-CNN, fast R-CNN, and faster R-CNN), R-CNN shows the best accuracy of tracking. Although fast R-CNN and faster R-CNN are very fast in the test phase, their tracking process is slow because they are very slow in the training phase. Since YOLO was implemented offline, it is the fastest tracker. YOLO is not qualified for online tracking, because it is very slow in the training phase.
- For human tracking from the front and side views, the combination of YOLO and the Kalman filter shows the best results. The use of the ACF tracker in unknown objects is recommended for tracking because YOLO does not detect these objects whereas the ACF tracker does.

- Compared to YOLO and ACF, the R-CNN based trackers show less accuracy because they do not have a very deep structure (i.e., 3 convolutional layers and 2 fully connected). By using a deeper convolutional neural network like YOLO the accuracy is dramatically increased. The training vector length was investigated and showed that the online tracker follows the recent appearances of the object of interest. The length should be set to an optimal value, for if it exceeds this value, the average accuracy decreases. On the other hand, the selection of shorter lengths leads to under-fitting and low accuracy.

- The results of this research are not limited to tracking. For instance, YOLO detects cars from the top view, but the precision of the object classification is low. For human detection, since YOLO was biased to the data from the front view, although the YOLO detection performance from this view is very good, the classification results are also disappointing.

For future work, YOLO detector will be trained using an updated data set to improve the detection results from the top view. The methods will be extended for multiple object tracking because all the detectors i.e., ACF, R-CNN, fast R-CNN, faster R-CNN and YOLO have the capability of multiple object detection. For the online trackers in each iteration of the training phase, instead of a single object, the future system will define multiple objects and the detectors will output different labels for different objects. In the case of offline tracking, YOLO is able to detect multiple objects as shown in this chapter. For each object, one Kalman filter will be used for tracking.

Chapter 3

Object Tracking in Polar Video

In this chapter, the proposed methods based on the author's contributions for object tracking in polar videos are described. Nowadays, the use of 360-degree cameras is intensively increasing. Object tracking by using these cameras is a potential task that needs to be viewed. The previous chapter proposed two methods for rectangular object tracking using an offline tracker and an online tracker based on training oriented object detectors. Due to the distortion and circularized shapes of the polar videos, the common methods for rectangular tackers do not apply to polar videos. This chapter presents the approaches for polar object tracking and is based on the author's following publications:

- The author's paper (C_5) has recently been published in 2018 in the journal of Multimedia Tools and Applications [Del+18]. The contributions of this paper to object tracking are a new circular polar object selection method, a region of interest selection technique, and the proposed color binary classifiers.
- The author contributes to object tracking in 360-degree videos by using SURF (Speeded Up Robust Features) and a training-based matching method (C_4) has been published in the International Symposium on Multimedia in 2017 [DG15].
- The author's paper has published a paper (C_2) [Del+16b] in the proceedings of the International Conference on Pattern Recognition in 2016. This paper modifies a famous tracker (TLD) [KMM12] to employ it for object tracking in 360-degree videos.
- The author's contribution to object tracking in 360-degree videos by using a polar trapezoid-shape method has been published in the proceedings of the International Symposium on Multimedia (C_3) [Del+16a].

This chapter has the following organization: Section 3.1 states the problem of object tracking in 360-degree videos and Section 3.2 then shows a literature review of object tracking in rectangular and polar videos. Videos from 360-degree and omni-directional cameras are compared in Section 3.3. Section 3.4 presents a modified training-learning-detection method to increase the robustness and speed of the TLD [KMM12] tracker especially against tracking challenges. A rectification-based object tracking method using YOLO (You Only Look Once) detector is then given in Section 3.5. Section 3.6 proposes a polar tracker which directly applies on polar videos and uses a trapezoid-shape technique for generating object candidates. Section 3.7 presents the author's proposed polar tracker based on SURF features and two training based classifiers for object matching and challenge detection. Section 3.8 presents the proposed polar tracker with a polar candidates generator and a combination of color information and binary features of videos. The experiments of the proposed polar trackers are given in Section 3.9. Section 3.10 concludes this chapter and states future work for polar object tracking.

3.1 Problem Statement

A 360-degree camera provides high-resolution videos with a wide field of view. The wide field of view allows the required number of installed cameras to be significantly decreased in comparison with the case of commonly used rectangular cameras. Thus, replacing conventional security systems with 360-degree cameras allows a significant reduction of hardware costs, software license, and maintenance costs. The wider field of view for conventional cameras increases their applications. For instance, the use of 360-degree cameras is growing in areas such as robotics, car traffic control, and intelligent surveillance systems.

With knowing the location of an object of interest in the first frame, the task of tracking this object becomes a fascinating topic for video processing from both scientific and industrial viewpoints. This task becomes scientifically more interesting when some complexities are involved in the videos. The complexities can include a moving-camera, uncertainty of the object, in- and out-of-the-plane rotation of the object, background clutter, small size and low resolution of the object, appearance changes, partial and full occlusion, and size variation within a video sequence caused by the object departing and approaching to the sensor. The object of interest is a unique object which the tacker is looking for. Figure 3.1 shows some objects of interest used in this chapter.

In this chapter, novel and scientific object tracking methods in the high convexity polar images from a high-resolution 360-degree camera are considered. This chapter

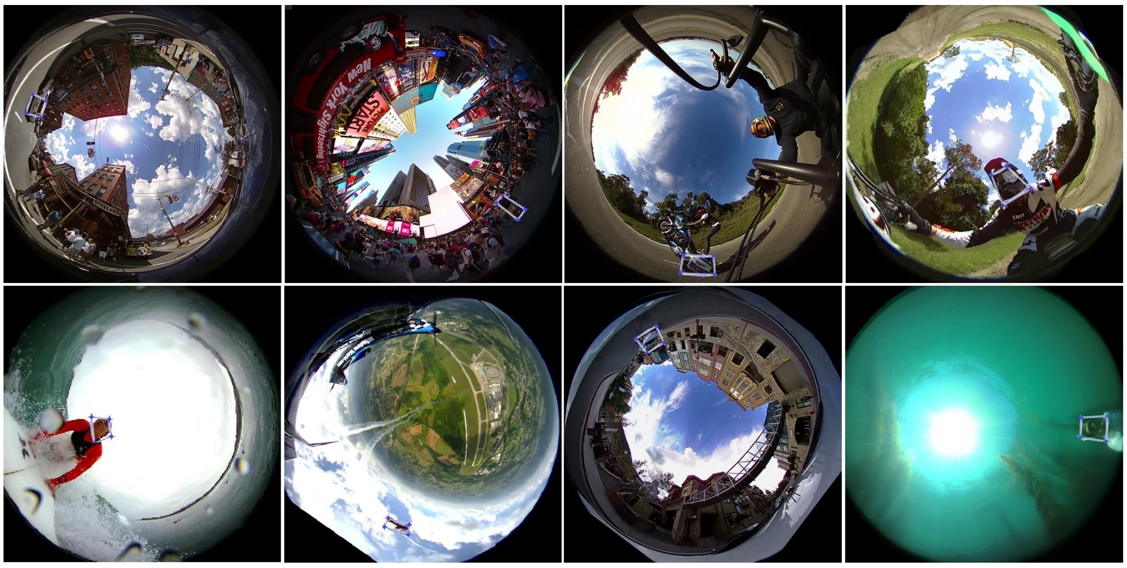

Figure 3.1: Some snapshots of 360-degree videos: Polar objects show the desired objects.

is composed of scientific projects with various techniques for object tracking in polar videos. A high-overlapping polar object selection is proposed to facilitate effective object selection (C_3,C_4). Also, a color binary classifier is introduced (C_5) to address the robustness and speed issues. This system can track arbitrary objects when the camera is moving and handle occlusion and scene leaving-returning issues. The first two proposed methods use an image rectification to convert the polar videos to rectangular ones and apply MTLD (C_2) and YOLO-based trackers to the rectified videos. The proposed polar methods take new approaches as compared to the other methods based on image rectification. The theoretical improvements are validated by experiments in Section 3.9.

3.2 Related Work

This section lists and shortly summarizes most related approaches for object tracking, starting with those using rectangular video followed by algorithms working with polar videos. Generally, object tracking methods based on their applications are grouped into the following categories:

1. Human Trackers: Most rectangular trackers have been fitted for human tracking. They usually simplify the object tracking problem for the easy case of human tracking.

2. Arbitrary Object Trackers: Some rectangular trackers aimed to track arbitrary objects. The object can be car, airplane, animal, human and so on.

3. Object Tracking for Robotics: Polar object trackers usually are used for navigating a robot, a vehicle, a helicopter and so on.

4. Object Tracking for Video Surveillance: Due to the wide field of polar cameras especially pan-tilt-zoom (PTZ) and fisheye cameras, the polar object trackers are also employed for the surveillance applications for monitoring humans or objects.

3.2.1 Rectangular Object Tracking for Human Following

Recently, some methods have been developed to overcome the challenges of object tracking in rectangular videos. Most of them focus on the same problem and the same data sets and gradually improve the tracking performance. There are many other methods for human detection and tracking in literature [Lin+07; WYX14; LSS05; WY10; MRS14; MSR15]. These methods often use very simple features to detect humans. The human body is usually described by some simple shapes such as a circle for the human's head and a cylindrical shape for other body parts. Thus, a very commonly used method is modeling the human appearance to 2d shapes [Lin+07] or 3d shapes [WY10]. Some other methods use offline-trained object detection for human tracking [LSS05]. In [X.W+16], tracking of different kinds of interacting objects is formulated as a network-flow integer program. This is made possible by tracking all objects simultaneously using intertwined flow variables and expressing the fact that one object can appear or disappear at locations where another is in terms of linear flow constraints.

There are some trackers focusing on human tracking like the tracker proposed in [XSL15]. This method uses an adaptive clustered decision tree for single object tracking and selects the minimum combination of necessary features to represent each target in frames. This method can successfully track various objects in the challenging condition of rectangular videos.

Correlation-based trackers (KCF) [Hen+15; Bol+10; Din+18] have gained an excellent performance and show robustness to challenging situations like motion blurring and illumination changes. In [Din+18], the authors recently improved the scalability of the KCF tracker. However, since KCF trackers depend strongly on the spatial layout of the tracked object, they are fragile to deformation, occlusion and camera shaking [Ber+16; Zha+17]. This drawback can be compensated by combining the correlation-based system to color information [Ber+16] or adding some output constraints [Zha+17] to the KCF tracker.

Common methods of object tracking are not able to track the object in the presence of some or all of the mentioned complexities. The problem of polar object tracking is much more complicated than rectangular tracking. Due to the distortion of polar images caused by natural convexity, concavity, and rotation of objects in the polar images, the conventional rectangular object selection [MRS14; MSR15; KMM12] does not apply. Moreover, since no information about the object of interest is available, template-based and offline-trained object tracking methods are not possible. Instead, other methods based on energy minimizing and online-training techniques are more practical [MRS14; MSR15]. The online-training method is usually fragile to the out-of-plane rotation and background clutter. A Gaussian dynamical model and an annealed particle filtering were used for human motion tracking in [Liu+12; Cui+13].

In [Wen+16], an algorithm is proposed that formulates the multi-object tracking task as one to exploit hierarchical dense structures on an undirected hyper-graph constructed based on tracklet affinity. The dense structures indicate a group of vertices that are interconnected with a set of hyper-edges with high affinity values. The appearance and motion similarities among multiple tracklets across the spatio-temporal domain are considered globally by exploiting high-order similarities rather than pairwise ones, thereby facilitating the distinguishability of spatially close targets with a similar appearance. In addition, the hierarchical design of the optimization process helps the proposed tracking algorithm handle long-term occlusions robustly.

In object tracking, it is critical to explore the data associations by exploiting the temporal information from a sequence of frames rather than the information from the adjacent two frames. Since straightforwardly obtaining data associations from multi-frames is an NP-hard multi-dimensional assignment (MDA) problem, most existing methods solve this MDA problem by either developing complicated algorithms or simplifying MDA as a 2D assignment problem based upon the information extracted only from adjacent frames. In [He+16], it is shown that the relation between associations of two observations is the equivalence relation in the data association problem, based on the spatial-temporal constraint that the trajectories of different objects must be disjoint. Therefore, the MDA problem can be equivalently divided into independent sub-problems by equivalence partitioning. In contrast to existing works for solving the MDA problem, the authors of [He+16] developed a connected component model (CCM) by exploiting the constraints of the data association and the equivalence relation on the constraints. Based upon CCM, the global solution of the MDA problem for object tracking can be efficiently obtained by optimizing a sequence of independent data association sub-problems.

The task of tracking multiple targets is often addressed with the so-called tracking-by-detection paradigm, where the first step is to obtain a set of target hypotheses for each frame independently. Tracking can then be regarded as solving two separate, but tightly coupled problems. The first is to carry out data association, i.e., to determine the origin of each of the available observations. The second problem is to reconstruct the actual trajectories that describe the spatio-temporal motion pattern of each individual target. The former is inherently a discrete problem, while the latter should intuitively be modeled in a continuous space. Dealing with an unknown number of targets, complex dependencies, and physical constraints are all challenging tasks on their own and thus, most previous work focuses on one of these sub-problems. In [MSR15], a multi-target tracking approach is presented that explicitly models both tasks as minimization of a unified discrete-continuous energy function. Trajectory properties are captured through global label costs, a recent concept from multi-model fitting, which is introduced to tracking. Specifically, label costs describe physical properties of individual tracks, e.g., linear and angular dynamics, or entry and exit points. Furthermore, the paper introduces pairwise label costs to describe mutual interactions between targets in order to avoid collisions. By choosing appropriate forms for the individual energy components, powerful discrete optimization techniques can be leveraged to address data association, while the shapes of individual trajectories are updated by gradient-based continuous energy minimization.

Many recent advances in multiple target tracking aim at finding a (nearly) optimal set of trajectories within a temporal window. To handle the large space of possible trajectory hypotheses, it is typically reduced to a finite set by some form of data-driven or regular discretization. In [MRS14], an alternative formulation of multitarget tracking as minimization of a continuous energy was proposed. Contrary to recent approaches, in [MRS14], authors focus on designing an energy that corresponds to a more complete representation of the problem, rather than one that is amenable to global optimization. Besides the image evidence, the energy function takes into account physical constraints, such as target dynamics, mutual exclusion, and track persistence. In addition, the partial image evidence is handled with explicit occlusion reasoning and different targets are disambiguated with an appearance model. Nevertheless to find the strong local minima of the proposed non-convex energy, a suitable optimization scheme that alternates between continuous conjugate gradient descent and discrete trans-dimensional jump moves is constructed. These moves, which are executed so that they always reduce the energy, allow the search to escape weak minima and explore a much larger portion of the search space of varying dimensionality.

3.2.2 Rectangular Object Tracking for Arbitrary Object Following

In [WK16] a multi-cue object tracking framework using the particle filter has been proposed to more efficiently handle various dynamic environment conditions such as partial or full occlusion, illumination changes, weather-related obstructions, and poor visibility. In [WZM12], the authors also use particle filter and Markov Chain Monte Carlo to improve the tracking robustness and speed. A method which uses Hough transform for hand detection and SIFT for generating and matching of key-points has been proposed in [WK16] for fingertip tracking. Authors in [Cho+15] propose a geogram-based feature descriptor to improve the tracking robustness and a hybrid technique of geogram-histogram to control the convergence of the mean-shift based tracker. In [Mia+11; SNH12] a SURF feature descriptor was used for object tracking in rectangular images.

Visual attention is a crucial indicator of the relative importance of objects in visual scenes to human viewers. In [Kar+15], an algorithm to extract objects which attract visual attention from videos is proposed. As human attention is naturally biased towards high level semantic objects in visual scenes, this information can be valuable to extract salient objects. The proposed algorithm extracts dominant visual tracks using eye tracking data from multiple subjects on a video sequence by a combination of mean-shift clustering and Hungarian algorithm. These visual tracks guide a generic object search algorithm to obtain candidate object locations and extend it to every frame. Further, [Kar+15] proposes a multiple object extraction algorithm by constructing a spatio-temporal mixed graph over object candidates. Bounding box based object extraction inference is performed using binary linear integer programming on a cost function defined over the graph. Finally, the object boundaries are refined using grab-cut segmentation. Feature pooling in a majority of sparse coding based tracking algorithms computes final feature vectors only by low-order statistics or extreme responses of sparse codes. The high-order statistics and the correlations between responses to different dictionary items are neglected. In [Ma+16], a more generalized feature pooling method for visual tracking is presented which utilizes the probabilistic function to model the statistical distribution of sparse codes. Since immediate matching between two distributions usually requires high computational costs, the Fisher vector to derive a more compact and discriminative representation for sparse codes of the visual target is introduced. The approach encodes target patches by local coordinate coding, utilizes a Gaussian mixture model to compute Fisher vectors, and finally trains semi-supervised linear kernel classifiers for visual tracking. In order to handle the drifting problem during the tracking process, these classifiers are updated online with current tracking results.

Under a tracking framework, the definition of the target state is the basic step for automatic understanding of dynamic scenes. More specifically, far object tracking raises challenges related to the potentially abrupt size changes of the targets as they approach the sensor. If not handled, size changes can introduce heavy issues in data association and position estimation. This is why adaptability and self-awareness of a tracking module are desirable features. The paradigm of cognitive dynamic systems (CDSs) can provide a framework under which a continuously learning cognitive module can be designed. In particular, CDS theory describes a basic vocabulary of components that can be used as the founding blocks of a module capable of learning behavioral rules from continuous active interactions with the environment. This quality is fundamental to deal with dynamic situations. In [Maz+16], a general CDS-based approach for tracking is proposed. It is shown that such a CDS-inspired design can lead to the self-adaptability of a Bayesian tracker in fusing heterogeneous object features, overcoming size change issues.

In [XSL15], the authors present a method for single target tracking of arbitrary objects in challenging video sequences. Targets are modeled at three different levels of granularity i.e., pixel level, parts-based level, and bounding box level, which are cross-constrained to enable robust model relearning. This work presents an adaptive clustered decision tree method which dynamically selects the minimum combination of features necessary to sufficiently represent each target part at each frame, thereby providing robustness with computational efficiency. The adaptive clustered decision tree is implemented in two separate parts of the tracking algorithm: firstly to enable robust matching at the parts-based level between successive frames and secondly to select the best super-pixels for learning new parts of the target.

Semantic object segmentation in a video is an important step for the large-scale multimedia analysis. In many cases, however, semantic objects are only tagged at video-level, making them difficult to be located and segmented. To address this problem, in [Zha+15] an approach to segment semantic objects in weakly labeled videos via object detection is proposed. In this approach, a video segmentation-by-detection framework is introduced, which first incorporates object and region detectors pre-trained on still images to generate a set of detection and segmentation proposals. Based on the noisy proposals, several object tracks are then initialized by solving a joint binary optimization problem with min-cost flow. As such tracks actually provide rough configurations of semantic objects, the object segmentation is refined while preserving the spatio-temporal consistency by inferring the shape likelihoods of pixels from the statistical information of tracks.

3.2.3 Polar Object Tracking for Robotics Application

An omnidirectional camera (from "Omni", meaning all) is a camera with a 360-degree field of view in the horizontal plane, or with a visual field that covers (approximately) the entire sphere. While it is not limited to a single viewpoint and can observe a large area, it distorts the objects in the scene. For this reason, existing methods for object tracking in 2D rectangular videos cannot be applied for these data without appropriate modifications. There are some methods using a similar type of polar camera (omnidirectional) in various applications. They are based on conventional low-resolution omnidirectional cameras. Many of them have achieved good performances in a simple scenario and special application like robotics. In [SS08] a SIFT-based tracking method is used to compute the motion (odometry) of a robot for image mosaicing outside. It uses image rectification (unwrapping) for low-resolution omnidirectional camera images. Due to application type (odometry), there is enough time to compensate for the likely errors of the tracker by using the next frames. Thus, the perfect precision of the tracker is not necessary in this case.

A technique for applying visual tracking algorithms to omnidirectional videos is presented in [Ram+07]. The method is based on a spherical image representation which allows taking into account the distortions and nonlinear resolution of omnidirectional images.

Using stereo disparity or depth information to detect and track moving objects has received increasing attention in recent years. However, this approach suffers from some difficulties, such as synchronization between two cameras and doubling of the image-data size. Besides, traditional stereo-imaging systems have a limited field of view (FoV), which means that they need to rotate the cameras when an object moves out of view. In [Xio+12], the authors present a depth-space partitioning algorithm for performing object tracking using a single-camera Omni-stereo imaging system. The proposed method uses a catadioptric omnidirectional stereo-imaging system to capture Omni-stereo image pairs. This imaging system has 360-degree FoV, avoiding the need for rotating cameras when tracking a moving object. In order to estimate Omni-stereo disparity, the authors present a depth-space partitioning strategy. It partitions three-dimensional depth space with a series of co-axial cylinders models the disparity estimation as a pixel-labeling problem and establishes an energy minimization function to solve this problem using graph cuts optimization. Based on the Omni-stereo disparity-estimation results, the authors detect and track-moving objects based on Omni-stereo disparity motion vector, which is the difference between two consecutive disparity maps.

Equipping mobile robots with an omnidirectional camera is very advantageous in numerous applications as all information about the surrounding scene is stored in a single image frame. [MCP14] is concerned with detection, tracking and following of a moving object with an omnidirectional camera. The camera calibration and image formation are based on the spherical unified projection model thus yielding a representation of the omnidirectional image on the unit sphere. Detection of moving objects is performed by calculating a sparse optical flow in the image and then lifting the flow vectors on the unit sphere where they are discriminated as dynamic or static by analytically calculating the distance of the terminal vector point to a great circle arc. The flow vectors are then clustered and the center of gravity is calculated to form the sensor measurement. Furthermore, tracking is posed as a Bayesian estimation problem on the unit sphere and the solution based on the Mises-Fisher distribution is utilized. Visual servoing is performed for the object following task where the control law calculation is based on the projection of a point on the unit sphere.

In a method presented in [ZLS17], the authors utilize background subtractor from OpenCV Library which creates a continuously updating background model for motion detection. The model is subtracted from the current frame leaving contours symbolizing the movement observed in the camera view. These contours are then analyzed and processed so that the system can track the largest contour. The tracked movement is outlined and directed to the user via virtual reality (VR) headset. The VR headset only displays a 60-degree portion of the camera view to the user which provides more realistic situational awareness of the surroundings for the user. These activities are a part of a larger effort to establish a foundation for autonomous unmanned vehicle systems with situational awareness capabilities.

A real-time algorithm for computing the ego-motion of a vehicle relative to the road is described in [SS08]. The algorithm uses as input only those images provided by a single omnidirectional camera mounted on the roof of the vehicle. The front ends of the system are two different trackers. The first one is a homography-based tracker that detects and matches robust scale-invariant features that most likely belong to the ground plane. The second one uses an appearance-based approach and gives high-resolution estimates of the rotation of the vehicle. This planar pose estimation method has been successfully applied to videos from an automotive platform. An example of camera trajectory estimated purely from omnidirectional images over a distance of 400m is given. For performance evaluation, the estimated path is superimposed onto a satellite image. In the end, image mosaicing is used to obtain a textured 2-D reconstruction of the estimated path.

The design and implementation of an omnidirectional vision system used for sideways-looking sensing on an autonomous helicopter are proposed in [HS03]. To demonstrate the capabilities of the system, a visual servoing task was designed which required the helicopter to locate and move towards the centroid of a number of visual targets. Results are presented showing that the task was successfully completed by a Pioneer ground robot equipped with the same omnidirectional vision system, and preliminary test flight results show that the system can generate appropriate control commands for the helicopter.

The authors in [KS07] present a moving obstacle detection method using optical flow in a mobile robot with an omnidirectional camera. Because an omnidirectional camera consists of a nonlinear mirror and CCD camera, the optical flow pattern in an omnidirectional image is different from the pattern in a perspective camera. The geometry characteristic of an omnidirectional camera has influences on the optical flow in an omnidirectional image. When a mobile robot with an omnidirectional camera moves, the optical flow is not only theoretically calculated in an omnidirectional image but also investigated in omnidirectional and panoramic images. The panoramic image is generalized from an omnidirectional image using the geometry of an omnidirectional camera. In particular, focus of expansion (FoE) and focus of contraction (FoC) vectors are defined from the estimated optical flow in omnidirectional and panoramic images. FoE and FoC vectors are used as reference vectors for the relative evaluation of optical flow. The moving obstacle is turned out through the relative evaluation of optical flows.

Detecting moving objects based on the camera attached in a mobile robot is not trivial since both background and object are moving independently. For moving object detection the movement of moving object needs to be extracted by considering the background which has also changed by the ego-motion of the mobile robot. Affine transformation is widely used to estimate the background transformation between images. However, when using an omnidirectional camera, the mixed motion of scaling, rotation, and translation appears in local areas and single affine transformation is not sufficient to describe those mixed nonlinear motions. The method presented in [Oh+12] divides the image into grid windows and obtains each affine transform for each window. This method can obtain a stable background transformation when the background has few corner features. The area of moving objects can be obtained from the background transformation-compensated frame difference using every local affine transformation for each local window.

A simple yet effective method for online and real-time multi-object tracking in 360-degree equirectangular panoramic videos is proposed in [LL18]. Based on the current state-of-the-art tracking-by-detection paradigm, several improvements have been made to overcome the challenges of full field-of-view (FoV) of spherical panoramic cameras (SPC). In addition, two data sets are presented for evaluation.

The nonuniform resolution of the omnidirectional camera has been used in [Sco+04] for human tracking based on a dual camera system. Multiple cues such as color, shape, and position are selected as human tracking features. The background of images in the fixed camera scenarios is still. Thus, by using background detection and subtraction, one can easily detect foreground objects. In such a case, the area of interest is dramatically limited. Therefore, finding the object of interest in the limited area can be precise and fast. For object tracking in omnidirectional videos, trackers in the literature [MCP14; Dem+12; KS07; Oh+12; SS08] are not applicable to the 360-degree images. In fact, they design their approach for a specific condition/application as it will be explained in the next paragraphs. The methods focus on an indoor environment for chasing objects [MCP14; Dem+12; KS07; Oh+12; HS03].

For the polar object tracking some methods [KS07; Oh+12; HS03; Del+16b; SS08] use image rectification for all frames prior to the tracking process and then an optical flow for tracking. The computational load for rectification is manageable in this case due to the limited resolution of the omnidirectional camera (less than 0.3 MP). However, the computation time becomes non-trivial for the image resolution of 2.25 MP in 360-degree data set[1], (it is around 2ms per frame in [Del+16b]). The computational load will become increasingly important as the resolution of this type of camera is also rapidly increasing (e.g., the 360fly-4k camera provides videos around 9MP resolution). Therefore, the need to optimize the algorithms is essential. The problem is even more relevant for non-optimized and non-parallel computational environments.

The object tracking methods presented in [KS07; Oh+12; MCP14] were based on the easy case of a low-resolution omnidirectional camera installed on a slow-moving robot in indoor environments. These methods are not applicable to more complicated conditions with outdoor filming, fast-moving objects, and occlusion. In [KS07] the authors exploited optical flow for object tracking. The method has been combined with a Bayesian prediction correction method for navigating a robot to an object [MCP14]. It can follow a human when the camera is moving without using image rectification [MCP14], but the distance between the person and camera is limited (around 1 meter). In addition, since this algorithm is sensitive to the object size, even in light-isolated environments in a room where the background is very smooth, the object must be non-moving in some intervals and turned to the camera to prevent missing the object of interest [MCP14]. Thus, they can be used only in simple applications like slow human chasing. In a

[1]https://www.youtube.com/channel/UCjS9DSZpPzfMwyDCp1Xai1Q/videos?viewas=subscriber

method presented in [LL18], YOLOv2 and Kalman filtering were combined for multi-human tracking in rectified 360-degree panoramic videos.

In [ZLS17], the authors utilize background subtraction and continuously updates the background model for motion detection in omnidirectional videos for an autonomous unmanned vehicle systems application. For omnidirectional object tracking, many works use normal object tracking techniques. For example in [Dem+12] cameras were installed in a way that the movement direction is similar to the easy case of a rectangular camera. Thus, most of the conventional object tracking methods can easily be applied to this data set. In [Dem+12], the authors exploit known tracking methods such as interest point tracking [Cal+10] and rectangular bounding box-based matching method [ZH06]. These methods cannot be applied to the 360-degree images. In addition, since the resolution is low, the application of this camera has been limited to indoor environments [Dem+12] where the distance is limited. In this case, a normal camera with a wide view and small focal-length lens could also be used.

3.2.4 Polar Object Tracking for Surveillance Application

Omnidirectional cameras have a lot of potential for surveillance and ambient intelligence applications, since they provide increased coverage with fewer cameras. A BOMNI data set is introduced in [Dem+12] and collected with two omnidirectional cameras simultaneously. The data set contains various scenarios, as well as actions relevant for ambient assisted living, such as falling down. The authors describe evaluation protocols on this data set and provide benchmarking baseline results for two tracking systems based on bounding box and interest point matching after foreground-background segmentation, respectively.

In a method proposed in [Dem+12], the authors proposed a method for multiple human tracking using omnidirectional cameras in the simple case of an indoor environment and still camera. It exploits variational Bayesian inference for tracking and uses rectangular object description similar to object tracking in normal images. It can handle occlusion well and perform tracking when the camera is fixed. In this work, a data set from a fixed and low-resolution omnidirectional camera for the indoor environment was introduced [Dem+12].

