Robust Real-Time Tracking in Smart Camera Networks

Robuust volgen in ware tijd in slimme cameranetwerken

Vedran Jelača

Promotor: prof. dr. ir. W. Philips
Proefschrift ingediend tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen

Vakgroep Telecommunicatie en Informatieverwerking
Voorzitter: prof. dr. ir. H. Bruneel
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2014 - 2015

ISBN 978-90-8578-761-7
NUR 965
Wettelijk depot: D/2015/10.500/5

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur
Vakgroep Telecommunicatie en Informatieverwerking

Promotor: prof. dr. ir. Wilfried Philips

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur

Vakgroep Telecommunicatie en Informatieverwerking
Sint-Pietersnieuwstraat 41, B-9000 Gent, België

Tel.: +32-9-264.34.12
Fax.: +32-9-264.42.95

Voorzitter: prof. dr. ir. Herwig Bruneel

Proefschrift ingediend tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen
Academiejaar 2014 - 2015

Members of the jury

Prof. dr. ir. Wilfried Philips (Ghent University, supervisor)
Prof. dr. ir. Aleksandra Pižurica (Ghent University)
Prof. dr. ir. Peter Veelaert (Ghent University)
Prof. dr. ir. Peter Lambert (Ghent University)
Prof. dr. ir. Tinne Tuytelaars (Katholieke Universiteit Leuven)
Prof. dr. ir. Richard Kleihorst (Ghent University, secretary)
Prof. dr. ir. Rik Van de Walle (Ghent University, chairman)

Affiliations

Research Group for Image Processing and Interpretation (IPI)
Independent Research Institute iMinds
Department of Telecommunications and Information Processing (TELIN)
Faculty of Engineering and Architecture
Ghent University

Sint-Pietersnieuwstraat 41
B-9000 Ghent
Belgium

Acknowledgements

If I have seen further, it is because I stood on the shoulders of giants.
—Sir Isaac Newton

I would like to take this opportunity to express my gratitude to colleagues, friends and family who helped and supported me throughout my PhD life and work.

First and foremost, I thank to my thesis supervisor Prof. Wilfried Philips for providing me the opportunity to undertake this exciting research in the very attractive field of computer vision and machine learning. Thank you, professor, for setting an example of rigorous and creative scientific thinking from which I learned a lot. It has been a challenge and a pleasure to work with you, and I am very grateful for your trust and support. Next, I would like to thank Prof. Aleksandra Pižurica for inviting me to come to Belgium and do my PhD at Ghent University. A big thank you, Aleksandra, for your support in the first two years of my PhD when we collaborated in the Vicats project. Thank you for your help with my first conference and journal publications.

I would also like to thank all the members of my PhD jury: prof. Wilfried Philips, prof. Aleksandra Pižurica, prof. Peter Veelaert, prof. Peter Lambert, prof. Tinne Tuytelaars, prof. Richard Kleihorst and prof. Rik Van de Walle, for being in my thesis committee, and for reading and commenting this thesis. Your questions, comments and suggestions have strengthen the final manuscript. I also gratefully acknowledge everyone who contributed to my education. It would be impossible to come this far without your contribution!

This whole research would not be possible without my collaborators in Vicats and iCocoon project, and I express my big gratitude to them: prof. Tinne Tuytelaars from Katholieke Universiteit Leuven and the ESAT-PSI-VISICS team, dr. Wouter Favoreel from Traficon-FLIR, prof. Peter Lambert and Aljosha Demeulemeester from ELIS-MMLab at Ghent University, the whole team of the video department at Alcatel-Lucent Bell Labs in Antwerp, and the iMinds institute. Of course, special thanks to all my dear friends and colleagues at TELIN department, especially my fellow researchers with whom I collaborated on a daily basis in different phases of my PhD research: Jorge Oswaldo Niño Castañeda, Sebastian Grünwedel, Andres Frías Velázquez, Peter Van Hese, Xingzhe Xie, Dimitri Van Cauwelaert, Francis Deboeverie and Dirk Van Haerenborgh. Big thanks to Francis Deboeverie, Jan Aelterman, Joris Roels and Jonas De Vylder for translating parts of this thesis into Dutch. Big thanks to Patrick, Annette and Sylvia, Philippe and Davy for providing me their precious administrative and IT support.

My dear friends have made sure that my PhD life was not just work. I thank Ljubomir, Liljana, Milan, Sašo, Tijana, Danilo, Milica, Stevan, Sebastian, Olivier, Joris, Mima, Jorge, Jelle, Dieter and Werner for all the fun we had in Ghent and around Belgium. I also thank my friends from Novi Sad, Boris, Marko, Duško, Bora, Neša, Nenad, Boba, Milan, Željko, Igor, Bojan, Goca, Nikola, Majda and Ceca for all the great time we have whenever we see each other. I hope we will stay in touch, wherever we are, for many years to come!

I am infinitely grateful to my grandparents, my parents Milenko and Marijana, and my sister Vanja for their love, trust and encouragement through all my life. Thank you for everything! Thank you my sweet little sister for all the fun we had from the day you were born. Thank you my sisters Jelena and Branka, my aunt Ljubica, and your families, for all the beautiful moments we have together.

Finally, I would like to thank my wife Ivana for her endless love, encouragement, immense understanding and unconditional support in all my doings. Thank you for loving me and being my strength and inspiration! I enjoy every day with you and love you to the bottom of my heart.

Vedran Jelača
Ghent, December 2014

Samenvatting

De laatste jaren worden videoverwerking en computervisie op grote schaal gebruikt om het dagelijkse leven te assisteren, beveiligen en vereenvoudigen op het gebied van veiligheid en inspectie, ouderenzorg, verkeersobservatie, videoconferencing, medische zorg, enz. Cameranetwerken op grote schaal zijn steeds wijder verspreid door de dalende kost van camera's en vooruitgang in cameraminiaturisatie. Het omgaan met en het analyseren van deze grote hoeveelheden videodata zijn redenen voor het ontwikkelen van nieuwe computervisie-algoritmen en samenwerkende multi-camerasystemen die de video lokaal verwerken op de camera's, en alleen compacte data uitwisselen om de gewenste taak te vervullen. Deze systemen worden slimme multi-camera netwerken genoemd.

Eén van de essentiële taken van zulke netwerken, en de focus van deze thesis, is het volgen van objecten en mensen om hun trajecten, gedrag en relaties vast te stellen. In levensechte omstandigheden zijn er verschillende significante uitdagingen bij het volgen van objecten.

- Grote variaties in belichtings- en weersomstandigheden (bij gebruik buitenshuis), zowel als frequente occlusie van de geobserveerde objecten, creëren een enorme uitdaging voor nauwkeurige tracking.

- De typische beeldkenmerken voor tracking, zoals kleur, vorm of textuur, zijn vaak niet discriminatief genoeg in werkelijke condities.

- Bandbreedtebeperkingen en moeilijkheden bij het opslaan en analyseren van grote hoeveelheden videodata maken tracking duur en technisch veeleisend. Deze moeilijkheden leggen een behoefte op slimme informatieselectie, slimme verspreiding en slimme fusie.

- Het gebruik van slimme visuele sensoren, zoals slimme camera's, creëren de behoefte voor lage complexiteit, en computationeel efficiënte real-time tracking.

De uitdagingen hangen af van de camerasetup die gebruikt wordt: een enkele camerasetup, of een multi-camerasetup met of zonder overlappende gezichtsvelden.

Als een omgeving geobserveerd wordt door een *enkele* camera, komt alle verworven informatie van een enkel gezichtspunt. In dit geval is het een uitdaging om voldoende data te verzamelen over de verschijning van geobserveerde objecten. Tracking is gevoelig aan poseveranderingen van het object en occlusies.

De enkele camerasystemen zijn nog steeds aanwezig in verschillende toepassingen, zoals perimeter en verkeerstoezicht, het buitenshuis volgen van mensen, enz.

In een netwerk met *overlappende* camerageziehtsvelden wordt elk deel van het interessegebied bekeken door minstens één camera en typisch met verschillende camera's die verschillende gezichtspunten hebben. Bij het volgen van objecten moet de informatie vanuit verschillende zichten gefuseerd worden om de performantie te verbeteren. Real-time tracking van mensen is vaak een essentiële taak in deze setups, in het bijzonder om het pad terug te vinden in beveiliging- en toezichttoepassingen [Pflugfelder 10], [Morris 08]. Verschillende andere toepassingen komen constant op. Bijvoorbeeld in telecommunicatie (meer specifiek videoconferencing) kan positionele data van elke aanwezige in de meeting gebruikt worden om interessegebieden die personen bevatten te definiëren, om zo meer gedetailleerde verwerking uit te voeren in deze gebieden. Het kan helpen om met pan-tilt-zoom (PTZ) camera's te focussen op specifieke personen [Aghajan 09], bij hette bepalen wanneer ze een kamer binnenkomen of buitengaan, hun identiteit (zelfs wanneer ze niet naar een camera kijken), en hun activiteiten afleiden [Fathi 11] zoals presenteren, kijken naar de presenteerder, en andere. In deze toepassingen, waarbij het doel is om de individuele trajecten van elke persoon vast te leggen, is het vermijden van tracking loss essentieel. Individuele trajecten mogen niet verloren gaan door occlusies en individuen mogen niet verward worden wanneer ze in elkaars buurt komen of wanneer hun paden elkaar kruisen.

Anderzijds, in een *niet-overlappend* cameranetwerk zijn er typisch "blinde" vlekken waar geen enkele camera een zicht heeft op het object. In deze netwerken is het belangrijk om niet alleen objecten te volgen vanuit elk camerastandpunt, maar ook om elk object te heridentificeren wanneer het bij andere camerazichten verschijnt, zodat trajecten uit verschillende zichten kunnen verbonden worden. Dit is een typisch scenario in verkeersobservatie, waarbij camera's in tunnels of langs wegen geplaatst worden.

In deze scriptie stellen we een tracking methode op voor deze drie camera setups. In de context van enkele camera tracking gebruiken we een multi-cue Kalman-filter-raamwerk. We stellen het gebruik van de Radontransformatiegebaseerde beeldprojectiekenmerken, wat we *signaturen* noemen, als computationeel efficiënt en verlichtingsinvariante kenmerken. We definiëren een tijdsvervormende techniek voor signatuur-matching, waar we randdetectie van gevolgde objecten gebruiken in elk videoframe. We demonstreren de voordelen van signatuur-gebaseerde kenmerken tegenover voorgrond blobs en optical flow kenmerken en analyseren vervolgens onze aanpak op verschillende verkeersobservatie sequenties, opgenomen in variabele weers- en belichtingsomstandigheden. Wanneer de lichtomstandigheden niet snel veranderen, zijn voorgrond blobs en optical flow geschikt voor tracking. In het geval van plotse belichtingsverandering hebben voorgrond blobs last van gaps- en ghosting-effecten, wat zorgt voor het uitzetten of inkrimpen van selectiekaders op de voorgrond, terwijl optical flow vectoren snel veranderen, zowel in magnitude, als in richting.

De voertuigsignaturen zijn echter relatief invariant aan deze verlichtingsveranderingen en dus zijn we hiermee beter in staat om betrouwbare metingen te bekomen.

In de context van multi-camera tracking met overlappende gezichtsvelden, afgezien van de nauwkeurigheid in realistische condities, focussen we op real-time uitvoering, lage latency en schaalbaarheid van tracking. Dit zorgt voor een extra complexiteitsniveau, vergeleken met state-of-the-art methoden zoals gepubliceerd in [Berclaz 11]. Een *real-time* en *laag-latency* operatie is in veel indoor trackers nodig, aangezien ze snel moeten reageren op positieveranderingen van personen. Aangezien deze toepassingen typisch gepaard gaan met monitoring op lange termijn, zijn *robuustheid* en tracking *nauwkeurigheid* ook van groot belang. *Schaalbaarheid* is een probleem dat veel over het hoofd gezien wordt in multi-camera-onderzoek: gecentraliseerde verwerking van meerdere videostromen creëert niet alleen een computationeel, maar ook een communicatieprobleem. Dit betekent dat het toevoegen van een camera aan het gecentraliseerd netwerk een significante impact kan hebben op het vermogen van het netwerk om waargenomen data te verdelen en verwerken. Daartoe richten we ons in deze scriptie op gedecentraliseerde en verdeelde tracking benaderingen, waarbij camera's gegroepeerd worden in clusters die communiceren met een lokaal fusiecentrum (gedecentraliseerd) of met elkaar (verdeeld), aangezien deze veel meer schaalbaarheid hebben dan gecentraliseerde benaderingen [Taj 11].

In onze aanpak, voert elke camera tracking met lage complexiteit uit, gebruik makend van beeldprojectiefeatures in combinatie met eenvoudige blobanalyse. Iedere camera representeert objecten (personen) als omhullende rechthoek (kuboiden) met respect tot een globaal coördinatensysteem. We ontwikkelden en vergeleken twee on-camera tracking benaderingen: histogram filtering en een aanpak zonder on-camera staatschatting. Aangezien we gekalibreerde camera's veronderstellen, schatten we de kuboiden corresponderend met de personen in globale coördinaten, en niet in beeldcoördinaten. Dit laat toe om een fysisch bewegingsmodel eenvoudiger op te stellen dan een model in het beelddomein waar schijnbare snelheden afhangen van de positie van de persoon relatief ten opzichte van de camera. Verder kunnen schattingen van een kuboide in globale coördinaten van verschillende camera's onmiddellijk gefuseerd worden in een fusiecentrum of zelfs in andere camera's zonder kennis van het verband tussen het beeldomein van de camera en het globaal coördinatiesysteem.

Een heel belangrijke component van onze multi-camera aanpak is feedback van het fusiecentrum. Deze feedback geeft de meest recente posities, snelheden en geometrie van de individuen in de scene terug aan de camera's. We passen probabilistische occlusieredenering toe in elke camera om te detecteren welke blobs behoren tot welke kuboide. De analyse zorgt ook voor aangepaste kuboide parameters (bv. posities). Onze methode vereist geen gesofisticeerde bewegingsestimatie voor iedere camera, maar we demonstreren de voordelen van het bezit van deze contextbewuste bewegingsmodellen.

We tonen de voordelen van onze aanpak aan door uitgebreide experimenten

waarbij personen getracked worden in vergaderzalen, d.w.z. dat de relatief eenvoudige analyse van beeldveranderingen betrouwbaar en accuraat kan zijn bij het tracken van meerdere objecten in een multi-camera netwerk. Dit wordt bereikt van zodra informatie, afkomstig van meerdere camera's, gefuseerd is en terug gecommuniceerd wordt naar elke camera, waardoor een krachtig mechanisme ontstaat om occlusieproblemen tegen te gaan. We tonen aan dat het aantal tracking losses in dit feedback gebaseerd raamwerk dicht bij nul ligt, zelfs in sequenties met overvloedige occlusie en moeilijke verlichtingsomstandigheden. De gemiddelde tracking error in deze sequenties is ongeveer 10 cm. Deze resultaten tonen een verbetering aan op state-of-the-art algoritmen zoals [Berclaz 11] en [Fleuret 08]. Ze tonen ook aan dat de toevoeging van beeldprojectie features aan voor- en achtergrondsignalen, voorgesteld door [Grünwedel 14], de tracking nauwkeurigheid verbetert, zonder daarbij de real-time prestaties en schaalbaarheid te schaden.

De communcatie overhead is zeer laag: een frequentie van 10 updates per seconde is voldoende om elke camera zijn parameters (positie, snelheid, breedte en hoogte) samen met een betrouwbare meting van iedere gevolgde kuboide naar het fusiecentrum uit te zenden. Deze geometrische descriptoren zijn geïntegreerd in het fusiecentrum, wat op zijn beurt gefuseerde descriptoren doorstuurt van alle gevolgde objecten naar alle slimme camera's. De transmissie-bandbreedte van de camera naar het fusiecentrum en terug ligt in de orde van 1 kilobyte/seconde per object voor elke camera. De lage communicatie overhead resulteert in een heel schaalbaar systeem. Bovendien is het een aanwinst bij slimme camera's op batterijen, waar de levensduur van deze batterijen meestal gelimiteerd is door communicatievermogen [Taj 11]. Het is eveneens een aanwinst in tijdelijke, ad-hoc setups, waar draadloze netwerken geprefereerd worden om bouwwerken op kabelinstallaties te vermijden.