An integrated multi-camera video-sensor (panoramic catadioptric vision tracker plus - PCVT+) is presented in [Sco+04] for surveillance systems. In this setup, an omnidirectional imaging device is used in conjunction with a PTZ camera leading to an innovative kind of sensor that is able to automatically track any moving object within the guarded area at a higher zoom level. In particular, the catadioptric sensor is firstly calibrated and used in order to track every single object that is moving within its 360-degree field of view. Omnidirectional image portions are eventually rectified and pan, tilt and zoom parameters of the moving camera are automatically adjusted by the image processing system in order to track detected objects. A cooperative strategy was developed for the selection of the object to be tracked by the PTZ sensor in the case of multiple targets.

Harabar et al. [HS03] utilized such an optic to automatically pilot a small traffic surveillance helicopter by using a GPS system and a low-resolution omnidirectional camera. Outside, they used GPS for navigation, while inside they used many large landmarks with a black and white "H" on it. They used simple image thresholding to detect the H's for stabilization.

Dual-camera systems have been widely used in surveillance systems because of the ability to explore the wide field of view (FoV) of the omnidirectional camera and the wide zoom range of the PTZ camera. Most existing algorithms require prior knowledge of the omnidirectional camera's projection model to solve the nonlinear spatial correspondences between the two cameras. To overcome this limitation, in [Che+08], two methods are proposed: geometry and homography calibration, where polynomials with automated model selection are used to approximate the camera projection model and spatial mapping, respectively. The proposed methods not only improve the mapping accuracy by reducing its dependence on the knowledge of the projection model but also feature reduced computations and improved flexibility in adjusting to varying system configurations. Although the fusion of multiple cameras has attracted increasing attention, most existing algorithms assume comparable FoV and resolution levels among multiple cameras. Different FoV and resolution levels of the omnidirectional and PTZ cameras result in another critical issue in practical tracking applications. The omnidirectional camera is capable of multiple object tracking while the PTZ camera is able to track one individual target at a time to maintain the required resolution. It becomes necessary for the PTZ camera to distribute its observation time among multiple objects and visit them in sequence. Therefore, this paper addresses a scheme where an optimal visiting sequence of the PTZ camera is obtained so that in a given period of time the PTZ camera automatically visits multiple detected motions in a target-hopping manner.

Table 3.1 shows the distribution of common object tracking properties over the presented methods.

Table 3.1: Distribution of object tracking properties over related methods.

	Multiple Target Tracking	Outdoor Application	Real-time Tracking	Moving Camera	Occlusion Handling	Rectangular Video	Polar Video
Wang et al. [X.W+16]	X	X	X	X	X	X	
Wen et al. [Wen+16]	X	X		X	X	X	
He et al. [He+16]	X	X		X	X	X	
Ma et al. [Ma+16]				X	X	X	
Mazzu et al. [Maz+16]		X	X	X		X	
Milan et al. [MSR15]	X	X		X	X	X	
Milan et al. [MRS14]	X	X		X	X	X	
Xiao et al. [XSL15]		X		X	X	X	
Zhang et al. [Zha+15]		X		X	X	X	
Karthikeyan et al. [Kar+15]	X					X	
Chen et al. [Che+08]	X	X	X				X
Scotti et al. [Sco+04]	X				X		X
Rameau et al. [Ram+07]		X			X		X
Xiong et al. [Xio+12]					X		X
Markovic et al. [MCP14]				X			X
Zirakchi et al. [ZLS17]			X			X	
Demiroz et al. [Dem+12]	X	X		X	X		X
Scaramuzza et al. [SS08]		X	X	X			X
Hrabar et al. [HS03]	X	X	X	X			X
Kim et al. [KS07]				X			X
Oh et al. [Oh+12]			X				X
Liu et al. [LL18]		X	X	X			X

3.3 360-degree versus Omnidirectional Camera

The samples of images of a 360-degree camera are presented in this section. From the data set, several videos are used whose snapshots are shown in Figure 3.2 and Figure 3.1. The characteristics of the videos are as follows:

1. Natural in-plane rotation,

2. Out-of-plane rotation,

3. Moving camera,

4. Various types of objects,

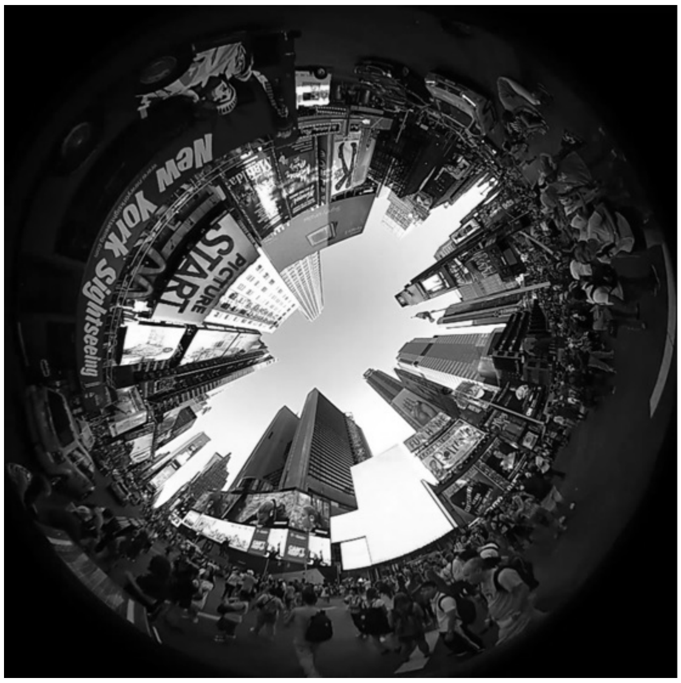

Figure 3.2: A 360-degree image

5. High image resolution,
6. Complex background.

The top region of a rectangular image is mapped to the center of the corresponding 360-degree image and other regions to the peripheral regions around the center. On the other hand, almost all videos contain out-of-plane rotation which means the camera is looking at the diverse sides of the same object in different times. The 360-degree data set consists of some videos from moving and fixed cameras wherein the objects of interest are human, airplane, car, and motorcycle with diverse shapes and features. Applying conventional polar and rectangular trackers to the 360-degree data set is very costly and slow.

The 360-degree data set has been considered in this section. Many challenging conditions are involved in these videos. The variety of objects of interest, diversity of occlusion, and illumination variation from the outdoor applications makes the tracking task more challenging than seen in the other data set in the literature.

Compared to the omnidirectional cameras (in Figure 3.3 (right)), the emerging 360-degree camera has a higher resolution and a wider FoV. 360-degree cameras do not have a blind region in the center of the image. 360-degree cameras also provide a wider field of view as compared to traditional fish-eye cameras providing maximum

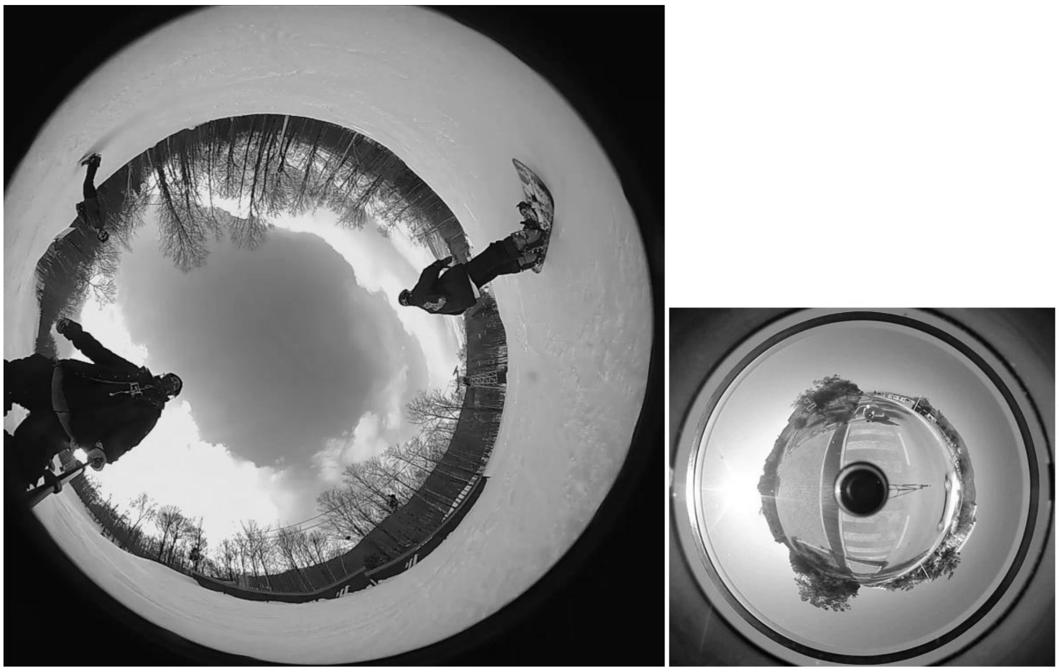

Figure 3.3: A 360-degree image (left) is bigger than a catadioptric omnidirectional image (Right). The catadioptric omnidirectional camera cannot cover the center region of the images.

180-degree view (Half spherical). This means that the fish-eye cameras cannot see the area behind the camera whereas the 360-degree cameras can. This ability opens up additional applications for these cameras as opposed to the traditional omnidirectional ones. However, the high resolution limits the algorithm speed necessary to maintain real-time capability. Without the capability, tracking of objects which rapidly move far away from the scene is not possible. From the output videos of the omnidirectional camera, it appears that a circle image has been cropped from a rectangular image. In contrast, 360-degree cameras have convexity even in the center part of the images. In other words; the field of view and the convexity of 360-degree cameras are higher than the fish-eye and omnidirectional cameras. Therefore, many omnidirectional-based tracking methods do not work on these videos, especially, when the objects are far away or have in-plane rotation. Since there are a wide variety of challenges in this data set, the mentioned methods are incompatible. In the next sections, new methods to allow tracking on this challenging data set are developed.

3.4 Object Tracking by Modified TLD

This section reviews a popular and robust approach in this category called Tracking-Learning-Detection [KMM12]. Then, the problem of object tracking in 360-degree videos is addressed. This method is based on the author's paper (C_2) published in International Conference on Pattern Recognition (ICPR2016) [Del+16b].

A very popular and robust approach for object tracking in rectangular videos is presented in [KMM12]. It is called Tracking-Learning-Detection (TLD). TLD is a framework designed for long-term tracking of an unknown object in a video stream. The main idea of [KMM12] is to let tracking and detection algorithms work together. Moreover, it also incorporates a learning module observing the performance of both tracker and detector.

The authors of [KMM12] start the motivation of their own approach stating the fact that most of the existing long-term tracking algorithms apply either tracking or detection and do not combine these two strategies. The trackers require only initialization, are fast and produce smooth trajectories. On the other hand, they accumulate errors during run-time (drift) and typically fail if the object disappears from the camera view. Research in tracking aims at developing increasingly robust trackers that track "longer". The post-failure behavior is not directly addressed. Detection-based algorithms estimate the object location in every frame independently. Detectors do not drift and do not fail if the object disappears from the camera view. However, they require an offline training stage and therefore cannot be applied to unknown objects. The components of the framework are characterized as follows:

Tracker estimates the object's motion between consecutive frames under the assumption that the frame-to-frame motion is limited and the object is visible. The tracker is likely to fail and never recover if the object moves out of the camera view.

Detector treats every frame as independent and performs full scanning of the image to localize all appearances that have been observed and learned in the past. As any other detector, the detector makes two types of errors: false positives and false negatives.

Learning observes the performance of both, tracker and detector, estimates the detector's errors and generates training examples to avoid these errors in the future. The learning component assumes that both the tracker and the detector can fail. By the virtue of learning, the detector generalizes to more object appearances and discriminates against the background.

In the next part of the current section, an approach is proposed that extends TLD towards object tracking in polar video.

Below, the MTLD [Del+16b], an accordingly modified version of the TLD summarized earlier in this section, is described. The general MTLD processing pipeline is depicted in Figure 3.4. Its modifications compared to TLD are as follows:

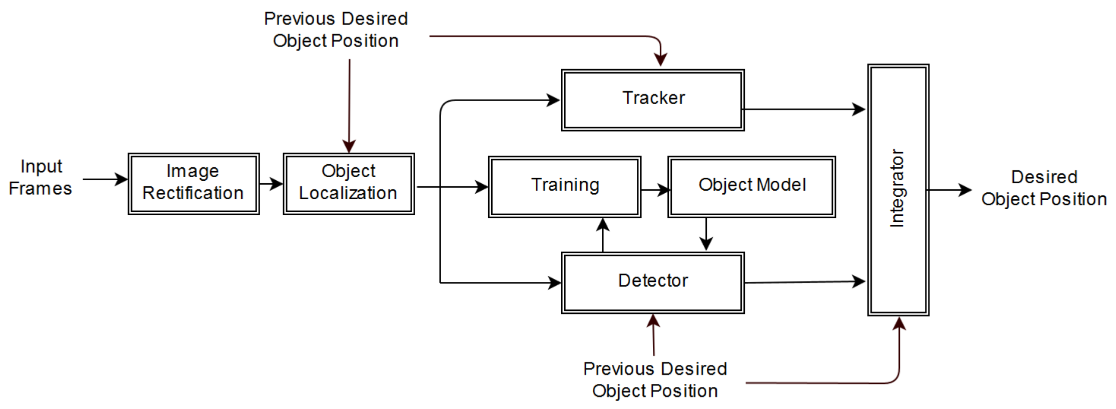

Figure 3.4: Block diagram of the proposed MTLD method.

Image Rectification: First, a rectification transformation is applied converting polar images into rectangular ones (see example in Figure 3.5).

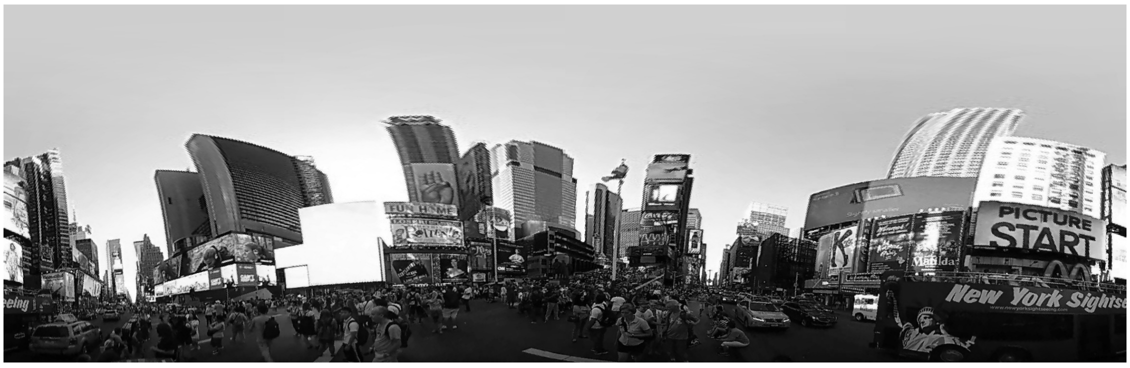

Figure 3.5: A rectified polar image

Classifier Modification: By the image rectification, in-plane rotation is mostly removed. However, in some cases, the desired object is not detected robustly due to the out-of-plane rotation problem. To resolve this issue, the nearest neighbor classifier performing this detection is modified in such a way to accept more variance in the appearance of the candidate objects. This strategy improves the detection performance of the objects of interest. However, it also increases the number of false positives. To tackle this problem, objects with a high replacement in consecutive frames are ignored. The block diagram of the proposed MTLD detector is depicted in Figure 3.6.

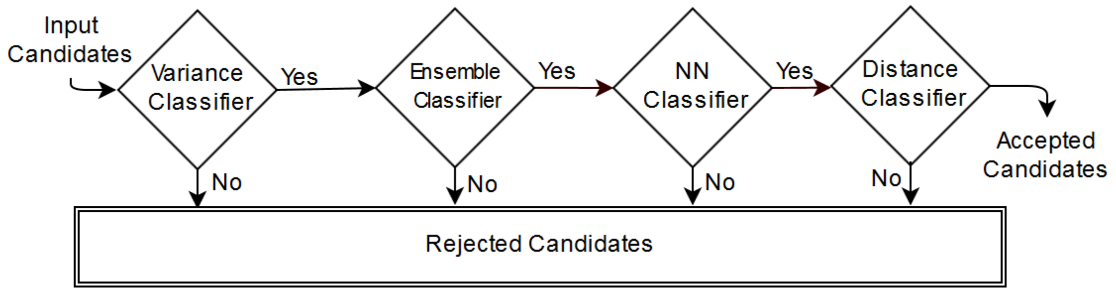

Figure 3.6: Block diagram of the modified object detector.

Search Area Restriction: The MTLD approach attempts to decrease the computational cost of the TLD method. For this reason, the object searching area in MTLD is restricted to a region around the object location in the previous frame. Due to the wide field of view in 360-degree images, even when the camera is moving, the object location in successive frames does not change significantly. Thus, the limitation of the searching area does not negatively influence the tracking performance.

Detection Strategy Modification: TLD uses the tracker output to train the detector. However, it has been observed that it does not play an important role in updating the detector output. To improve the detection performance, the currently detected object is included in the positive example set rather than the tracker output. Usually, the object size varies in a very limited way between two successive frames of a polar video. MTLD assumes the size difference to be lower than 25% which additionally optimizes the algorithm.

3.5 Object Tracking using YOLO Detector

This section presents the proposed deep learning based tracker. This method uses YOLO (You Only Look Once) to find the location of the object of interest in an image. Most of the object detection systems fail to detect objects directly in polar images because they were trained using data sets of rectangular images. When this object detector is applied to a polar image, the precision falls down. Assuming that the objective conditions are realistic and correspond to those of a rectangular image, the object recognizer will make correct decisions about the presence and location of the object. However, this does

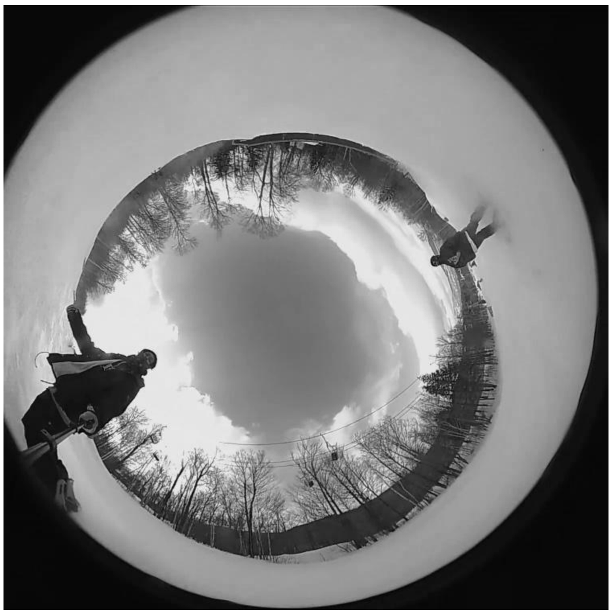

Figure 3.7: When YOLO is applied directly to the polar image, it cannot detect any object due to the object orientation.

not happen in the case of polar images. In 360-degree videos, the pixels are arranged in a circle, where the top of a scene is mapped to the center of the polar image. The circular orientation of the pixels causes an object distortion (Figure 3.7). To overcome the difficulty related to the polar object representation, all pixels of a polar image need to be transformed to a rectangular arrangement. (see Figure 3.8)

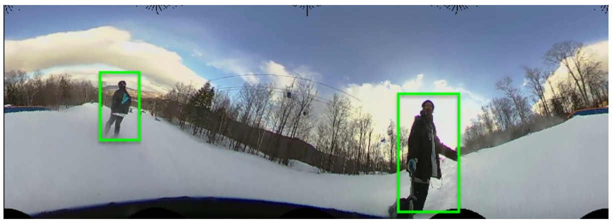

Figure 3.8: After image rectification, YOLO is able to state the correct position and size of the objects.

The rectangular version of the polar image can be obtained by converting polar coordinates to Cartesian coordinates using an unwrapping approach [KM11]. However, during this unwrapping process, the resolution of the output image is not uniform, negatively affecting the object detection precision. To counteract this issue, a bilinear interpolation is used to estimate the value of empty pixels in the image. For image rectification first, a polar coordinate system is created by shifting the image origin from the top left towards the center of the image. Each point is initially identified by its coordinates *(x, y)*. After translating the origin, each point in the image is redefined based on the new origin. The points are now described with *(x′, y′)* and calculated by subtracting the coordinates of the center: x'=x-x_c and y'=y-y_c, where x_c and y_c are the coordinates of the image center. In the next step, corresponding polar coordinates describing the points are created. A polar coordinate system is two-dimensional, in which every point is described by a distance from a reference point and an angle from a reference direction. In this case, the reference point is the image center. The image pixels are described using an angle, θ and a distance p between the image center and the pixels in the image. p and θ are calculated using the following equations [KM11].

$$p = \sqrt{x'^2 + y'^2} \tag{3.1}$$

$$\theta = tan^{-1}(x'/y') \tag{3.2}$$

The new Cartesian coordinates are obtained according to the following equation [KM11]:

$$x_n = \alpha * (r_m - p),\ y_n = \beta * (p * \theta) \tag{3.3}$$

Where r_m is the polar image radius, α and β are scaling factors affecting the outcoming resolution and are in the range of [0,1], where 1 means that the output image has the maximum resolution. After the unwrapping step, the outcome image has a rectangular form; However, it still has distortion caused by missing pixels. To overcome the distortion issue, this system uses the information of four known neighboring pixels. After this step, the image borders are cropped, so that the border distortion is resolved.

To overcome challenges like occlusion or disappearance of the object of interest in the data set, the proposed system uses the tracking-by-detection paradigm. Obscuration and disappearances of objects often arise in complex videos, due to the movement of multiple objects or non-stationary camera. The recognition of objects in a video is done by a state-of-the-art object detector YOLO [RF17] proposed by Redmon *et al.* . YOLO stands for *You Only Look Once* referring to the evaluation for predicting bounding boxes and class probabilities over the entire image. YOLO uses a single convolutional neural network for object detection while previous object detectors are applied to several locations on the image or use a sliding window approach, where a classifier is applied on evenly spaced locations over the image. Therefore, YOLO is able to predict the objects' presence and location significantly faster and is avoiding a complex pipeline. It takes a look at the entire image once and rates it globally to decide whether the object is a part of the background. YOLO uses pre-trained features to predict all bounding boxes of objects in the image. YOLO segments the input image into an *sxs* grid. Every grid cell detects an object if the center of the object falls into that cell. Then, the grid cell predicts the bounding box of the object and a confidence score for the bounding box. The confidence score shows the confidence of the system for the object presence and the accuracy of the predicted bounding box. If a cell does not contain an object, the confidence score should be zero. YOLO provides the object coordinates (i.e., location), an output class (i.e., the object type) and the confidence score. The visual output of YOLO is represented in Figure 3.10. Each cell also predicts conditional class probabilities, YOLO is evaluated using the PASCAL VOC data set [Eve+15]. YOLO is composed of 24 convolutional layers followed by 2 fully connected layers. A convolutional layer basically convolves an image in a kernel to form a filtered image. The kernel is a matrix containing specific object features. By convolving the original image with the kernel, the output is a filtered image containing statements about where a possible object can be and how confident the system is. Fully connected layers are then used to vote whether an object exists at a certain position. Fast YOLO uses 9 convolutional layers and is faster than Yolo, but has lower accuracy. YOLO uses ImageNet 1000-class data set [Rus+15] for training. In this data set, each image was manually labeled, describing the depicted object. YOLO pretrains a convolutional neural network containing 20 convolutional layers followed by an average-pooling layer and a fully connected layer. Pooling layers are used to shrink the images based on certain window size. To improve the recognition performance, four additional convolutional layers and two fully connected layers with randomly initialized weights are added. The final layer then predicts class probabilities and bounding box coordinates. YOLO9000 is another version of YOLO detecting over 9000 object categories in real time. YOLOv3 is also an updated version of YOLO that has a bigger network than YOLO and more accuracy. For the proposed tracker, the information about higher-level object categories such as people, cars, or airplanes is sufficient and the specific object categories like car models are not necessary. That is why YOLO9000 was not used in the proposed tracker. After the rectification step, the resolution shrinks. Due to the high field of view of 360-degree cameras, objects in the correspondent image become relatively small and have an abnormal aspect ratio compared to the data set YOLO trained on. In this case, YOLO occasionally fails to recognize objects. This drawback can, however, be improved by a tracker (i.e., Kalman filter). The pure Kalman filter cannot generate any information about the object position and size. Therefore, this tracker needs the information extracted by YOLO in order to facilitate the later tracking process.

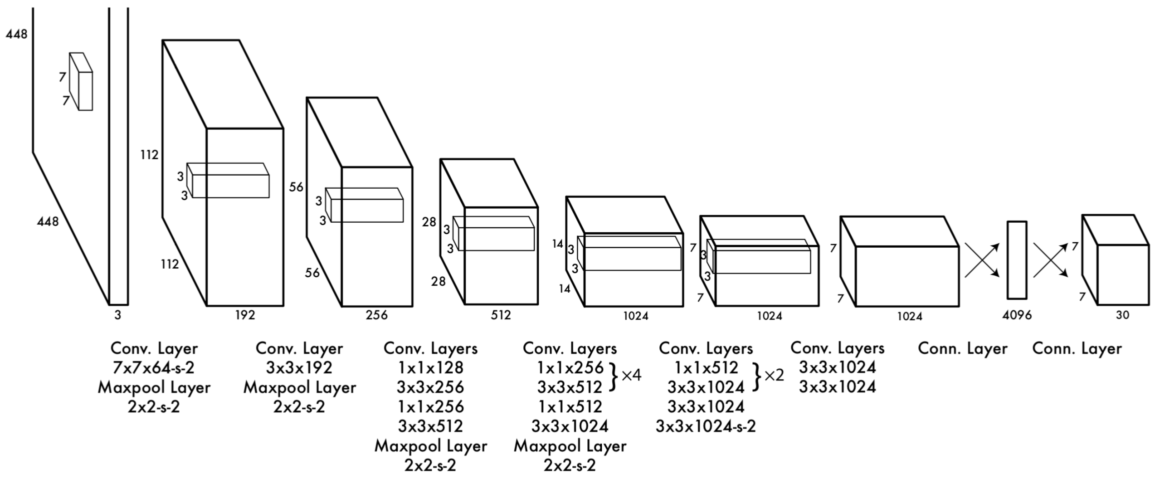
448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 ×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 ×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

Figure 3.9: Convolutional neural network with 24 convolutional layers followed by 2 fully connected layers [RF17]

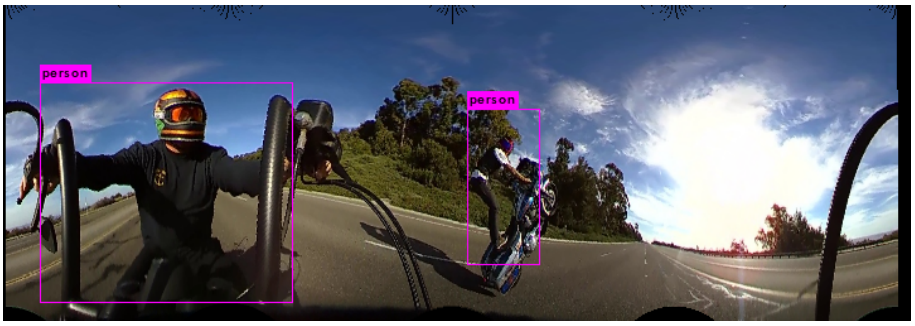

Figure 3.10: An example of the YOLO visual output with correct labeling

The proposed object tracking system uses two separate tracking approaches namely Kalman filter and Lucas-Kanade methods. The Kalman filter uses the previously extracted information of the object of interest to determine the object's position. The Lucas-Kanade method, however, requires no prior information for object tracking. In the following, first the procedure of the two mentioned tracker is explained and then the combination of them is presented.

The Kalman filter is a linear state estimator receiving measurements over a time period to describe the object state. It is used in different application areas like guidance, vehicles control and navigation, robots, autonomous driving, interaction with persons or object tracking. The Kalman filter is an optimal recursive data processing algorithm [May82]. Optimal in this context means that it incorporates all of the provided series of measurements, regardless of their precision to estimate unknown variables that are more accurate than those based on a single measurement. Unknown variables are

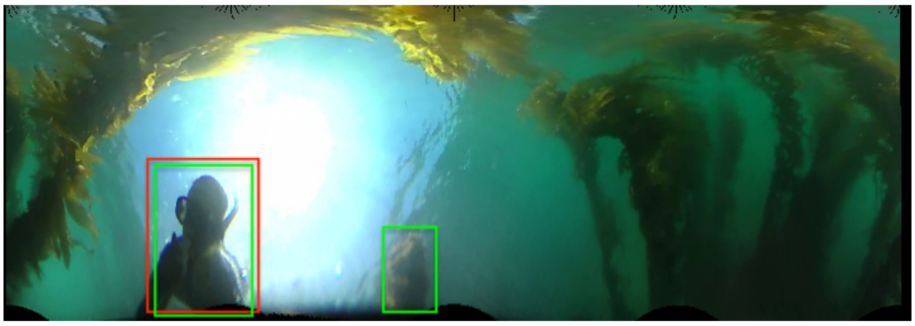

Figure 3.11: The visual outputs of the Kalman filter and YOLO are shown. Green rectangles show the YOLO output and red rectangles are the Kalman filter estimation.

able to describe the past, present and future state, even if there is no information about the true nature of the model [BW01]. By combining multiple measurements, mean squared estimation errors are minimized. In order to estimate one of the aforementioned states, the Kalman filter considers two phases. In the first step, a predictor estimates a state, based on the previously given data. Then, the update phase, the Kalman filter corrects the prediction when a new measure is available. Thus, the Kalman filter is alternating between the update and the prediction phase (see Figure 3.13).

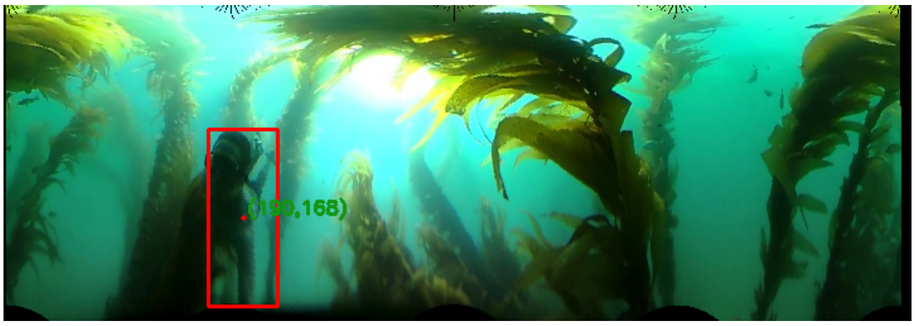

Figure 3.12: YOLO failed to find the object of interest. Nevertheless, the position of the object could be estimated with a slight deviation by the Kalman filter.

The proposed system uses the Kalman filter to predict the object's future location using a history of objects' location and speed. The Kalman filter uses the location information extracted by YOLO. The proposed system uses a six-dimensional state vector, with the following values: x, y, w, h, v_x, v_y where x and y are the position coordinates, v_x, v_y are the object's velocity in horizontal and vertical directions and w, h are the size of the object. There is also a four-dimensional measurement vector containing $\{i_x, i_y, i_w, i_h\}$ only used when YOLO provides information about the location of the object of interest, during the update phase.

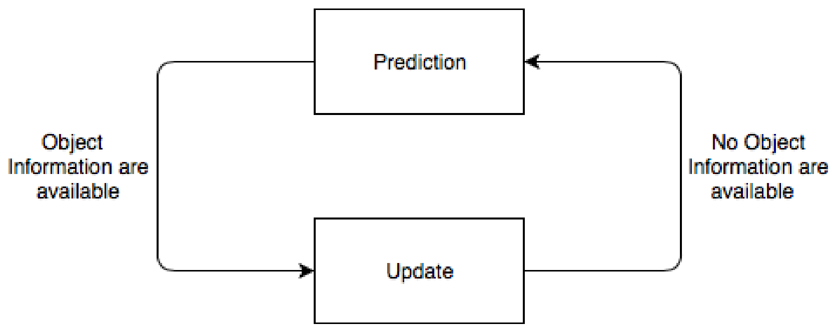
Prediction
Object Information are available
No Object Information are available
Update

Figure 3.13: The cycle of the Kalman filter: If object information to correct the prediction is available, the update phase is started. The prediction phase estimates the object position using its history.

Due to the fact that the Kalman filter does not rely on object information continuously and can estimate the future object state even without measurement, it provides good results in case of the detector's failure. Even if an object is occluded over a short period, the filter predicts the future position based on a linear model and can update the measurements as soon as the object becomes visible again. Visual output of the filter is shown in Figure 3.11.

Lucas-Kanade is a well-known tracker relying on the optical flow of a video sequence [Roj10]. It assigns a two-dimensional movement vector to a point of interest to track it from one frame to the next. In order to calculate this vector, first, unique points are found. These points are edges or corners within the image. There is a strong gray-scale intensity change near the edges of an image. A corner pixel is characterized by two dominant edge directions in its neighborhood. The corner and edge can be calculated using different methods like Canny edge detector. Corners are locally unique, distinguishable from the background and insensitive to noise. These features make the corners better candidates than edges. Lucas-Kande considers a small region around the desired point and assumes that the flow within this region is constant. If the interval between two successive images is too long or the object moves too fast so that the position has changed significantly, Lucas-Kanade may lose the interesting point. Lucas-Kanade calculates the optical flow using the gradients inside a region of movement and the movement direction is a two-dimensional vector. This method works best when the object moves slowly. The interesting points are only found at edges or preferably corners. However, if an object has a uniform surface, such points cannot be found and therefor Lucas-Kanade misses the object of interest. The Kalman filter and Lucas-Kanade methods have their own advantages and disadvantages. The Kalman filter has very good results, but only if the YOLO detector regularly detects the desired object and provides associated information. Thus, if the detector misses the object, the overall tracking performance falls down. The quality of the image can have a strong influence on the results of the Lucas-Kanade method. If the points of interest over several images cannot be clearly identified, Lucas-Kanade may either track a wrong point or lose the object of interest. The different combinations of Lucas-Kanade and the Kalman filter were tested. In the first scenario, Lucas-Kanade performs mainly detection of the object by continuously calculating the optical flow and the Kalman filter runs in the background and predicts the object's size and position.

Based on initial experiments, the previously extracted information is more important since the tracking-by-detection paradigm is used and YOLO offers a higher precision than Lucas-Kanade for most videos. Therefore, the combination of the Kalman filter and YOLO shows better results than the single Lucas-Kanade tracker. In the second scenario, the combination of YOLO, Kalman filter and Lucas-Kanade is proposed. The Kalman filter performs the prediction phase and YOLO updates the Kalman filter. However, if YOLO misses the interesting object over several frames, the Lucas-Kanade method is used for detection instead of YOLO. Because if the estimated values are not renewed over a longer time and the object of interest does not move linearly, the tracker may drift off and miss the object. This combination shows the best overall performance.

3.6 Polar Object Tracking in 360-degree Video

In this section, a polar high-overlapping object selection method is proposed to allow the object selection and tracking in 360-degree videos. Also, a color classifier in an energy minimizing framework is introduced to concern the complex background issue and out-of-plane rotation. The author's contribution to object tracking in 360-degree videos by using a polar trapezoid-shape method has been published in the proceedings of the International Symposium on Multimedia in 2016 (C_3) [Del+16a].

This method selects an object of interest in the first frame. Based on the location of this object, an area of interest is selected in the second frame, in which some overlapped candidates are generated. These are candidates for the object of interest. Finally, the candidates are forwarded to the object detection module to find the object of interest in the current frame. This process is automatically repeated for the next frames.

The contribution of this section is defining a polar area of interest and a polar trapezoid-shape candidate selection. To have an efficient and fast object tracker, a region of interest (RoI) around the object of interest in the first frame is extracted. The proposed area of interest is a region surrounded by a sector containing the center of mass (CoM) of the object of interest in the bisecting radius indicated by a reference line (see Figure 3.14). Using this approach, only this area of interest is considered for searching the object of interest reducing the computational load significantly. For the sake of candidate generation, similar to RoI selection, a polar-based strategy is proposed. The region which is surrounding the object of interest, is represented by using a sector made by two concentric circles. Two radii and two circles are selected in a way that the region completely circumscribes the object of interest. A candidate is represented by two radii r_1, r_2, and the angle α_1 between the two radii. As depicted in Figure 3.14 the region of interest is highlighted with blue color and the rest with green. The RoI containing the selected object of a video is shown at the bottom of Figure 3.14.

For the sake of candidate generation, the polar objects can be approximated by trapezoid ones and the difference between polar and trapezoidal objects is ignorable. Figure 3.15 shows some polar overlapped candidates. To have a clear presentation of the polar object selection, only 5 candidates have been shown. In real practice, the number of candidates is much more than 5 and they are strongly overlapped. According to Figure 3.15, the candidate (a) shows the last desired object. Candidates (b) and (c) have the same aspect ratio and radius but different angles. On the other hand, candidates (d) and (e) have the same aspect ratio and angle but different radii. To increase the chance of correct object detection, the scanning process is repeated by $\{0.8, 0.9, 1.1, 1.2\}$ as coefficient of the aspect ratio of the last desired object. This procedure compensates probable out-of-plane rotation of the object.

By using the same angle α and radii r_1 and r_2 as the first frame, overlapped candidates are selected by an incremental step $\Delta\Theta$ degree relative to the horizontal radius.

Also by increasing the radius length by Δr steps, the region of interest can be scanned. The choice of these parameters is due to optimality.

The building blocks of the proposed method are shown in Figure 3.16, showing the proposed hierarchical construction for object detection. The aim of this module is to select the object of interest among the polar candidates. Prior to the first step, the size of all candidates, as well as the last desired object, is normalized to allow comparison. The candidates are normalized in a way that the size of its pad image becomes a square of size 60×60 (see Figure 3.17)

To extract the features of the frames first, the polar candidates are fed to the variance classifier, wherein the variance of a gray-scale candidate is calculated. Then, this value is compared to the image variance of the object of interest in the last frame or last detected object. The comparison is performed by using the following condition:

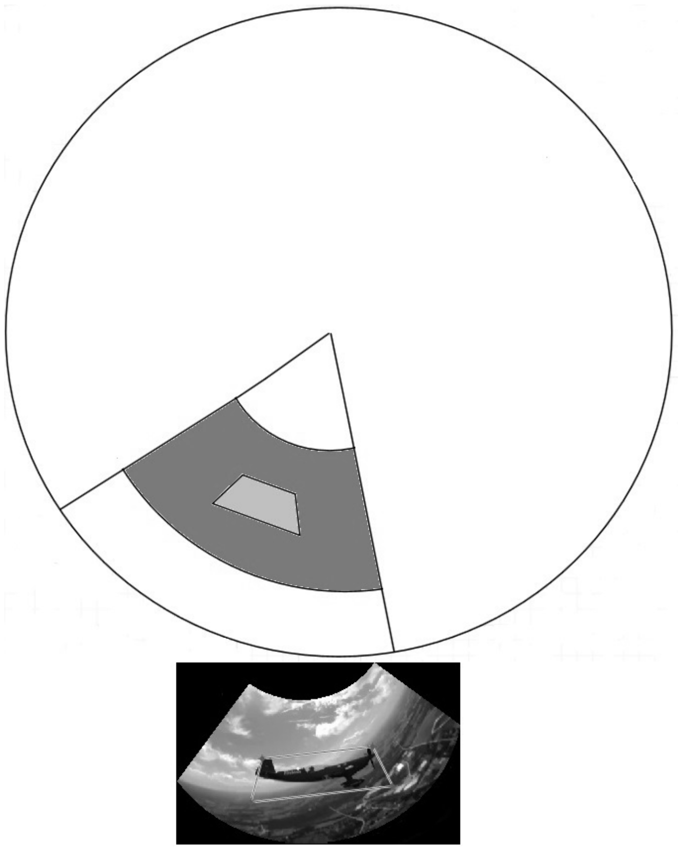

Figure 3.14: Polar ROI selection

$$\alpha_1 \times v^d_{i-1} < v_i < \alpha_2 \times v^d_{i-1} \tag{3.4}$$

Where v^d_{i-1} and v^d_i are the variance of the last object of interest and variance of the image candidate in the current frame respectively and α_1 and α_2 are adjusting parameters specified experimentally. If the input candidate satisfies the above equation, it will be forwarded to the next block; otherwise, it is rejected by the classifier. By using this classifier, the smoother regions like sky, street, snow, etc in one side, and candidates from very complex backgrounds like trees, bushes, and leaves on the other side are removed. The experiments show that this classifier is implemented faster than two other classifiers. Thus, to increase the performance of the system, this classifier is used in the first step. In the next two steps, each candidate is divided into 9 segments as shown in Figure 3.18.

To proceed with the classification, a color-max classifier as the second classifier is applied on the rest of the candidates. To this aim, each segment of the above candidates is converted to HSV space and 6 colors (channel) of green, white, red, brown + purple, blue + green and yellow. The exploited thresholds for color identification are as follows.

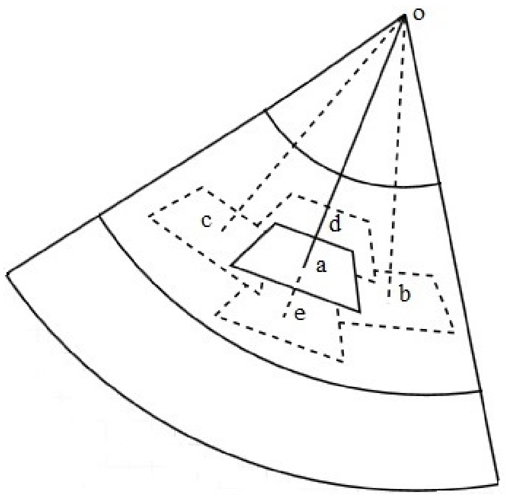

Figure 3.15: Overlapping trapezoid-shape candidate selection method is shown. Each candidate is defined by a trapezoid-shape region around the object of interest.

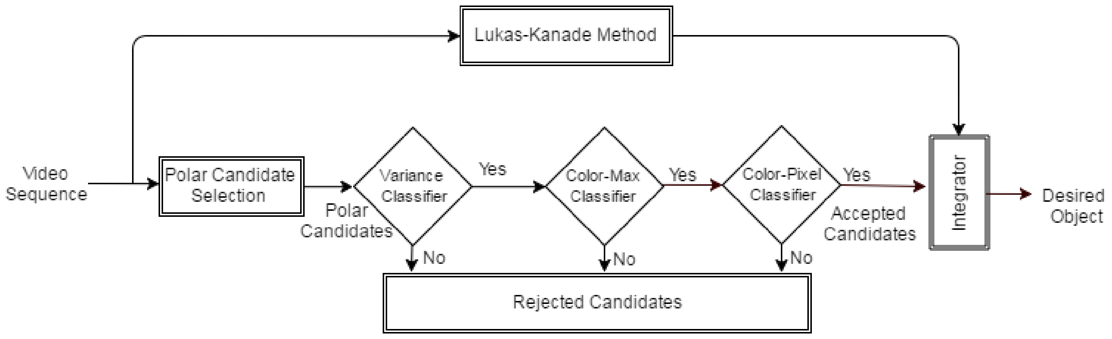

Figure 3.16: Block diagram of the proposed trapezoid-shape tracker is shown.

Furthermore, the number of each resulted channels is counted and the color which is used by the majority number of pixels is considered for that segment. The system repeats this procedure for other segments. Therefore, for each candidate there is a codeword. The code alphabet comes from the color number presented in the last row of Table 3.2. Finally, by using the following rule, many wrong candidates are rejected:

$$\sum_{j=1}^{9} S^j > t \tag{3.5}$$

Where

$$S^j = \begin{cases} 1, & \text{if } C_i^j = C_d^j \\ 0, & \text{otherwise} \end{cases}$$

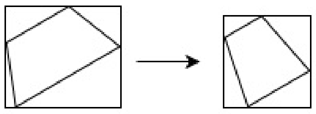
Figure 3.17: Object normalization by their pad

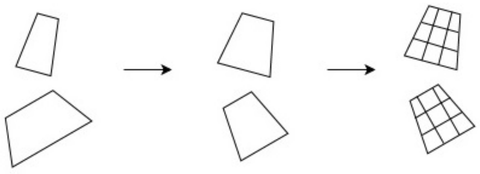
Figure 3.18: Object normalization and segmentation

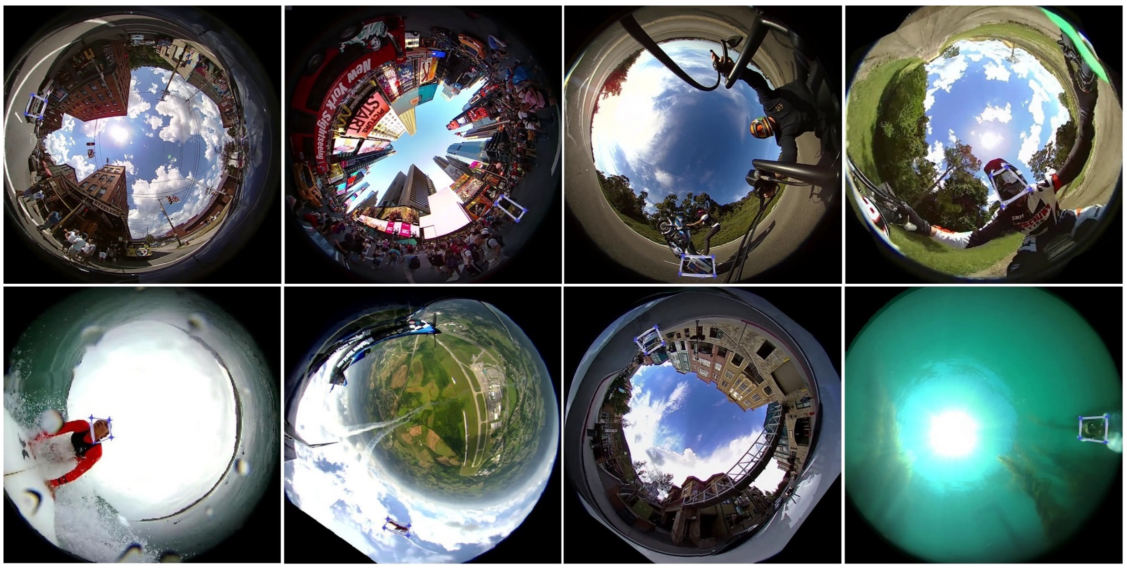
Figure 3.19: Object tracking true positive results

In this equation C_i^j and C_d^j are the codeword of the segment j (the color of the given segment) for the current candidate and last desired object and t is the user-defined threshold value respectively. By this classifier, most of the wrong candidates are rejected and then few remaining candidates are forwarded to the next block. In the next classifier, i.e. color-pixel classifier, the decision on the correct candidate will be finalized where in each mentioned segment, the area or the number of pixels of the representative color using its codewords is counted. As a result of each candidate i, there are nine segments j. By using the following rule, the object of interest is detected:

$$d^* = \{i | d_i < t_s\} \tag{3.6}$$

Table 3.2: Thresholds of different colors

	Yellow	Green	White	Red	Purple	Blue
HueLowTh	0.035	0.15	0	0.05	0.76	0.4
HueHighTh	0.14	0.48	1	0.97	0.94	0.75
SatLowTh	0.4	0.03	0	0.3	0.33	0.21
SatHighTh	1	0.97	0.1	1	0.67	0.98
ValLowTh	0.15	0.04	0.6	0.01	0.1	0.35
ValHighTh	1	0.8	1	1	0.7	1
Colornom	1	2	3	4	5	6

Where

$$d_i = \sum_{j=1}^{9} |A_i^j - A_d^j|$$

In the latter equation d^* is the index of the new desired object and A_d^j is the area of the last desired object for the specific color in the jth segment and t_s is a user-defined threshold value respectively. In fact, in Equation 3.6, the energy function between two frames is minimized. In cases where there are multiple accepted candidates or no accepted candidates at all, another tracking method called Lucas-Kanade [Bou00] is used to estimate the location of the desired object in the next frame. The Lucas-Kanade (LK) method supposes that the displacement of the image contents between two successive frames is small and approximately constant. The integrator compares the output of this tracker and the proposed hierarchical classifiers to make the final output.

If the hierarchical processing routine does not find any object, it is supposed that the desired object either has been gone out of the scene or occluded by other objects. In such a case, the system keeps the characteristics of the last detected object to compare it with the coming frames in the next frames. Therefore, the desired object is detected. This procedure will be repeated for all of the remaining frames. In the section of experiments, the results of this method are presented. If the hierarchical processing routine finds one object, the integrator outputs this object and otherwise the object with the minimum distance with the LK branch is outputted.

3.7 Object Tracking using Feature Descriptor and Matching

In this section the proposed method for object tracking in polar videos using SURF (Speeded Up Robust Features) extraction and matching is presented. This method is based on the author's paper (C_4) published in the proceedings of the International Symposium on Multimedia in (ISM 2017) [DG15]. The problem of object tracking is formulated as finding a given query object in the next frame (scene). For this aim, SURF first finds interesting points in both the query image and the frame and extracts a couple of features from each point. Then, by minimizing a distance function, finds the matched points between the images. A polar object is selected manually in the first frame and it is searched automatically in the next frames. The steps of the proposed method are as follows. In the first frame, a trapezoid-shape polar shape is drawn around the manually selected object by a user. Based on the location of the object, a region of interest (RoI) is drawn around the object location in the next frame. The polar RoI and trapezoid-shape object selection are similar to the methods presented in Section 3.6. But here the overlapped candidate selection is not used. The RoI is represented by a sector limited between two concentric arcs and two radii. The radii and arcs are selected in such a way that the RoI completely circumscribes the object of interest and is located in the center of RoI. In this section, the angle between two radii of RoI is set to 90 degrees. The RoI and object selection method are shown in Figure 3.20. Both the object and the scene (the RoI) are then enhanced by using histogram equalization and resized by factor 3. By using SURF descriptor, interesting points are detected from the object and the scene and then from each interesting point, a couple of SURF features are extracted. For each interesting point in the object and scene, the SURF features as well as the interesting points location and intensity are calculated and fed to a supervised classifier (Random Forest) to find the matching points in the object and scene. In the scene, a polar trapezoid-shape object region is then drawn around the points and therefore a new object is obtained. In each tracking iteration, whenever two points are matched, the features of these points are saved in the challenging vector. During tracking, an SVM (Support Vector Machine) classifier is used to see if a challenging situation (i.e., occlusion, out-of-plane-rotation, departing or approaching the camera) occurs or if it is a normal tracking situation. The classifier uses the intensity and location information of the current object as well as the same information from all members of the challenging vector. For each challenge, a correspondent algorithm is exploited to handle it. The tracker repeats the above steps for the next frames of the video. The updating method of the challenging vector is explained later.

In the proposed SURF-tracker scenario, the precise location of the object region is

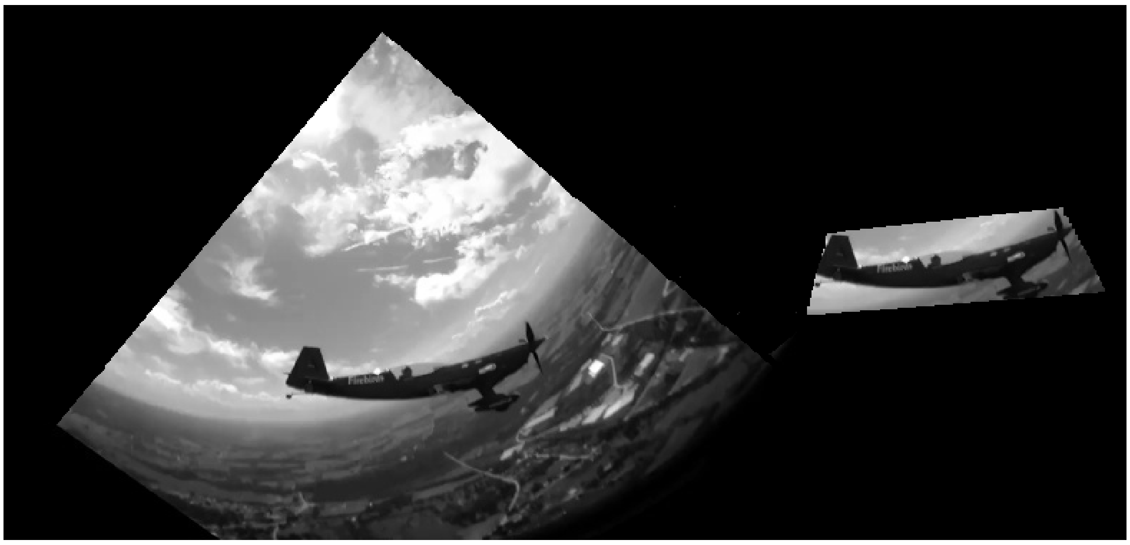

Figure 3.20: Polar RoI selection (left) and polar object selection (right) for the proposed SURF-tracker

very important. Because if a region with a smaller area than the real object (ground truth) is selected, some SURF interesting points might be missed. In contrast, if the cropped region is larger than the object of interest, irrelevant points in the background might be detected as the interesting point. In this case, instead of the object of interest, an irrelevant region in the background is tracked and the tracker misses the object. When interesting points within the object match to the correspondent points in the scene, a region around the detected points in the scene is depicted which forms the new object of interest in the next frame. To form the new object of interest, the tracker first calculates center of mass (CoM) of the matched points in the query object using the points mean (function). Then, 4 polar distances (i.e., radius differences (δr_i) and angle differences ($\delta\theta_i$) between the 4 corners and their corresponding CoM (r^{cm}, θ^{cm})) are calculated. The 4 distances are then used to make the newly found object in the scene. By using the following equation, polar locations of the corners (r^{ci}, θ^{ci}) of the new object of interest in the scene are obtained:

$$r^{ci} = \delta r_i + r^{cm}, \theta^{ci} = \delta\theta_i + \theta^{cm}, i = 1,2,3,4 \tag{3.7}$$

To find the matched points between scene and object, traditional matching methods like Sum Squared Distance (SSD) and Sum Absolute Distance (SAD) were examined. The drawback of these methods is that even by using the best parameter set and high image quality, there is a considerable mismatching rate (Figure 3.21 shows an example).

In other words, the matching problem does not have a linear solution in the feature space. Thus, the information from SURF features seems insufficient. To handle the mismatching problem, a random forest classifier is used, because this classifier can solve the challenging classification problems. Also, this classifier has fast response during training and test phases which is important for fast object tracking. Other classifiers like neural networks (even deep structure) and support vector machines were also tested, the best results were obtained using the random forest classifier. Suppose, there are l and m interesting points in the object and the scene respectively.

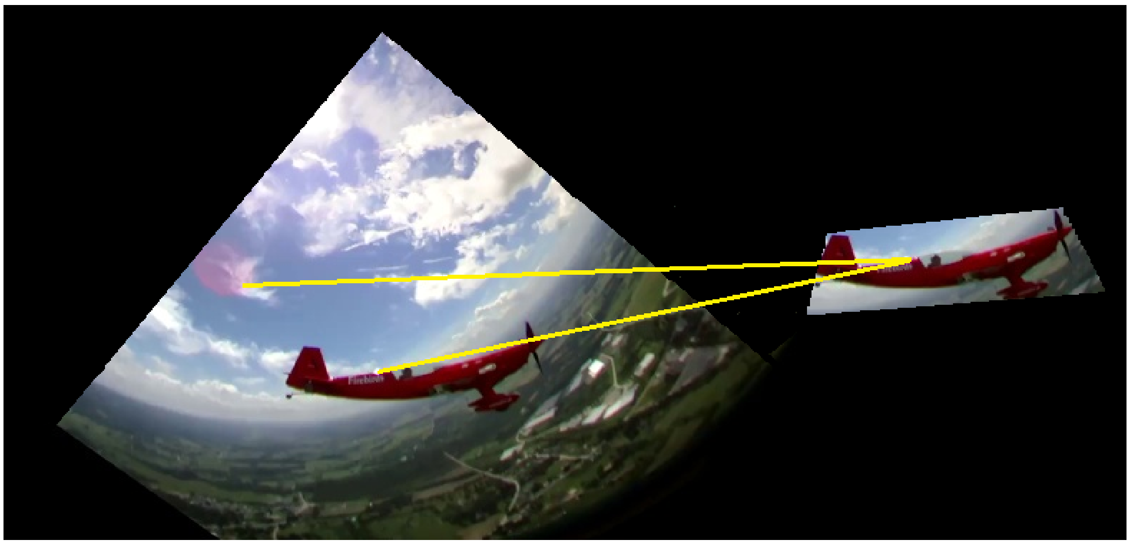

Figure 3.21: An example of mismatching for SAD. Two near points in an object mismatched to two far points in the scene.

The total number of possible pair-points would be $l \times m$. For each pair-point, totally 150 features are extracted. The feature vector is expressed as $\{s_o, s_s, i_o, i_s, l_o, l_s\}$ where s_o and s_s denote 64 SURF features for the object and scene, i_o and $i_s$show intensity of interesting pair-points as well as their 9 neighbor points (18 features) and l_o, l_s show the interesting point location respectively (x, y of the interesting points of the object and the scene form 4 features). To the best knowledge of the author, it is the first time that the information of location and intensity is fused to SURF features for object tracking. This combination is able to handle the matching problem in challenging situations like illumination variation, environment change, and background clutter very well.

In this section a supervised classifier is used to detect and classify the tracking. The system classifies the situation into tree states including out-of-plane rotation, occlusion and a normal situation.

Based on the initial experiments, when the object of interest rotates in an out-of-plane direction, the matched points quantity decreases, because some interesting points become hidden from the camera view. This, in turn, increases the object missing chance. This situation needs a precise mechanism. To handle the situation, each object is compared to the last frame and the members of the occlusion vector which contain diverse appearances of the desired object. The parameters of the matching classifier are modified in a way that the matching sensitivity increases so that even with a minimum number of matching points (i.e., one pair) the query object and the scene is considered as matched. As a result, the tracker's overall precision rises. By modifying $\delta\alpha_i$ in (3.9), the aspect-ratio of the surrounding trapezoid is accordingly modified to completely surround the object. When the normal situation is detected by the challenging classifier (SVM), this process is truncated and the normal tracker (i.e., the matching classifier (RF) with the original parameters) is activated.