In de context van multi-camera object tracking in niet-overlappende camerazichten is de grootste uitdaging het accuraat heridentificeren van objecten wanneer ze in elk camerazicht verschijnen. We pakken dit probleem aan met een conceptueel andere benadering dan state-of-the-art methoden [Yu 11], [Shan 08], [Guo 07], [Hou 09], [Rios Cabrera 12]. Ten eerste, gebruiken we de voorgestelde signatuur features als eenvoudige descriptoren van het objectvoorkomen, wat eenvoudig is om te berekenen en vergelijken, en toch heel informatief is bij laag-resolutie beelden. Het matchen van voorkomensmodellen wordt verkregen door een eenvoudige combinatie van 1-D correlaties in een grof-naar-fijne procedure. De signaturen worden ook gebruikt om schaalverschillen tussen observaties van verschillende camera's te leren, wat belangrijk is voor hun uitlijning. De tweede nieuigheid is het gebruik van signaturen van meerdere beelden om een voorkomensmodel, gebaseerd op meerdere observaties en automatische selectie van goede observaties voor matching (d.w.z. informatieve observaties met weinig tot geen storingen), op te stellen. Dergelijk voorkomensmodel laat toe om objecten voor the stellen vanuit meerdere standpunten, online verzameld terwijl ze bewegen in de multi-camera-omgeving. Dit is in het bijzonder voordelig wanneer objecten van vorm veranderen (bv. door

richtingsverandering of door het weglopen van de camera). Tenslotte gebruiken we het Hungarian algoritme om matching te optimaliseren en onduidelijkheden op te lossen.

Samenvattend, stellen we in deze scriptie een uitgebreid raamwerk voor om objecten te tracken in slimme cameranetwerken. We benaderden tracking problemen van laag tot hoog niveau, i.e. van objectdetectie en feature extractie tot hoog-niveau contextueel redeneren, informatieselectie en fusie. De belangrijkste contributies van deze scriptie zijn:

- computationele en data-efficiënte descriptoren van het objectvoorkomen, gebaseerd op de 1-D Radon-transformatie-achtige beeldprojecties (signaturen) [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13];
- modellering van objectvoorkomen, gebaseerd op objectsignaturen, robuustheid naar vorm- en belichtingsveranderingen toe en occlusies [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13];
- computationeel efficiënt matchen van voorkomen voor objectherkenning, gebruik makend van vervormbare curve-uitlijning, dynamische tijdsvervorming en globale en locale 1-D correlatie [Jelača 13];
- gedistribueerde multi-view modellering van voorkomen met automatische selectie van informatieve observaties voor tracking [Jelača 13];
- robuust, real-time tracking met één enkele camera, gebruik makend van beeldprojectie features in een Kalman-filter-raamwerk [Jelača 12], [Jelača 14];
- een gedecentraliseerd multi-camera raamwerk voor tracking met een feedback loop van het fusie centrum [Jelača 11a], [Grünwedel 14], [Grünwedel 12], [Xie 12].

We demonstreren deze bijdragen in meerdere, realistische scenario's:

- verschillende verkeersscenario's onder verschillende belichtings- en weersomstandigheden,
- tracken van voertuigen in een tunnel, gebruik makend van een cameranetwerk met niet-overlappende zichten, en
- tracken van personen in een vergaderzaal, gebruik makend van een cameranetwerk met overlappende zichten.

In totaal heeft dit doctoraatsonderzoek geleid tot drie publicaties in internationale peer-reviewed tijdschriften: twee gepubliceerd [Jelača 13], [Grünwedel 14], en een artikel in review fase [Jelača 14]. Verder werden dertien papers gepubliceerd in de notulen van internationale conferenties [Jelača 08], [Despotović 10], [Jelača 11b], [Jelača 11a], [Grünwedel 11a], [Van Hese 11], [Demeulemeester 11], [Niño Castañeda 11], [Frías Velázquez 11], [Jelača 12], [Grünwedel 12], [Maćešić 12], [Xie 12].

Summary

In recent years video processing and computer vision have become widely used to assist, protect and simplify the daily life in areas such as security and surveillance, elderly care, traffic monitoring, video conferencing, medical care and many more. Large-scale camera networks have become increasingly widespread due to decreasing costs of cameras and advances in camera miniaturization. Handling and analysis of these vast amounts of video data are reasons for development of new computer vision algorithms and cooperative multi-camera systems that process video data locally, on the cameras, and share only compact and informative representation of these data to fulfil the desired application task. These systems are called smart multi-camera networks.

One of the essential tasks of such networks, and the focus of this thesis, is tracking of objects and people to determine their trajectories, behaviour and relations. In real-world environments, there are several significant challenges to accomplish the object tracking task.

- Big variations in illumination and weather conditions (in outdoor use), as well as frequent occlusions of the viewed objects create a tremendous challenge for accurate tracking.
- Some of the typical image features for tracking, such as colour, shape or texture, are often not discriminative enough in real-world conditions.
- Bandwidth constraints and difficulties in storing and analysing large amounts of video data make tracking costly and technically demanding, imposing the need for smart information selection, distribution and fusion.
- Deployment of smart visual sensors, such as smart cameras, creates the need for low complexity, and data and computationally efficient real-time tracking.

The challenges also depend on the camera setup that is used: a single camera setup, or a multi-camera setup with or without overlapping views.

If the environment is observed by a *single* camera, all acquired information comes from a single viewpoint. In this case it is challenging to collect enough data about the appearance of viewed objects and tracking is sensitive to object pose changes and occlusions. The single camera systems are still present in many applications, such as perimeter and traffic surveillance, outdoor people tracking, and many others.

In a network with *overlapping* camera views, each part of the area of interest is viewed by at least one camera and typically with several cameras having different viewpoints. Object tracking needs to fuse the information from multiple views to enhance its performance. Real-time tracking of people is usually an essential task in these setups, especially for path-retracing in security and surveillance applications [Pflugfelder 10], [Morris 08]. Many other applications are constantly emerging. For instance, in telecommunications (more specifically video-conferencing), positional data of each meeting attendant can be used to define regions of interest containing people, to limit more detailed processing to these areas. It can be helpful to focus pan-tilt-zoom (PTZ) cameras on specific people [Aghajan 09], to determine when they enter and leave the room, their identity (even when they do not face a camera), and infer their activities [Fathi 11] such as presenting, looking at the presenter, and others. In these applications, in which the goal is to determine individual trajectories of each person, avoiding tracking losses is essential. Trajectories of individuals should not be lost due to occlusions and individuals should not be mixed up when they get close together or when their paths intersect.

On the other hand, in a *non-overlapping* camera network there are typically "blind" areas where neither of the cameras has a view on the object. In these networks it is necessary to not only track objects in each camera view, but also to re-identify each object when it appears in the other views so that the trajectories in different views can be connected. This is a typical scenario in traffic surveillance when cameras are placed in tunnels or along roads.

In this thesis we propose tracking methods for all these three camera setups. In the context of single camera tracking we use a multi-cue Kalman filter framework. We propose using Radon transform like image projection features, which we call *signatures*, as computationally efficient and illumination invariant cues. We define a time warping technique for signature matching, which we use to find boundaries of tracked objects in each video frame. We demonstrate the advantages of such signature based cues versus foreground blobs and optical flow cues, and evaluate our approach on several traffic surveillance sequences recorded in different weather and illumination conditions. When the lighting conditions do not change rapidly, foreground blobs and optical flow are suitable measurements for tracking. In the cases of sudden illumination changes foreground blobs suffer from gaps and ghosting effects, which leads to stretching or shrinking of foreground bounding boxes, while the optical flow vectors change rapidly, both in magnitude and direction. However, the vehicle signatures are relatively invariant to such illumination changes so with them we are able to obtain more reliable measurements.

In the context of multi-camera tracking with overlapping views, beside on the accuracy in real-world conditions, we focus on real-time, low-latency and scalability of tracking. This adds another level of complexity compared to state-of-the-art methods as published by [Berclaz 11]. *Real-time* and *low-latency* operation is needed in many indoor trackers, because they need to react quickly to changes in people's positions. Since these applications typically involve long-

term monitoring, *robustness* and tracking *accuracy* are also critical. *Scalability* is an often overlooked problem in multi-camera research: centralized processing of multiple video streams creates not only a computing but also a communication bottleneck. This means that adding a camera to the centralized network can significantly impact the network's ability to distribute and process the acquired data. Therefore, in this thesis we focus on decentralized and distributed tracking approaches, which group cameras into clusters that communicate with a local fusion centre (decentralized) or with each other (distributed), since these are much more scalable than centralized approaches [Taj 11].

In our approach, each camera performs low-complexity tracking, using simple image projection features in combination with simple blob analysis. Each camera represents objects (people) as bounding boxes (cuboids) with respect to a world coordinate system. We developed and compared two approaches for on-camera tracking: histogram filtering and an approach without on-camera state estimation. Since we assume calibrated cameras, we are estimating the person's cuboid in world coordinates rather than in image coordinates. This allows a physical motion model to be expressed more easily than a model in the image domain where apparent speeds depend on the position of the person with respect to the camera. Furthermore, the estimates of a cuboid in world coordinates from different cameras can be directly fused in a fusion center or even in other cameras without knowing the relationship between a camera image domain and the world coordinate system.

A very important component of our multi-camera approach is feedback from the fusion centre. This feedback returns to the cameras the most recent positions, speeds and geometries of individuals in the scene. Based on this feedback, we perform probabilistic occlusion reasoning in each camera to identify which blobs belong to which cuboid. The analysis also yields updated cuboid parameters (e.g. positions). Our method does not require sophisticated motion estimation in each camera, but we demonstrate the benefits of having the context aware motion models.

By extensive experiments of people tracking in meeting rooms, we demonstrate the advantages of our approach, i.e. that a relatively simple analysis of changes in images can be reliable and accurate for tracking multiple objects in a multi-camera network. It is achieved when information from multiple cameras is fused and communicated back to each camera, creating a powerful mechanism to overcome many issues with occlusions. We demonstrate that the number of tracking losses in such a feedback based framework is close to zero, even in sequences with abundant occlusion and difficult illumination conditions. The average tracking error in these sequences is around 10 cm. These results show an improvement on state of the art algorithms as reported by [Berclaz 11] and [Fleuret 08]. They also show that adding image projection features to foreground/background cues proposed by [Grünwedel 14] improves tracking accuracy, without affecting the real-time performance and scalability.

The communication overhead is very low: a frequency of 10 updates per second is sufficient for each camera to transmit the parameters (position, speed,

width and height) together with a reliability measure of each tracked cuboid to the fusion centre. These geometrical descriptors are integrated by the fusion centre, which then returns fused descriptors for all tracked objects to all smart cameras. The transmission bandwidths from the camera to the fusion centre and back are in the order of 1 kilobyte/second per object for each camera. The low communication overhead results in a highly scalable system. Moreover, it is an asset in battery operated smart cameras, where battery lifetime is mostly limited by communication power [Taj 11]. It is also an asset in ad-hoc temporary setups, where wireless networks are preferred to avoid building works on cable installations.

In the context of multi-camera object tracking in non-overlapping camera views, the biggest challenge is to accurately re-identify objects when they appear in each camera view. We address this problem by a conceptually different approach then state-of-the-art methods [Yu 11], [Shan 08], [Guo 07], [Hou 09], [Rios Cabrera 12]. Firstly, we use the proposed signature features as simple descriptors of object appearance, which are easy to compute and compare, yet highly informative in low resolution images. Matching of the appearance models is obtained by a simple combination of 1-D correlations in a coarse-to-fine procedure. The signatures are also used to learn scale differences between the observations from different cameras, which is important for their alignment. The second novelty is to use signatures from multiple images for creating a multiple observation appearance model and automatic selection of good observations for matching (i.e. informative observations with few or no disturbances). Such an appearance model enables representation of objects from multiple views, collected online as they move through the multi-camera environment. This is especially beneficial when objects change pose (e.g. by changing movement direction or moving away from the camera). Finally, we use the Hungarian algorithm to optimize the matching and resolve ambiguities.

To summarize, in this thesis we proposed a comprehensive framework for object tracking in smart camera networks. We addressed the tracking problems from the low to the high level, i.e. from object detection and feature extraction to the high level contextual reasoning, information selection and fusion. The main contributions of this thesis are:

- computationally and data efficient descriptors of the object appearance, based on 1-D Radon-transform like image projections (signatures) [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13];
- object appearance modelling based on the object's signatures, robust to pose and illumination changes, and occlusions [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13];
- computationally efficient appearance matching for object recognition using deformable curve alignment, dynamic time warping, and global and local 1D correlation [Jelača 13];
- distributed multi-view appearance modelling with automatic selection of informative observations for tracking [Jelača 13];

- robust real-time single camera tracking using image projection features in a Kalman filter framework [Jelača 12], [Jelača 14];
- a decentralized multi-camera framework for tracking with a feedback loop from the fusion centre [Jelača 11a], [Grünwedel 14], [Grünwedel 12], [Xie 12].

We demonstrate these contributions in multiple real-world scenarios:

- various traffic scenarios under different illumination and different weather conditions,
- vehicle tracking in a tunnel using a camera network with non-overlapping views, and
- people tracking in a meeting room using a camera network with overlapping views.

In total, the research during this PhD resulted in three publications in international peer-reviewed journals: two published [Jelača 13], [Grünwedel 14], and one article under review [Jelača 14]. Furthermore, thirteen papers have been published in the proceedings of international conferences [Jelača 08], [Despotović 10], [Jelača 11b], [Jelača 11a], [Grünwedel 11a], [Van Hese 11], [Demeulemeester 11], [Niño Castañeda 11], [Frías Velázquez 11], [Jelača 12], [Grünwedel 12], [Maćešić 12], [Xie 12].

Contents

List of Abbreviations

ARMA	Autoregressive moving average
ASIFT	Affine scale-invariant feature transform
BG	Background
BraMBLe	Bayesian multiple-blob tracker
DPDM	Dynamic point distribution model
DTW	Dynamic time warping
EKF	Extended Kalman filtering
EM	Expectation maximization
FAST	Features from accelerated segment test
FG	Foreground
FN	False negative
FP	False positive
GC	Graph Cut
GMM	Gaussian Mixture Model
GT	Ground truth
HF	Histogram filtering
HMM	Hidden Markov model
HOG	Histogram of oriented gradients
ICA	Independent component analysis
iCocoon	Immersive communication by means of computer vision
JPADF	Joint probabilistic data association filter
LBP	Local binary patterns
MAP	Maximum a posteriori probability
MHT	Multiple hypotheses tracking
MIL	Multiple instance learning
MOG	Mixture of Gaussians
MSE	Mean squared error
NLE	No local state estimation
NN	Nearest neighbour
NoOL	Number of object losses
NoOS	Number of object switches
PCA	Principal component analysis
PDA	Probabilistic data association
PDF	Probability density function
PF	Particle filtering
PGM	Probabilistic graphical model
RANSAC	Random sample consensus

ROI	Region of interest
SIFT	Scale-invariant feature transform
SURF	Speeded up robust features
SVD	Singular value decomposition
SVM	Support vector machines
TATE	Total average tracking error
TBD	Track-before-detect
TP	True positive
ViBe	Visual background extractor
VICATS	Video content analysis for traffic/tunnel surveillance

1

Introduction

In recent years video processing and computer vision have become widely used tools to assist, protect and simplify the daily life by deployment in areas such as security and surveillance, elderly care, traffic monitoring, video conferencing, medical care and many more. Large-scale camera networks have become increasingly widespread due to decreasing costs of cameras and advances in camera miniaturization. Handling and analysing these vast amounts of video data are reasons for development of new computer vision algorithms and cooperative multi-camera systems that process video data locally, on the cameras, and share only compact and informative representation of these data to fulfil the desired application task. These systems are called decentralized or distributed multi-camera networks.

One of the essential tasks of a multi-camera network is tracking of viewed objects (in most cases humans and vehicles) to determine their trajectories, behaviour and relationships to each other. Here, security and surveillance for path-retracing is among the best known applications [Pflugfelder 10], [Morris 08]. More recently, telecommunication applications are also emerging. For instance, in video-conferencing, positional data of each meeting attendant can be used to define regions of interest containing people, to limit more detailed processing to these areas. It can be helpful to focus pan-tilt-zoom (PTZ) cameras on specific people [Aghajan 09], to determine when they enter and leave the room, their identity (even when they do not face a camera), and infer their activities [Fathi 11] such as presenting, looking at the presenter, and others.

There are several significant challenges on the way to accomplish the object tracking task in real-world environments (see Figures 1.1 and 1.2).

- Big variations in illumination and weather conditions (in outdoor use), as well as frequent occlusions of the viewed objects create a tremendous challenge for accurate tracking.

- Some of the typically used image features for tracking, such as colour, shape or texture, are often not discriminative enough in real-world conditions.

- Bandwidth constraints and difficulties in storing and analysing large amounts of video data make tracking costly and technically demanding, imposing the need for smart information selection, distribution and fusion.
- Deployment of smart visual sensors, such as smart cameras, creates the need for low complexity, and data and computationally efficient real-time tracking.

Networks of smart cameras and low resolution visual sensors attract a lot of research attention [Soro 09a]. These are cameras with on-board processing and communication hardware. They allow construction of more flexible and scalable camera networks because the required image processing can be distributed over the cameras. The collaborative processing of the output data of the smart cameras can take place either on a central station or on one of the cameras. However, data processing in a smart camera network entails some specific challenges. The hardware embedded with the image sensor is usually designed specifically for image processing (high degree of parallelism), which is an advantage, but it also has some limitations in terms of memory and processing power. If the amount of output data of the smart cameras is kept low, wireless operation also becomes possible. This is a huge advantage for the flexibility of the system. Battery operation is in this case also desirable. A battery life on a single charge is extended if the image processing algorithms are computationally efficient and require less communication between the cameras or with the central station.