As the tracker is going on, the challenging vector becomes bigger and consequently, tracking robustness increases. To handle occlusion and out-of-plane rotation, a memory-based algorithm is used. Based on a SAD threshold, a data structure is defined. During tracking in diverse frames, the object in the first frame, as well as other frames which qualified in the following circumstance are saved in an occlusion vector.

$$\sum_i s(O_c, O_i)/n < th \tag{3.8}$$

Where n is the length of the occlusion vector, s is SAD (Sum of Absolute Difference), O_i and O_c denote the center of an object in the vector and current object respectively. It means that the objects with minimum similarity have been saved in the occlusion vector. There is an upper limit for $n (n = 20)$ and a FIFO updating method is exploited.

When an occlusion occurs, the RoI extends to the whole image because the object can reappear everywhere in the image. Instead of the last object of interest, all members in the challenging vector are compared with the current object. Whenever an object has been matched i.e., an object has been found, the searching process (over the challenging vector) is truncated and in the next frames, the normal tracking procedure is taken.

In the case of departing or approaching the camera, the object size is changed. SURF misses some interesting points when the object gets smaller. Based on the remaining interesting points, the interesting object is located and tracked.

3.8 Polar Model for Fast Object Tracking in 360-degree Video

The proposed tracker in this section has been established on the polar object tracking in Section 3.6. This section is based on the author's paper C_5 published in the Journal of Multimedia Tools and Application [Del+18] in 2018. In this section, modifications to improve the robustness and the speed of the tracker are presented. The trapezoid-shaped candidate generation method in Section 3.6 is improved by concentric-arcs shape describer in this section. The candidate generation for the central part of the image is completely reconstructed in this section. To the best knowledge of the author, binary features have not previously been exploited in the color planes. This section presents a pixel comparison method for color binary features which is different from the previous work in Section 3.6 which compares the pixel-groups in HSV color channels. The HSV based tracker proposed in Section 3.6 is slower than the proposed method in this section. The building blocks of the proposed color binary tracker have been shown in Figure 3.22. The proposed tracker selects a region of interest (RoI) using the position of the last object of interest. Then, within this RoI, it selects overlapping polar candidates. All polar candidates are then forwarded to hierarchical classifiers. The task of the classifiers is to select more similar candidates to the last object of interest. The proposed tracker uses two classifiers namely variance and color-based binary classifiers for object detection. Finally, the tracker applies an ancillary tracker and selects the object of interest using an integrator.

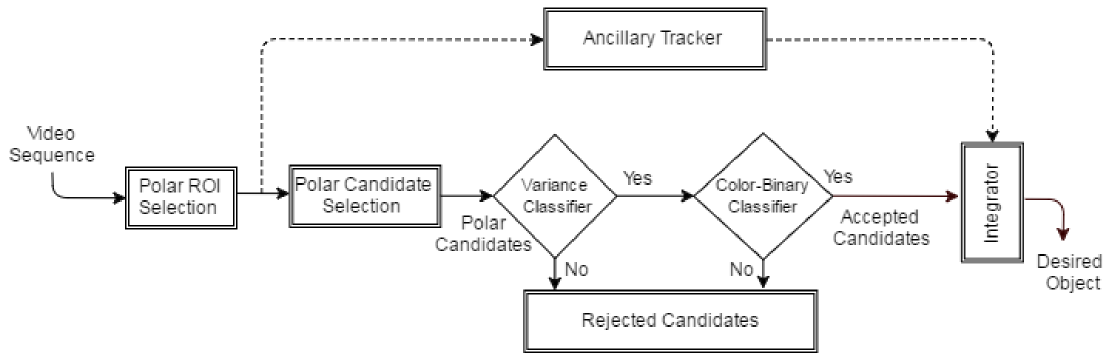

Figure 3.22: Block diagram of the proposed tracker is shown. Polar candidates are selected in a RoI. Then, they are fed to the hierarchical classifiers including a variance classifier and the proposed color-based binary classifier. In some cases, an ancillary tracking method is applied on the video frames. Finally, integrator outputs the object of interest.

One of the commonly-used methods to decrease the computational load is to restrict the search area of the image. In this regard, instead of the whole image, the tracker searches for the object of interest a small region (RoI) with a high probability of object presence. This RoI is similar to the RoI presented in Section 3.6, but they are different when the object of interest is located in the center of the image. The RoI is represented by $\{r_1^{roi}, r_2^{roi}, \theta_1^{roi}, \theta_2^{roi}\}$ which is composed of two radii r_1^{roi}, r_2^{roi} and their correspondent polar angles of θ^{roi} and θ^{roi}. To achieve more robust and faster object tracking, the tracker draws in the second frame an area of interest around the location of the object of interest in the first frame. The proposed area of interest is a region surrounded by a sector wherein the center of mass of the object of interest is located along the bisecting radius. Using this approach, only this area is considered for searching the object of interest reducing the computational load significantly. Figure 3.23 shows the RoI containing the selected object. In this case, the center and angle of the RoI and the last object of interest are the same. During tracking, the RoI is updated by using the following formula:

$$\begin{aligned} r_1^{roi} &= Max(r_1^d - 0.5 \times r_1^d, 0) \\ r_2^{roi} &= Min(r_2^d + 0.5 \times r_2^d, 750) \\ \theta_1^{roi} &= \theta_1^d + 15 \\ \theta_2^{roi} &= \theta_2^d - 15 \end{aligned} \tag{3.9}$$

Where r_1^d and r_2^d are two radii ($r_1^d < r_2^d$), representing the small and big arcs of the object of interest and θ_1^d and θ_2^d are their correspondent polar angles, and 750 is the image radius. In the case of the full occlusion, the whole image is considered as RoI. This method is different from the last work presented in Section 3.6 which contains some drawbacks when the object is located in the center of the polar image. In this work, this drawback is resolved by exploiting a circular RoI in a center with the same center as the image and a radius, less or equal to twice of the radii surrounding the object of interest (See Figure 3.27).

Similar to the RoI selection method, a polar-based strategy is proposed for object selection. Each candidate is a confined region defined by two radii and two concentric circles and therefore can be expressed by $\{r_1^i, r_2^i, \theta_1^i, \theta_2^i\}$. Where θ_1^i and θ_2^i are the angle of the right and left radii representing the polar candidate. By using the same angles and radii r_1 and r_2 as the first frame, overlapped candidates are selected by the incremental step of $\Delta\Theta$ degree relative to the horizontal radius. Also, by incrementing the radius length by Δr pixels steps, the region of interest can be scanned. Figure 3.24 and Figure 3.25 show polar object detection for overlapped sample objects. Figure 3.25 shows the methods of overlapped polar object selection and normalization by their pad image. In this figure, object (1) shows the last object of interest. Objects of (2) and (3) have the same angle but different radii, and objects of (4) and (5) have the same radius but different angles. To increase the chance of object detection, the scanning process is repeated by

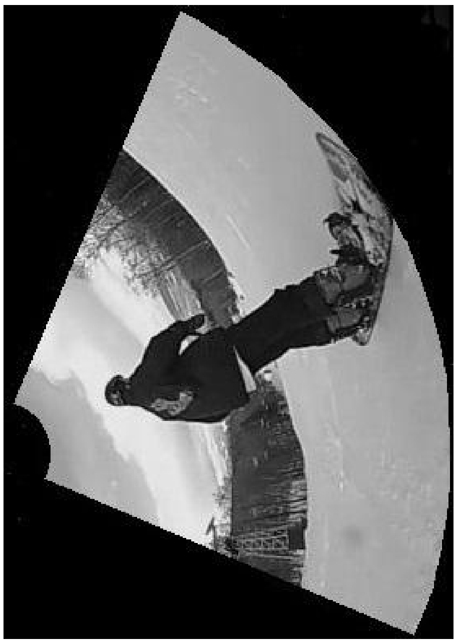

Figure 3.23: A polar sector around the object of interest shows the polar region of interest (RoI).

{0.8,0.9,1.1,1.2} as a coefficient of the aspect ratio of their pad image. This procedure facilitates a stable tracking in two cases of out-of-plane rotation and abrupt size variation (when a target is approaching the camera). Once the candidates are identified, they are forwarded to classifiers for object detection. To have a clear presentation of polar object selection in Figure 3.25, only 5 candidates have been shown. In real practice, the number of candidates is normally higher and they are strongly overlapped. To facilitate the comparison, the size of all candidates is normalized. To perform this process, all the candidates are grouped according to their angle $\theta^i = \theta_1^i + (\theta_2^i/2)$. Each group is labeled with its angle θ^i. In each group, the object with the same radii $\{r_1^i, r_2^i\}$ is considered as index object of the group. Then, the objects with the same angle are normalized to the size of the index object of the group j. Figure 3.26 shows this normalization process. First, the size of the pad image (shown in Figure 3.25) is resized to pad of the index object (see Figure 3.26). By using the normalization in the group, the rotation of candidates is unnecessary. Therefore, the computation load is reduced.

When the object lies in the center of the image (i.e., a circle with the same center as the image and a radius with a length of one-third of the image), a circle with a radius maximum two times longer than the longest radius of $Max(r_1^i, r_2^i)$ is considered as the RoI and the center of the RoI is the same as image center. For candidate selection, instead

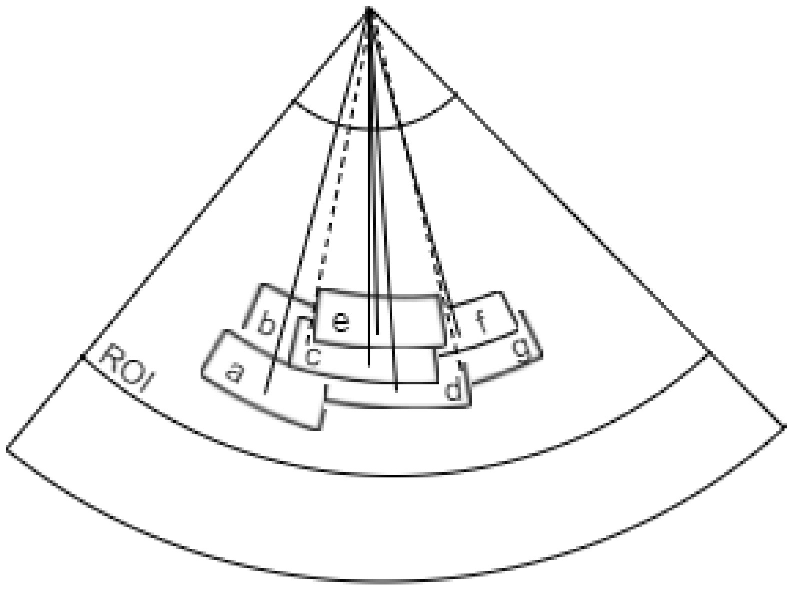

Figure 3.24: Overlapped polar candidate generation is shown. Some overlapped candidates have been shown. In this case, the region of interest (RoI) is the surrounded region among two arcs and two radii.

of two radii, two non-center-crossing chords are selected. This prevents the candidates from deformation. Figure 3.27 visualizes this technique.

The aim of engaging two classifier modules is to select the object of interest among the polar candidates. Figure 3.22 shows the proposed hierarchical construction for object detection. The hierarchical classifiers provide robustness without sacrificing computational efficiency. To extract the features of the candidates, they are first fed to a variance classifier similar to Section 3.6, wherein the input image is converted to a gray-scale image and the variance of the image is calculated. Then, this value is compared to the image variance of the last detected object.

In fact, using this classifier, smoother regions like the sky, street, snow, etc. as well as complicated candidates (with high variance images) like trees, bushes, and leaves are removed. The experiments show that this classifier can be implemented faster than the color classifier. Thus, to increase the performance of the architecture, the variance classifier is used in the first step.

Algorithm 1 explains the steps of the color binary feature extraction. In algorithm 1 $R(.), G(.), B(.)$ are the R, G, and B planes of an RGB image respectively. Due to their simplicity and robustness, binary features are well-suited for fast processing tasks of video processing including object matching and tracking. In this section, the tracker robustness is improved by combining the binary features with the color information of

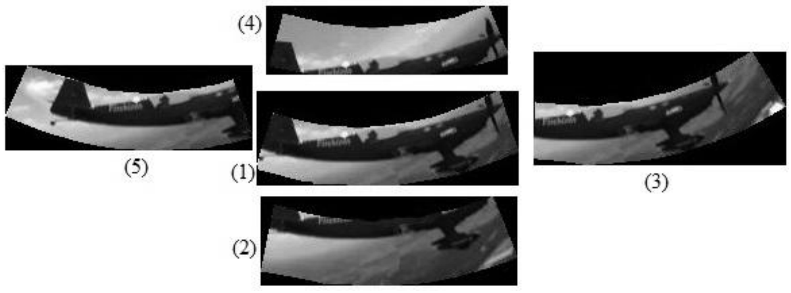

Figure 3.25: Five overlapped objects of "Rob" with their pads are shown. The pad is an inscribed rectangle surrounding the object. Objects are normalized by resizing their pad.

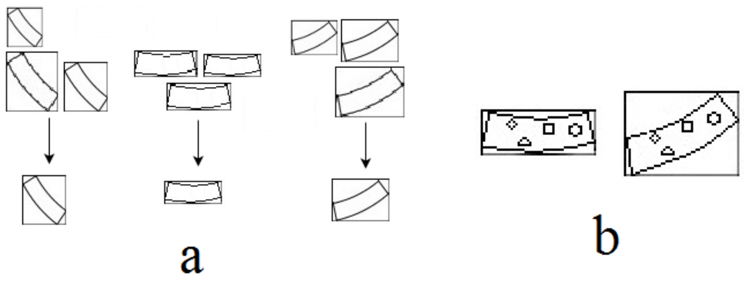

Figure 3.26: (a) Normalization process for three groups of candidates are shown. Candidates are categorized with their angle. In each group, the index candidate (i.e., the candidate with the same radii as the last object of interest) is selected. All candidates of the same group are then resized to the size of the index image. (b) shows the pixel selection process for comparison. Correspondent pixels are shown with similar shapes (i.e., square, circle, triangle, and diamond), when two objects are compared, the pair pixels are compared according to their position to the reference point of the object.

the candidates. Giving two objects to be compared, first pixel pairs within two objects are selected using a pseudo random set of pixel locations, then a set of pixel comparisons are performed (See Figure 3.26(b)). The number of comparisons m is set to 40% of the total number of points in the object N. Higher values increase the probability of object identification but also increase the computational load. The pseudo random generator generates a new set of pixels for each frame. In this way, the probable error of the pseudo random generator is minimized and the generality of the procedure is preserved. Due to non-conformity of the points of the polar (non-real) images to the Cartesian system in digital (real) images, in each line, the point with the nearest Euclidean distance in the Cartesian system to the corresponding polar point is chosen as the binary feature. Suppose Po is a single plane (i.e., color planes of red, green or blue) of a polar candidate, the pixel comparisons for binary features can be expressed as follows:

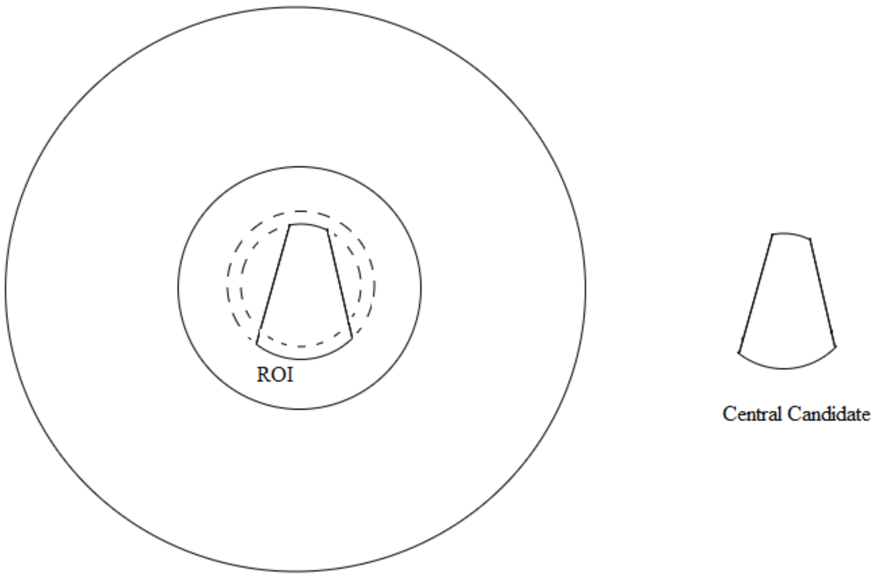

Figure 3.27: The selection method for a central candidate is shown. The central candidate is shown on the right, is circumscribed among two chords and two concentric arcs as shown in the left image.

$$B^k = \begin{cases} 1, & \text{if } Po^k > Po^{k+1} \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

Where BP is the binary feature of a polar plane Po and k is the pixel index which has been pseudo randomly selected from the all possible candidate pixels by using a pseudo random generator. By using the pseudo random generator, the same order of pixels are chosen in all candidates and the candidates are correctly compared. Suppose the total number of selected pixels is m and the total number of pixels of candidates is n. For a given pixel, if the output of all planes (B_r^k,B_g^k, and B_b^k) is equal to one, the final output pixel BO^k is set to one. As a result, B would be a binary vector [0,1] with the length of m. To compare two objects, the similarity score s is used. Given two objects with the same length of n, the similarity score is the numbers of 1 in $B(k)$ divided by the total numbers of pixel comparisons m.

$$s(Bo^k, O_d^k) = \sum_k (Bo^k \;\&\; \mathbf{O}_d^k)/m \tag{3.11}$$

This similarity score s is a basis for a decision strategy including detection of the object

Algorithm 1: color binary classifier

```
1: **Input:** Polar candidates from variance classifier $C_v$ and last detected object of interest $\mathbf{O}_d$
2: $C_r \leftarrow R(C_v), \mathbf{O}_r \leftarrow R(\mathbf{O}_d)$
3: $C_g \leftarrow G(C_v), \mathbf{O}_g \leftarrow G(\mathbf{O}_d)$
4: $C_b \leftarrow B(C_v), \mathbf{O}_b \leftarrow B(\mathbf{O}_d)$
5: for k=1: m do
   $\triangleright$ $M$ is the number of comparing pixels.
6: $B_r^k \leftarrow BP(C_r)$
7: $B_g^k \leftarrow BP(C_g)$
8: $B_b^k \leftarrow BP(C_b)$ $\triangleright$ $BP()$ is the pixel comparison with Eqn. 3.10.
9: $Bo^k \leftarrow 0$
10: **if** $B_r^k = 1 \& B_g^k = 1 \& B_b^k = 1$ **then**
11:     $Bo^k \leftarrow 1$
12: **end if**
13: end for
14: **Output:** $Bo^k$ $\triangleright$ Color Binary Feature
```

of interest or the case of object occlusion or scene departure. For a given candidate, if the similarity score s is more than 0.8, the candidate is accepted.

To address the case of multiple accepted candidates, another tracker is employed. For this ancillary tracker, the well-known tracking methods of Lukas-Kanade (LK) [Bou00] and (Speeded Up Robust Features) SURF-tracker [SNH12] were separately investigated. To improve the speed of the tracker, the mentioned trackers are applied only on the polar RoI. LK is a famous method for object tracking in rectangular videos. It supposes that the displacement of the objects in the image between two successive frames is small and approximately constant. SURF descriptors were originally proposed in [Bay+08]. They are fast and robust. Since a fast and rotation-robust method is needed for the high resolution and polar videos, the SURF descriptors were chosen as the ancillary tracker. The experiments show that the SURF-tracker in general has better results than LK. In addition, the experiments show that whenever the ancillary methods or the hierarchical branch are employed individually, tracking performance declines. Thus, the two classifiers should be integrated.

The integrator compares the output of the ancillary tracker and the proposed hierarchical classifiers to select the final object candidate. Let n denote the number of the detected object by the hierarchical branch. To decide on the final object of interest $\mathbf{O}_d$, the following rule is proposed.

$$\mathbf{O}_d = \begin{cases} NoObject, & \text{if } n = 0 \\ \mathbf{O}_h, & \text{if } n = 1 \\ \mathbf{O}_h^*, & \text{if } n > 1 \end{cases} \tag{3.12}$$

Where $\mathbf{O}_h^* = \{\mathbf{O}_h | Min(d(\mathbf{O}_h^i, \mathbf{O}_a))\}$ and $d(.,.)$ is the Euclidean distance and $\mathbf{O}_h^i$ and $\mathbf{O}_a$ are the output(s) of hierarchical and ancillary tracker branches respectively. Since the ancillary trackers show less robustness compared to the hierarchical branch, they are not recommended to be used in the two other cases of a single candidate or no detected candidate. To improve the robustness of the tracker an object model Mo containing a history of the object appearances is proposed. Let $\mathbf{O}_d^i$ denote the current object of interest, the object model Mo is then expressed as $\mathbf{M}o = \{\mathbf{O}_d^1, ..., \mathbf{O}_d^{j-1}, \mathbf{O}_d^j, \mathbf{O}_d^{j+1}, ...\}$. If $n = 0$, the tracker assumes no object was found by the hierarchical processing routine. In this case, the object of interest possibly has been gone out of the scene, occluded by other objects, rotated in an out-of-plane direction or its appearance has changed. In this case, each candidate with a similarity score s more than 0.7 (i.e., $0.7 < S < 0.8$) is compared to all members of an object model as will be discussed later. The candidate with the maximum sum of similarity scores to the model (i.e., $O_d = \{O_q | Max(\sum_{k=1}^{20} s(\mathbf{O}_d^k, \mathbf{O}_q))\}$ and $\mathbf{O}_d^k$ and $\mathbf{O}_q$ are a member of the object model and a query object respectively) is selected as a retrieved object $\mathbf{O}_d$. If the similarity score s is less than 0.7, the object of interest is assumed to have either gone out of the scene or been occluded by other objects. In this case, the system keeps the characteristics of the last detected object to compare it with the next frames and the RoI is extended to the whole image as the occluded or exited object may reappear in any other region of the image. The tracker updates the model when it satisfies the following criterion:

$$Mo \leftarrow \mathbf{O}_d^i \quad if \quad \{\forall \mathbf{O}_d^j \in Mo | s(\mathbf{O}_d^j, \mathbf{O}_d^i) < 0.6\} \tag{3.13}$$

By using this model, the tracker preserves the diversity of the object's appearance in the model and increases the overall precision. As a result, rather than only information from two adjacent frames, the temporal information from a sequence of frames is exploited. Once the number of the model members reaches 20, the second member is omitted, preserving the object of interest in the first frame i.e., $\mathbf{O}_d^1$. This reduces the deleterious effect of incorrect candidates or target drifting. The maximum model length can be set to preserve both accuracy and speed of the tracker.

3.9 Experiments and Results

In order to validate the polar object tracking methods, one set of 15 360-degree videos[2] and one set of 10 catadioptric omnidirectional videos[3] including more than 30000 frames have been selected. In the data sets, the video lengths are different and the image size is 1500×1500 pixels for 360-degree videos and 850×850 for omnidirectional videos. They include various objects of interest like a car, a motorcycle, a pedestrian, the human head, the human hand, the human body, an airplane and a balloon. The videos contain both in-plane and out-of-plane rotations. Except for "NYC" (a video with some pedestrians in New York City), in other videos, the camera is moving (some of them have drastic moving). Almost all videos have diverse degrees of occlusion, from partial to full occlusion. Sometimes the object moves out the scene and then comes back.

Table 3.3: Comparison results between the proposed MTLD and TLD in terms of recall and precision are shown. The proposed modified TLD method outperforms original TLD.

Video	TLD Recall	TLD Precision	MTLD Recall	MTLD Precision
Snorkeling	0.038	1.0	0.84	0.87
Snowboard	0.516	0.516	0.67	0.67
Rob	0.76	0.76	0.76	0.74
Street1	0.65	1.0	1.0	1.0
NYC	0.64	0.64	0.86	0.86
Harley	1.0	1.0	1.0	1.0
Car Racing	1.0	1.0	1.0	1.0
Street2	0.92	1.0	1.0	1.0
Motocross	1.0	1.0	1.0	1.0
Ice sailing	0.78	0.96	0.85	0.88
A Lap	1.0	1.0	0.89	0.94
Balloon	0.95	0.97	0.93	0.91
Caio Afeto	0.75	0.90	0.58	0.60
200 MPH	0.94	0.98	0.87	0.92
Shredding	1.0	1.0	0.95	0.92

The experiments showed that TLD cannot track objects in 360-degree images due to the lack of the rectification step. Thus, at first the input frames have been rectified and

[2]https://www.youtube.com/channel/UCjS9DSZpPzfMwyDCp1Xai1Q/videos?viewas=subscriber
[3]http://cvrg.iyte.edu.tr/datasets.htm

then inputted by TLD as done in MTLD modules. To evaluate the performance, recall $r = t_p/(t_p + f_n)$ and precision $p = t_p/(t_p + f_p)$ have been used.

Where t_p, f_n, and f_p indicate true positive, false negative and false positive respectively. In this case, precision p is the number of true positives divided by the number of all responses and recall is the number of true positives divided by the number of object occurrences that should have been detected [KMM10]. The proposed method is still sensitive to the background clutter. The results of the evaluation of the MTLD method and the TLD method in terms of recall and precision measures are listed in Table 3.3. To evaluate the proposed methods in this chapter, the F-measure (or f_1) is also used. It is calculated by using the following equation.

$$f_1 = \frac{2 \times r \times p}{p + r} \tag{3.14}$$

According to the information on this table, the proposed MTLD method outperforms the TLD method significantly. The mean recall rate has been improved by more than 20% while the precision rate stays in the same range as for the TLD method. Also, TLD is not successful in object tracking in the case of a high rate of out-of-plane rotation in videos of "Snowboard", "Street1", "Street2", and "Pedestrians". However, MTLD shows better results in these sample videos. Moreover, the TLD method fails to track the diver's head when he goes underwater. Therefore, TLD is very sensitive to environment changing, while the MTLD method can handle this variation. In the video of "Rob", in some frames, the object of interest goes to a region with a complex background which has some patterns similar to the object. Thus, neither MTLD nor TLD can track the object in those frames. The proposed MTLD method has a discriminating capability of spatially close targets with similar appearance especially in crowded scenes like "NYC". In the long videos, the proposed method can track the object with a high-performance rate. Therefore, MTLD is efficient in handling long-term occlusions. As one can see, MTLD outperforms TLD significantly. While the average precision has remained similar in both cases, the average recall has been improved by 20%.

To evaluate individual effect of the proposed modifications, each module was applied on "Snorkeling", because it has maximum recall difference between the MTLD and TLD. Restricting the search area does not have any negative effect on the recall rate and just increases the implementation speed. Let threshold modifier, distance classifier, integrator modification, changing the input source of the trainer and using the FIFO strategy for fulling the training queue be denoted by *NN*, *Dis*, *Int*, *T* and *FIFO* respectively. Table 3.4 shows the result of the above experiment.

Table 3.4: Effect of each modified module on the recall rate of "Snorkeling" is given. *NN*, *Dis*, *Int*, *T* and *FIFO* show threshold modifier, distance classifier, integrator modification, changing the input source of the trainer and using the FIFO strategy for fulling the training queue respectively.

Modules	Recall
TLD	0.038
TLD + NN	0.282
TLD + NN + Dis	0.761
TLD + NN + Dis + Int	0.770
TLD + NN + Dis + Int + T	0.821
TLD + NN + Dis + Int + T + FIFO	0.840

The results of the evaluation of recall measure for both methods of TLD and MTLD have been shown in a bar graph in Figure 3.28. According to Figure 3.28, MTLD has improved the recall variable for most of the video samples. For other video samples the recall value remains unchanged.

As indicated, the implementation speed has been efficiently increased in MTLD for all video samples except for the "Snorkeling" video. Since TLD mostly rejects all candidates of "Snorkeling" in the first steps of the detector, the candidates do not pass through all steps of the detector. Thus, some modules of TLD are not involved in most frames of this video and therefore the total time consumption is limited.

In another experiment, the average computational time per frame for both MTLD and TLD was measured. The comparison result is shown in Figure 3.29. According to the figures, MTLD searches the object of interest in a more limited area than TLD. By searching the object of interest in a limited area of the image, the number of candidates is dramatically reduced. Thus, MTLD tracks objects in a lower period as shown in Figure 3.29.

Figure 3.30 shows the whole image and the area of interest for "Rob" respectively. The probability of the presence of the object of interest in this region of interest (RoI) is higher than the rest of the image. By using this ROI the FP rate of the proposed MTLD tracker is decreased and the computation speed is increased.

To comparatively evaluate the TLD and the proposed MTLD method, the set of 360-degree videos has been used. Initial experiments showed that TLD cannot track objects in 360-degree images due to the lack of the rectification step. For this reason, all frames have first been rectified and then passed to the TLD and the MTLD method for processing.