Various methods for multi-camera tracking have been proposed in literature, such as [Kim 06], [Fleuret 08], [Khan 09], [Taj 09] to name a few. Most of them focus on computer vision tasks such as foreground segmentation, motion analysis and 3-D position estimation, but achieving real-time, scalable and robust collaborative tracking, with optimal use of multi-camera resources remains insufficiently explored. A low-latency problem is also often overlooked in multi-camera networks (centralized processing of multiple video streams creates not only a computing but also a communication bottleneck). One way to address these problems is to shift the computation load towards the smart camera and to limit the data exchange within the camera network by transmitting compact and representative data, instead of whole images. In this way, no video transmission is needed for the purpose of object tracking, not even for regions of interest within the camera views. When video transmission is needed for other purposes, the positional information provided by the tracking system can help to reduce the overall video bandwidth by restricting transmission to regions of interest. However, even many detailed image analysis algorithms, e.g. face recognition, can run on a single (smart) camera and do not require video transmission.

Taking into account all aforementioned aspects, in this PhD thesis we address the following multi-camera tracking problems.

- **The information selection problem:** Which image and video features

are data and computationally efficient, and at the same time robust to illumination changes and object occlusions? Which features create the most informative object descriptors?

- **The distribution of work within the network:** Which part of the analysis can be done in the smart camera and which within the network? How to avoid processing overload in any given camera by distributing work to other cameras? How to achieve real-time processing within a camera network?

- **The information distribution problem:** What is the optimal amount of information to send from each camera, avoiding that some cameras send redundant information? What information needs to be sent from each camera to maximally contribute to the system?

- **The information integration problem:** What is the most likely estimate of the current position of any object given the data from all cameras? How can the integrated information be sent back to the cameras to improve the accuracy of on-camera trackers?

We focus on finding solutions to address all these problems by deeply integrating different tracking levels into a single framework.

Low-level: Low-level tracking is based directly on features extracted from raw video data. These approaches include algorithms for foreground/background segmentation, low-level object detection (determination of object boundaries based on the extracted features), and rough estimation of object's appearance, pose and position. The processing is usually performed on, but not limited to, a single camera and operates on a frame-by-frame basis.

Mid-level: At the middle level, the trajectories obtained at the low-level are associated into longer trajectories. This is often done considering not only initialization, termination and transition of trajectories, but also hypotheses of trajectories being false, i.e. not belonging to any object of interest. At this level the appearance and motion model are typically refined to characterize the tracked object (target) more accurately. A modified transition matrix can be computed and sent to an optimization algorithm to obtain optimal associations. The processing at this level is more beneficial when performed in a multi-camera network. In particular, multi-camera networks with overlapping views provide substantial advantages over a single fixed viewpoint camera in terms of accuracy and precision of mid-level tracking.

High-level: High-level tracking typically operates on an abstract level and combines several low-level and mid-level cues, such as object detection, object recognition, object pose estimation, and others. At the high level the contextual information is also included into tracking. A scene structure model is often reconstructed, including maps for scene entries and exits, occluders and relationships between the objects. Kinematics of objects is used to constrain trajectory associations. All this high-level information is then used to assign

the long range trajectories and reduce trajectory fragmentation and possible identity switches.

Each of these levels poses already challenges on its own. Therefore, combining approaches of each level for development of applications such as multi-camera vehicle and people tracking is highly non-trivial. In this thesis, we deploy techniques ranging from low-level to high-level, specifically designed for smart multi-camera networks. Our focus is on robust, real-time and scalable tracking of multiple vehicles and people simultaneously. The developed algorithms evolve from techniques for pattern recognition of speech and text, and probabilistic modelling used in automation and robotics. We demonstrate the proposed techniques using a single camera and smart camera networks with overlapping and non-overlapping views. This research has been performed in projects VICATS (Video Content Analysis for Tunnel Surveillance) and ICOCOON (Immersive Communication by means of Computer Vision). More information about these projects we give in Chapters 6 and 5, where we explain also how this PhD research has contributed to these industrial applications.

1.1 Contributions and publications

The main novelties and contributions presented in this thesis are as follows.

- **Computationally and data efficient descriptors of the object appearance, based on 1-D Radon-transform like image projections.** These descriptors, which we call *signatures*, similarly as edge based descriptors capture discontinuities in image brightness, but without a thresholding step in the feature calculation. Therefore, there is no dependence on threshold parameters and the proposed signatures can be calculated more consistently than edges in various illumination conditions. These features are also robust against occlusions and inaccurate or false object detections, common in real-world tracking scenarios. They are suitable for implementation on smart cameras and can be calculated for all objects in an image in a single reading of the image. We represent the appearance of viewed objects with the signatures computed along different image axes. This contribution was demonstrated in applications for traffic surveillance and people tracking in a meeting room. It was a basis for further novelties in our work and has been published as a part of several publications: [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13].

- **An object appearance modelling based on the object's signatures.** We represent the appearance of viewed objects with the signatures along different image axes. We demonstrate that horizontal and vertical signatures capture most of the information about the appearance. The appearance model is constructed over time to capture information from different viewpoints as the object moves through the scene, collecting the most informative signatures into the model and neglecting the signatures from bad observations (sudden illumination changes, false or inaccurate

detections and occlusions). Moreover, we extended this appearance modelling approach to multi-camera environments where models can be created collaboratively by the cameras along the object trajectory. These contributions were demonstrated in applications for traffic surveillance in different weather and illumination conditions, and people tracking in a meeting room. They have been published as a part of several publications: [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13].

- **A computationally efficient appearance matching for object recognition.** We propose a coarse to fine procedure based on global and local 1-D correlations to compare 1-D signatures and compute the appearance matching measure. We also use a dynamic time warping technique to match the signatures that originate from the observations in successive frames. The Hungarian optimization algorithm is used for many-to-many matching when the observations are taken from different cameras. This work has been published in a journal [Jelača 13].

- **A distributed multi-view appearance modelling** with automatic selection of informative observations for tracking in a single camera view and camera networks with overlapping and non-overlapping views. This appearance modelling enables scalable real-time tracking of multiple objects simultaneously (the scalability means that adding more cameras into the network has low impact on real-time performance of tracking). This work has been published in a journal [Jelača 13].

- **Robust real-time single camera tracking.** We incorporated the proposed appearance models into a multi-cue Kalman filter framework to robustly track vehicles and people even in cases of challenging illumination conditions, low resolution images with various artefacts and different camera viewpoints. We use tracking by recognition approach, where the objects are redetected and reidentified at each time instance. This work has been published in a conference [Jelača 12] and a journal paper [Jelača 14] has been submitted.

- **A decentralized multi-camera framework for tracking with a feedback loop from the fusion centre.** In this framework, the main part of the low-level video processing takes place in the cameras. Each camera transmits a compact high-level description of moving objects to the fusion center, which fuses these data using a Bayesian approach. Possible errors in estimations done by cameras are corrected using the feedback made by a cooperative decision from all available cameras in the network. The performance of the proposed system is evaluated in terms of precision and accuracy on indoor and meeting scenarios. In meeting scenarios we construct a context aware motion model that takes into account position of furniture in the room and people behaviour (walking, standing in place, sitting). This work has led to one journal publication [Grünwedel 14] and three conference publications [Jelača 11a], [Grünwedel 12]

and [Xie 12].

We demonstrate these contributions in multiple real-world scenarios:

- various traffic scenarios under different illumination and different weather conditions,
- vehicle tracking in a tunnel using a camera network with non-overlapping views, and
- people tracking in a meeting room using a camera network with overlapping views.

In total, the research during this PhD resulted in three publications in international peer-reviewed journals: two published [Jelača 13], [Grünwedel 14], and one article under review [Jelača 14]. Furthermore, thirteen papers have been published in the proceedings of international conferences [Jelača 08], [Despotović 10], [Jelača 11b], [Jelača 11a], [Grünwedel 11a], [Van Hese 11], [Demeulemeester 11], [Niño Castañeda 11], [Frías Velázquez 11], [Jelača 12], [Grünwedel 12], [Maćešić 12], [Xie 12].

The work on multi-camera tracking of vehicles in tunnels has been performed in collaboration with my colleague Jorge Niño Castañeda. Therefore, parts of the general concept of vehicle tracking might also appear in his PhD dissertation. However, his work has been mainly focused on vehicle detection and single camera tracking using local binary patterns and optical flow, so tracking based on image projection features, vehicle identification and multi-camera tracking with non-overlapping views are not the focus of his thesis.

The work on multi-camera people tracking has been performed in collaboration with my colleague Sebastian Grünwedel. Our joint work focused on the conceptual design of a distributed multi-camera system for collaborative tracking (information selection, distribution and fusion), but from different perspectives. In his work and PhD thesis the focus is on a system design of collaborative trackers, mainly the information distribution and fusion within the system. In his work he also tackles problems of foreground/background segmentation robust to lighting changes and occupancy mapping in centralized multi-camera systems. In this thesis, however, a higher focus is on information selection, optimizing tracking features and incorporating low-level and mid-level tracking cues to create more robust and more accurate tracking data.

Some other colleagues in the Image Processing and Interpretation (IPI) research group at Ghent University, and in Vision Systems (VIS) group at Hogeschool Gent have also worked on the multi-camera system for people tracking, though more from an engineering perspective. Their contribution will be explained in Chapter 6, in the more detailed explanation of the iCocoon project.

Figure 1.1: Examples of people surveillance videos from real meeting room environments. We see that significant illumination changes are possible when indoor lighting is turned on or off, or when ambient lighting varies in different parts of the room. There are also significant occlusions of people by the room furniture or between people themselves. When objects are viewed by low resolution visual sensors there is an additional challenge to find tracking features (see the bottom row).

Figure 1.2: Examples of traffic surveillance videos from real-world environments. We see there is a big variation in illumination and weather conditions, as well as camera viewpoints. Vehicle images often have low resolution and there are frequent and significant occlusions.

1.2 List of publications

International peer-reviewed journals

V. Jelača, A. Pižurica, J.O. Niño Castañeda, A. Frías Velázquez and Philips W. *Vehicle matching in smart camera networks using image projection profiles at multiple instances.* Image and Vision Computing, vol. 31, no. 9, pages 673-685, 2013.

S. Grünwedel, V. Jelača, J.O. Niño Castañeda, P. Van Hese, D. Van Cauwelaert, D. Van Haerenborgh, P. Veelaert and W. Philips. *Low-Complexity Scalable Distributed Multi-Camera Tracking of Humans.* ACM Transactions on Sensor Networks, vol. 10, no. 2, May 2014.

International peer-reviewed conferences

V. Jelača, A. Pižurica and W. Philips. *Computationally efficient algorithm for tracking of vehicles in tunnels.* In Proceedings of the 19th Annual Workshop on Circuits, Systems and Signal Processing, pages 335-338, 2008.

I. Despotović, V. Jelača, E. Vansteenkiste and W. Philips. *Noise-robust method for image segmentation.* In Advanced Concepts for Intelligent Vision Systems, Lecture Notes in Computer Science, volume 6474, pages 153-162, 2010.

V. Jelača, J.O. Niño Castañeda, A. Frías Velázquez, A. Pižurica and W. Philips. *Real-time vehicle matching for multi-camera tunnel surveillance.* In Proceedings of SPIE, the Society of Photo-Optical Instrumentation Engineers, volume 7871, 2011.

V. Jelača, S. Grünwedel, J.O. Niño-Castañeda, P. Van Hese, D. Van Cauwelaert, P. Veelaert and W. Philips. *Demo: Real-time indoors people tracking in scalable camera networks.* In Proceedings of the Fifth ACM/IEEE International Conference on Distributed Smart Cameras, 2011.

S. Grünwedel, V. Jelača, P. Van Hese, R. Kleihorst and W. Philips. *PhD forum: Multi-view occupancy maps using a network of low resolution visual sensors.* In Proceedings of the Fifth ACM/IEEE International Conference on Distributed Smart Cameras, 2011.

A. Demeulemeester, V. Jelača, C. Hollemeersch, S. Grünwedel, P. Lambert, J.O. Niño Castañeda, D. Van Cauwelaert, P. Van Hese, P. Veelaert, R. Van de Walle and W. Philips. *Demo : real-time 3D visualization of multi-camera room occupancy monitoring for immersive communication systems.* In Proceedings of the Fifth ACM/IEEE International Conference on Distributed Smart Cameras, 2011.

A. Frías Velázquez, J.O. Niño Castañeda, V. Jelača, A. Pižurica and W. Philips. *A mathematical morphology based approach for vehicle detection in road tunnels.* In Proceedings of SPIE, the International Society for Optical Engineering, volume 8135, 2011.

J.O. Niño Castañeda, V. Jelača, R. Rios Cabrera, A. Frías Velázquez, A. Pižurica, T. Tuytelaars and W. Philips. *Non-overlapping multi-camera detection and tracking of vehicles in tunnel surveillance.* In Proceedings of the International Conference on Digital Image Computing Techniques and Applications, pages 591-596, 2011.

P. Van Hese, S. Grünwedel, J.O. Niño Castañeda, V. Jelača and W. Philips. *Evaluation of background/ foreground segmentation methods for multi-view occupancy maps.* In Proceedings of the 2nd International Conference on Positioning and Context-Awareness, pages 37-42, 2011.

V. Jelača, J.O. Niño Castañeda, A. Pižurica and W. Philips. *Image projection clues for improved real-time vehicle tracking in tunnels.* In Proceedings of SPIE, the International Society for Optical Engineering, volume 8301, 2012.

S. Grünwedel, V. Jelača, J.O. Niño Castañeda, P. Van Hese, D. Van Cauwelaert, P. Veelaert and W. Philips. *Decentralized tracking of humans using a camera network.* In Proceedings of SPIE, the International Society of Photo-Optical Instrumentation Engineers, volume 8301, 2012.

M. Maćešić, V. Jelača, J.O. Niño Castañeda, N. Prodanović, M. Panić, A. Pižurica, V. Crnojević and W. Philips. *Real-time detection of traffic events using smart cameras.* In Proceedings of SPIE, the International Society for Optical Engineering, volume 8301, 2012.

X. Xie, S. Grünwedel, V. Jelača, J.O. Niño Castañeda, D. Van Haerenborgh, D. Van Cauwelaert, P.Veelaert, W. Philips and H. Aghajan. *Learning about Objects in the Meeting Rooms from People Trajectories.* In Proceedings of the 6th ACM/IEEE International Conference on Distributed Smart Cameras, 2012.

1.3 Thesis outline

The remainder of the thesis is organized as follows.

Chapter 2: Object tracking. In this chapter we give background on object tracking. We start with a brief overview of object representation and object detection as two important factors that influence the choice and design of object trackers. We also explain and motivate the object representation and detection choices we have made in our work. Next, we describe various methods for object tracking, with comparison of their advantages and disadvantages. We focus in more detail on the methods we use in our work.

Chapter 3: Image projection features for object tracking. This chapter we begin with an overview of features that are typically used for object tracking. We explain their advantages and disadvantages in the context of smart camera networks and real-world conditions. Then, we give an overview of appearance modelling methods and relate them to the underlying features. After this we introduce the image projection features that we proposed in our work to model the appearance of tracked objects. We show characteristics of these features and demonstrate their advantages in cases of illumination changes, changes in object pose (viewpoint at the object), and inaccurate and false object detections.

Chapter 4: Tracking in a single camera view. In this chapter we give a formal definition of the object tracking problem in a single camera view and show the challenges of real-time tracking in real-world conditions. We give an overview of the common methods that address this problem. Then we define our approach by introducing appearance modelling using the image projection features proposed in Chapter 3. We show how to use these appearance models in a Kalman filter framework for multi-cue tracking. The evaluation of our approach is done both qualitatively and quantitatively on several traffic surveillance videos recorded in different weather conditions, during night and in a tunnel.

Chapter 5: Tracking in non-overlapping camera views. This chapter addresses the problem of tracking objects in environments that are not completely observed by cameras. The objects need to be identified in each camera view and therefore, after giving an overview of state-of-the-art work in this area, we define our approach for object recognition suitable for real-time tracking on smart cameras. We define a method to construct a multi-view multi-template appearance model and use it for object recognition and tracking. We demonstrate the robustness of our approach on tracking vehicles in harsh tunnel environments and compare our results with several other methods.

Chapter 6: Tracking in overlapping camera views. In this chapter our focus is on people tracking using a network of smart cameras with overlapping views. People are tracked in a video conference scenario, hence the tracker has to be robust to frequent occlusions by room furniture and between people themselves, as well as to global and local lighting changes. Before we introduce our approach we give an overview of the state-of-the-art in multi-camera tracking with overlapping camera views. Then, we show how to use

image projection profiles in a multi-cue fashion for people tracking, how to construct a people motion model in a furnished room, build a collaborative auto-corrective multi-camera tracker, and deal with occlusions by high level reasoning. We demonstrate the accuracy, real-time performance and scalability of our approach using several video sequences recorded in a meeting room. Finally, we show potential of our approach in real video conference applications.

Chapter 7: Conclusion This chapter concludes this PhD thesis, giving an overview of the main findings and proposals for the future work.

2

Object Tracking

Object tracking can be defined as the problem of estimating the trajectory of an object as it moves through the scene. In other words, a tracker assigns consistent labels to the tracked objects (targets) in different frames of a video. Additionally, depending on the tracking domain, a tracker can also provide object-centric information, such as orientation, area, or shape of an object. Some of the issues that make object tracking complex are:

- loss of information caused by projection of the 3D world on a 2D image,
- noise in images or low image resolution,
- scene illumination changes,
- non-rigid nature of objects,
- complex object motion,
- changes of the object appearance,
- partial and full object occlusions, and
- real-time processing requirements.

Many approaches for object tracking have been proposed. They primarily differ based on the way they address the following questions: Which object representation is suitable for tracking? Which image features should be used? How should the motion, appearance, and shape of the object be modelled? The answers to these questions depend also on the context in which the tracking is performed. This chapter provides the necessary background on object tracking.

Tracking algorithms in almost all cases require object detection as an input information, i.e. instances of the objects to track in the video. Depending on the way objects are represented (as points, rectangles, or some more precise shapes) object detection can also deliver information about the object boundaries, in the best case to fully extract the object from the rest of the image. In this context, object detectors might be better called object extractors, but the

term *detector* is commonly used in literature so we also use it in this thesis. In the context of tracking, detection methods are very important because they can significantly influence on design of object tracking. There is also a whole group of tracking methods called tracking-by-detection in which tracking is performed by redetecting objects and establishing correspondences with previous detections. Therefore, in this chapter where we give an overview on object tracking, we pay significant attention to object detection as well. For more details we refer the reader to very good surveys of object detection and tracking made by Yilmaz *et al.* [Yilmaz 06], and of local feature detectors made by Mikolajczyk and Schmid [Mikolajczyk 05a], and Tuytelaars and Mikolajczyk [Tuytelaars 08]. We also refer the reader to the book of Thrun *et al.* [Thrun 05], which explains in detail the tracking methods applied to intelligent robotics and autonomous agents.

This chapter is structured as follows. In Section 2.1 we give an overview of various types of object representation. In Section 2.2 we explain different object detection methods. Object tracking methods that are relevant for our work are explained in Section 2.3. Section 2.4 concludes this chapter.

2.1 Object representation overview

In this section we describe the object shape and appearance representations commonly used for detection and tracking.

2.1.1 Object shape representation

Objects are typically represented in one of the following ways.

Points. In general, the point representation is suitable for tracking objects that occupy small regions in an image. The object can be represented either by a single point, e.g. the centroid, (Figure 2.1b) [Veenman 01], or by a set of points (Figure 2.1c) [Serby 04]. In our work we used the point representation to represent objects in the Kalman filter framework. We represent the position of tracked people by a point (a projection of their centroid) on the ground plane.

Primitive geometric shapes. In this type of representation, object shape is represented by a rectangle, ellipse (Figure 2.1d, 2.1e) [Comaniciu 03], or some other geometric shape. Object motion for such representations is usually modelled by translation, affine, or projective (homography) transformations. Though primitive geometric shapes are more suitable for representing simple rigid objects, they are also used for tracking non-rigid objects. In our work we used this type of object representation. In 2D (images) we represented vehicles and people by rectangular bounding boxes or polygons, while for representation in the 3D space (real-world) we used cuboids.

Object silhouette and contour. Contour representation defines the boundary of an object (Figure 2.1h, 2.1i). The region inside the contour is called the silhouette of the object (see Figure 2.1j). Silhouette and contour representations are typically suitable for tracking complex non-rigid shapes [Yilmaz 04].

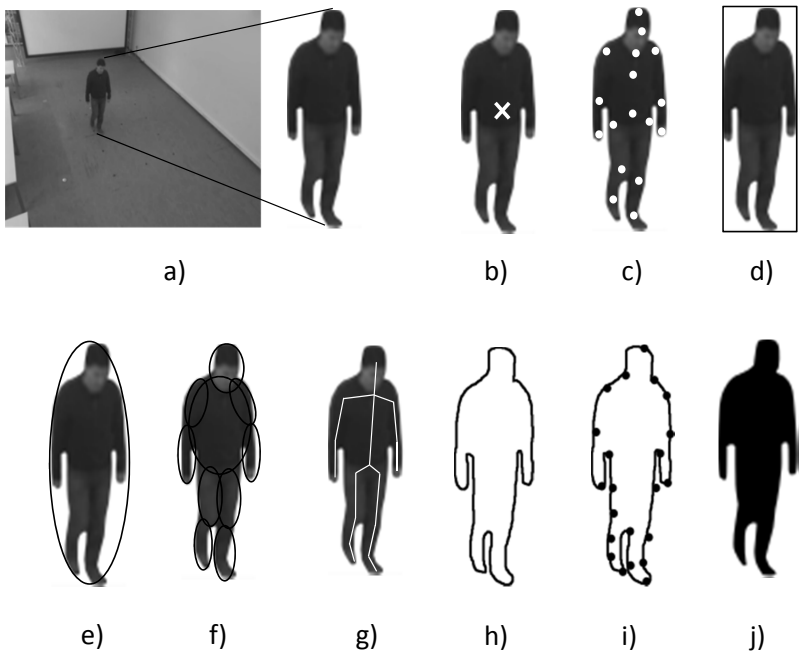


Figure 2.1: *An example video frame from a people tracking scenario (a), and different types of object representations*: b) Centroid; c) Multiple points; d) Rectangular patch (bounding box); e) Elliptical patch; f) Part-based multiple patches; g) Object skeleton; h) Complete object contour; (i) Control points on the object contour; (j) Object silhouette.

Articulated shape models. Articulated objects are composed of connected body parts. For example, the human body is an articulated object with torso, legs, hands, head, and feet connected by joints. The relationship between the parts are typically governed by kinematic motion models. In order to represent an articulated object, the constituent parts can be modelled using, for instance, cylinders or ellipses as shown in Figure 2.1f.

Skeletal models. Object skeleton can be extracted by applying the medial axis transform to the object silhouette [Ballard 82]. This model can be used to model both articulated and rigid objects (see Figure 2.1g). In some works, e.g. [Ali 01], this model has also been used as a shape representation for recognizing objects.

2.1.2 Object appearance representation

There are several common appearance representations in the context of object tracking.

Probability densities of object appearance. The probability density estimates of the object appearance can either be parametric, such as Gaussian [Zhu 96]

and a mixture of Gaussians [Paragios 02], or non-parametric, such as Parzen windows [Elgammal 02] and histograms [Comaniciu 03]. The probability densities of object appearance features (e.g. colour or texture) can be computed from the image regions specified by the selected shape models (e.g. the interior region of an ellipse or a contour).

Templates. Templates are formed using simple geometric shapes or silhouettes [Fieguth 97]. An important advantage of a template is that it carries both spatial and appearance information. Traditionally, templates encode the object appearance generated from a single view. In such cases they are only suitable for tracking objects whose poses do not vary considerably during the course of tracking. In this thesis we use templates for object appearance representation and a significant portion of our work has been dedicated to creating and matching templates from multiple camera views.

Active appearance models. Active appearance models are generated by simultaneously modelling the object shape and appearance [Edwards 98]. In these models, the object shape is typically defined by a set of landmarks. Similarly to the contour-based representation, the landmarks can reside on the object boundary or, alternatively, inside the object region. For each landmark an appearance vector is stored, which is usually in the form of colour, texture, or gradient magnitude. Active appearance models typically require a training phase to learn both the shape and its associated appearance from a set of samples. To create such a sample set one can use, for instance, the principal component analysis (PCA).

Multi-view appearance models. These models encode different views of an object. One way to represent these different object views is to generate a subspace from the given views. Subspace approaches, for example, principal component analysis (PCA) and independent component analysis (ICA), have been used for both shape and appearance representation [Black 98]. Some other possible approaches to learn the different views of an object is by training a set of classifiers, for example, Bayesian networks [Park 04] or the support vector machines [Avidan 01].

In general, there is a strong relationship between the object representations and the tracking algorithms. Object representations are usually chosen according to the application domain. For tracking objects that appear very small in an image, a point representation is usually appropriate. For the objects whose shapes can be approximated by rectangles or ellipses, more appropriate are primitive geometric shape representations. For instance, Comaniciu *et al.* [Comaniciu 03] for tracking football players used an elliptical shape representation and for appearance modelling they employed a colour histogram computed from the elliptical region. Black and Jepson [Black 98] used eigenvectors to represent the appearance. The eigenvectors were generated from rectangular object templates. For tracking objects with complex shapes, for example humans, a contour or a silhouette based representation is often appropriate.

In our work presented in this PhD thesis, we use rectangular bounding boxes to represent vehicles and people in a single camera view. In a multi-view

setup we represent these objects as cuboids in a 3D (real-world) environment, and as a polygonal projection of the cuboids onto the image plane in each of the camera views. In the low level tracking steps (e.g. tracking based on foreground blobs), the objects are represented by silhouettes. We represent the appearance of objects by Radon transform like image projection profiles, which we call signatures. We will define these signatures in Chapter 3 of this thesis. From the signatures we construct appearance templates and multi-view appearance models, which we then use for object tracking and recognition.

2.2 Object detection overview

Tracking methods typically require object detection either in every frame of the video, in multiple frames while object moves throughout the scene, or at least when the object first appears in the scene. Object detection methods can use information from a single frame or the temporal information computed from a sequence of frames. This temporal information is often based on frame differencing, to detect changing regions and moving objects in consecutive frames, which can reduce the number of false detections. Given the object detections (object regions in the image), the tracker then performs object correspondence from one frame to the next to generate the tracks (trajectories).

Some of the common object detection methods can be divided into several groups: detectors based on local features, segmentation based detectors, detectors based on foreground or background modelling, supervised or unsupervised classifiers, and others.

2.2.1 Detectors based on local features

These detectors find interest points (keypoints) and regions in images, which have an expressive texture in their localities. An object of interest is then detected based on the correspondence of these features in this new versus previous observations. There are very good surveys of local feature detectors made by Mikolajczyk and Schmid [Mikolajczyk 05a], and Tuytelaars and Mikolajczyk [Tuytelaars 08]. As formulated by Tuytelaars and Mikolajczyk [Tuytelaars 08], good features should have the following properties.

- *Repeatability*: given two images of the same object or scene, even when taken under different viewing conditions, there should be a high percentage of the corresponding features detected in both images. This can be achieved if the features are invariant to the image transformations that can occur between different views.
- *Distinctiveness/informativeness*: the intensity patterns underlying the detected features should show a lot of variation, such that features can be distinguished and matched.
- *Quantity*: the number of detected features should be sufficiently large, such that a reasonable number of features are detected even on small

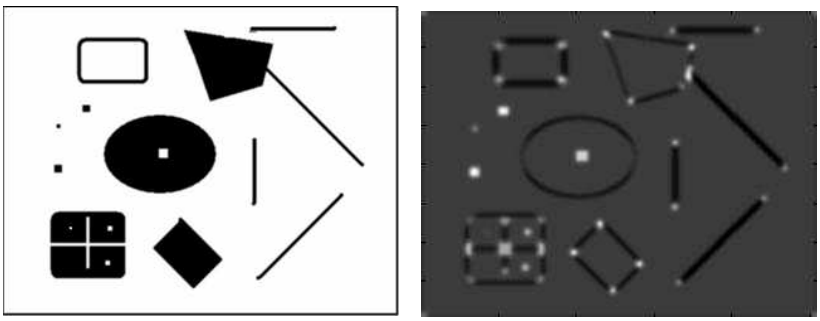


Figure 2.2: *Example of Harris corners detection.* Left: the original image; Right: the illustration of Harris cornerness measure. We see that Harris corners (highlighted by white dots in the image on the right) are found as locations where the image signal varies significantly in both x and y directions.

objects. The density of features should reflect the information content of the image to provide a compact image representation.

- *Locality*: the features should be local, to enable simple model approximations of the geometric and photometric deformations between two images taken under different viewing conditions.
- *Accuracy*: the detected features should be accurately localized, both in spatial location, as with respect to scale and possibly shape.
- *Efficiency*: preferably, the detection of features in a new image should allow for time-critical applications.

Some common interest point detectors are the Harris [Harris 88], KLT [Shi 94], SIFT [Lowe 04], SURF [Bay 08] and FAST [Rosten 05] interest point detectors. For a comparative evaluation of interest point detectors, we refer the reader to the surveys [Mikolajczyk 05a] and [Tuytelaars 08].

The *Harris* detector computes the first order image derivatives in x and y directions to find the directional intensity variations. Then, a second moment matrix, which encodes this variation, is evaluated for each pixel in a small neighbourhood. Corners are found as locations in the image where the image signal varies significantly in both directions (see Figure 2.2). Interest points are similarly computed by the *KLT* detector, with an additional criterion that enforces a predefined spatial distance between detected interest points (points that are spatially close to each other are eliminated). The Harris and KLT detectors are invariant to rotation and translation, but not to affine or projective transformations.

In order to introduce robust detection of interest points under different transformations, Lowe [Lowe 04] introduced the *SIFT* (Scale Invariant Feature Transform) method. This method detects interest points based on the peaks in the histograms of gradient directions in a small neighbourhood around a

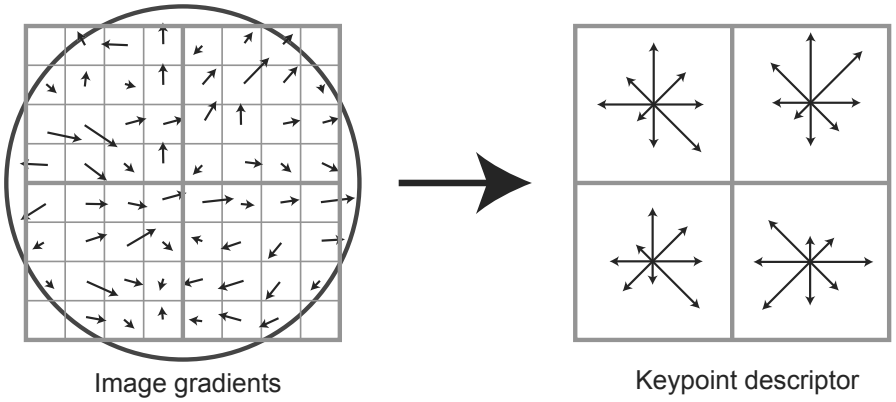


Figure 2.3: *An example of a SIFT descriptor.* A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over subregions, as shown on the right, with the length of each vector (arrow) corresponding to the sum of the gradient magnitudes within the region. This figure shows a 2×2 descriptor array computed from an 8×8 set of samples.

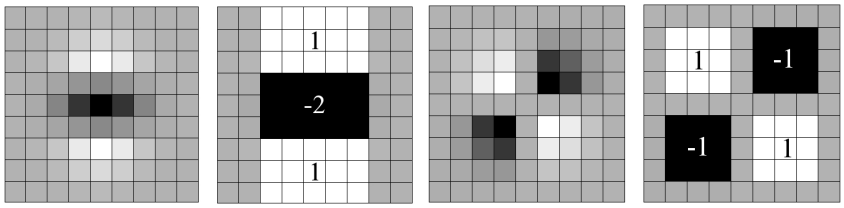


Figure 2.4: *Illustration of box-type SURF filters.* These filters can be quickly applied on images using integral images as illustrated in Figure 2.5.

point, and creates a keypoint descriptor as shown in Figure 2.3. The points are detected at different image scales and resolutions. Therefore, SIFT detector generates a greater number of interest points compared to other point detectors. It has also been shown by Mikolajczyk and Schmid [Mikolajczyk 05a] that SIFT outperforms most point detectors and is more resilient to image deformations. On the other hand, SIFT detector is not very computationally efficient, so there are recent point detectors such as SURF and FAST that are made with the goal to achieve high computational efficiency.