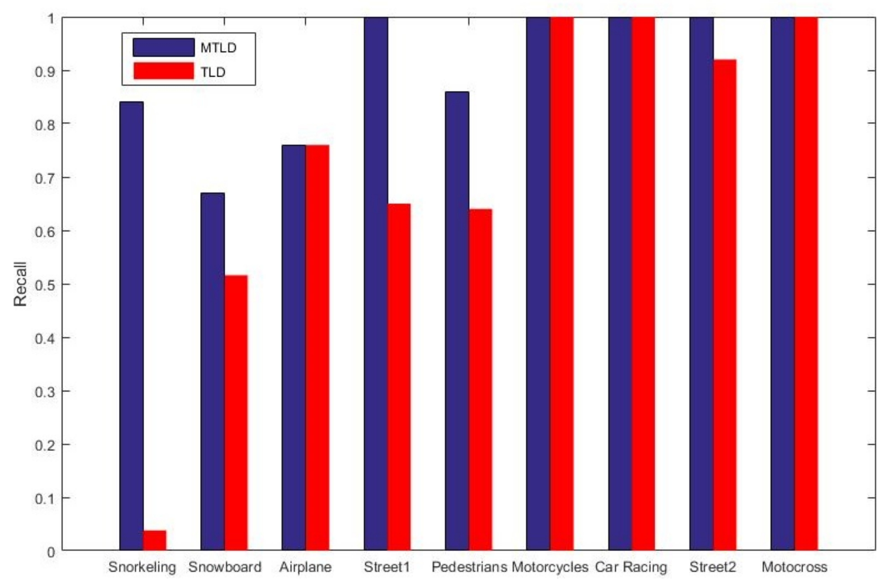

Figure 3.28: Results of recall measures for MTLD and TLD are shown.

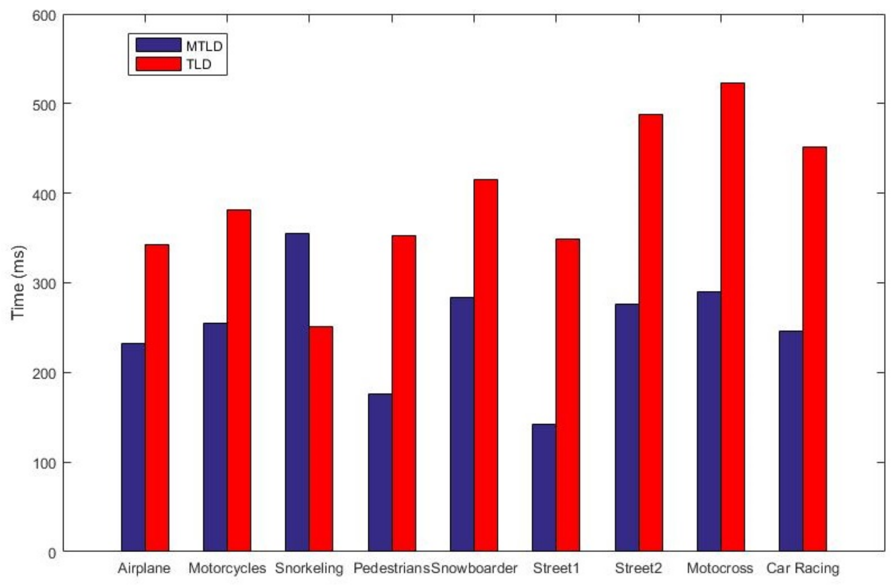

Figure 3.29: Computation time comparison between MTLD and TLD is given. MTLD is faster than TLD for all videos except for "Snorkelling". Because for this video TLD cannot track the object of interest and first steps of the detector reject all candidates in the image.

Figure 3.31 shows some examples of polar images used for the experiments. The objects of interest are represented by a polar representation.

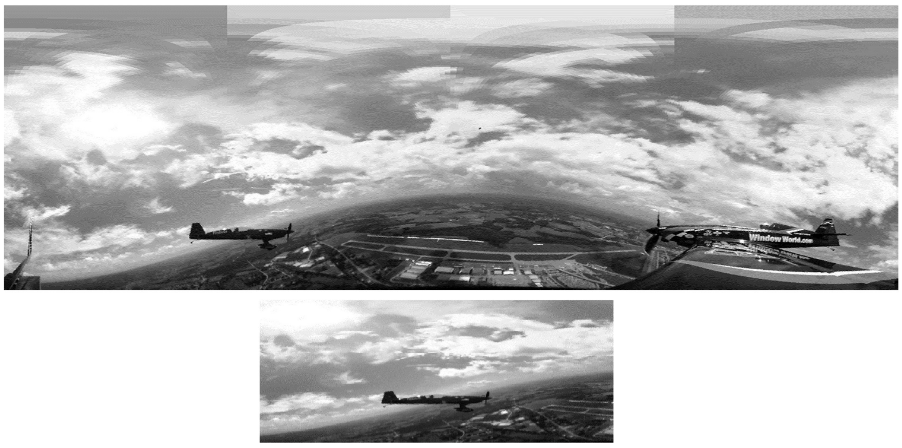

Figure 3.30: TLD searches the whole image to find the object of interest. This image shows rectified frame from the "Rob" video (top image). Limited searching area in MTLD is shown (bottom image). The probability of the presence of the object of interest in this region of interest (RoI) is higher than the rest of the image.

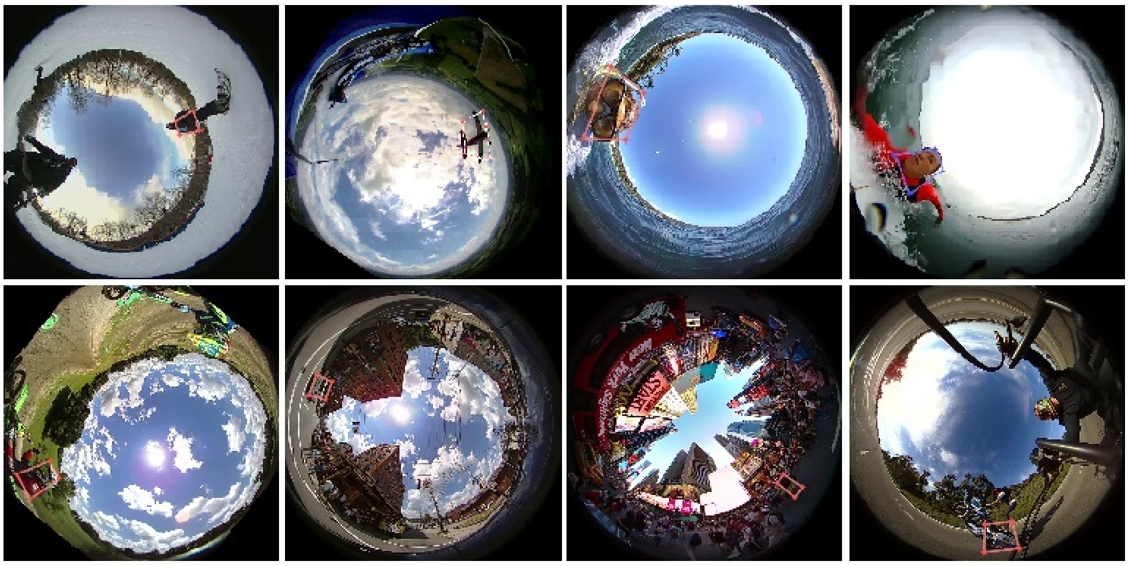

Figure 3.31: Examples of 360-degree images and objects of interest are given.

The YOLO-based trackers proposed in Section 3.5 have been implemented in C++ using OpenCV. For the YOLO-based trackers, the 360-degree data set has been used.

The snapshots of the videos used are shown in Figure 3.32, where the following videos are shown respectively:
first row (from left to right): "200MPH", "A Lap", "Balloon", "Caio Afeto", second row: "Ice sailing", "Motor", "NYC", "Park", third row: "Rob", "Shredding", "Snorkeling", "Snowboarding" and finally the fourth row shows "Sonoma" and "TPittsburgh".

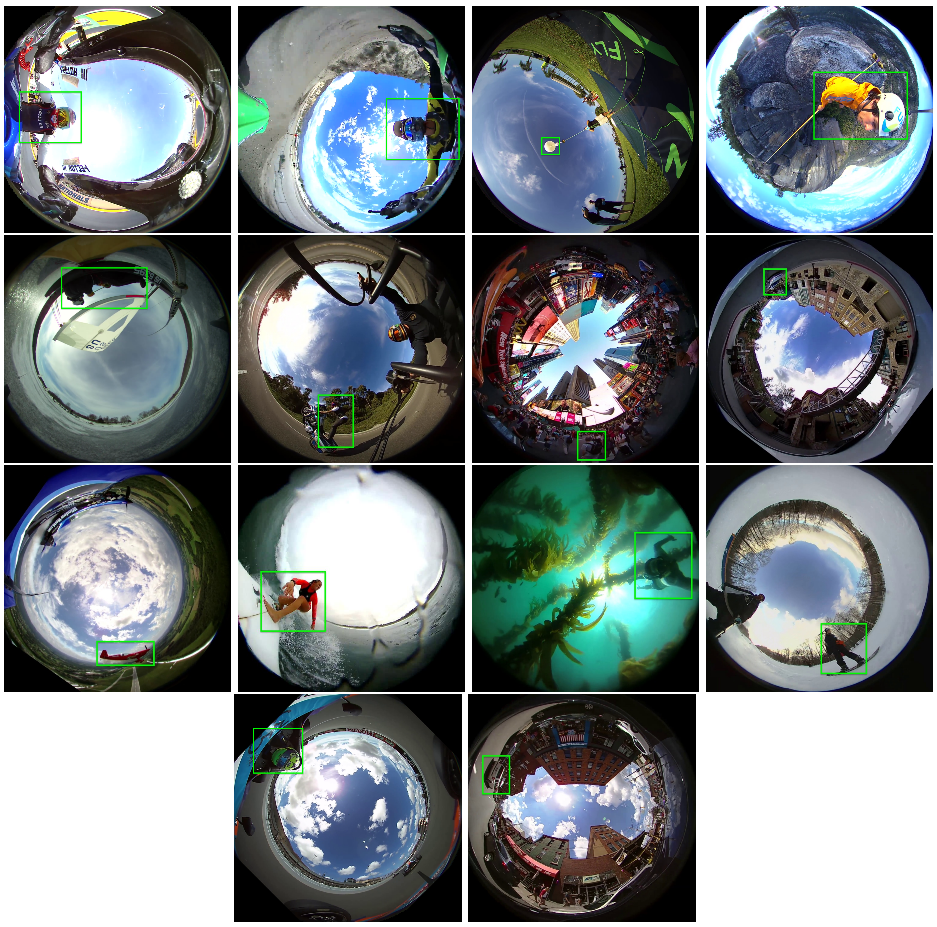

Figure 3.32: The object of interest is marked by green rectangles in 360-degree videos.

The results of the Lucas-Kanade tracker, the combination of YOLO and the Kalman filter, and the combination of the Lucas-Kanade and Kalman-YOLO are shown in Figure 3.33.

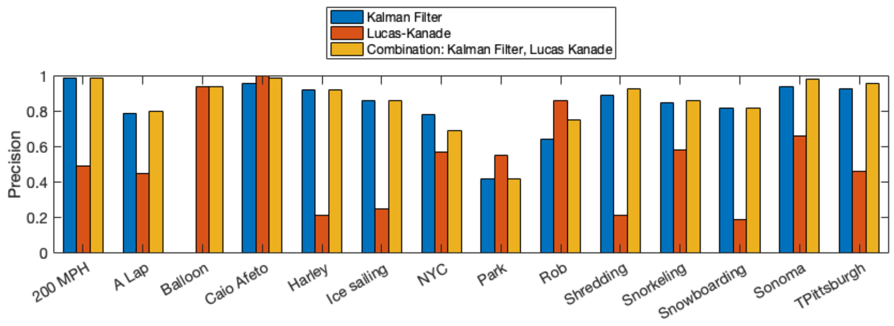

Figure 3.33: Results of the Kalman filter, Lucas-Kanade and their combination are shown.

According to the information in the table and figure, the combination of the Kalman filter and YOLO outperforms the Lucas-Kanade tracker.

Table 3.5: Tracking results for the Kalman-YOLO (left), the Lucas-Kanade (middle) and Kalman-YOLO-LK (right) are given.

Video	Desired Object	Frames	Recall	Precision	F-Measure
Rob	Red Airplane	600	0.57/1.00/1.00	0.64/0.86/0.75	0.60/0.92/0.86
Snorkeling	Diver	920	0.97/0.64/1.00	0.85/0.58/0.86	0.91/0.61/0.91
Shredding	Woman	897	1.00/1.00/1.00	0.89/0.21/0.93	0.94/0.35/0.96
Park	Car	60	1.00/1.00/1.00	0.42/0.55/0.42	0.59/0.71/0.59
NYC	Person	94	1.00/1.00/1.00	0.78/0.57/0.69	0.85/0.73/0.82
Harley	Driver	307	1.00/1.00/1.00	0.92/0.21/0.92	0.96/0.35/0.96
TPittsburgh	Car	302	1.00/1.00/1.00	0.93/0.46/0.96	0.98/0.63/0.98
Sonoma	Driver	1100	1.00/1.00/1.00	0.94/0.66/0.98	0.97/0.80/0.98
200_MPH	Driver	998	1.00/1.00/1.00	0.99/0.49/0.99	0.99/0.66/0.99
A Lap	Driver	1655	0.98/0.63/1.00	0.79/0.45/0.80	0.87/0.53/0.89
Ice_sailing	Person	1252	1.00/1.00/1.00	0.86/0.25/0.86	0.92/0.40/0.92
Caio_Afeto	Person	998	1.00/1.00/1.00	0.96/1.00/0.99	0.98/0.98/0.99
Snowboarding	Person	797	0.95/1.00/1.00	0.82/0.19/0.82	0.88/0.32/0.90
Balloon	Balloon	400	0.00/1.00/1.00	0.00/0.94/0.94	0.00/0.97/0.97
Mean	-	10380	0.93/0.91/1.00	0.83/0.51/0.89	0.88/0.61/0.94

Because the YOLO detector provides better results than the Lucas-Kanade tracker in terms of object detection. On the other hand, the Lucas-Kanade tracker cannot track objects with uniform texture, because the interesting points with strong corners do not exist. In addition, LK misses the objects when large leaps occur within a few frames due to fast movements and non-stationary cameras. This issue is understood from Table 3.5 in the videos "Snowboarding", "Shredding" and "Harley" where the object of interest has sudden movement, Lucas-Kanade loses it.
However, the Lucas-Kanade method performs slightly better than the Kalman filter for some videos. For instance, YOLO does not distinguish the object category of "balloon" and therefore, it cannot give any object information to the Kalman filter. For the "Park" and "Rob" videos, YOLO is also not able to give useable information about the object of interest to the Kalman filter and thus, the accuracy falls down. However, the Lucas-Kanade tracker is able to track the object well due to the constant movement of the objects of interest and unique surface.
Both methods contain their own advantages and disadvantages; the Kalman filter is bound to the results of YOLO, while the Lucas-Kanade method is sensitive to the movement speed and the texture of the object of interest. To improve the overall tracking accuracy, the two mentioned trackers are combined. The results of the experiment are shown in Table 3.5. According to this table, the performance is improved by the combi-

nation of both trackers. On average, the precision and F-Measure have been improved in comparison to the Kalman filter based tracker or the Lucas-Kanade tracker individually. This combination improves the results especially in a case such as the "Rob" video when YOLO misses the object, Lucas-Kanade tracks the object. In the combination case, even videos with good results using the Kalman filter show better results or the same results.

The parameters of the proposed polar object tracking in Section 3.6 including the number of overlapping candidates and the thresholds of the variance classifier and color classifiers specify the precision and recall rate. The optimal parameters have been found to give the best tradeoff between the precision and recall variables. By selecting more overlapping candidates in a specific region, the F-measure rate increases. However, the processing time also increases showing a trade-off between the accuracy and the computational cost of the proposed method.

To find the optimal value for $\Delta\Theta$ and Δr, the "Rob" video is chosen to perform the experiment. Figure 3.34 shows the calculated F-measure for this experiment. According to Figure 3.34, the optimal values for $\Delta\Theta$ and Δr are 1 degree and 4 pixels respectively. The higher values decrease the F-measure. However, the lower values of Δr do not change the F-measure significantly but the computational cost. Similar results are achieved for the other videos. Accordingly, the parameters concerning the variance classifier are set to $\alpha_1 = 0.7$ and $\alpha_2 = 1.5$ respectively. Finally the threshold for the number of matched segments is set to $tr = 6$ and the threshold for the area difference to $ts = 50$ as well.

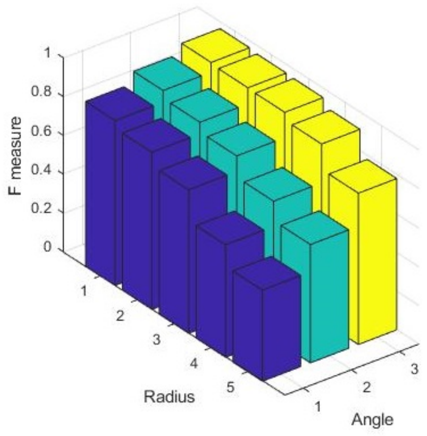

Figure 3.34: Scanning parameters for "Rob" for trapezoid-shape tracker are shown.

In the proposed trapezoid-based tracker, LK tracker is used in parallel which increases the total mean detection rate of individual frames and therefore, facilitates the long-term tracking ability. Compared to MTLD, this method shows faster implementation, because the rectification process could be removed [Del+16b]. Yet, the recall measure has been

Table 3.6: Evaluation results for the proposed trapezoid-shape tracker are shown.

	Video	Frames	Recall	Precision	F-measure
1	Rob	734	0.97	0.95	0.95
2	Car Racing	330	0.91	0.89	0.89
3	Motocross	250	0.76	0.84	0.76
4	Motor cycles	293	0.93	0.87	0.93
5	NYC	465	0.64	0.53	0.57
6	Snorkeling	683	0.63	0.67	0.64
7	Street1	58	1.0	1.0	1.0
8	Street2	294	0.92	0.87	0.89
9	Snowboarding	524	0.71	0.67	0.68
10	Surfboarder	974	0.91	0.96	0.93
	Mean	-	0.82	0.81	0.81

a little degraded. According to the information given in Table 3.6, the proposed polar object tracking method shows promising performance on various objects and scenes. The mean of F-measure for the data set is more than 81%. This value is different for different video samples, due to the image quality, the size of the object of interest in the videos, their length, and background.

The parameters concerning the SURF detector have been set in a manner in which the maximum interesting points are extracted (i.e., $sft = 10$). sft is the strongest feature threshold. Since the database contains various type of objects, objects with different sizes and shapes, diverse scenes, variable illuminations and different degrees of occlusion, the generality of the selected parameters is guaranteed. To train the matching classifier of the proposed SURF-tracker, 9486 pair-objects from 2081 frames were selected and then features were extracted for training. The frames for the training were selected from "Snorkeling", "Snowboard", "Rob" and "NYC" videos. These frames were not used

Table 3.7: Performance evaluation of the proposed SURF based tracker is shown.

Video	**Desired Object**	**Frames**	**Recall**	**Precision**	**F-Measure**
Snorkeling	Diver's head	2450	0.88	0.84	0.85
Snowboard	Snowboarder	1087	0.82	0.86	0.83
Rob	Airplane	7468	0.81	0.82	0.81
NYC	Pedestrian	341	0.94	0.96	0.94
Park	Car	58	1.0	1.0	1.0
Motorcycles	Motor cycle	293	1.0	0.98	0.98
Car Racing	Driver's head	330	1.0	1.0	1.0
TPittsburgh	Car	294	1.0	1.0	1.0
Ride	Driver's head	4194	0.94	0.93	0.93
Shredding	Woman's face	974	0.87	0.85	0.85
Average			0.868	0.865	0.860

during the evaluation. The SURF features were extracted and then matched by using *SSD* and *SAD*, and mismatched pair points were used as negative samples and matched points were used as positive samples. This selection improves the performance of the matcher classifier in the real world. To make a uniform sample set, some other negative points manually created and added to the previous negative set.

The detailed results of the proposed SURF tracker in terms of Recall, Precision and F-measure is shown in Table 3.7. According to this table, the proposed method has a discriminating capability of spatially close targets. In the video of "Snorkeling", even when the scene environment is changed several times, the tracker does not miss the object. Therefore, the proposed method is robust with respect to the environment change.

The proposed SURF tracker was implemented on an Intel Core i7-4600 CPU @ 2.10 GHz, using Matlab code (linked also to some C++ components and OpenCV functions). This non-optimized implementation on a standard PC machine achieves near the real-time with the mean performance of approximately 10 fps (varying somewhat with different test videos). The proposed method displays promising results even for the videos which did not use in the training phase. It shows the generality of the classifier.

For the "Snowboard video", the proposed tracker is robust when the object departs the camera and when it rotates in an out-of-plane direction.

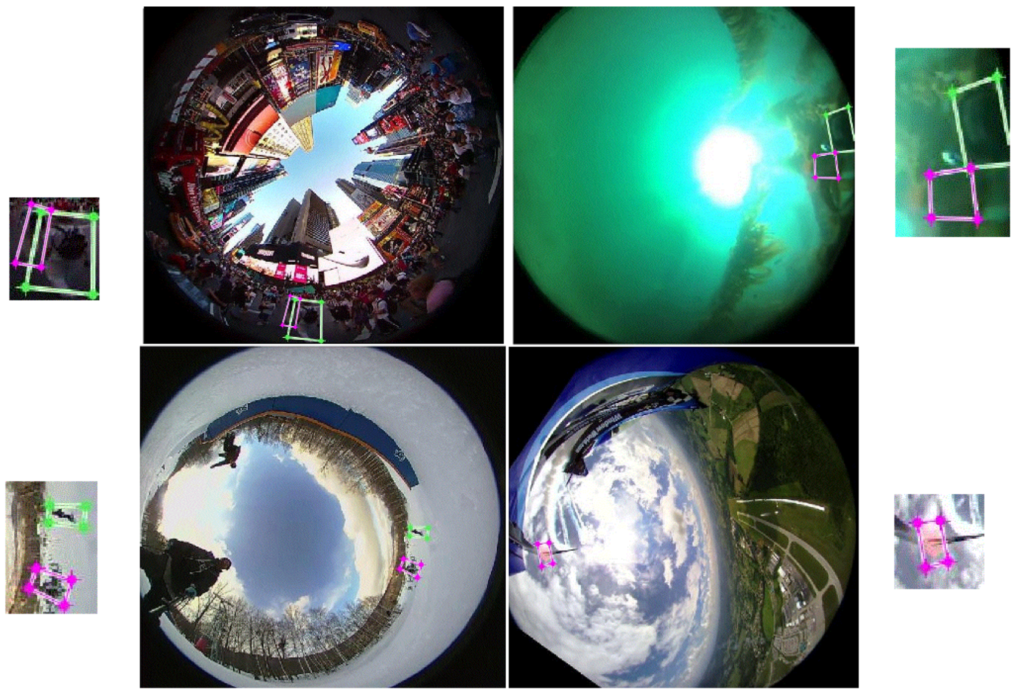
Figure 3.35: Examples where the color-binary tracker (red frame) cannot locate the object of interest (green frame), are shown.

Almost all videos have diverse degrees of occlusion from partial to full occlusion. In addition, for some videos, the object goes out of the scene and then comes back. The parameters concerning the variance classifier for trapezoid-shape tracker are set to $\alpha_1 = 0.7$ and $\alpha_2 = 1.5$ respectively. These parameters were selected using initial experiments to preserve both accuracy and speed of the tracker. Since the data sets contain various types of objects with different sizes and shapes, in diverse scenes, backgrounds, illumination conditions and different degrees of occlusion, the generality of the selected parameters is guaranteed. Figure 3.35 shows some failure detections of the proposed tracker. From left to right, in the first image ("NYC"), another person with a similar shirt and bag color is mistaken as the object of interest, is located near to the object of interest (within the polar RoI), although in the next frames when it disappears from the scene, the real object of interest is tracked. For the second and the third examples ("Snorkeling" and "Snowboard"), the same thing happens, i.e., some pseudo objects with a similar color in the proximity of the last object of interest are identified. However, when the object is moving and the background changes, the object of interest is identified. For the fourth case ("Rob"), the tracker mistakenly chooses a region with a similar color to the object of interest. This non-desired object could pass the color binary classifier but this problem is rare.

Table 3.8: Evaluation results of the proposed color binary tracker are shown.

Video	Desired Object	Frames	Recall	Precision	F-measure
Snorkeling	diver's head	2652	0.81	0.75	0.78
Snowboard	snowboarder	1319	0.71	0.66	0.68
Rob	airplane	9000	0.78	0.75	0.76
Park	silver car	58	1.00	1.00	1.00
NYC	pedestrian	465	0.90	0.87	0.88
Harley	motor cycle	293	0.99	0.95	0.97
Sonoma	driver's head	330	1.00	1.00	1.00
TPittsburgh	white car	294	1.00	1.00	1.00
Ride	driver's head	4194	1.00	0.89	0.94
Shredding	woman's face	974	0.99	0.95	0.97
Balloon	white balloon	4761	0.99	0.96	0.97
A Lap	driver's head	2149	0.98	0.97	0.97
200 MPH	driver	1173	0.99	0.99	0.99
Caio Afeto	person's hand	1347	0.72	0.59	0.65
Ice sailing	person's body	1583	0.93	0.87	0.90
Mean		-	0.885	0.837	0.857

The evaluation results of the proposed method when SURF ancillary tracker is used, in terms of the recall, the precision and the F-measure are listed in Table 3.8. Color features

Table 3.9: Results of the color binary tracker on omnidirectional videos are shown.

Video	Frames	Recall	Precision	F-measure
car101	83	0.98	0.94	0.962
car105	83	1	0.88	0.936
car108	83	1	0.92	0.963
car110	83	1	0.96	0.980
car112	83	0.98	0.91	0.947
car114	83	1	0.94	0.972
car117	82	0.98	0.92	0.955
car119	83	1	0.89	0.946
car123	83	1	0.85	0.918
car124	83	0.98	0.92	0.954

increase the discriminating capability of spatially close targets with similar appearance, especially in crowded scenes like NYC. In addition, the proposed SURF tracker is effective in handling long-term occlusions as is contained in some parts of "Rob". Generally, this method has excellent performance in short videos and fair performance in long videos like "Rob". In the video of "Snorkeling", even when the scene environment changes several times, the tracker does not miss the object. Therefore, the proposed method is robust with respect to the environment change. In "Caio Afeto", when a small object like a person's hand has a significant change in appearance, the performance of the tracker is fair.

The results of applying the binary tracker on the omnidirectional data set are listed in Table 3.9. Since this data set has fewer challenges as compared to the 360-degree data set (e.g., the camera is fixed and the object is a non-rigid car and without occlusion), the average tracker F-measure is higher than 360-degree data set.

With the aim of demonstrating the efficiency of the color binary classifier and preference of SURF-tracker for the ancillary branch, within the proposed framework various combinations of SURF-tracker versus LK-tracker as the ancillary branch and gray vs. color binary classifier as the hierarchical branch classifier were investigated to identify the best combination. The results in terms of F-measure are listed in Table 3.10. The table shows that the best results are achieved from the integration of the color binary classifier with the SURF tracker. Generally, the ancillary tracker is necessary to identify an object in the case of multiple candidate detections by the hierarchical branch. Without the ancillary tracker there is no reference to select the object and therefore the tracker misses the object and tracker performance is substantially degraded for most of the videos in the data set.

To compare the performance of the color binary classifier with the gray-based one, the candidates within a given frame are first filtered by the variance classifier and then both gray and color-based binary classifiers were applied to all of the remaining candidates. The similarity value S is calculated and plotted in Figure 3.36.

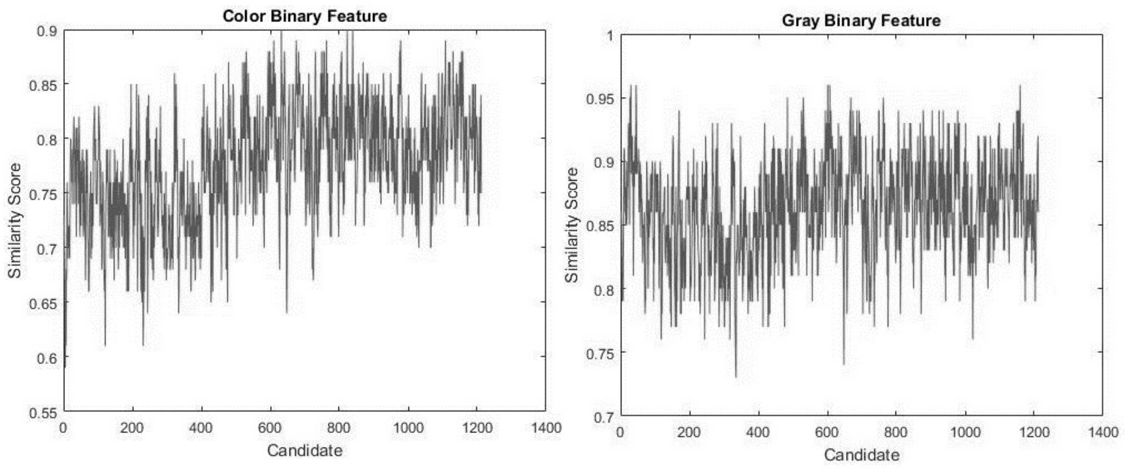

Figure 3.36: The similarity score for all candidates (after variance classifier) of "Snowboarding" for color and gray binary features is presented.

The plot shows that color binary yields less S-variation than the gray-one. In this example, for the same candidates, there are three maximums for color binary S-vector while there are five maximums for the gray one. Therefore, the discrimination capability of color-based binary features is higher than the gray one. Similar results are achieved from analysis of other videos, thus confirming the preference of the color-based classifier.