SURF (Speeded Up Robust Features) have been proposed by Bay *et al.* [Bay 08]. It is a scale-invariant feature detector based on the Hessian-matrix. It uses the determinant of the Hessian matrix both for selecting the location and scale. The Hessian matrix is roughly estimated using a set of box-type filters, as

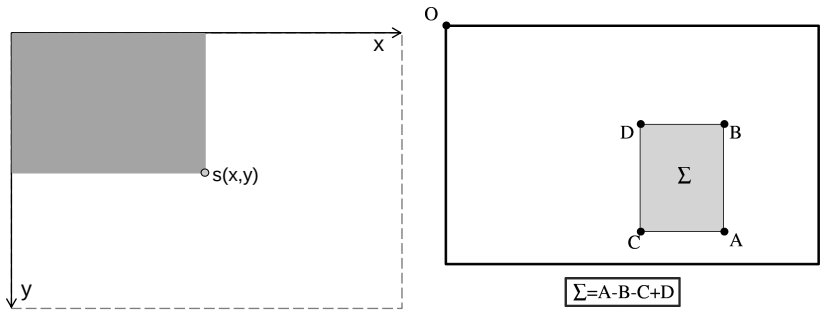


Figure 2.5: *Illustration of the integral image and the sum calculation of pixel values in an arbitrary image area.* Left: The integral image is a summed area image in which each pixel is equal to the sum of all pixel values with lower or equal x and y coordinates; Right: Having the integral image, a sum of pixel values in any arbitrary image area can be calculated in only three basic arithmetic operations (two subtractions and one addition).

shown in Figure 2.4, similar to the rectangular filters proposed in [Viola 01] for face detection. These box filters approximate second-order Gaussian derivatives and can be evaluated very fast using integral images [Viola 01], independently of their size, see Figure 2.5. This enables using SURF features in real-time video processing.

The *FAST* detector was introduced by Rosten and Drummond in [Rosten 05, Rosten 06]. This detector compares pixels only on a circle of fixed radius around the point. A circle of 16 pixels around the corner candidate is considered (see Figure 2.6). The pixels are classified into dark, similar, and brighter subsets. A decision tree algorithm from [Quinlan 86] is used to select the pixels which yield the most information about whether the candidate pixel is a corner. This is measured by the entropy of the positive and negative corner classification responses based on this pixel. This process is applied recursively on all three subsets and terminates when the entropy of a subset is zero. The decision tree resulting from this partitioning is used as a corner detector. Finally, non-maxima suppression is applied on the sum of the absolute difference between the pixels in the circle and the center pixel. This results in a very efficient detector, applicable in real-time video processing. The FAST detector can produce a large number of interest point candidates, but there are many unstable ones that exist only in some viewing conditions and need to be filtered out during the process of feature matching in different views.

In our work we used KLT, FAST and SIFT detectors as some of the cues for people and vehicle tracking and as a comparison with our proposed approaches.

2.2.2 Detectors based on foreground/background models

Detection of moving objects can be achieved by building a representation of the scene called the background model and then finding deviations from the

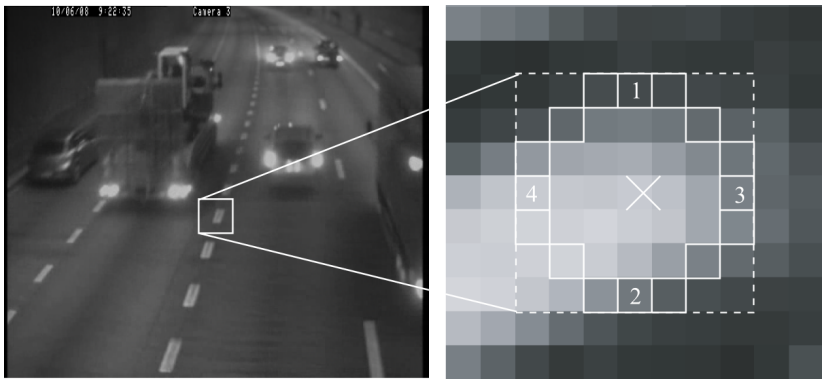


Figure 2.6: *Illustration of pixels examined by the FAST detector.* Initially pixels 1 and 2 are compared with a threshold, then 3 and 4 as well as the remaining ones at the end. The pixels are classified into dark, similar, and brighter subsets.

model. Any significant change from the background model is typically a result of a moving object. Usually, a connected component algorithm is applied to obtain connected regions corresponding to the objects. This process is called the background subtraction. Background subtraction became popular following the work of Wren *et al.* [Wren 97]. They proposed modelling the colour of each pixel by a single 3D Gaussian in a Y-U-V colour space (it can be generalized to any colour space). The model parameters, the mean and the covariance, are learned from the colour observations in a predefined number of consecutive frames. Once the background model is computed, the pixels that deviate from this model are labelled as foreground pixels. This approach was later extended by Stauffer and Grimson [Stauffer 00] to use a mixture of Gaussians (MoG) instead of a single Gaussian. Zivkovic *et al.* [Zivkovic 06] added also modelling of shadow casts and a variable number of Gaussians to model the pixel colour value distribution. This approach adapts the number of Gaussian components on-line to statistical changes to improve the processing time and make it suited for real-time applications.

One of the recent extension of the MoG method is the Visual Background Extractor (ViBe) [Barnich 09], [Barnich 11], [Zhu 12]. This is a sample-based approach for modelling the pixel distribution. Instead of using a statistical model for the unknown pixel distribution, Barnich *et al.* approximate this distribution by a set of representative pixel samples. The sample set is updated by a random process that substitutes old pixel values by new ones if there are enough pixels similar to the new value in the neighbourhood around the modelled pixel. In this way ViBe exploits spatial information and adapts to lighting changes. As shown in [Barnich 09], this method is more robust to noise than MoG. Van Droogenbroeck *et al.* made a variant of ViBe with adaptive parameters and blob filtering [Van Droogenbroeck 12]. ViBe has several parameters that adapt its performance to different conditions. There is an extension named

Pixel-Based Adaptive Segmenter (PBAS) [Hofmann 12], which adds heuristics to adaptively change the parameters.

Some other approaches to model background per pixel incorporate region-based (spatial) scene information using non-parametric kernel density estimation. One of such approaches is proposed by Elgammal and Davis [Elgammal 00]. This method can handle small camera jitter or small movements in the background. Liyuan and Maylor [Liyuan 02] fuse the texture and intensity (colour) features to perform background subtraction. Since texture does not vary greatly with illumination changes, the method is less sensitive to illumination. Methods such as [Heikkila 06] use local binary patterns as a model of local texture characteristics, which are calculated over a circular region around the pixel.

There are also background subtraction methods that represent pixel intensity variations as discrete states corresponding to the events in the environment. For instance, for tracking vehicles on a highway image pixels can be in the background state, the foreground (vehicle) state, or the shadow state (which can be both foreground and background). Rittscher *et al.* [Rittscher 00] use Hidden Markov Models (HMM) to classify small blocks of an image as belonging to one of these three states. In the context of detecting light on and off events in a room, Stenger *et al.* [Stenger 01] use HMMs for the background subtraction. The advantage of using HMMs is that certain events, which are hard to model correctly using unsupervised background modelling approaches, can be learned using training samples.

Instead of modelling the variation of individual pixels, Oliver *et al.* [Oliver 00] propose a holistic approach using the eigenspace decomposition. The background is represented by the most descriptive eigenvectors, which capture possible variations in illumination in the field of view. The foreground objects are detected by projecting the current image to the eigenspace and finding the difference between the reconstructed and actual images.

One limitation of the aforementioned approaches is that they require a relatively static background. There are methods, such as the ones proposed by Monnet *et al.* [Monnet 03], and Zhong and Sclaroff [Zhong 03], which address this limitation. Both of these methods are able to deal with time-varying background (e.g., the waves on the water, waving trees, moving clouds, escalators, etc.). These methods model the image regions as autoregressive moving average (ARMA) processes which provide a way to learn and predict the motion patterns in a scene. An ARMA process is a time series model that is made up of sums of autoregressive and moving-average components, where an autoregressive process can be described as a weighted sum of its previous values and a white noise error.

Illumination changes in the observed scene pose a big challenge to most background subtraction methods. Therefore, Grünwedel *et al.* [Grünwedel 11b] recently proposed using edge features (image gradients) for background subtraction, since edges are theoretically insensitive to illumination changes. Their method detects moving edges, i.e. the edges that belong to moving objects.

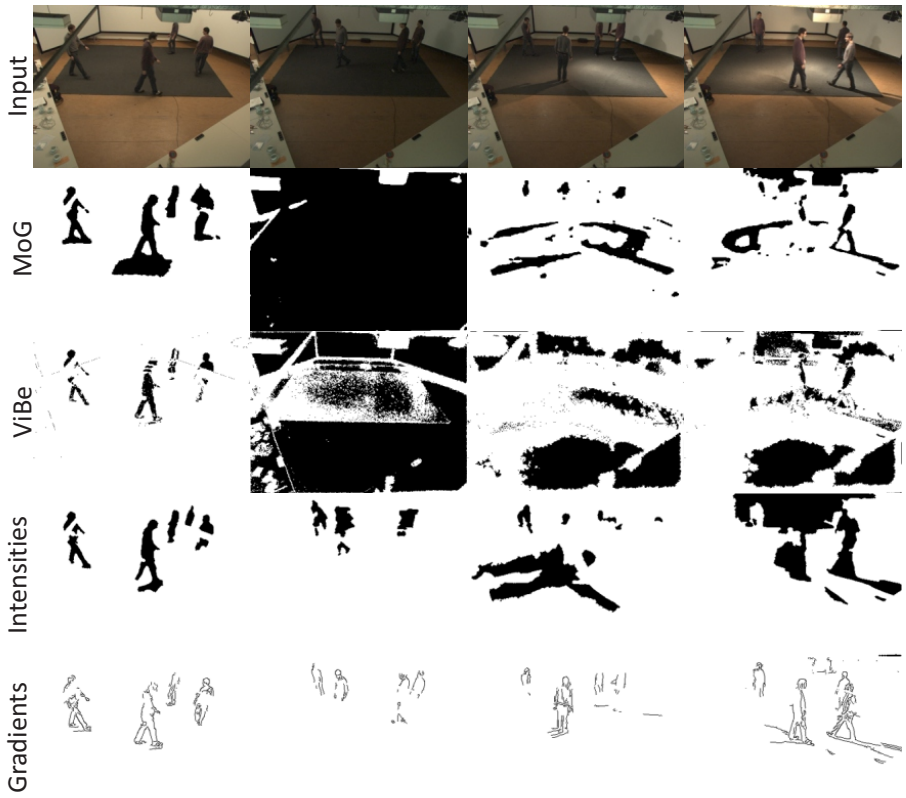


Figure 2.7: *Comparison of different foreground/background segmentation methods.* We see that the edge based method of [Grünwedel 11b] is the most robust to illumination changes and outperforms other methods.

An essential component of this method is combining a short-term and a long-term background model to include both fast and slow background changes in the model. In this way mostly only edges that belong to real moving objects remain in the foreground, and not edges of "ghost objects" (e.g. the edges that appear when an object moves and reveals the background behind it). The work of [Grünwedel 11b] also includes comparison with other background subtraction methods, namely ViBE, MoG, and short plus long term background modelling based on pixel intensities, showing that edge based approach is the most robust one for environments with intensive illumination changes, see Figure 2.7. In his PhD thesis Grünwedel S. goes further and shows that this edge based method performs better than the other methods in the context of occupancy monitoring (monitoring a presence of people in certain areas). Therefore, we use this method to detect people in our work on people tracking.

One of the biggest limitation of background subtraction as an object detection method is the requirement of stationary cameras. Camera motion usually

distorts the background models so background subtraction is not suitable for applications with moving cameras. Background subtraction in practice also provides incomplete object regions in many instances. The objects may be spilled into several regions, or there may be holes inside the object since there is no guarantee that the object features will be different from the background features. On the other hand, the advantage of background subtraction methods is their computational efficiency. Since the scope of our work is on real-time tracking in stationary camera networks, we find background subtraction a useful element as one of the cues for object detection and tracking in our work.

2.2.3 Segmentation based detectors

The aim of image segmentation algorithms is to partition the image into perceptually similar regions. In our work we used segmentation based detectors to extract high-level information about the observed scenes. For instance, in traffic surveillance applications vehicles are expected to be seen on roads, so by segmenting the scene to detect roads and adding this information to the vehicle tracker, we reduce the number of false trajectory associations. In this section, we will give an overview of fundamental segmentation techniques relevant to object tracking, and explain what methods we use and why.

Mean-shift clustering. For the image segmentation problem, Comaniciu and Meer [Comaniciu 02] propose the mean-shift approach to find clusters in the joint spatial-colour space. Given an image, a large number of hypothesized cluster centres randomly chosen from the data are used to initialize the algorithm. Then, each cluster centre is moved to the mean of the data lying inside the multidimensional ellipsoid centred on the cluster centre. The vector defined by the old and the new cluster centres is called the *mean-shift vector*. This vector is computed iteratively until the cluster centres do not change their positions. Note that some clusters may get merged during the mean-shift iterations. In Figure 2.8 there is an example of the mean-shift segmentation. Mean-shift based segmentation requires fine tuning of various parameters to obtain better segmentation. For instance, selection of the colour and spatial kernel bandwidths, and the threshold for the minimum size of the region can considerably effect the resulting segmentation. However, mean shift segmentation gives good results in segmenting roads and pavements (due to relative uniformity of their colour, brightness or texture) so we use it to establish additional cues in vehicle tracking scenarios.

Image segmentation using graph-cuts. Image segmentation can also be formulated as a graph partitioning problem, where the vertices (pixels) of a graph (image), are partitioned into disjoint subgraphs (regions), by pruning the weighted edges of the graph (see Figure 2.9). The total weight of the pruned edges between two subgraphs is called a cut. The weight is typically computed by colour, brightness, or texture similarity between the nodes. Wu and Leahy [Wu 93] used the minimum cut criterion, where the goal is to find

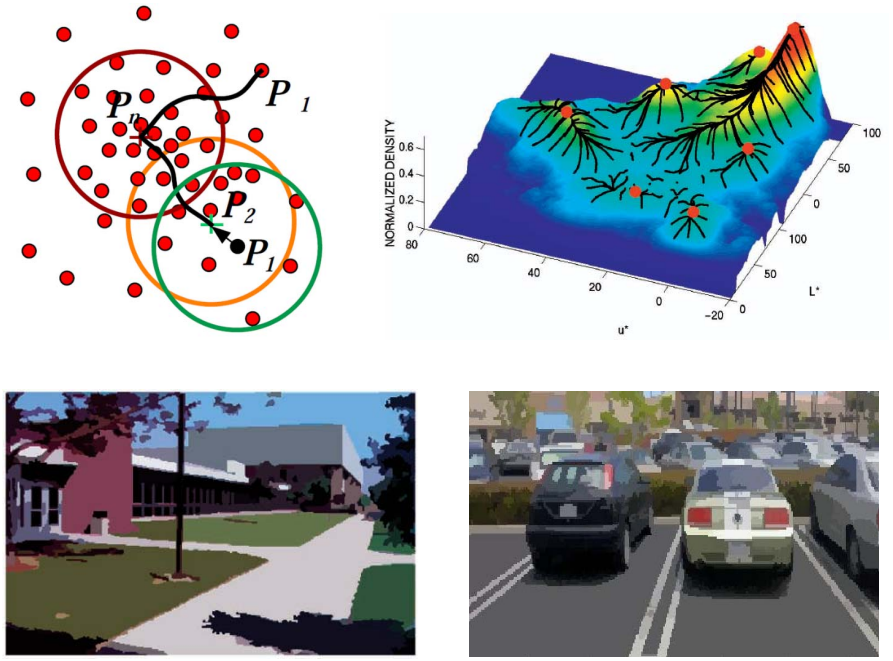


Figure 2.8: *Illustration of the mean shift procedure and result examples. Top left*: Principle of the mean shift analysis for one cluster- to find the cluster center for points P_1, repeatedly find the centroid of points inside a sphere (initially at P_1) and recentre the sphere on the centroid until the sphere is stationary (when centroid reaches the point P_n in this example); *Top right*: the mean shift procedure for multiple clusters in the L*u*v* colour space [Comaniciu 02]- gradient ascent trajectories and centroids of each cluster (marked with red dots); *Bottom*: Two result examples of outdoor scenes- we see that roads and pavements are typically very well segmented due to their relatively uniform colour.

the partitions that minimize a cut. In their approach, the weights are defined based on the colour similarity. One limitation of the minimum cut is its bias toward oversegmenting the image. This effect is due to the increase in cost of a cut with the number of edges going across the two partitioned segments.

Shi and Malik [Shi 00] proposed the normalized cut to overcome the over-segmentation problem. In their approach, the cut not only depends on the sum of edge weights in the cut, but also on the ratio of the total connection weights of nodes in each partition to all nodes of the graph. In normalized cuts-based segmentation, the solution to the generalized eigensystem for large images can be expensive in terms of processing and memory requirements. On the other hand, this method requires fewer manually selected parameters, compared to the mean-shift segmentation, which is an advantage in application where many clusters need to be extracted. In our applications the processing power and

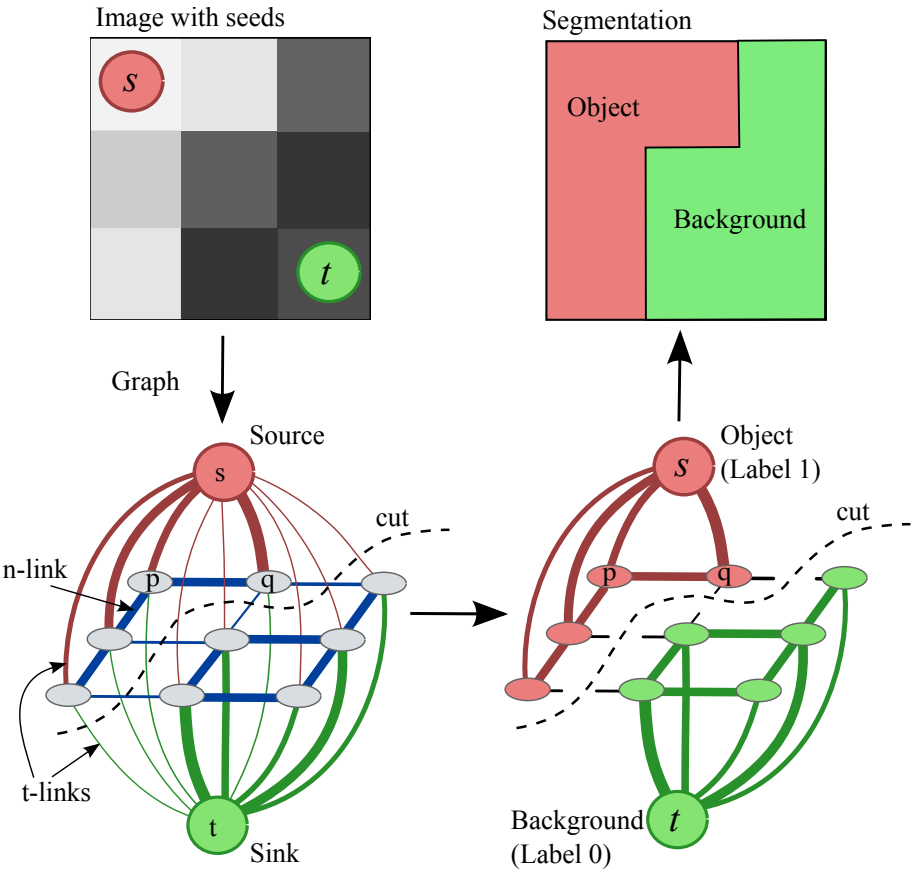


Figure 2.9: *Graph cut segmentation procedure.* Two clusters are separated by cutting the weighted edges of the graph to minimize the cost of a cut.

memory requirements are more critical so mean shift segmentation is a better choice.

Beside graph-cuts there are other inference methods with Markov Random Field (MRF) priors that could be used for image segmentation, such as methods based on random search [Grady 06], (loopy) belief propagation [Felzenszwalb 06], and others.

Active contours. In an active contour framework, object segmentation is achieved by evolving a closed contour to the object's boundary, such that the contour tightly encloses the object region. The evolution of the contour is governed by an energy functional which defines the fitness of the contour to the hypothesized object region. Typically, image gradient and image region based information is used to construct the energy terms. Image gradients provide very local information, while regional terms do not result in good contour

localization so it is best to use them in combination.

Important issues in contour-based methods are the contour initialization and representation. *Initialization* in image gradient-based approaches is typically done by placing a contour outside the object region and then the contour is shrunk until the object boundary is encountered. In region-based methods the contour can be initialized either inside or outside the object so that the contour can either expand or shrink, respectively, to fit the object boundary. However, these approaches require prior object or background knowledge. Contour *representation* can be explicit, by control points, or implicit, by level sets. In the explicit representation, the relations between the control points are defined by spline equations. In the level sets representation, the contour is represented on a spatial grid, which encodes the signed distances of the grids from the contour with opposite signs for the object and the background regions. The most important advantage of implicit representation over the explicit representation is its flexibility in allowing topology changes.

2.2.4 Classification based detectors

Object detection can also be performed by learning different object appearances automatically from a set of examples by means of supervised learning, and then finding new observations of the same categories based on the learned examples. Given a set of learning examples (templates), supervised learning methods generate a function that maps inputs to desired outputs. Inputs are typically features computed from object images, while outputs are object class labels (e.g. "a vehicle" or "not a vehicle"). In the context of object detection, the learning examples are composed of object features and their associated object class, where both are manually defined.

Selection of features has an important role in the performance of classification. It is important to use a set of features that discriminate one class from the other(s). In Chapter 3 we give an overview of the features that could be used for object classification. Other good candidates are the features mentioned in Sections 2.2.1 and 2.2.2 of this chapter. Once the features are selected, different appearances of an object can be learned through supervised learning. Some of the supervised learning approaches are neural networks [Rowley 98], adaptive boosting [Viola 03], decision trees [Grewe 95] and support vector machines [Papageorgiou 98]. These learning methods compute a hyperplane that separates one object class from the other in a high dimensional space.

The biggest drawbacks of supervised learning methods are the requirements of a large collection of samples from each object class and the need for manual labelling of the samples. A possible approach to reducing the amount of manually labelled data is to accompany co-training with supervised learning [Blum 98]. The main idea behind co-training is to train two classifiers using a small set of labelled data where the features used for each classifier are independent. After training is achieved, each classifier is used to assign unlabelled data to the training set of the other classifier. It was shown that, starting from a small set of labelled data with two sets of statistically

independent features, co-training can provide a very accurate classification rule [Blum 98]. Co-training has been successfully used to reduce the amount of manual interaction required for training in the context of Adaboost [Levin 03] and Support Vector Machines (SVM) [Kockelkorn 03]. In the remainder of this section we will give more details about adaptive boosting and support vector machines since we use these two methods in our work.

Adaptive boosting. Boosting is an iterative method of finding a very accurate classifier by combining many weak classifiers, each of which may only be moderately accurate [Freund 95], see Figure 2.10. Proper training is crucial to the performance of the Adaboost algorithm. The underlying idea is to train each of the weak classifiers to perform better than the others at classifying data using a particular feature. A strong classifier is then constructed by integrating the best trained weak classifiers. The Adaboost training process consists of several steps. First, an initial distribution of weights over the training set is constructed. Secondly, the boosting mechanism selects a base classifier that gives the least error proportional to the weights of the misclassified data. Then the weights associated with the data misclassified by the selected base classifier are increased. In this way, in the next iteration the algorithm encourages the selection of another classifier that performs better on the misclassified data.

In the context of object detection, weak classifiers can be simple operators such as thresholding scalar object features extracted from the image. In 2003, Viola *et al.* [Viola 03] used the Adaboost framework to detect pedestrians. In their approach, perceptrons were chosen as the weak classifiers which are trained on image features extracted by a combination of spatial and temporal operators. The operators for feature extraction are in the form of simple rectangular filters shown in Figure 2.11. In the temporal domain, the operators can be in the form of frame differencing, which captures motion information. This can reduce the number of false detections by enforcing object detection in the regions where the motion occurs. In the context of vehicle tracking, for vehicle detection we use the method of [Rios Cabrera 12], which is based on Adaboost.

Support vector machines. As a classifier, Support Vector Machines (SVM) are used to cluster data into two classes by finding the maximum marginal hyperplane that separates one class from the other [Boser 92], as shown in Figure 2.12. The margin of the hyperplane, which is maximized, is defined by the distance between the hyperplane and the closest data points. The data points that lie on the boundary of the margin of the hyperplane are called the support vectors. In the context of object detection, these classes correspond to the object class (positive samples) and the non-object class (negative samples). The computation of the hyperplane is carried out from manually generated training examples labelled as object and non-object. Despite being a linear classifier, SVM can also be used as a non-linear classifier by applying a kernel function to the input feature vector. The kernel function transforms the

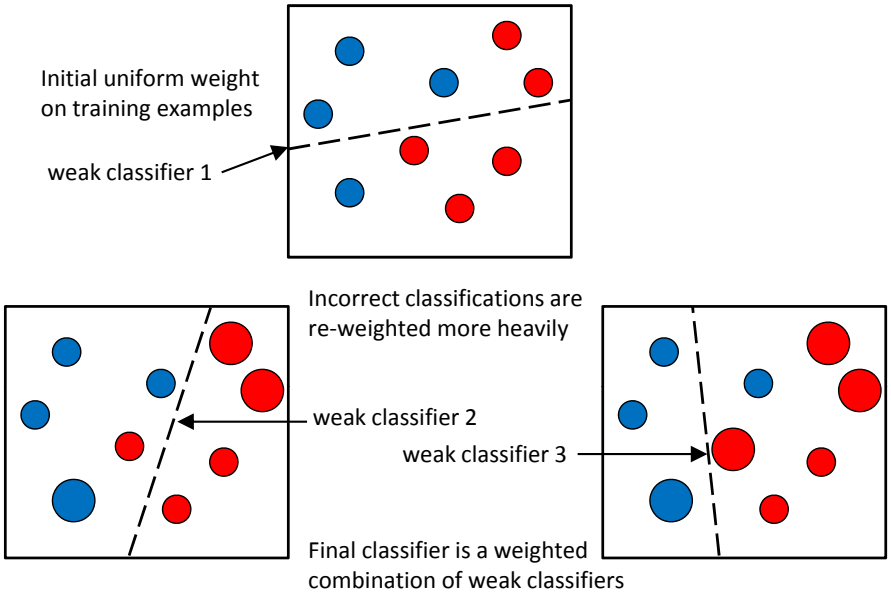


Figure 2.10: *Adaboost classification principle.* The strong classifier is obtained as a weighted liner combination of many weak classifiers. In this way it is possible to have highly accurate classification and real-time performance.

data that is not linearly separable to a higher dimensional space which is likely to be separable. The kernels used for this purpose are typically polynomial kernels or radial basis functions, for example, Gaussian kernel or a sigmoid function. However, the selection of the right kernel for the problem at hand is not easy. Once a kernel is chosen, one has to test the classification performance for a set of parameters, but there is no guarantee that the performance will remain good when new observations are introduced to the sample set. Nevertheless, SVM classifiers offer high flexibility and in many applications achieve higher accuracy than other classifiers.

2.3 Object tracking overview

The aim of an object tracker is to generate the trajectory of an object over time by locating its position in video frames. Object trackers may also provide the complete region in the image that is occupied by the object at a given time instance. The tasks of detecting the object and establishing correspondence between the object instances across frames can either be performed separately or jointly. In the first case, possible object regions in every frame are obtained

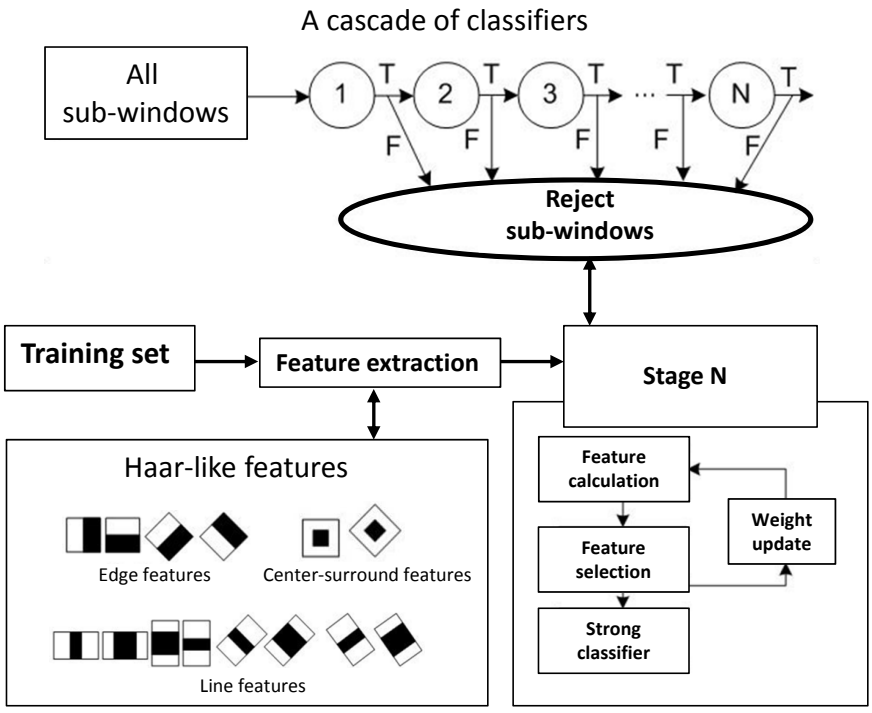


Figure 2.11: *Adaboost classification and features often used for object detection.* Classifiers are organized in a cascade so that each classifier specializes for one feature and at each stage many false hypotheses are rejected. Only the hypotheses classified as true pass to the next classifier.

by means of an object detection algorithm, and then the tracker associates objects across frames. This approach is usually referred to as tracking-by-detection. For this purpose the detection methods explained in Section 2.2 can be used. In the case when the object region and correspondence are jointly estimated, this is typically done by iteratively updating object location and region information obtained from previous frames. In both approaches, the objects are represented using the shape and/or appearance models described previously in this chapter, in Section 2.1.

The selected model that represents object shape limits the type of motion or deformation the object can undergo. For example, if an object is represented as a point, then only a translational model can be used. In the case where a geometric shape representation like a polygon or a cuboid is used for the object, parametric motion models like affine or projective transformations are appropriate. These representations can approximate the motion of rigid objects in the scene. For a non-rigid object, its silhouette or contour is the most

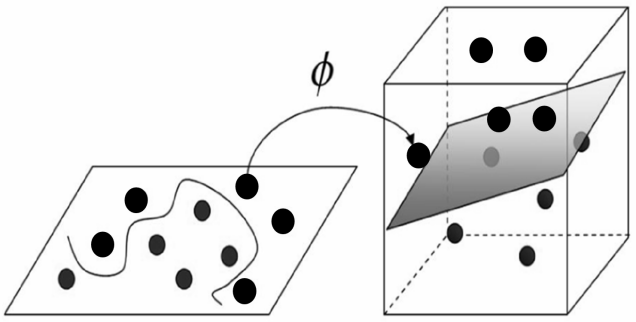


Figure 2.12: *Support vector machines classification example for two classes. Left*: Input space in which data is not linearly separable; *Right*: Higher dimensional feature space in which data becomes linearly separable. The kernel function ϕ transforms the input space into feature space.

descriptive representation and both parametric and non-parametric models can be used to specify their motion. However, if there is no interest to determine motion of specific parts of a non-rigid object or there is no a close up view of the object, a non-rigid object can as well be represented by a rectangle, a polygon or a cuboid.

We now briefly introduce the main categories of tracking methods, followed by a detailed section on the two categories we use extensively in our work.

Point tracking. Objects detected in successive frames are represented by points. The association of the points is based on the previous object state, which can include object position and motion (see Figure 2.13a). This approach usually requires an external detection of the objects in every frame.

Kernel tracking. Kernel refers to the object shape and appearance. For example, the kernel can be a rectangular template or an elliptical shape with an associated histogram. Objects are tracked by computing the motion of the kernel in consecutive frames (see Figure 2.13b). This motion is usually in the form of a parametric transformation such as translation, rotation, and affine.

Silhouette tracking. Tracking is performed by estimating the object region in each frame. Silhouette tracking methods use the information encoded inside the object region. This information can be in the form of appearance density and shape models which are often in the form of edge maps. Given the object models, silhouettes are tracked by either shape matching or contour evolution (see Figures 2.13c and 2.13d). Both of these methods can essentially be considered as object segmentation applied in the temporal domain using the priors generated from the previous frames.

In our work we focus on point and kernel tracking approaches since silhouette tracking approaches typically require a close up view of the object and are less computationally efficient. Therefore, in the following sections we explain point and kernel tracking in more detail.

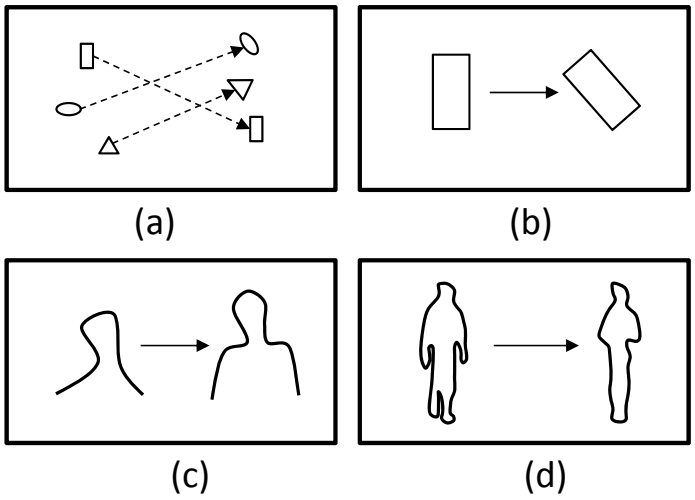


Figure 2.13: Different tracking approaches: a) Multipoint correspondence; (b) Parametric transformation of a rectangular patch; c, d) Two examples of contour evolution.

2.3.1 Point tracking

Point tracking can be formulated as associating detected objects represented by points across frames. Point correspondence is a complicated problem, especially in the presence of occlusions, misdetections, entries, and exits of objects. Overall, there are two broad categories of point correspondence methods: deterministic and statistical methods. Deterministic methods typically use qualitative motion heuristics [Veenman 01] to constrain the correspondence problem. On the other hand, probabilistic methods establish correspondence by explicitly taking into account the object measurement and uncertainties.