Table 3.10: Comparison results in terms of F-measure for LK and Gray-based and Color-based for the proposed color binary tracker are shown.

Video	Gray(SURF)	Color(LK)	Color(SURF)
Snorkeling	0.74	0.77	0.78
Snowboard	0.65	0.68	0.68
Rob	0.73	0.74	0.76
Park	1.0	1.0	1.0
NYC	0.86	0.87	0.88
Harley	0.92	0.96	0.97
Sonoma	1.0	1.0	1.0
TPittsburgh	1.0	1.0	1.0
Ride	0.89	0.93	0.94
Shredding	0.95	0.95	0.97
Balloon	0.95	0.97	0.97
A Lap	0.95	0.94	0.97
200 MPH	0.99	1	0.99
Caio Afeto	0.62	0.57	0.65
Ice sailing	0.90	0.92	0.90
Mean	0.829	0.844	0.857

To compare the primal object tracking algorithms, the algorithms presented in [KMM10; KMM12; Del+16b; Del+16a; SNH12; Hen+15] have been selected. These methods were selected since they are similar to the proposed method and have better performance on the polar data set than the other conventional rectangular trackers. Other open-source methods can be added to this list.

Table 3.11: Comparison of similar tracking methods in terms of F-measure is shown. The proposed color binary tracker has the best performance for most videos.

Video	SURF	PNL	TLD	MTLD	POT	KCF	Proposed tracker
Snorkeling	0.46	0.19	0.19	0.74	0.59	0.32	**0.78**
Snowboard	0.52	0.43	0.51	0.67	0.63	0.65	**0.68**
Rob	0.54	0.56	0.43	0.63	0.69	0.39	**0.76**
Street1	0.89	0.76	0.78	1.0	1.0	1.0	1.0
NYC	0.63	0.64	0.62	0.81	0.57	**0.96**	0.88
Harley	0.84	0.91	0.93	0.95	0.93	0.95	**0.97**
Car Racing	0.78	1.0	1.0	1.0	0.89	1.0	1.0
TPittsburgh	0.74	0.92	0.95	1.0	0.89	1.0	1.0
Ride	0.87	0.86	0.94	**0.96**	0.78	0.57	0.94
Shredding	0.89	0.89	0.96	1.0	0.93	1.0	0.97
Balloon	0.79	0.97	0.97	0.96	0.92	0.43	0.97
A Lap	0.76	0.90	0.91	1.0	0.91	1.0	0.97
200 MPH	0.83	0.92	0.91	0.96	0.89	**1.0**	0.99
Caio Afeto	0.21	0.64	0.69	**0.81**	0.59	0.28	0.65
Ice sailing	0.69	0.64	0.68	0.86	0.86	0.84	**0.9**
Mean	0.656	0.696	0.679	0.822	0.767	0.557	**0.857**

For selected rectangular methods [KMM10; KMM12; Del+16b; Hen+15], the images were first rectified [Del+16b] to fairly compare the results to those of the polar methods, as the results were very poor without rectification. In Table 3.11 the comparison results among PNL (Positive-Negative-Learning) [KMM10], TLD (Training-Learning-Detection) [KMM12], KCF (Kernelized Correlation Filters) [Hen+15], MTLD (Modified-TLD) [Del+16b] and POT (Polar-Object-Tracking) [Del+16a] as well as the proposed scheme and SURF-tracker(Speeded Up Robust Features) [SNH12] are considered for the comparison. Table 3.11 lists the results of the comparisons. As shown in the table, the proposed color binary tracker obviously outperforms PNL and TLD which use gray-binary features for all of the videos. The proposed method even outperforms MTLD for most videos. Although, for some videos, MTLD shows a better performance. When the ancillary trackers are used independently, (i.e., LK and SURF) the tracker becomes weak.

Especially in the case of occlusion or disappearance, the false positive detection rate increases. On the other hand, when the hierarchical detector branch is used separately,

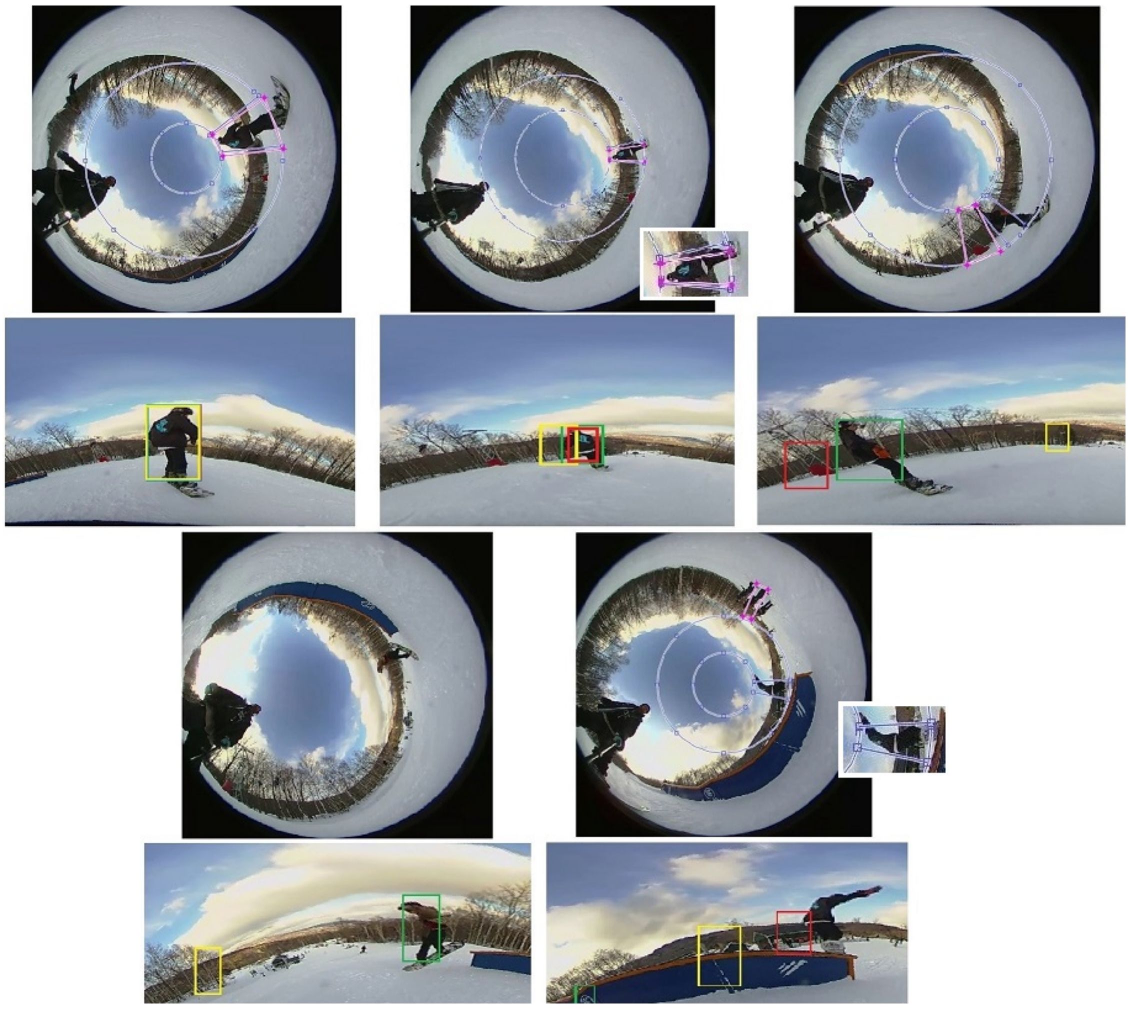
Figure 3.37: Visualizations for PNL, TLD, MTLD, POT and the color binary tracker are shown by yellow, red, green, pink and blue respectively. The color binary tracker has the best performance.

the average of F-measure falls down around 5 percent. Because in this case, when more than two objects are detected by this branch, the tracker misses the object in the current frame and also some of the next successive frames until it can retrieve the object again. Here, the importance of a hybrid architecture appears. The comparison of the proposed method with POT demonstrates preference of binary features and polar candidate selection proposed in this section. For a certain pixel, by color binary features, the probability of the false object detection when 3 pair values (from red, green and, blue planes) are compared, is less than a single comparison in gray binary features. The visualization comparisons of the methods for selected frames of the Snowboard video are shown in Figure 3.37. According to this figure, the proposed method can track the object of interest better than the other methods.

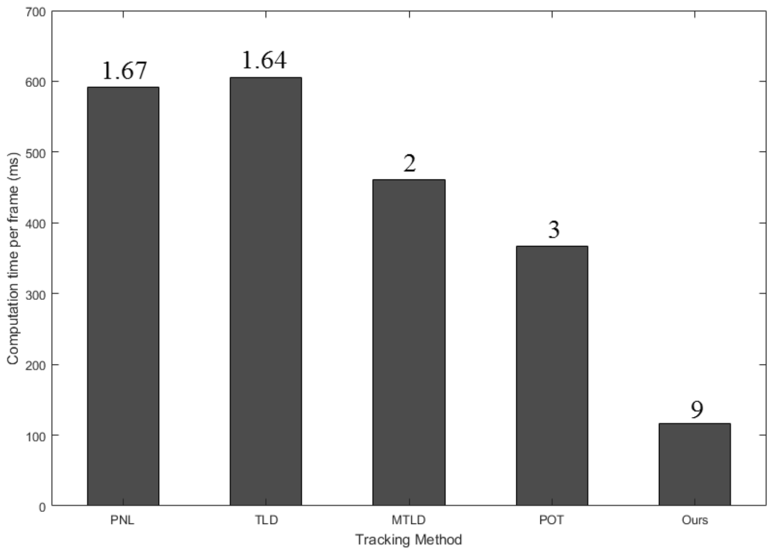

Figure 3.38: Algorithm speed comparison in terms of computation time and fps (written above the bars) is given. The color binary tracker is faster than the tested trackers.

The proposed color binary tracker has been implemented on an Intel Core i7-4600 CPU @ 2.10 GHz, using Matlab code (linked also to some C++ components and OpenCV functions). This non-optimized implementation on a standard PC machine using a single core achieves near real-time performance with a mean speed of approximately 9 fps (varying somewhat with different test videos). The results of the speed comparison among the mentioned methods are shown in Figure 3.38.

The comparison was performed under similar conditions, i.e., with the same hardware features and hybrid development environment of Matlab and C++ and with the same level of optimization. Unlike MTLD, TLD and PNL, the proposed color binary tracker does not require the costly rectification process and is therefore faster. It is also faster than POT because it extracts the point-based fast binary features from candidates instead of extracting information from the complete area of a candidate as used in POT. It should also be noted that POT does not use binary features. Finally, the color binary tracker method is three times faster than MTLD which is the third fastest in this comparison.

3.10 Conclusions and Future Trends

In Section 3.4, an efficient method for tracking unknown objects in 360-degree images has been proposed. The state-of-the-art method of TLD has been improved to overcome the tracking problems in the challenging conditions of 360-degree images. The resolution of the 360-degree images is much higher than the TLD data set, thus the searching area is restricted to decrease the computation load in the MTLD method. The experimental results show that proposed MTLD method outperforms the state-of-the-art method of TLD. This method can track objects even when they have out-of-plane rotation and varying environment. However, like TLD, the MTLD tracker is fragile to complex backgrounds when the desired object is similar to a part of the background. Therefore, future work will address this concern.

Section 3.6 has proposed a polar tracker to track polar objects directly in 360-degree images. This tracker restricts the searching area to decrease the computation load. Moreover, a hierarchical structure with color-based classifiers has been presented. The experimental results show that the proposed method can track various objects in diverse scenes in a promising level.

A training-based matching method for SURF-based object tracking in 360-degree videos has also been proposed in Section 3.7. This method has another training-based mechanism for automatic detection of challenging situations like out-of-plane-rotation, occlusion and departing the camera. The algorithm adapts itself to handle these situations. The experimental results demonstrate the robustness of the proposed tracker.

A polar model for object definition in the 360-degree data set was also proposed in this chapter. This method first segments the input frame into a polar region around the labeled object and then selects overlapped polar candidates within the polar region. The polar candidates are then classified by two hierarchical classifiers of variance and color binary. The discriminatory of binary features was improved by combining them with color information. The experiments on the 360-degree and a catadioptric omnidirectional data set show that the proposed color binary method boosts the tracking performance for diverse objects in challenging real-world scenarios and outperforms similar trackers (i.e., TLD, MTLD, POT, KCF, SURF, and PNL). The proposed method has comparable tracking performance to MTLD but is faster than MTLD. The proposed method is also faster and more precise than POT because it uses binary features.

In the future, the proposed trackers will be extended to multiple object tracking. The parallel processing will be employed to reduce the computational cost. In addition, the use of HSV color space in binary features will be considered. Furthermore, the tracker sometimes mistakes a similar object near the desired object. This fault can be improved by adding more features to the feature vector. For rigid objects, the tracker is often able to retrieve the desired object in the next frames; however, this is not the case for articular objects. The future work will also focus on articular object tracking by dividing the articular object into smaller parts (like body parts in humans) and applying the tracker to each part. The real-time capability can be achieved by optimizing the implementation of this approach. In fixed-camera scenarios, the use of background detection and subtraction methods between the RoI selection and object detection parts is suggested.

Chapter 4

Railway Crack Detection

This chapter proposes scientific methods for crack detection and classification on concrete sleepers. The chapter is based on the author's papers (C_6) and (C_7) recently published in the International Journal of Pattern Recognition and Artificial Intelligence, 2018 [Tab+18] and the proceedings of the 10th International Conference on Computer Recognition Systems, 2017 [Del+17]. This paper received the best paper award at this conference.

The organization of this chapter is as follows: Section 4.1 states the problem of crack detection on the concrete sleepers of the railway system. Section 4.2 shows a literature overview of the crack detection methods on streets, bridges and wooden sleepers. Two proposed algorithms for sleeper location and crack detection and classification based on image processing and machine learning are introduced in Section 4.3. Experiments and results of the proposed methods are presented in Section 4.4. Finally, conclusions are drawn and further possible research directions are listed in Section 4.5.

4.1 Problem Statement

Railway structure is subjected to dis-formation and damage because of weather condition, traffic and, topographic situation. Sleepers are the infrastructure under the railroad made from wood or concrete. The damage occurs in the form of cracks on the sleepers and can introduce dangerous situations depending on the daily traffic load and the crack type which would require immediate actions. Conventionally, the railway sleepers are inspected manually which is an extremely time-consuming and labor-intensive-maintenance task, because the inspection areas include non-accessible locations or limited visible points due to their geometry. Similar problems exist in huge and tall structures like cable bridges, high rising towers and dams as well [JWKN15]. To overcome the aforementioned problems, several approaches have been presented in the literature in the form of research works and industrial solutions. In this chapter an integrated system including hardware and software solutions based on a vehicle called "rail mapper" [1] is employed to acquire data followed by the proposed crack detection process. Data acquiring and crack detection include capturing the sleeper frames in several locations and automatic detection of the sleepers and the cracks throughout. The final output of the detection module is an identifier which indicates whether the sleeper is a healthy one, otherwise the cracks are detected and classified for further actions. Some pre-processing methods enhancing the captured frames quality are applied prior to sleeper location. The sleeper body is extracted in the first step and the cracks are detected and classified in the second step. To extract the sleeper body, different methods are employed in this chapter in which each comes with some cons and pros. Template matching and a generic feature-based method are the main approaches for the first step. The second step detects the possible cracks and classifies them using geometric features and supervised classifiers. The SVM and random forest classifiers are employed to select the right candidates in both steps.

4.2 Related Work

Few industrial works exist in the literature addressing the problem of quality control for crack and damage detection and classification with different use-cases. The applications and use-cases include crack detection in the texture of the structures like streets, pavement, bridges and wires [YNH08; Pra+16; OC13; FMH06; SF06].

In this section, an overview of the available structures' quality control methods is given. Based on the quality control methods, they can be categorized in the following two groups:

1. learning-based quality control methods
2. Non learning-based quality control methods

In the following sections, some quality control methods including the state of the art based on the aforementioned classes are explained.

4.2.1 Learning-based Quality Control Methods

Detection of cracks on bridge decks is a vital task for maintaining the structural health and reliability of concrete bridges. Robotic imaging can be used to obtain bridge surface image sets for automated on-site analysis. In [Pra+16], a novel automated crack

[1]http://www.igi-systems.com/railmapper.html

detection algorithm using the STRUM (Spatially Tuned Robust Multi-feature) classifier and results on real bridge data with a state-of-the-art robotic bridge scanning system are presented. By using machine learning classification, the need for manually tuning threshold parameters is eliminated. The algorithm uses a robust curve fitting method to spatially localize potential crack regions even in the presence of noise. Multiple visual features that are spatially tuned to these regions are computed. Feature computation includes examining the scale-space of the local feature in order to represent the information of the crack. The classification results are obtained with real bridge data from hundreds of crack regions over two bridges. In order to create a composite global view of a large bridge span, an image sequence from the robot is aligned together to create a continuous mosaic. A crack density map for the bridge mosaic provides a computational description as well as a global view of the spatial patterns of bridge deck cracking. The bridges surveyed for data collection and testing include Long-Term Bridge Performance program's (LTBP) pilot project bridges at Haymarket, VA, USA, and Sacramento, CA, USA [Pra+16].

A fully integrated system for the automatic detection and characterization of cracks in road flexible pavement surfaces, which does not require manually labeled samples, is proposed in [OC13] to minimize the human subjectivity resulting from traditional visual surveys. The first task addressed, i.e., crack detection, is based on learning from samples paradigm, where a subset of the available image database is automatically selected and used for unsupervised training of the system. The system classifies non-overlapping image blocks as either containing crack pixels or not. The second task deals with crack type characterization, for which another classification system is constructed, to characterize connect components of the detected cracks. Cracks are labeled, with appropriate labels. Moreover, a novel methodology for the assignment of crack severity levels is introduced, estimating for the width of each detected crack.

Rail inspection is a very important task in railway maintenance for traffic safety issues and in preventing dangerous situations. Monitoring railway infrastructure is an important aspect in which the periodical inspection of the rail rolling plane is required. Railway structure can have different types of anomalies such as defects of rail surface and sleepers, missing of fastening elements and deviations in the contour of the ballast. Up to the present days, the inspection of the railroad is operated manually by trained personnel. A human operator walks along the rail track searching for rail anomalies. This monitoring way is not more acceptable for its slowness and subjectivity. A vision based technique is presented in [Maz+05] to automatically detect the presence or absence of the fastening elements (also named bolts) that fix the rail to the sleepers. The images are acquired by a digital line scan camera installed under a train. Subsequently these images are pre-processed by using the wavelet transform with Haar and Daubechies approximation coefficients. The obtained coefficients are fed as input to two different neural networks: the first one identifies the bolts candidates and the second one validates the bolt recognition process. The final detecting system has been applied to a long sequence of real images showing high robustness and good performances.

In [Ste+02], another vision-based technique for automatically detecting the absence of the fastening bolts that secure the rails to the sleepers is presented. The inspection system uses images from a digital line scan camera installed under a train. The images are preprocessed by using several combinations of wavelet transform and principal component analysis methods. Two different types of classifiers analyze the images in order to evaluate the pre-processing technique that gives the highest rate in detecting the presence of the bolts. The final detecting system (the best combination pre-processing technique and classifier) was applied on a long sequence of real images showing a high reliability and robustness.

In [Moh08], a machine vision algorithm is designed and applied for the visual inspection of railway sleepers to identify the flaws associated with the sleepers such as cracks on the sleepers, and condition of the rail fastenings in the scene, etc. This machine vision algorithm is experimentally evaluated using 200 real images of the railway sleepers to determine the condition of those wooden railway sleepers. In this process image acquisition, the first stage of any vision system, is carried to capture the real images. Next to the image acquisition process, a machine vision algorithm using various methods of image processing techniques is applied to the image data collected in order to extract its features such as the number of cracks, length of the crack, width of the crack, and length of the metal plate. Finally, these extracted features using a machine vision algorithm are then stored in separate feature vectors and were used further for the classification task using pattern recognition techniques. Classification of the features extracted is carried out by using classifiers like Support vector machines (SVM) and Radial Basis function (RBF), though there are several other classifiers available.

4.2.2 Non Learning-based Quality Control Methods

Crack detection on concrete surfaces is the most popular subject related to the inspection of the concrete structures. The conventional method of crack detection is performed by experienced human inspectors who sketch the crack patterns manually. Some automated crack detection techniques utilizing image processing have been proposed. Although most of the image-based approaches pay attention to the accuracy of the crack detection results, the computation time is also important for the practical use, because the size of the digital images reaches even more than 10-mega pixels, negatively affecting the processing time. In [YNH08], an efficient and high-speed method for crack detection employing image processing is proposed. To reduce the computation time, the authors use the ideas of the sequential similarity detection algorithm and active search (SSDA). According to the concept of SSDA, some parts of the process are terminated or skipped.

Interest in automatic crack detection on concrete structure images for non-destructive inspection has been increasing. In general, there are various noises such as irregularly illuminated conditions, shading, blemishes and divots in the concrete images. These lead to difficulties for automatic crack detection. Two pre-processings in order to remove such noises for crack detection are presented in [FMH06]. First, slight variations like irregularly illuminated conditions and shading are removed from concrete images by the subtraction pre-processing with the smoothed image. Secondly, a line filter based on the Hessian matrix is used to emphasize line structures associated with cracks. Finally, thresholding processing is used to separate cracks from the background.

The problem of deteriorating pipeline infrastructure is widely apparent. Since a complete rebuilding of the piping system is not financially realistic, municipal and utility operators require the ability to monitor the condition of buried pipes. Thus, reliable pipeline assessment and management tools are necessary to develop long term cost effective maintenance, repair, and rehabilitation programs. In [SF06] a simple, robust and efficient image segmentation algorithm for the automated analysis of scanned underground pipe images is presented. The algorithm consists of image pre-processing followed by a sequence of morphological operations to accurately segment pipe cracks, holes, joints, laterals, and collapsed surfaces, a crucial step in the classification of defects in underground pipes. The approach can be completely automated and has been tested on five hundred scanned images of buried concrete sewer pipes from major cities in North America.

In [Bab09], computer vision methods for measurement of rail gauge, and reliable identification and localization of structural defects in railroad tracks are presented. The rail gauge is the distance between the innermost sides of the two parallel steel rails. Two methods for evaluation of rail gauge are proposed which were designed for different hardware setups: the first method works with two pairs of unaligned video cameras while the second method works with depth maps generated by paired laser range scanners. A method for detection of rail defects such as damaged or missed rail fasteners, tie clips, and bolts, based on correlation and MACH filters is developed. Lastly, to make the algorithms perform in real-time, the GPU based library for parallel computation of the above algorithms is developed. Rail gauge is the most important measurement for track maintenance, because deviations in gauge indicate where potential defects may exist. A vision-based method for rail gauge estimation from a pair of industrial laser range scanners is developed. In this approach, the method starts with building a 3D panorama of the rail out of a stack of input scans. After the panorama is built, the FIR circular filtering and Gaussian smoothing are applied to the panorama buffer to suppress the noise component. In the next step, the rail heads in the panorama buffer are segmented. The method which detects railroad crossings or forks in the panorama buffer is used. If they are not present, the rail edge using robust line fit is found. If they are present an alternative solution is taken: the rail edge positions are predicted using the Kalman filter. In the next step, common to both fork/crossings conditions, the adjusted positions of rail edges using additional clustering in the vicinity of the edge are found. The rail head surface is approximated by the third degree polynomial and then two plane surfaces are fitted to find the exact position of the rail edge. Lastly, using rail edge information, the rail gauge is calculated and it smoothes it with 1D Gaussian filter.

Over the last few years research has been oriented toward developing a machine vision system for automatically locating and identifying defects on rails. Rail defects exhibit different properties and are divided into various categories related to the type and position of flaws on the rail. Several kinds of interrelated factors cause rail defects such as type of rail, construction conditions, and speed and/or frequency of trains using the rail. In [Man+04] an experimental comparison among three filtering approaches based on texture analysis of rail surfaces is presented to detect the presence/absence of a particular class of surface defects (corrugation).

Periodic inspections are necessary to keep railroad tracks in the state of good repair and prevent train accidents. Automatic track inspection using machine vision technology has become a very effective inspection tool. Because of its non-contact nature, this technology can be deployed on virtually any railway vehicle to continuously survey the tracks and send exception reports to track maintenance personnel. However, as appearance and imaging conditions vary, false alarm rates can dramatically change, making it difficult to select a good operating point. In [GPC15] extreme value theory (EVT) is used within a Bayesian framework to optimally adjust the sensitivity of anomaly detectors. By approximating the lower tail of the probability density function (PDF) of the scores with an Exponential distribution (a special case of the Generalized Pareto distribution), and using the Gamma conjugate prior learned from the training data, it is possible to reduce the variability in false alarm rate and improve the overall performance.

This method has shown an increase in the defect detection rate of rail fasteners in the presence of clutter.

An automated video inspection system of joint bars is developed in [oTr06] to detect cracked joint bars. The system utilizes high-resolution scan line cameras and two joint bar detection laser sensors mounted on a hi-railer. In two field demonstrations, the system detected cracks in joint bars with acceptably low false alarm rates (40 percent of detected cracks were confirmed and 60 percent were rejected by the system operators). Though the system missed 15 percent of the cracks, none of the missed cracks were center cracks. The crack detection algorithm is being refined and tested to reduce the number of missed cracks and false detections caused by high ballasts and vegetation, grease, mud, or other conditions on or near joint bars.

4.3 Sleeper and Crack Detection

The proposed sleeper processing method consists of two steps namely sleeper detection and crack detection. First, this module detects the sleeper within the input image and the second module then finds the cracks on the sleeper and classifies it according to the crack width. For each module, 2 different algorithms are proposed and compared. An overall map linking the proposed algorithms is presented in Figure 4.1. The input frames are captured by using a high-resolution camera installed on the rail mapper, the frames are then forwarded to image processing based modules for detecting the sleepers and possible cracks on them. After the sleeper and crack detection, a partial classification task is performed. The tasks of detection and partial classification of the cracks are performed in two different and sequential steps. First, the valid images wherein the whole sleeper is contained are labeled. In the proposed application a valid image is an image in which the whole sleeper concerning the position of the camera appears and is not covered by gravel, garbage or other road metal. Also, each frame must contain just one full sleeper. After automatic labeling of the valid images, the possible existing cracks are detected and classified further from the cropped sleeper images. Figure 4.4 shows some sample images of valid and invalid images in terms of the presence of the sleeper.

Different technical and scientific challenges emerge in each aforementioned step. The first challenge is the illumination problem in which the light of the illumination system is not uniformly emitted and distributed on the whole sleeper surface due to environmental effects (e.g., weather and light condition). To solve this problem, besides the physical adjustment of the illumination set, the proposed system uses a pre-processing step to enhance the quality of the input images. The camera frame rate is tuned so that the

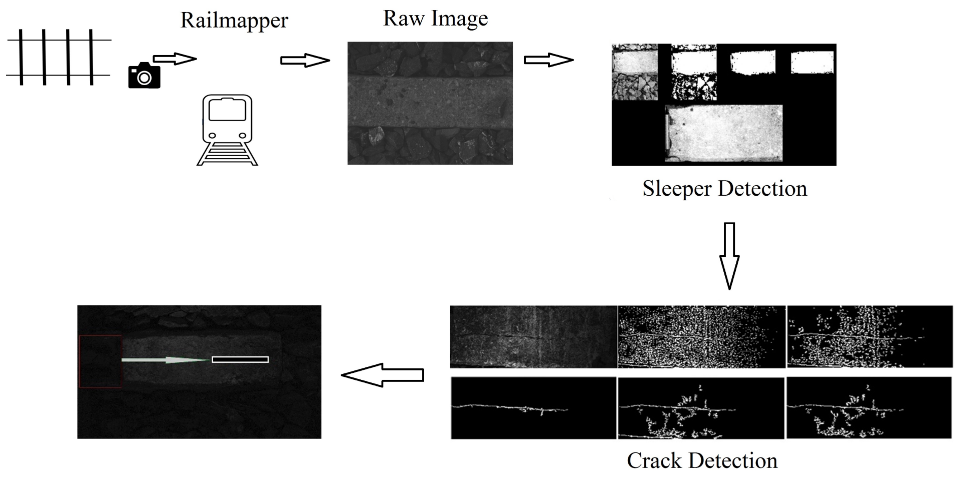

Figure 4.1: An overall view linking the proposed algorithms and general framework is shown. The input frames are captured using a camera installed on the rail mapper, the frames are then forwarded to the next modules for detecting the sleepers and possible cracks on them. After the sleeper and crack detection, the partial classification task is performed.

full sleeper appears in more than one frame and the camera setup has been adjusted to achieve high quality image sequences with the lowest possibility of blurring and to maximize the image brightness. Another challenge is the existence of background clutter due to gravels, leaves, garbage or coating algae. This problem cannot be avoided easily. However, the proposed training-based approach for sleeper detection addresses this problem by increasing the accuracy of the candidate selection phase and mitigating false decisions.

Two different methods have been used for filtering the valid images and detecting the sleepers. The methods use some image pre-processing techniques in order to reduce the image noise and adjust the contrast for further detection tasks. The description of each method is given sequentially.

The first method is a generic feature-based method in which the sleeper is extracted based on some geometrical and statistical features. In this method, three sequential operations are applied to each frame after the image pre-processing step:

- image binarization,
- binary object extraction,
- extracting geometrical features.