2.3.1.1 Deterministic methods for correspondence

Deterministic methods for finity point correspondences define a cost of associating each object in frame $t-1$ to a single object in frame t using a set of motion constraints. Minimization of the correspondence cost is formulated as a combinatorial optimization problem. A solution consists of one-to-one correspondences among all possible associations. Such a solution can be obtained by optimal assignment methods, such as Hungarian or the Kuhn-Munkres algorithm, [Kuhn 55] and [Munkres 57], or by greedy search methods. The correspondence cost is usually defined by using a combination of the following constraints.

- *Proximity* assumes the location of the object does not change notably between successive frames (see Figure 2.14a).
- *Maximum velocity* defines an upper bound on the object velocity and

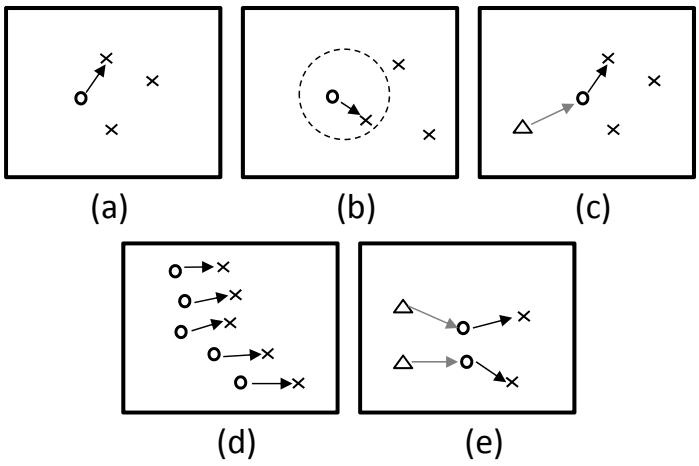


Figure 2.14: Different motion constraints: a) Proximity; b) Maximum velocity (r denotes radius); c) Small velocity change; d) Common motion; e) Rigidity constraints.

limits the possible correspondences to the circular neighbourhood around the object (see Figure 2.14b).

- *Small velocity change (smooth motion)* assumes the direction and speed of the object does not change drastically between successive frames (see Figure 2.14c).
- *Common motion* constrains the velocity of objects in a small neighbourhood to be similar. This constraint is suitable for object represented by multiple points (see Figure 2.14d).
- *Rigidity* assumes that objects in the 3D world are rigid so the distance between any to points in the actual object will remain unchanged (see Figure 2.14e).
- *Proximity uniformity* is a combination of the proximity and the small velocity change constraints.

Note that these constraints are not specific to the deterministic methods. They can also be used in the context of point tracking using statistical (probabilistic) methods.

Here we present a sample of different methods proposed in the literature in the category of deterministic point trackers. Sethi and Jain [Sethi 87] solve the correspondence by a greedy approach based on the proximity and rigidity constraints. Their algorithm is initialized by the nearest neighbour criterion and uses the information from two consecutive frames for tracking. The correspondences are exchanged iteratively to minimize the cost. A modified version of the same algorithm which computes the correspondences in the backward

direction (from the last frame to the first frame) in addition to the forward direction is also analysed. This method cannot handle occlusions, entries, or exits. Salari and Sethi [Salari 90] handle these problems by first establishing correspondence for the detected points and then extending the tracking of the missing objects by adding a number of hypothetical points. Rangarajan and Shah [Rangarajan 91] propose a greedy approach, which is constrained by proximal uniformity. Optical flow in the first two frames is used to find initial correspondences. This method does not address entry and exit of objects. If the number of detected points decrease, the method assumes occlusion or misdetection. Occlusion is handled by establishing the correspondence for the detected objects in the current frame. For the remaining objects, position is predicted based on a constant velocity assumption.

In the work by Intille *et al.* [Intille 97], which uses a slightly modified version of [Rangarajan 91] for matching object centroids, the objects are detected by using background subtraction. The authors explicitly handle the change in the number of objects by examining specific regions in the image, for example, a door, to detect entries/exits before computing the correspondence. Veenman *et al.* [Veenman 01] extend the work of [Sethi 87], and [Rangarajan 91] by introducing the common motion constraint for correspondence. The common motion constraint provides a strong constraint for coherent tracking of points that lie on the same object. However, it is not suitable for points lying on isolated objects moving in different directions. The algorithm is initialized by generating the initial tracks using a two-pass algorithm, and the cost function is minimized by Hungarian assignment algorithm [Munkres 57] in two consecutive frames. This approach can handle occlusion and misdetection errors, however, it is assumed that the number of objects are the same throughout the sequence, that is, no object enters or exits.

Shafique and Shah [Shafique 03] propose a multi-frame approach to preserve temporal coherency of the speed and position. They formulate the correspondence problem as a graph theoretic problem. Multiple frame correspondence relates to finding the best unique path for each point. For misdetected or occluded objects, the path will consist of missing positions in corresponding frames. The correspondence is then established by a greedy algorithm. They use a window of frames during point correspondence to handle occlusions whose durations are shorter than the temporal window used to perform matching.

2.3.1.2 Statistical methods for correspondence

Measurements obtained from video sensors always contain noise. Moreover, the object motion can undergo random perturbations. For instance, when people walk they can suddenly change their moving direction or stop walking. Statistical correspondence methods solve these tracking problems by taking the measurement and the uncertainties into account during the object state estimation. The statistical correspondence methods use the state space approach to model the object properties such as position, size, velocity, and acceleration. Measurements usually consist of the object position in the image, which is obtained by

an object detection mechanism. In this section we will give an overview of the state estimation methods in the context of point tracking. However, it should be noted that these methods can be used in general to estimate the state of any time varying system. For example, these methods have extensively been used for tracking contours [Isard 98], activity recognition [Vaswani 03], object identification [Zhou 03], and structure from motion [Matthies 89]. We also extensively use them in our work, both for single and multi-camera tracking.

Consider a moving object in the scene. The information representing the object, for example, location, is defined by a sequence of states $\mathbf{x}_t : t = 1, 2, ...$ The change in state over time is governed by the dynamic equation, $\mathbf{x}_t = f(\mathbf{x}_{t-1}) + \mathbf{w}_t$, where $\mathbf{w}_t : t = 1, 2, ...$ is white noise. The relationship between the measurement and the state is specified by the measurement equation $\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{n}_t)$, where $\mathbf{n}_t$ is white noise independent of $\mathbf{w}_t$. The objective of tracking is to estimate the state $\mathbf{x}_t$ given all the measurements up to that moment or, equivalently, to construct the probability density function $p(\mathbf{x}_t|\mathbf{z}_{1,...,t})$. A theoretically optimal solution is provided by a recursive Bayesian filter which solves the problem in two steps. The prediction step uses a dynamic equation and the already computed probability density function of the state at time $t-1$ to derive the prior probability density function of the current state, $p(\mathbf{x}_t|\mathbf{z}_{1,...,t-1})$. Then, the correction step employs the likelihood function $p(\mathbf{z}_t|\mathbf{x}_t)$ of the current measurement to compute the posterior probability density function $p(\mathbf{x}_t|\mathbf{z}_{1,...,t})$. There are several filtering techniques typically used for this purpose.

Kalman filtering. Kalman filtering (KF) was invented in the 1950*s* by Rudolph Emil Kalman [Kalman 60], as a technique for filtering and prediction in linear systems. At time t, the state is represented by the the mean μ_t and the covariance σ_t. It is assumed that the state has a Gaussian distribution and that state transitions and measurements are linear with added Gaussian noise. These limitations are rarely fulfilled in practice, but in many applications they can be good approximations if it is possible to keep the state and measurement uncertainties very small.

Figure 2.15 illustrates the Kalman filter algorithm for a simplistic one-dimensional localization scenario. Suppose an object moves along the horizontal axis in each diagram in Figure 2.15. Let the prior over the object location be given by the normal distribution shown in Figure 2.15a. Various sensors (e.g. a GPS system, video cameras, etc.) are used to get the information about the object's locations, and those return a measurement that is centred at the peak of the bold Gaussian in Figure 2.15b. This bold Gaussian illustrates this measurement: its peak is the value predicted by the sensors, and its width (variance) corresponds to the uncertainty in the measurement. Combining the prior with the measurement by the Kalman filter algorithm, yields the bold Gaussian in Figure 2.15c. This belief's mean lies between the two original means.

Next, assume the object moves towards the right, as represented by the

Gaussian shown in bold in Figure 2.15d. This Gaussian is shifted by the amount the object moved, and it is also wider due to a higher uncertainty in the location. Next, the robot receives a second measurement illustrated by the bold Gaussian in Figure 2.15e, which leads to the posterior shown in bold in Figure 2.15f. As this example illustrates, the Kalman filter alternates a measurement update step, in which sensor data is integrated into the present belief, with a prediction step (or control update step), which modifies the belief in accordance to an action. The update step decreases and the prediction step increases uncertainty in the belief in object's location.

Extended Kalman filtering. Extended Kalman filtering (EKF) overcomes the linearity limitations of KF by taking the assumption that the next state probability and the measurement probabilities are governed by non-linear functions. These non-linear functions are then typically approximated using linear Taylor expansions. In this way EKF represents the belief by a multivariate Gaussian distribution, opposed to a single variate in KF. This also enables EKF to be computationally efficient. There are also extensions of EKF that enable multi-modal representations of the posterior belief by a mixture of Gaussians (very useful in cases when the object can be in multiple states with considerable probabilities, but the arithmetic mean of these hypotheses is not a likely state). These extensions are typically called *multi-hypotheses extended Kalman filter*. In the case of high non-linearity of prediction and update functions, it is possible to sample a set of points around the mean and propagate them through the non-linear functions to compute the mean and covariance of the state estimate. This technique is called *unscented Kalman filter*.

In our work and in this thesis we extensively use Kalman filtering methods. As the state of a tracked object (target) at time instance t we use as a six-dimensional vector $\mathbf{x}_t = (x_t, y_t, \dot{x}_t, \dot{y}_t, w_t, h_t)$, which contains the position of the target along the x and y image axes (x_t, y_t), its apparent velocity ($\dot{x}_t$ and $\dot{y}_t$), and apparent size, i.e. the width and height of its bounding box (w_t and h_t). We combine multiple cues (measurements) to improve quality and robustness of state estimations. Depending on the application domain, vehicle or people tracking, and camera views, close-up or wide view, we set differently the parameters of the filter. Between consecutive video frames (observations) the state of people can change more abruptly than the state of vehicles, so we use higher Kalman gain for people tracking. The Kalman gain is a function of the relative certainty of the measurements and current state estimate, and can be tuned to achieve particular performance. With a high gain, the filter places more weight on the measurements, and thus follows them more closely. With a low gain, the filter follows the model predictions more closely, smoothing out noise but decreasing the responsiveness. At the extremes, a gain of one causes the filter to ignore the state estimate entirely, while a gain of zero causes the measurements to be ignored. In this way, by using high gain values for people tracking we rely more on camera observations and incorporate new measurements faster than for vehicle tracking. The optimal value of Kalman gain we determine experimentally.

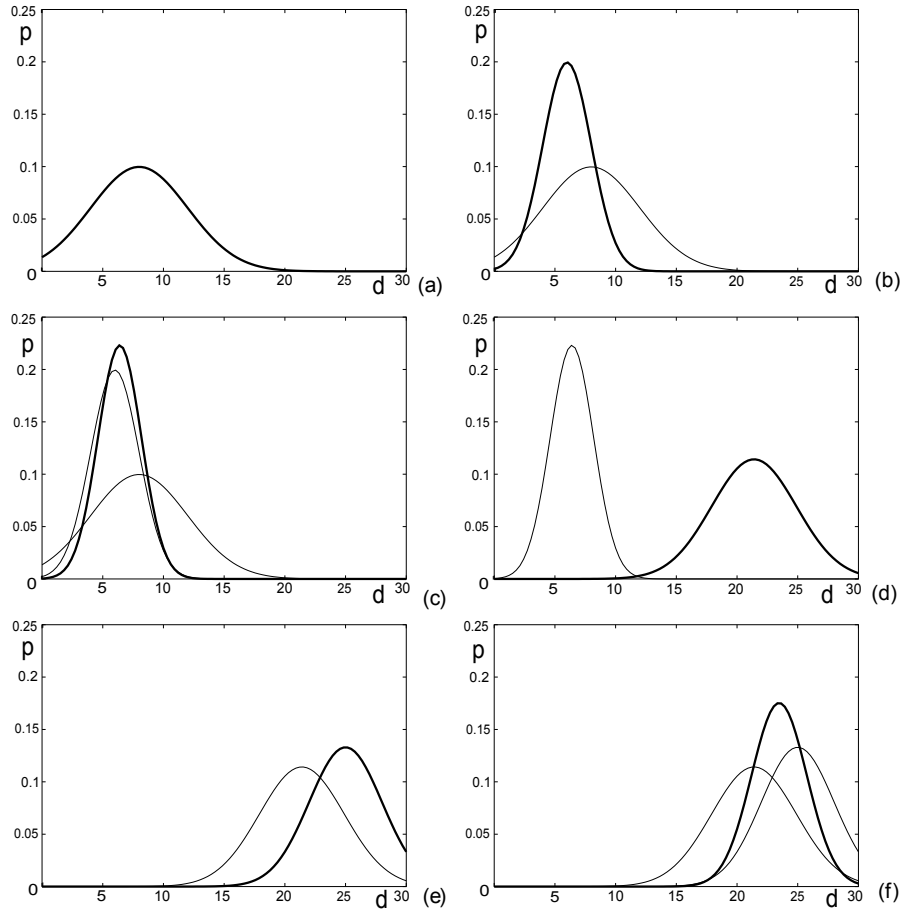


Figure 2.15: Illustration of a Kalman filter for a simplistic one-dimensional robot localization scenario [Thrun 05]: the vertical axis (p) represents state (location) probability, while the horizontal axis (d) represents the location of the robot. a) Initial belief; b) Measurement (in bold) with the associated uncertainty; c) Belief after integrating the measurement into the belief using the Kalman filter algorithm; d) belief after motion to the right (which introduces uncertainty); e) A new measurement with associated uncertainty; f) the resulting belief.

Histogram filtering. Histogram filters (HF) decompose the continuous state space into finitely many regions, and represent the cumulative posterior for each region by a single probability value. When applied to discrete spaces, such filters are known as *discrete Bayesian filters*. HF are well-suited to represent complex multi-modal beliefs. For this reason, they are often the method of choice when a tracker has to cope with phases of global uncertainty, and when it faces hard data association problems that yield separate, distinct hypotheses.

Particle filtering. Particle filtering (PF) is an alternative non-parametric implementation of the Bayesian filter. Just like histogram filters, particle filters approximate the posterior by a finite number of parameters. However, they differ in the way these parameters are generated, and in which they populate the state space. The key idea of the particle filter is to represent the posterior belief by a set of random state samples (particles) drawn from this posterior, instead of representing the belief by a parametric form. Such a representation is approximate, but it is non-parametric, and therefore can represent a much broader space of distributions than parametric methods. PF is a popular substitute for the KF in presence of non-Gaussianity of the noise statistics and non-linearity of the relationships between consecutive states and between state and measurements. PF can use multi-modal likelihood functions and propagate multi-modal posterior distributions like those occurring in case of temporary occlusions and background clutter. However, the number of samples required and therefore the complexity of the algorithm grows exponentially with the dimensionality of the estimated state space. This limits the application of the PF variety where samples are drawn from the prior to relatively small tracking problems.

To better and practically understand how particle filter works, let us show an example of algorithm iteration with 10 particles, which represent 10 possible filtered values (Figure 2.16):

1. Starting step - there are 10 possible filtered values, all with the same weight;

2. Importance weight step - by exploiting state estimate probability obtained from measurements, the algorithm assigns a weight at each filtered value;

3. Re-sampling step - values with high weight are spread over different values with the same weight, while low weight values are discarded;

4. Sampling/prediction step - filtered particles are randomly perturbed.

Note that when tracking multiple objects using these filters, one needs to deterministically associate the most likely measurement for a particular object to that object's state, i.e. the correspondence problem needs to be solved before these filters can be applied. The simplest method to perform correspondence is to use the nearest neighbour approach. We exploit it also in our work. However, if the objects are close to each other, then there is always a chance that the correspondence is incorrect. An incorrectly associated measurement can cause the filter to fail to converge. There are several statistical data association techniques to tackle this problem. A detailed review of these techniques can be found in the survey by Cox [1993]. *Joint probability data*

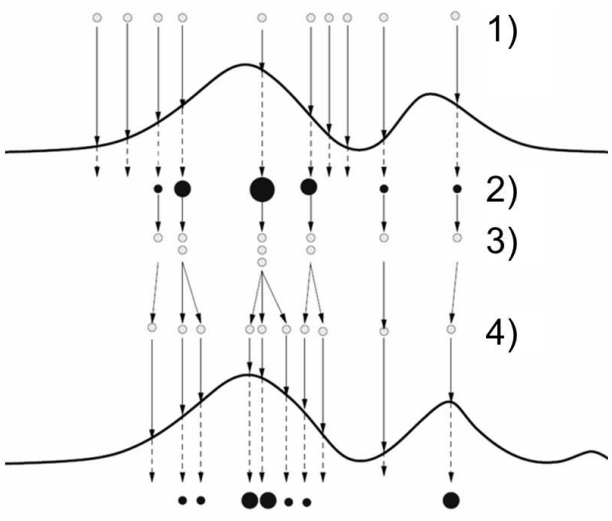


Figure 2.16: An example of particle filter iteration: 1) Starting step; 2) Importance weight step; 3) Re-sampling step; 4) Sampling/prediction step.

association filtering and *multiple hypotheses tracking* are two widely used techniques for data association. Here we give a brief description of these techniques.