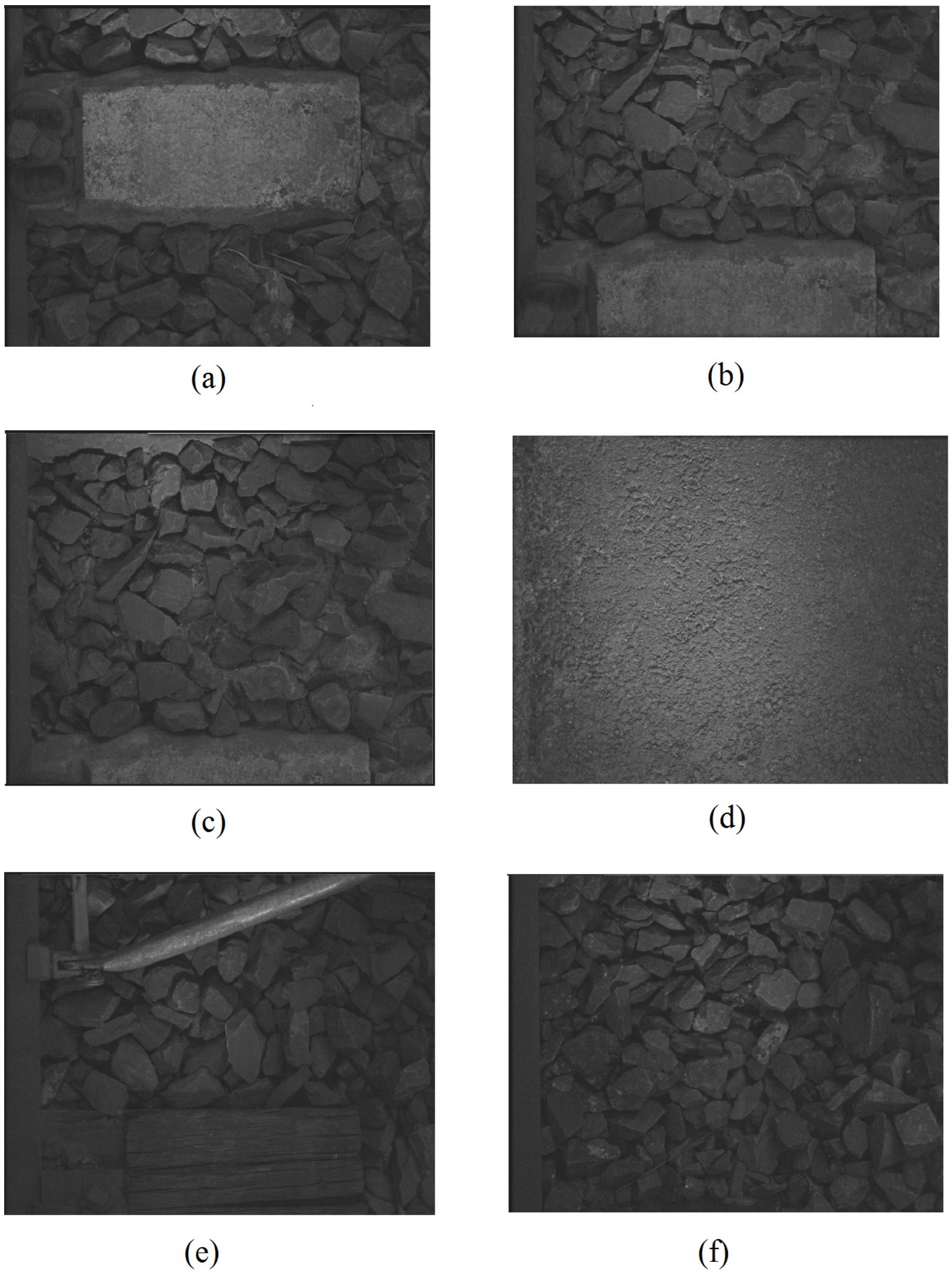

Figure 4.2: Samples of valid and invalid images are shown. (a) shows a valid sleeper, (b) an image with a part of sleeper and stones around it, and (c) a part of a wooden sleeper as well as a metal object, (d) a part of a sleeper and stones around it, (e) only a part of a sleeper and, (f) gravel without a sleeper.

In the image binarization part, the common widely used Otsu's method is employed [Ots79] wherein after applying the threshold, a binary image is generated. The biggest connected object out of the generated binary image is extracted throughout the next operation. In the valid de-noised images, the sleeper is supposed to be the biggest connected object. Thus, the approximate area containing the sleeper is specified. The exact location of the biggest detected object is determined based on some thresholds imposed by the sleeper size. In practice, different threshold values corresponding to the size and the shape of the sleeper are used to filter out the non-sleeper objects. The description of this method is given in Algorithm 2.

Algorithm 2: Sleeper detection and extraction based on the shape features

```
1 Input: Moving camera frame $\mathbf{I}$, threshold values $t_1, t_2, ...$
2 Output: Full sleeper frame from sleeper containing frames or NULL
    $\mathbf{J} = imgPre(\mathbf{I})$
    $\mathbf{I}_b = imgBin(\mathbf{J})$
    $\mathbf{I}_e = imgObj(\mathbf{I}_b)$
    $[x_1, y_1, x_2, y_2] = imgSmooth(\mathbf{I}_e)$
    if $\mathbf{I}_e$ satisfies the sleeper size characteristic specifies by $t_1$, $t_2$, ... then
      $\mathbf{I}_o = imgCrop(\mathbf{I}, x_1, y_1, x_2, y_2)$
      Return $\mathbf{I}_o$
    else
      Return NULL
    end if
```

The main drawback of the above algorithm emerges in the connected binary object extraction ($imgObj(I_{bin})$) function due to a high intrinsic similarity between the sleeper and ballast textures. As a result, the extracted connected object might include ballast texture partially or fully. To prevent this situation, some morphological operations are applied before object selection. According to the performed experiments the combination of closing and opening operations outperforms each operation individually. However, this modification does not diminish the drawback completely.

To enhance the accuracy by strengthening the detection part, a candidate selection phase is added based on the support vector machine (SVM) classifier on top of the Algorithm 2. During this phase, the frames containing the sleepers are tried to be selected from the output of the first part. Several features are used to build the feature vector as follows:

1. mean of the frame pixel intensities,
2. standard deviation of the frame pixel intensities,
3. mean of the gradient magnitude values of the frame,
4. standard deviation of the gradient magnitude values of the frame,
5. mean of the gradient directional values of the frame,
6. standard deviation of the gradient directional values of the frame.

The selected frames are then forwarded to the crack detection module which is described later.

The second solution for the sleeper detection problem searches a common template object among the sleepers and performs the task of template matching. The template matching is performed on each pixel of the input image and similarity between the template and the source image is computed. To calculate the similarity (matching) score, the sum of square differences (SSD) and cross correlation (CC) based matching algorithms are used. The selected 2D template used for sleeper detection is a sleeper fastener shown in Figure 4.3 (right). The fastener is always a part of the sleeper even for the damaged sleepers. To detect a sleeper in a frame, the template is traversed over the entire input image from top-left to bottom-left. The matching measure is determined on each pixel of the input image lying under the template. An exhaustive search (i.e., on entire image) is performed only in the first frame of the video. In the preceding frames, the temporal information of the detected location of the sleeper region is exploited to limit the search area in the succeeding frames.

Let the input image be indicated as $\mathbf{I}$ and template image as $\mathbf{T}$ with size $h \times w$. In the template image, a pixel is indicated as $\mathbf{T}(i, j)$ where i and j represent the horizontal and vertical positions of the pixel respectively. Assume that $\mathbf{T}$ is being matched with a rectangular region in the input image $\mathbf{I}$, where a pixel in $\mathbf{I}$ is represented as $\mathbf{I}(x, y)$. SSD is stated as:

$$\mathbf{S}(x, y) = \sum_{i=1}^{m} \sum_{j=1}^{n} \Big(\mathbf{I}(x+i, y+j) - \mathbf{T}(i, j)\Big)^2 \tag{4.1}$$

The template scans the fastener side of the sleeper from top to bottom, wherever this equation reaches its maximum, it would be the location of the sleeper.

The second step of the proposed approach concerns crack detection and classification. In this step, the main sleeper image from the sleeper detection module is enhanced using

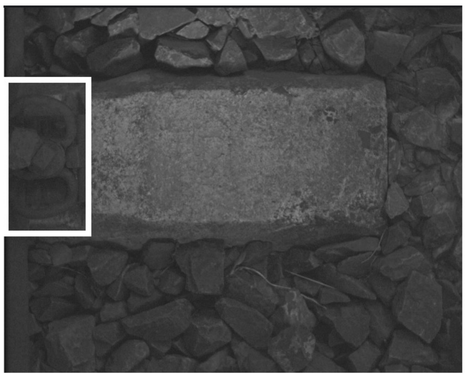
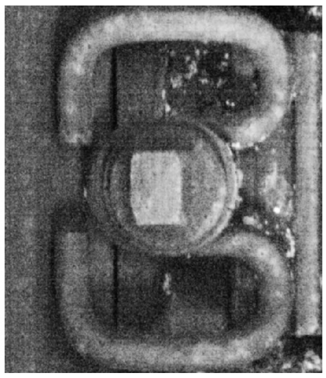

Figure 4.3: The fastener of a sleeper is shown inside a rectangle in the left side of the left image. The template matching algorithm finds the location of this fastener. The right image shows the template used for sleeper detection. It is the image of a railway fastener. The proposed algorithm searches for the location of the sleeper by scanning this template in the input image.

the histogram equalization method and low pass filtering. Then, the enhanced image is converted to a binary image by using an adaptive threshold method and then, tiny noisy particles of the resulting binary image are removed by using some area circumstances (i.e., filtering out the particles with a small area). The resulted binary image can include a crack. To remove non-crack holder candidates, two different decision scenarios are proposed below.

1. hierarchical structure method,
2. learning based method.

In the hierarchical scenario, the enhanced image is covered using the binary image and consequently some crack candidates are generated and the normalized density pixel intensity (D) of the enhanced image in the regions where the correspondent binary mask pixels are equal to one (i.e., $\mathbf{B}(x,y) = 1$), is calculated. The NPDI of candidate i is expressed as follows.

$$D_i = \sum_{x,y} \frac{\mathbf{E}_i(x,y)}{A_i}, \quad \text{if} \quad \mathbf{B}(x,y) = 1 \tag{4.2}$$

In this equation, $\mathbf{B}$ is the binary image and A_i is the area of the ith crack candidate. Therefore, for each candidate in the binary image, the D number is calculated from the enhanced gray-scale image. To enhance the accuracy of the crack detection module and avoid false rejection, an auxiliary morphological opening operation with a window size of [10 1] is applied to separate the sticking non-crack component(s) from the real crack in the vertical direction. Suppose the sleeper image is **I** and $O_1, ..., O_k$ are the crack candidates of **I**, H_i stands for the height and X_i is the bounding box of O_i, the algorithm for crack detection based on the hierarchical structure decision scenario is given in Algorithm 3.

Algorithm 3: Hierarchical method for filtering the crack holder sleepers

```
1 Input: Sleeper frame $I$ and threshold values $t_A, t_N, t_{HU}, t_{HL}, t_{BU}, t_{BL}$
2 Output: 0, 1
    3 for i=1: k doif $A_i \leq t_A$ then  Return 0
    else if $D_i > t_N$ then
      Return 0
    else if $H_i \leq t_{HL} \vee H_i \geq t_{HU}$ then
      Return 0
    else if $t_{BL} \leq X_i \leq t_{BU}$ then
      Return 0
    else
      Return 1
    end if
```

It is worth mentioning that the two classes of the cracks and non-cracks are not linearly discriminated in the feature space which causes the hierarchical method to incorrectly classify non-crack sleepers as crack-holders. This problem can be mitigated by using a supervised classifier for the detection task. In this approach, after applying adaptive thresholding and morphological operations, a connected component analysis (CCA) is performed for the candidate generation (i.e., to give a unique label to each candidate) and then from each candidate some features are calculated and fed to a pre-trained classifier. Suppose, *imgEnh* is the image enhancement function, *imgAdp* is the adaptive threshold function, *imgMor* is the function for applying morphological operations and *imgCCA* applies CCA on $I_m or$ for labeling the candidates, the description of this approach comes in Algorithm 4.

The list of features is as follows:

1. Location of top-left point each object on the sleeper, normalized by the height and width of the sleeper,

2. Location of right-down point each object on the sleeper, normalized by the height and width of the sleeper,

Algorithm 4: Supervised learning-based method for crack detection

```
1 Input: Sleeper frame $\mathbf{S}$
2 Output: The candidate is a crack (1) or non-crack (0)
    $\mathbf{S}_e = imgEnh(\mathbf{S})$
    $\mathbf{S}_a = imgAdp(\mathbf{S}_e)$
    $\mathbf{S}_m = imgMor(\mathbf{S}_a)$
    $\mathbf{S}_c = imgCCA(\mathbf{S}_m)$
    Extract the features of $I_m$ as vector $V^t_{I_m} = (v_1, ...v_n)$.
    Apply supervised classifier on $V_t$.
    if $\mathbf{S}_c$ contains cracks
    Return 1
    else
    Return 0
```

3. Area of the objects,
4. NPDI of the objects,
5. Aspect-Ratio of the objects,
6. Length of the objects.

For detecting the cracks, two types of the classifiers namely a random forests and a SVM with the radial basis function kernel (SVM-RBF) are applied. After the crack detection process, a crack classification method is performed using the maximum thickness of cracks and the traffic of the railroad (loading weights per day) as the main determining parameters for both horizontal and vertical cracks. Due to a fixed distance between the camera and sleeper, and the fixed focal length of the lens, each pixel covers a fixed area on the sleeper surface. Based on this observation, the damaging class thresholds are set in terms of the number of pixels.

4.4 Experiments and Results

All experiments have been performed on the visual data sets provided by the rail mapper vehicle. Different data sets have been generated in several rounds however, few data sets have been found useful for processing, applying image processing operations, and training. The cameras have been installed in five locations of the railroad including outer right, inner right, center, inner left and, outer left of the vehicle to capture the

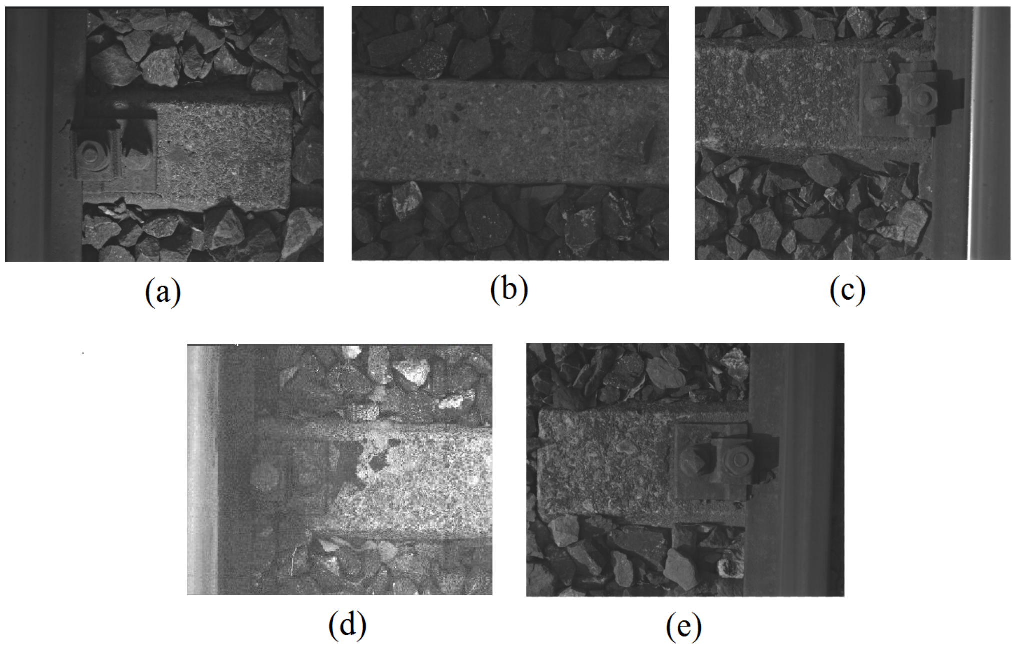

Figure 4.4: Sample of enhanced image frames captured from five different locations in order from outer right (a), center (b), inner right (c), inner left (d), and outer left (e). The original frames quality are too poor to be visible.

whole area of the sleeper as shown in Figure 4.4. The first approach for sleeper detection has been implemented on a sample set of 1500 images wherein 1000 images contain the full sleeper.

According to the experiment, the task of sleeper body detection in the central frames is not challenging, so that the first method is applied achieving a perfect accuracy. However, sleeper detection task in the other frames (i.e., right and left frames) is not as straightforward as in the central frames due to the existence of different objects (e.g., railway, fastener and so on.). Here, for visualizing the research results, the most challenging part (i.e., the sleeper bodies captured on the outer right side of the rail tracks) is investigated. As mentioned before, the sleeper body detection step can be implemented via two different approaches. The intermediate results of a successful detection example are shown in Figure 4.5. The achieved accuracy rate for sleeper detection via Algorithm 2 is not efficient and has not crossed 75%. To increase the detection rate and make the features more discriminative, an auxiliary morphological opening operation has been applied. To filter out the wrong frames, a candidate selection step was applied. To train the SVM classifier, 936 images from both classes including 520 sleeper container frames and 416 non-sleeper or partial sleeper container frames are

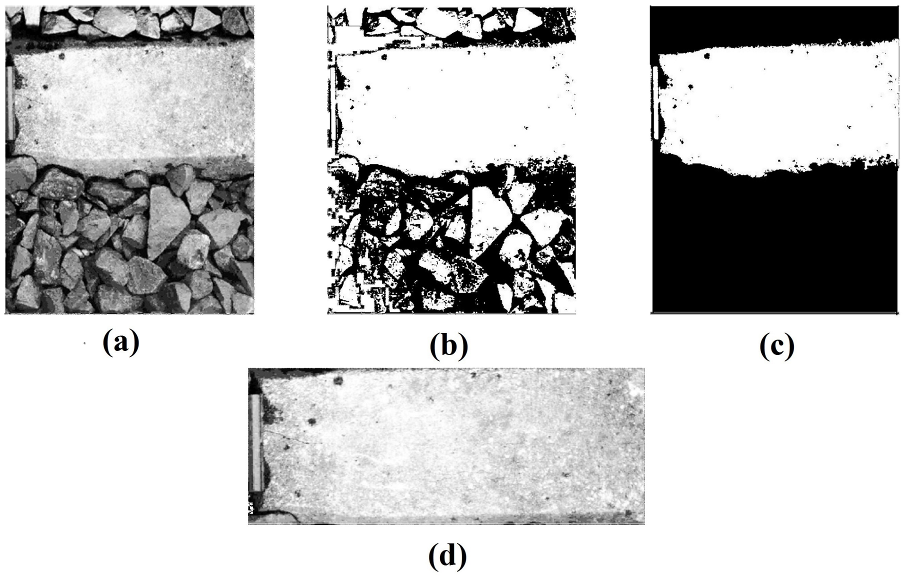

Figure 4.5: Intermediate and final results of a successful sleeper detection in the feature based approach are shown. (a) shows the enhanced image of input using Otsu thresholding method, (b) shows the binary image of (a), (c) presents the candidate with maximum area of (b) and (d) shows the detected sleeper.

used as a training sample whose aforementioned features have been extracted. A set of 564 frames including both classes were used to test the algorithm. The kernel function used in the classifier is RBF with a Gamma equal to 1.81. The lower values for Gamma decrease the accuracy whereas higher values do not change it. The second approach for the sleeper detection module is based on the template matching method. The selected template is the sleeper fastener which is common in all sleepers as shown in Figure 4.3. This method outperforms the rule-based method (without the candidate selection step) in terms of accuracy and detection rate however its computational cost is much higher. The summary result indicating the accuracy performance of the candidate selection step and first phase is shown in Table 4.1. In this scenario a true positive (T_P) occurs when a full sleeper exists in the image and the system detected it correctly and a false positive (F_P) happens when the system detects a non-sleeper object as the sleeper and a false negative (F_N) occurs when a sleeper exists in the image and the system cannot detect it. To compare the performances of sleeper detection methods, recall $R = T_P/(T_P + F_N)$ and precision $P = T_P/(T_P + F_P)$ are used.

Two classifiers SVM and random forest have been trained to detect the cracks on sleepers for the candidate selection procedure of the second step. Similar to the first step,

Method	Recall	Precision
SVM-RBF	0.968	0.978
Template matching	0.923	0.906

Table 4.1: A performance comparison of the sleeper detection step between the template matching and and the training-based methods is shown.

Training parameter	Value
Max number of trees in the forest	100
Min sample count	5
Max depth	25
Forest accuracy	0.01
Min sample count	15
Number of variables randomly selected to find the best split	4

Table 4.2: Training parameters for the random forest classifier are shown. Various parameter sets were examined and with this set the best result was obtained.

RBF kernel parameter (Gamma) is set to 0.1 found by a greedy search. Also the training parameters for the random forest classier are shown in Table 4.2.

To train the classifiers, a set of 1042 samples is used wherein nearly 50% of the set contains the crack holder images while the rest do not. More than 95% of the cracked sleepers came from wooden sleepers. The test set includes 649 images, 235 out of which are cracked. The performance result of the second step is shown in Table 4.3. The visual results of the second step of the detection mechanism including the intermediate results are shown in Figure 4.6.

To evaluate the whole pipeline of the crack detection mechanism, the accuracy analysis and detection rate of each step are taken into account and linked to each other. A false positive happens when a wrong frame (non-sleeper body in the first step or a healthy sleeper in the second step) is detected and identified. A false negative happens when a right frame (a sleeper-body containing frame in the first step or a crack holder sleeper in

Classifier type	Recall	Precision	F-measure
SVM-RBF	93%	97%	94.9%
Random Forest	94%	97%	95.5%

Table 4.3: Performance results of the crack detection step are shown. Two classifiers namely random forest and SVM were used. Their results are almost identical.

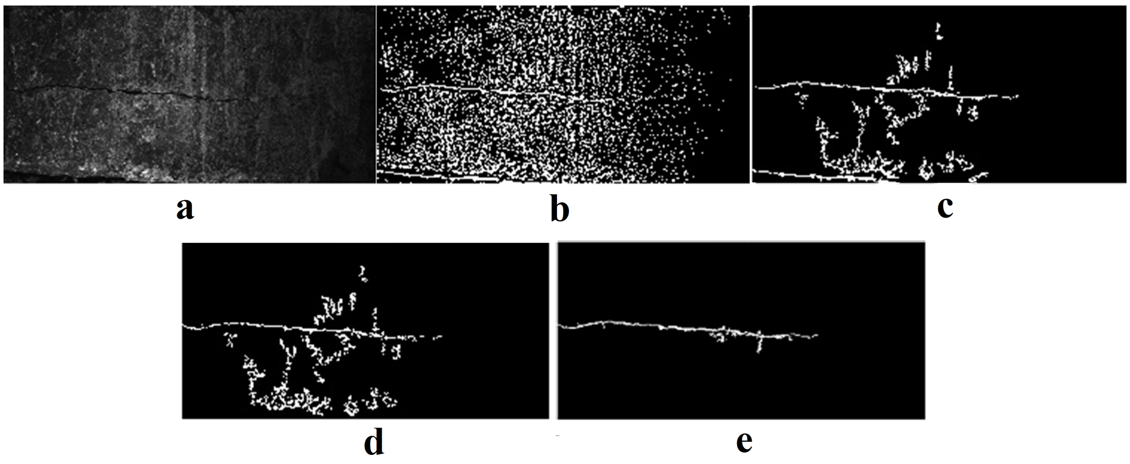

Figure 4.6: Intermediate and final results of a successful crack detection (second step) are shown. (a) shows a sleeper image, (b) shows binary image using adaptive thresholding, (c) shows the result of applying the area conditions (removing small candidates), (d) presents removing border candidates and (e) shows a found crack using machine learning.

Step 1	Step 2	Recall	Precision	Accuracy
Algorithm 2-SVM-RBF	Algorithm 4-RF	0.726	0.735	74.1%
Template matching method	Algorithm 4-RF	0.923	0.885	87.4%

Table 4.4: Performance comparison of the whole system pipeline is shown. Step 1 shows sleeper detection using the rule-based method and the template matching method. Step 2 presents crack detection using random forest.

the second step) is not detected. The latter event is a dangerous event depending on the damage level and generally must be avoided. The parameters are set in such an order to minimize the false negative event. The modest performance (lower bounds) of the whole system is shown in Table 4.4 wherein RF stands for the random forest classifier. Since RF shows slightly better results than SVM, it is chosen for this experiment.

It must be noted that occurring the cracks in the sleepers is not expected in the majority of the sleepers and in fact, it is a rare event. Also, according to the second approach of the sleeper detection step, a sleeper body can be ignored if the fastener is covered or broken. Similarly, this is also a rare event.

The proposed algorithm has been implemented on an Intel Core i7-4600 CPU @ 2.10 GHz, using Matlab code. The hardware setup has been adjusted to achieve high quality image sequences with the lowest possible overlap between the frames suitable for machine learning tasks. The non-optimized software implementation on a standard PC machine using a single core CPU takes 37 (ms), 87 (ms) and 24 (ms) for sleeper detection (template matching method), crack detection, and crack classification respectively in average per image. The software can be significantly improved towards an optimal implementation on the same platform.

4.5 Conclusions and Future Trends

In this chapter, a vision-based method for detecting the cracks in concrete sleepers of the railway tracks has been presented. The presented method has been given via different approaches in two successive steps corresponding to sleeper detection and crack detection respectively. For the first step, two techniques namely template-based and rule-based sleeper detection methods have been used. The template matching technique uses a railway fastener as the template image.

The performance of the method has been analyzed and a comparison result between the approaches has been shown. The template-based sleeper detector and RF-based crack detector show better results than the other examined methods. To the best of the author's knowledge, this work is the only work which scientifically tackles the problem of crack detection in the concrete sleepers. The use of the fastener template and candidate generation and the type of the features in the crack detection part are the other contributions of this work.

Expanding the method toward a more accurate, improved, scalable and interoperable system and including a wider range for crack type classification will be considered as future work. This track can be followed by adapting the classifiers or using new classifiers and sophisticated features for a better performance as well. The improvement of the crack detection generality using deep neural networks will be considered in particular.

Chapter 5

Conclusion and Future Work

In this dissertation, scientific algorithms for object tracking and an automatic system for railway sleepers inspection were proposed. The details of the proposed methods, as well as their results were given in the previous chapters. This chapter concludes the whole thesis and collects the most interesting solutions and findings based on the author's contributions obtained during the development of the projects reflected in the last chapters. In addition, an overview of the possible future works for both methodological and application based parts are given at the end of this chapter.

This chapter starts with Section 5.1 which presents a summary of the algorithms presented in this thesis. The author's proposed methods for rectangular and polar object tracking, as well as vision-based sleeper monitoring are summarized in Section 5.1. Then, Section 5.2 concludes this thesis. The points obtained from the results of the previous chapters are given in this section. Finally, the book closes with future visions towards improving or generalizing the proposed methods, in Section 5.3.

5.1 Thesis Summary

The dissertation is summarized as follows:

Object Tracking in Rectangular Videos: A comparative study among five famous training based detectors (i.e., Aggregated Channel Feature (ACF), RCNN (Region-based Convolutional Neural Network), FastRCNN, FasterRCNN and You Only Look Once (YOLO)) in the object tracking context were undertaken. For the comparison, two methods of online and offline trackers were proposed. The online tracker generated synthetic data using the object of interest in the first frame and segmented the video frames into certain parts. Then, it employed synthetic data to train the object detectors. The online tracker used the trained detectors to detect the object of interest in each frame of the first group in video frames. The tracker used the detected objects as well as synthetic data to train the detectors in the second iteration. The online tracker updated/trained the detectors at the end of each video segments and this procedure continued to the end of the video. The experiments showed that ACF tracker is the fastest and the most stable tracker among the online trackers. The proposed online tracker and the comparative study are two contributions to this part. In the second part of rectangular object tracking, an offline tracker was proposed. For a given frame, the YOLO detector first generated some candidates in the scene. The offline tracker then used a two dimension Kalman filter to select the object of interest among the object candidates. This method uses tracking by detection algorithm. The experimental results showed the YOLO tracker outperforms all other investigated online and offline rectangular trackers. The combination type of the deep-learning based object detector (i.e., YOLO version2) with a 2D Kalman filter is the contribution of offline tracker.

Polar Object Tracking Using Video Unwrapping: The rectangular trackers on the polar videos have very weak performance and a process before the rectangular tracking methods is necessary to increase the tracking precision. The proposed tracker in this part first rectified the polar video and then applied a modified version of a famous tracker i.e., Training-Learning-Detection (TLD). The modified training-learning-detection tracker (MTLD) was proposed to increase the tracker robustness and speed compared to the TLD tracker, especially robustness against out of plane rotation. It composed of modules which were added to TLD such as a rectangular region of interest (ROI) definition, the nearest neighbor (NN) threshold modification in TLD, proposing a distance classifier, the integrator modification, changing the input source of the trainer in the original TLD and using the FIFO strategy for fulling the training queue. The results confirmed the accuracy and speed improvement of MTLD compared to the TLD tracker. The mentioned modifications (i.e., ROI, NN and integrator modification, distance classifier and FIFO defecation and changing the trainer input) are contributions of this part.

Polar Object Tracking Directly on Polar Videos: Although, the MTLD was much better than TLD from speed and precision perspectives, it is still slow. To increase the tracking speed, three polar object tracking methods were proposed to directly follow the object of interest in polar videos and bypass the slow rectification (unwrapping) process.

Both trackers selected a region of interest around the object of interest using a proposed polar region of interest (PROI) selection. The first tracker extracted interesting points using SURF descriptor from the enhanced version of the last object of interest and scene. Then, it fed the SURF features as well as the location and intensity of the interesting points in the object and in the scene to a Random Forest classifier. The classifier found matched points in the object and the scene and drew a polar object region around the points to obtain a new object. The tracker used an SVM classifier to detect challenges type (i.e., out of plane rotation, occlusion and etc.) and used a correspondent algorithm to handle the challenge. The feature vector used in this part, the training based matching classifier and the training based challenge detector classifier were three contributions of this part. The second tracker generated trapezoid-shape candidates in the polar region of interest (PROI) and selected the object of interest using two proposed color-based classifiers. The first classifier i.e., color max classifier segmented each candidate into 9 polar segments. Each segment is then converted to HSV space and 6 colors (channels) of green, white, red, brown + purple, blue + green and yellow. Furthermore, the number of each resulted channels is counted. The color used by the majority number of pixels is considered for that segment. The second color classifier (color pixel classifier) found the desired object based on the area for the specific color. The third tracker generated the polar candidates (exactly polar shapes) in the polar region of interest (PROI) and combined binary features with RGB color information of pixels. The combination incenses the correctness of the correct candidate finding among all the candidates. This method outperforms the other three polar trackers (i.e., MTLD, SURF tracker and the tracker with trapezoid-shape candidates) from both accuracy and speed viewpoints as the results showed. The polar selection of area of interest, polar candidate selection, the using type of color information i.e., the color pixel classifier, the color max classifier as well as the color binary classifier are the contributions of the proposed color-based trackers. The mentioned contributions were effective to increase the precision and speed of the polar trackers.

Railway Sleeper Inspection: A system was proposed to automatically capture valuable frames from the top view of railway sleepers and a machine learning based system to detect the possible cracks and find their locations and to classify the cracks according to a standard of predefined categories of the crack damaging level. The proposed algorithm first enhanced the input video frames using classical histogram equalization; then, it finds the sleeper location by finding the fastener location using an object matching technique. It then converted the resulting sleepers to binary images using an adaptive thresholding method to generate some crack candidates. It extracted some features from the binary image and used a supervised classifier like Random Forest or SVM for the detection of the cracks among the crack candidates. After crack detection, it classified the cracks according to their thickness. It is an application of image processing which provides a very precise system for the concrete sleeper inspection for railroads.

5.2 Thesis Conclusion

This section reviews the contribution of the proposed methods and concludes the thesis. The contributions are listed in the following three parts namely rectangular object tracking, polar object tracking and railway quality control.

1. A comprehensive comparative study in the context of object tracking among 5 famous recently proposed detectors was executed. Two trackers based on online and offline tracking were used. A comparison of the five train-based detectors namely ACF, RCNN, FastRCNN, FasterRCNN and YOLO was proposed for the first time in this thesis to find out the best detector for tracking. The ACF tracker has the best results among the online trackers from both accuracy and speed viewpoints. ACF has effective implementation, because it runs on the CPU machine instead of GPU for the other online trackers. Among the RCNN based trackers (i.e., RCNN, FastRCNN, and FasterRCNN), RCNN has the best tracking accuracy. FastRCNN and FasterRCNN are very fast in the test phase, their tracking process is slow because they are very slow in training phase. Since the YOLO tracker was trained offline, it is the fastest tracker. YOLO is not qualified for online tracking, due to its very slow training phase. For human tracking from the front and side views, the combination of YOLO and Kalman filter shows the best results. For tracking of unknown objects, the ACF tracker is better than the YOLO tracker. Compared to YOLO and ACF, the RCNN based trackers show less accuracy because they do not have a very deep structure (i.e., 3 convolutional layers and 2 fully connected). By using a deeper CNN like YOLO, the accuracy is increased. The training vector length was investigated and showed that the online tracker follows the recent appearances of the object of interest. The length should be set to an optimal value; if it exceeds the value, the average accuracy decreases. On the other hand, selection of the shorter lengths leads to the under-fitting and low accuracy.

   The results of this research are not limited to tracking. They have interesting points for object detection. For instance, from top view YOLO detects cars, but it often cannot identify them. Although YOLO detects the humans well from the top view, its classification results are disappointing. In the training phase, YOLO was biased to the human data from the front view. Although object detection based on deep learning has recently improved, further improvement is still necessary. For instance, YOLO detector should be trained using a big data set including more various views of the objects. ACF detector with a non-deep-learning structure shows better results than some of the deep-learning-based detectors namely RCNN, FastRCNN, and FasterRCNN. Extensive experiments showed that the YOLO tracker outperforms the rest of the trackers.

2. To solve the problem of object tracking in 360-degree videos a couple of methods with different contributions were proposed. An efficient polar tracker was proposed to follow unknown objects in 360-degree videos. A state-of-the-art tracker i.e., Tracking-Learning-Detection (TLD) was promoted to overcome the tracking problems in the challenging conditions of 360-degree videos. The resolution of these videos is much higher than the TLD data set, thus the searching area was restricted to decrease the computational load in the TLD method. The experimental results showed that the proposed MTLD method outperforms TLD. MTLD tracks objects even when they have an out-of-plane rotation and varying environment in which cases TLD losses the object of interest in the cases. A polar tracking scheme (POT) was proposed to follow polar objects in 360-degree videos and to bypass the slow rectification process in MTLD. A polar area was proposed to limit the searching region and to decrease the computational load. The removing of the rectification process and the color-based classifiers are two contributions of this tracker. A training-based matching method was proposed for SURF-based object tracking. This method proposed a train-based classifier for object matching and another train-based mechanism for automatic detection of challenges namely out-plane-rotation, occlusion and departing the camera. The tracker then adapted itself to handle these situations. The proposed training-based object matcher and challenging classifier find the correspondent interesting points in the scene and object of interest which could not be found using traditional distance functions. The other polar tracker first segmented the input frame into a polar region around the labeled object and overlapped polar candidates were selected within the region. The polar candidates were then classified using two hierarchical variance and color-binary classifiers. The discriminatory of binary features was improved by adding color information. Experiments on the 360-degree and catadioptric omnidirectional data sets showed that the proposed color-binary tracker boosts the

tracking performance for diverse objects in various and challenging real-world scenarios and outperforms similar trackers i.e., TLD, MTLD, POT, and SURF-tracker. This method has comparable tracking performance to MTLD but it is much faster than MTLD. It is also faster and more precise than POT because it uses binary features.

3. An automatic visual system for railway monitoring was proposed. To the best of the author's knowledge, this work is the only scientific work which tackles the problem of crack detection in the concrete sleepers. The use of the fastener template and the type of the features in the crack detection part are the other contributions of this work.

5.3 Future Trends

For each proposed method a couple of improvements are planned as future work. For the proposed rectangular trackers, YOLO detector will be trained using an updated data set to improve the detection results from the top view. The experiments will be extended to other videos in the VOT benchmark. The proposed trackers will be extended for multiple object tracking because all the detectors i.e., ACF, RCNN, FastRCNN, FasterRCNN, and YOLO can detect multiple objects. For the online trackers in each iteration of the training phase, instead of a single object, multiple objects will be defined and the detectors will output different labels for different objects. In the case of offline tracking, YOLO is able to detect multiple objects for each object.

For the proposed trackers in polar videos the following is planned for future work. For the MTLD tracker, the future work will address tracking in complex backgrounds. Because like TLD, MTLD is also fragile to background clutter when the object of interest is similar to a part of the background. The color binary tracker will be extended to multiple object tracking. Parallel processing will be employed to reduce the computational cost. In addition, the use of HSV color space in binary features will be considered. For rigid objects, the tracker is often able to retrieve the object of interest in the next frames; however, this is not the case for articular objects. The future work will also focus on articular object tracking by dividing the articular object into smaller parts (like body members in human) and applying the tracker to each part. Real-time processing can be achieved by optimizing the implementation of this approach. In fixed-camera scenarios, the use of background detection and subtraction between the RoI selection and object detection parts is suggested. This process improves the speed of the tracker and reduces the probable similar objects tracking.

For the proposed railway monitoring method, expanding towards a more accurate, improved, scalable and interoperable system and with a wider range for crack type classification will be considered as future work. This track can be followed by adapting the classifiers or engaging new classifiers and sophisticated features for a better performance as well.

List of Figures

List of Tables

Bibliography

[AG17] B. Akok and F. Gurkan. "Robust Object Tracking by Interleaving Variable Rate Color Particle Filtering and Deep Learning". In: *IEEE International Conference on Image Processing* (2017), pp. 3665–3669.

[Bab09] P. Babenko. *VISUAL INSPECTION OF RAILROAD TRACKS*. [online] Available: http://crcv.ucf.edu/papers/theses/Babenko$_{pavel.pdf}$, 2009.

[Bay+08] H. Bay, A. Ess, T. Tuytelaars, and L. Gool. "Speeded up robust features (Surf)". In: *Journal of Computer Vision and Image Understanding* 110.3 (2008), pp. 346–359.

[Ber+16] L. Bertinetto, J. Valmadre, S. Golodetz, O.Miksik, and P.H.S. Torr. "Staple: Complementary learners for real-time tracking". In: *IEEE International Conference on Computer Vision and Pattern Recognition* (2016), pp. 1401–1409.

[Bol+10] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. "Visual object tracking using adaptive correlation filters". In: *IEEE International Conference on Computer Vision and Pattern Recognition* (2010), pp. 2544–2550.

[Bou00] J. Y. Bouguet. "Pyramidal implementation of the Lucas Kanade feature tracker". In: *Intel Corporation, Microprocessor Research Labs* (2000).

[BW01] G. Bishop and G. Welch. "An introduction to the Kalman filter". In: *Proc of SIGGRAPH, Course* 8.27599-3175 (2001), p. 59.

[BY18] S. H. Bae and K. J. Yoon. "Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018), pp. 2239–2252.

[Cal+10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. "Brief: Binary robust independent elementary features". In: *European Conference on Computer Vision* (2010), pp. 778–792.

[Che+08] C. H. Chen, Y. Yao, D. Page, B. Abidi, A. Koschan, and M. Abidi. "Heterogeneous Fusion of Omnidirectional and PTZ Cameras for Multiple Object Tracking". In: *IEEE Transactions on Circuits and Systems for Video Technology* 18.8 (2008), pp. 1052–1063.

[Cho+15] G. Choe, T. Wang, F. Liu, G. Li, H. O, and S. Kim. "Moving object tracking based on geogram". In: *Multimedia Tools and Applications* 74.21 (2015), pp. 9971–9794.

[CT18] K. Chen and W. Tao. “Learning Linear Regression via Single Convolutional Layer for Visual Object Tracking”. In: *IEEE Transactions on Multimedia* (2018), pp. 1–13.

[Cui+13] J. Cui, Y. Liu, Y. Xu, H. Zhao, and H. Zha. “Tracking Generic Human Motion via Fusion of Low- and High-Dimensional Approaches”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 43.4 (2013), pp. 996–1002.

[Del+16a] A. Delforouzi, S. A. H. Tabatabaei, K. Shirahama, and M. Grzegorzek. “Polar Object Tracking in 360-Degree Camera Images”. In: *International Symposium on Multimedia* (2016), pp. 347–352.

[Del+16b] A. Delforouzi, S. A. H. Tabatabaei, K. Shirahama, and M. Grzegorzek. “Unknown Object Tracking in 360-Degree Camera Images”. In: *23rd International Conference on Pattern Recognition* (2016), pp. 1799–1804.

[Del+17] A. Delforouzi, S. A. H. Tabatabaei, M. H. Khan, and M. Grzegorzek. “A Vision-Based Method for Automatic Crack Detection in Railway Sleepers”. In: *10th International Conference on Computer Recognition Systems CORES* (2017), pp. 130–139.

[Del+18] A. Delforouzi, S. A. H. Tabatabaei, K. Shirihama, and M. Grzegorzek. “A Polar Model for Fast Object Tracking in 360-degree Camera Images”. In: *Multimedia Tools and Applications* (2018).

[Dem+12] B. E. Demiroz, I. Arı, O. Eroglu, A. A. Salah, and L. Akarun. “Feture-Based Tracking on a Multi-Omnidirectional Camera Dataset”. In: *International Symposium on Communications, Control and Signal Processing* (2012), pp. 1–5.

[DG15] A. Delforouzi and M. Grzegorzek. “An Object Tracking System Based on SIFT and SURF Feature Extraction Methods”. In: *18th International Conference on Network-Based Information Systems* (2015), pp. 561–565.

[DH16] J. Ding and Y. Huang. “Severely Blurred Object Tracking by Learning Deep Image Representations”. In: *IEEE Transactions on Circuits and Systems for Video Technology* (2016), pp. 319–331.

[Din+18] G. Ding, W. Chen, S. Zhao, J. Han, and Q. Liu. “Real-Time Scalable Visual Tracking via Quadrangle Kernelized Correlation Filters”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.1 (2018), pp. 140–150.

[Dol+14] P. Dollar, R. Appel, S. Belongie, and P. Perona. “Fast Feature Pyramids for Object Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.8 (2014), pp. 1532–1545.

[DPG18] "A. Delforouzi, B. Pamarthi, and M. Grzegorzek". “Training-based Methods for Comparison of Object Detection Methods for Visual Object Tracking”. In: *Sensors* (2018).

[Eve+15] M. Everingham, S. M. Eslami, L. V. Gool, C. K. Williams, J. Winn, and A. Zisserman. “The pascal visual object classes challenge: A retrospective”. In: *International journal of computer vision* 111.1 (2015), pp. 98–136.

[FMH06] Y. Fujita, Y. Mitani, and Y. Hamamoto. “A Method for Crack Detection on a Concrete Structure”. In: *18th International Conference on Pattern Recognition* 3 (2006), pp. 901–904.

[Gir+14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 580–587.

[Gir15] R. Girshick. “Fast R-CNN”. In: *IEEE International Conference on Computer Vision* (2015), pp. 1440–1448.

[GPC15] X. Gibert, V.l M. Patel, and R. Chellappa. “Sequential Score Adaptation with Extreme Value Theory for Robust Railway Track Inspection”. In: *CoRR* abs/1510.05822 (2015).

[Gup+13] A. M. Gupta, B. S. Garg, C. S. Kumar, and D. L. Behera. “An on-line visual human tracking algorithm using SURF-based dynamic object model”. In: *IEEE International Conference on Image Processing* (2013), pp. 3875–3879.

[GW18] M. Guan and C. Wen. “Real-Time Event-Triggered Object Tracking in the Presence of Model Drift and Occlusion”. In: *IEEE Transactions on Industrial Electronics* (2018), pp. 1–11.

[He+16] Z. He, X. Li, X. You, D. Tao, and Y. Y. Tang. “Connected Component Model for Multi-Object Tracking”. In: *IEEE Transactions on image processing* 25.8 (2016), pp. 3698–3711.

[Hen+15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. “High-Speed Tracking with Kernelized Correlation Filters”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.3 (2015), pp. 583–596.

[HS03] S. Hrabar and G. S. Sukhatme. “Omnidirectional vision for an autonomous helicopter”. In: *IEEE International Conference on Robotics and Automation* 1 (2003), pp. 558–563.

[JWKN15] J. C. Park J. W. Kim S. B. Kim and J. W. Nam. “Development of Crack Detection System with Unmanned Aerial Vehicles and Digital Image Processing”. In: *Advances in Structural Engineering and Mechanics* (2015), pp. 25–29.

[Kar+15] S. Karthikeyan, T. Ngo, M. Eckstein, and B.S. Manjunath. “Eye tracking assisted extraction of attentionally important objects from videos”. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3241–3250.

[KM11] O. E. Kadmiri and L. Masmoudi. “An omnidirectional image unwrapping approach”. In: *Multimedia Computing and Systems (ICMCS), 2011 International Conference on*. IEEE. 2011, pp. 1–4.

[KMM10] Z. Kalal, J. Matas, and K. Mikolajczyk. “P-N Learning: Boot-strapping Binary Classifiers by Structural Constrains”. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2010), pp. 49–56.

[KMM12] Z. Kalal, K. Mikolajczyk, and J. Matas. “Tracking-Learning-Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7 (2012), pp. 1409–1422.

[KP18] H. I. Kim and R. H. Park. “Residual LSTM Attention Network for Object Tracking”. In: *IEEE Signal Processing Letters* 25 (2018), pp. 1029–1033.

[KS07] J. Kim and Y. Suga. “An omnidirectional vision-based moving obstacle detection in mobile robot”. In: *International Journal of Control, Automation, and Systems* 5.6 (2007), pp. 663–673.

[Li+11] J. Li, J. Zhang, Z. Zhou, W. Guo, B. Wang, and Q. Zhao. “Object tracking using improved Camshift with SURF method”. In: *IEEE International Workshop on Open-source Software for Scientific Computation* (2011), pp. 136–141.

[Li+18] F. Li, K. Shirahama, M. A. Nisar, L. Koeping, and M. Grzegorzek. “Comparison of Feature Learning Methods for Human Activity Recognition Using Wearable Sensors”. In: *Sensors* 18 (2018).

[Lin+07] Z. Lin, L. S. Davis, D. Doermann, and D. DeMenthon. “Hierarchical Part-Template Matching for Human Detection and Segmentation”. In: *IEEE 11th International Conference on Computer Vision* (2007), pp. 1–8.

[Liu+12] Y. Liu, J. Cui, H. Zhao, and H. Zha. “Fusion of Low-and High-Dimensional Approaches by Trackers Sampling for Generic Human Motion Tracking”. In: *21st International Conference on Pattern Recognition* (2012).

[LL18] G. Liu and S. Liu. “Object Tracking in Vary Lighting Conditions for Fog Based Intelligent Surveillance of Public Spaces”. In: *IEEE Access* 6 (2018), pp. 29283–29296.

[LSS05] B. Leibe, E. Seemann, and B. Schiele. “Pedestrian Detection in Crowded Scenes”. In: IEEE Computer Society, 2005, pp. 878–885.

[Ma+16] B. Ma, H. Hu, J. Shen, Y. Liu, and L. Shao. “Generalized Pooling for Robust Object Tracking”. In: *IEEE Transactions on image processing* 25.9 (2016), pp. 4199–4208.

[Man+04] C. Mandriota, M. Nitti, N. Ancona, E. Stella, and A. Distante. “Filter-based feature selection for rail defect detection”. In: *Machine Vision and Applications* 15.4 (2004), pp. 179–185.

[May82] P. S. Maybeck. *Stochastic models, estimation, and control*. Vol. 3. Academic press, 1982.

[Maz+05] P. L. Mazzeo, M. Nitti, E. Stella, and A. Distante. “Visual Recognition of fastening Bolt in Railway Maintenance Context by using Wavelet Transform”. In: *International Journal on Graphics, Vision and Image Processing* SI1 (2005), pp. 25–32.

[Maz+16] A. Mazzu, P. Morerio, L. Marcenaro, and C. S. Regazzoni. “A Cognitive Control-Inspired Approach to Object Tracking”. In: *IEEE Transactions on image processing* 25.6 (2016), pp. 2697–2711.

[MCP14] I. Markovic, F. Chaumette, and I. Petrovic. “Moving object detection, tracking and following using an omnidirectional camera on a mobile robot”. In: *IEEE International Conference on Robotics and Automation* (2014), pp. 5630–5635.

[Mia+11] Q. Miao, G. Wang, C. Shi, X. Lin, and Z. Ruan. “A new framework for on-line object tracking based on SURF”. In: *Pattern Recognition Letters* 32.13 (2011), pp. 1564–1571.

[Moh08] S. P. Mohammad. “Machine Vision for Automating Visual Inspection of Wooden Railway Sleepers”. In: *Department of Computer Science, Dalarna University* (2008).

[MRS14] A. Milan, S. Roth, and K. Schindler. “Continuous Energy Minimization for Multitarget Tracking”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.1 (2014), pp. 58–72. ISSN: 0162-8828. DOI: `10.1109/TPAMI.2013.103`.

[MSR15] A. Milan, K. Schindler, and S. Roth. “Multi-Target Tracking by Discrete-Continuous Energy Minimization”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.10 (2015), pp. 2054–2068.

[NH16] H. Nam and B. Han. “Learning Multi-domain Convolutional Neural Networks for Visual Tracking”. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4293–4302.

[OC13] H. Oliveira and P. L. Correia. “Automatic Road Crack Detection and Characterization”. In: *IEEE Transactions on Intelligent Transportation Systems* 14.1 (2013), pp. 155–168.

[Oh+12] C.M. Oh, Y.C. Lee, D.Y. Kim, and C.W. Lee. “Moving object detection in omnidirectional vision-based mobile robot”. In: *IEEE Annual Conference of Industrial Electronics Society* (2012), pp. 4232–4235.

[ON15] K. OShea and R. Nash. “An Introduction to Convolutional Neural Networks”. In: *CoRR* (2015), pp. 1–11.

[OP16] P. Ondruska and I. Posner. “Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks”. In: *Proceedings of the Thirtieth Conference on Artificial Intelligence* (2016), pp. 3361–3367.

[oTr06] U.S. Department of Transportation. “Research Results: Video System for Joint Bar Inspection”. In: *Federal Railroad Administration* RR06-03 (2006).

[Ots79] N. Otsu. “A Threshold Selection Method from Gray-Level Histograms”. In: *IEEE Transactions on Systems, Man and Cybernetics* 9.1 (1979), pp. 62–66.

[Pra+16] P. Prasanna, K. J. Dana, N. Gucunski, B. B. Basily, H. M. La, R. S. Lim, and H. Parvardeh. “Automated Crack Detection on Concrete Bridges”. In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 591–599.

[Ram+07] F. Rameau, D. Sidibe, C. Demonceaux, and D. Fofi. “Tracking moving objects with a catadioptric sensor using particle filter”. In: *IEEE 11th International Conference onComputer Vision Workshops* (2007), pp. 328–334.

[RF17] J. Redmon and A. Farhadi. “YOLO9000: Better, Faster, Stronger”. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 6517–6525.

[Roj10] R. Rojas. “Lucas-kanade in a nutshell”. In: *Freie Universit at Berlinn, Dept. of Computer Science, Tech. Rep* (2010).

[Rus+15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision* 115.3 (2015), pp. 211–252.

[Sak+15] Y. Sakai, T. Oda, M. Ikeda, and L. Barolli. “An Object Tracking System Based on SIFT and SURF Feature Extraction Methods”. In: *18th International Conference on Network-Based Information Systems* (2015), pp. 561–565.

[Sco+04] G. Scotti, L. Marcenaro, C. Coelho, F. Selvaggi, and C. S. Regazzoni. “A novel dual camera intelligent sensor for high definition 360 degrees surveillance”. In: *IEE Intelligent Distributed Surveilliance Systems* (2004), pp. 26–30.

[SF06] S. K. Sinha and P. W. Fieguth. “Segmentation of buried concrete pipe images”. In: *Automation in Construction* 15.1 (2006), pp. 47 –57.

[Sha+17] S. Shaoqing, K. He, R. Girshick, and J. Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2017), pp. 1137–1149.

[SNH12] H. Shuoa, W. Nab, and S. Huajunc. “Object Tracking Method Based on SURF”. In: *Applied Mechanics and Materials* (2012), pp. 351–356.

[SS08] D. Scaramuzza and R. Siegwart. “Appearance-Guided Monocular Omnidirectional Visual Odometry for Outdoor Ground Vehicles”. In: *IEEE Transactions on Robotics* 24.5 (2008), pp. 1015–1026.

[Ste+02] E. Stella, P. Mazzeo, M. Nitti, G. Cicirelli, A. Distante, and T. D’Orazio. “Visual recognition of missing fastening elements for railroad maintenance”. In: *IEEE 5th International Conference on Intelligent Transportation Systems* (2002), pp. 94–99.

[SZ14] K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2014), arXiv preprint arXiv:1409.1556.

[Tab+18] "S. A. H. Tabatabaei, A. Delforouzi, M. H. Khan, T. Wesener, and M. Grzegorzek". “Automatic Detection of the Cracks on the Concrete Railway Sleepers”. In: *International Journal of Pattern Recognition and Artificial Intelligence* (2018).

[Uij+13] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. “Selective search for object recognition.” In: *International Journal of Computer Vision* (2013).

[Wan+18] C. Wang, L. Zhang, L. Xie, and J. Yuan. "Kernel Cross-Correlator". In: *arXiv:1709.05936* (2018).

[Wen+16] L. Wen, Z. Lei, S. Lyu, S. Z. Li, and M. H. Yang. "Exploiting Hierarchical Dense Structures on Hypergraphs for Multi-Object Tracking". In: *IEEE Transactions on pattern analysis and machine intelligence* 38.10 (2016), pp. 1983–1996.

[WK16] G. S. Walia and R. Kapoor. "High-Speed Tracking with Kernelized Correlation Filters". In: *Multimedia Tools and Applications* 75.23 (2016), pp. 583–596.

[WN07] B. Wu and R. Nevatia. "Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet based Part Detectors". In: *International Journal of Computer Vision* 75.2 (2007), pp. 247–266.

[WY10] L. Wang and N. H. C. Yung. "Extraction of Moving Objects From Their Background Based on Multiple Adaptive Thresholds and Boundary Evaluation". In: *IEEE Transactions on Intelligent Transportation Systems* 11.1 (2010), pp. 40–51.

[WY13] N. Wang and D. Y. Yeung. "Learning a Deep Compact Image Representation for Visual Tracking". In: *Advances in Neural Information Processing Systems* (2013), pp. 809–817.

[WYX14] L. Wang, N. H. C. Yung, and L. Xu. "Multiple-Human Tracking by Iterative Data Association and Detection Update". In: *IEEE Transactions on Intelligent Transportation Systems* 15.5 (2014), pp. 1886–1899.

[WZM12] D. Wang, Q. Zhang, and J. Morris. "Distributed Markov Chain Monte Carlo kernel based particle filtering for object tracking". In: *Multimedia Tools and Applications* 56.2 (2012), pp. 303–314.

[X.W+16] X.Wang, E. Tueretken, Franc, O. Fleuret, and P. Fua. "Tracking Interacting Objects Using Intertwined Flows". In: *IEEE Transactions on pattern analysis and machine intelligence* 38.11 (2016), pp. 2312–2326.

[Xio+12] Z. H. Xiong, I. Cheng, W. Chen, A. Basu, and M. J. Zhang. "Depth space partitioning for omni-stereo object tracking". In: *IET Computer Vision* 6.2 (2012), pp. 153–163. ISSN: 1751-9632.

[XSL15] J. Xiao, R. Stolkin, and A. Leonardis. "Single target tracking using adaptive clustered decision trees and dynamic multi-level appearance models". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 4978–4987.

[YC18] S. Yun and J. Choi. "Action-Driven Visual Object Tracking With Deep Reinforcement Learning". In: *IEEE Transactions on Neural Networks and Learning Systems* 29 (2018), pp. 2239–2252.

[YL18] R. Yao and G. Lin. "Semantics-Aware Visual Object Tracking". In: *IEEE Transactions on Circuits and Systems for Video Technology* (2018), pp. 1–14.

[YNH08] T. Yamaguchi, S. Nakamura, and S. Hashimoto. "An efficient crack detection method using percolation-based image processing". In: *3rd IEEE Conference on Industrial Electronics and Applications* (2008), pp. 1875–1880.

[ZH06] Z. Zivkovic and F. v. d. Heijden. "Efficient adaptive density estimation per image pixel for the task of background subtraction". In: *Pattern Recognition Letters* 27.7 (2006), pp. 773–780.

[ZH13] D. Zhou and D. Hu. "A robust object tracking algorithm based on SURF". In: *International Conference on Wireless Communications and Signal Processing* (2013), pp. 1–5.

[Zha+15] Y. Zhang, X. Chen, J. Li, C. Wang, and C. Xia. "Semantic Object Segmentation via Detection in Weakly Labeled Video". In: 2015, pp. 3641–3649.

[Zha+17] B. Zhang, Z. Li, X. Cao, Q. Ye, C. Chen, L. Shen, A. Perina, and R. Ji. "Output Constraint Transfer for Kernelized Correlation Filter in Tracking". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47.4 (2017), pp. 693–703.

[ZLS17] A. Zirakchi, C. L. Lundberg, and H. E. Sevil. "Omni Directional Moving Object Detection and Tracking With Virtual Reality Feedback". In: *Conference on Dynamic Systems and Control* (2017).

[Zor17a] M. Zorzi. "Convergence analysis of a family of robust Kalman filters based on the contraction principle". In: *SIAM Journal on Optimization Control* 55 (2017), pp. 3116–3131.

[Zor17b] M. Zorzi. "Robust Kalman Filtering Under Model Perturbations". In: *IEEE Transactions on Automatic Control* 62 (2017), pp. 2902–2907.

[ZS18] F. Zheng and L. Shao. "A Winner-Take-All Strategy for Improved Object Tracking". In: *IEEE Transactions on Image Processing* 27 (2018), pp. 4302–4313.