Joint probability data association filter (JPADF). If we assume there are N tracks (tracked objects) and M, $M \neq N$, measurements at time instance t, the problems is to assign these measurements to the tracks. The JPDAF associates all measurements with each track, assigning a weight factor to each of the associations and assuming that the number of tracks will remain constant over time. The major limitation of the JPDAF algorithm is its inability to handle new objects entering the field of view or already tracked objects exiting the observed area. Since the JPDAF algorithm performs data association of a fixed number of objects tracked over two frames, serious errors can arise if there is a change in the number of objects.

Multiple hypotheses tracking (MHT). If motion correspondence is established using only consecutive frames, there is always a chance of an incorrect correspondence. Better tracking results can be obtained if the correspondence decision is deferred until several frames have been examined. The MHT algorithm maintains several correspondence hypotheses for each object at each time frame [Reid 1979]. The final track of the object is the most likely set of correspondences over the time period of its observation. The algorithm has the ability to create new tracks for objects entering the field of view and terminate tracks for exiting objects. It can also handle occlusions by continuation of a track even if some of the measurements from an object are missing. MHT is an

iterative algorithm. An iteration begins with a set of current track hypotheses. Each hypothesis is a collection of disjoint tracks. For each hypothesis, a prediction of each object's position in the next frame is made. The predictions are then compared with actual measurements by evaluating a distance measure. A set of correspondences (associations) are established for each hypothesis based on the distance measure, which introduces new hypotheses for the next iteration. Each new hypothesis represents a new set of tracks based on the current measurements. Note that each measurement can belong to a new object entering the field of view, a previously tracked object, or a spurious measurement. Moreover, a measurement may not be assigned to an object because the object may have exited the field of view, or a measurement corresponding to an object may not be obtained. The latter happens because either the object is occluded or it is not detected due to noise. Note that MHT makes associations in a deterministic sense and exhaustively enumerates all possible associations. To reduce the computational load, Streit and Luginbuhl [Streit 94] proposed a *probabilistic MHT* in which the associations are considered to be statistically independent random variables and thus there is no requirement for exhaustive enumeration of associations.

2.3.1.3 Point tracker evaluation

Point tracking methods can be evaluated on the basis of whether they generate correct point trajectories. Given a ground truth, the performance can be evaluated by computing precision and recall measures. In the context of point tracking, precision (p) and recall (r) measures can be defined as:

$$p = \frac{N_c}{N_e}, \tag{2.1}$$

where N_c and N_e are respectively the number of correct and established correspondences, and

$$r = \frac{N_c}{N_{gt}}, \tag{2.2}$$

with N_c and N_{gt} being the number of correct and ground truth correspondences, respectively.

Additionally, a qualitative comparison of object trackers can be made based on their ability to deal with entries of new objects and exits of existing objects, handle the missing observations (occlusions), and provide an optimal solution to the cost function minimization problem used for establishing correspondence. To handle missing or noisy observations, it is often necessary to use motion based constraints as explained in Section 2.3.1.1.

In our work and in this thesis we extensively use statistical point tracking methods. Therefore, in Chapters 4 and 6 we will explain the methods we use in more detail. We combine multiple cues to improve quality and robustness of measurements. We also pay significant attention to computational and data efficiency.

2.3.2 Kernel tracking

Kernel tracking is typically performed when the object is represented by a primitive object region (e.g. a rectangle or an ellipse). The object motion is computed in the form of parametric motion (translation, conformal, affine, etc.) or the dense flow field computed in subsequent frames. These algorithms differ in terms of the appearance representation used, the number of objects tracked, and the method used to estimate the object motion. We divide these tracking methods into two subcategories based on the used appearance representation: templates and density-based appearance models, and multi-view appearance models.

2.3.2.1 Tracking using templates and appearance models

Templates and density-based appearance models (see Section 2.1.2) are widely used because of their relative simplicity and low computational cost.

The most common approach in this category is *template matching*. Template matching is a brute force method of searching the image for a region similar to the object template defined in one of the previous frames. The position of the template in the current image is computed by a similarity measure, for example, cross correlation. Usually image intensity or colour features are used to form the templates. Since image intensity is very sensitive to illumination changes, image gradients [Birchfield 1998] can also be used as features. A limitation of template matching is its high computation cost due to the brute force search. To reduce the computational cost, one of typical solutions is to limit the object search to the vicinity of its previous position. Also, more efficient algorithms for template matching have been proposed [Schweitzer 02].

Note that instead of templates, other object representations can be used as well. For instance, colour histograms or mixture models can be computed by using the appearance of pixels inside the rectangular or ellipsoidal regions. Fieguth and Terzopoulos [Fieguth 97] generate object models by finding the mean colour of the pixels inside the rectangular object region. To reduce computational complexity, they search the object in eight neighbouring locations. The similarity between the object model and the hypothesized position is computed by evaluating the ratio between the colour means. The position which provides the highest ratio is selected as the current object location. Comaniciu and Meer [Comaniciu 03] use a weighted histogram computed from a circular region to represent the object. For histogram generation, the authors use a weighting scheme defined by a spatial kernel that gives higher weights to the pixels closer to the object center. Instead of performing a brute force search for locating the object, they use the mean-shift method (Section 2.2.3). The mean-shift tracker maximizes the appearance similarity iteratively by comparing the histograms of the object and the window around the hypothesized object location. However, a disadvantage of the mean-shift tracking is its sensitivity to initialization. Birchfield and Rangarajan [Birchfield 05] introduced the concept of a spatiogram, which is a generalization of a histogram that includes poten-

tially higher order moments. They show how to use spatiograms in kernel-based trackers, deriving a mean-shift procedure in which individual pixels vote not only for the amount of shift but also for its direction. They used both mean shift and exhaustive search and showed that spatiograms improve results in comparison with histograms.

There are methods, such as [Jepson 03], that propose classifying object features into stable, transient and noise components. The stable component identifies the most reliable appearance for motion estimation, that is, the regions of the object whose appearance does not quickly change over time. The transient component identifies the quickly changing features, while the noise component handles the outliers in the object appearance. The parameters of these three-component mixture can be learned online, for instance by the *expectation maximization (EM)* algorithm. The advantage of learning stable and transient features is that one can give more weight to stable features for tracking. For example, if the face of a person who is talking is being tracked, then the forehead or nose region can give a better match to the face in the next frame as opposed to the mouth of the person.

Another approach to track a region defined by a primitive shape is to compute its translation by use of an optical flow method. Optical flow methods are used for generating dense flow fields by computing the flow vector of each pixel under the brightness constancy constraint. This computation is always carried out in the neighbourhood of the pixel, either algebraically [Lucas 81] or geometrically [Schunk 86]. Extending optical flow methods to compute the translation of a rectangular region is trivial. Shi and Tomasi [Shi 94] proposed the KLT tracker which iteratively computes the translation of a region (typically $n \times n$ patch) centered on an interest point (for interest point detection see Section 2.2.1). If the sum of square difference between the current patch and the projected patch is small, they continue tracking the feature, otherwise the feature is eliminated.

In our work we extensively use template matching with selection of stable features and highly informative templates. We also use optical flow, both as a tracking cue and as a comparison to better understand the value of our proposed methods.

2.3.2.2 Tracking using multi-view appearance models

In the previous tracking methods, the appearance models (e.g. histograms, templates, etc.) are usually generated online. Thus, these models represent the information gathered about the object from the most recent observations. The objects may appear different in different views, and if the object view changes dramatically during tracking, the appearance model may no longer be valid, and the object track might be lost. To overcome this problem, in some applications different views of the object can be learned offline and used as templates for tracking. In our work we strongly refer to these multi-view appearance methods and frequently compare the results of our proposed methods with these ones. Here we give a brief overview of the most common methods.

Black and Jepson [Black 98] proposed a subspace-based approach, eigenspace, to compute the affine transformation from the current image of the object to the image reconstructed using eigenvectors. First, a subspace representation of the appearance of an object is built using Principal Component Analysis (PCA). Then the transformation from the image to the eigenspace is computed by minimizing the so-called subspace constancy equation, which evaluates the difference between the image reconstructed using the eigenvectors and the input image. Note that the use of eigenspace for similarity computation is a useful alternative to standard template matching techniques, such as normalized correlation. The eigenspace-based similarity computation is equivalent to matching with a linear combination of eigen templates. This allows for distortions in the templates, for example, distortion caused by illumination changes in images.

In a similar vein, Avidan [Avidan 01] used a Support Vector Machine (SVM) classifier for tracking. SVM is a general classification scheme that, given a set of positive and negative training examples, finds the best separating hyperplane between the two classes (see Section 2.2.4 for more details on SVM). During testing, the SVM gives a score to the test data indicating the degree of membership of the test data to the positive class. For SVM-based trackers, the positive examples consist of the images of the object to be tracked, and the negative examples consist of all things that are not to be tracked. Generally, negative examples consist of background regions that could be confused with the object. Avidan's tracking method, instead of minimizing the intensity difference of a template from the image regions, maximizes the SVM classification score over image regions in order to estimate the position of the object. One advantage of this approach is that knowledge about background objects (negative examples that are not to be tracked) is explicitly incorporated in the tracker. Instead of the SVM classifier, other classifiers can be used as well, e.g. Adaboost [Rios Cabrera 12].

2.3.2.3 Kernel tracker evaluation

The main goal of the kernel trackers is to estimate object motion. With the object representation based on primitive geometric models, the computed motion implicitly defines the object region as well as the object orientation. Depending on the context in which these trackers are being used, their evaluation can be performed in the following ways.

In the case of analysing the object behaviour based on the object trajectory, only the motion is adequate. In this case, the evaluation can be performed by computing a distance measure between the estimated and actual motion parameters. However, to identify an object, the region assigned to it is also important, not only the motion. Therefore, in this case the tracker's performance is evaluated by computing the precision and the recall measures. Both of these measures are defined in terms of the intersection of the hypothesized and correct object region. In particular, precision is the ratio of the intersection to the hypothesized regions. Recall is the ratio of the intersection to the ground truth

region. A qualitative comparison of kernel trackers can also be obtained, based on tracking single or multiple objects, ability to handle occlusion, requirement of training, type of motion model, and requirement of a manual initialization.

The use of primitive geometric shapes to represent objects is very common due to the real-time applicability of the state-of-the-art methods. Because of the rigidity constraint, tracking methods in this category compute parametric motion of the object. This motion is usually in the form of translation, conformal affine, affine, or projective. Motion of the object can be estimated by maximizing the object appearance similarity between the previous and current frame. The estimation process can be in the form of a brute force search, or by using gradient ascent (descent)-based maximization (minimization) process. Object trackers, based on the gradient ascent (descent) approach, require that at least some part of the object is visible inside the chosen geometric shape whose location is defined by the previous object position. To eliminate such requirements, a possible approach is to use Kalman filtering or particle filtering discussed in the context of point trackers to predict the location of the object in the next frame. Given the object state defined in terms of velocity and acceleration of the object centroid these filters will estimate the position of the object centroid such that the likelihood of observing part of the object inside the kernel is increased [Comaniciu 03]. This requirement can also be met by performing global motion compensation, assuming that the objects are further from the camera and camera motion can be estimated by affine or projective transformation [Yilmaz 03].

One of the limitations of primitive geometric shapes for object representation is that parts of the objects may be left outside of the defined shape while parts of the background may reside inside it. This phenomena can be observed for both the rigid objects (when the object pose changes) and non-rigid objects (when local motion results in changes in object appearance). In such cases, the object motion estimated by maximizing model similarity may not be correct. To overcome this limitation, one approach is to force the kernel to reside inside the object rather than encapsulating the complete shape, but this can often lead to loss of valuable information. Another approach is to regularly update the appearance model and assign weights to the components of the model based on the conditional probability of observed components.

2.3.2.4 Tracking objects in real-world scenarios

In this section, we discuss issues that arise in tracking objects in realistic scenarios. These include locating objects as they undergo occlusion and keeping unique tracks of objects as they are viewed through multiple cameras. Here we give only a brief overview of these issues, while more details are given in Chapters 4, 5 and 6.

Occlusion can be classified into three categories: self occlusion, inter-object occlusion, and occlusion by the background scene structure. Self occlusion occurs when one part of the object occludes another. This situation most frequently arises while tracking articulated objects. Inter-object occlusion occurs

when two objects being tracked occlude each other. Similarly, occlusion by the background occurs when a structure in the background occludes the tracked objects.

Generally, for *inter-object occlusion*, the multi-object trackers like [Maccormick 00] and [Elgammal 02] can exploit the knowledge of the position and the appearance of the occluder and occludee to detect and resolve occlusion. *Partial occlusion* of an object by a scene structure is hard to detect since it is difficult to differentiate between the object changing its shape and the object getting occluded, but in our work we did some effort in this direction as well. A common approach to handle *complete occlusion* during tracking is to model the object motion by linear dynamic models or by non-linear dynamics and, in the case of occlusion, to keep on predicting the object location until the object reappears. For example, a linear velocity model and a Kalman filter is a common approach for estimating the location and motion of objects. A non-linear dynamic model is used in [Isard 01] and a particle filter employed for state estimation. Some works have also utilized other features to resolve occlusion, for example, silhouette projections [Haritaoglu 00] (to locate persons' heads during partial occlusion), and optical flow [Munkres 01] (assuming that two objects move in opposite directions).

Free-form object contour trackers employ a different occlusion resolution approach. These methods usually address occlusion by using shape priors which are either built ahead of time [Cremers 02] or built online [Yilmaz 04]. In particular, Cremers *et al.* [Cremers 02] built a shape model from subspace analysis (PCA) of possible object shapes to fill in missing contour parts. [Yilmaz 04] built online shape priors using a mixture model based on the level set contour representation. Their approach is able to handle complete object occlusion.

The probability of occlusion can be reduced by an appropriate selection of camera positions. For instance, when a birds-eye view of the scene is available, occlusions between objects on the ground do not occur. Multiple cameras viewing the same scene can also be used to resolve object occlusions during tracking [Dockstader 01a], [Mittal 03].

The need for using multiple cameras for tracking arises for two reasons. The first reason is the use of depth information for tracking and occlusion handling. The second reason for using multiple cameras is to increase the area under view since it is not possible for a single camera to observe large areas because of a finite sensor field-of-view. An important issue in using multiple cameras is the relationship between the different camera views which can be manually defined [Collins 01], [Cai 99] or computed automatically [Lee 00], [Khan 03] from the observations of the objects moving in the scene. For the tracking algorithms in these multi-camera environments high computational cost is another concern. It is necessary to optimize used features, and efficiently and accurately fuse information from different views.

In many situations, it is not possible to have overlapping camera views due to limited resources or large areas of interest. Methods for tracking in such a scenario inherently have to deal with sparse object observations due to

non-overlapping views. Therefore some assumptions have to be made about the object speed and the path in order to obtain the correspondences across cameras [Huang 97], [Kettnaker 99], [Javed 03]. The performance of these algorithms depends greatly on how much the objects follow the established paths and expected time intervals across cameras. For scenarios in which spatio-temporal constraints cannot be used, for example, objects moving arbitrarily or with significant freedom in the non-overlap region, it is necessary to use tracking-by-recognition approach. This approach uses the appearance and the shape of the object to recognize it when it reappears in a camera view.

2.4 Conclusion

In this chapter we presented an extensive overview of object tracking methods and also gave a brief review of related topics: object detection and object representation. We divided the tracking methods into three categories based on the use of object representations: methods establishing point correspondence, methods using primitive geometric models, and methods using contour evolution. Note that all these tracking methods require object detection at some point. For instance, the point trackers require detection in every frame, whereas geometric region or contours-based trackers require occasional detections or only when the object first appears in the scene.

In this chapter we also described the context of use, degree of applicability and evaluation criteria of the tracking algorithms. Our biggest focus was on the algorithms that we use in our work and throughout this thesis.

3

Image Projection Features for Object Tracking

Selecting the right features plays a critical role in tracking. In general, the most desirable properties of a visual feature are its *uniqueness* so that they can be distinguished in the feature space, and *repeatability* so that these features can be found in different object observations. Feature selection is closely related to the object representation. For example, colour is used as a feature for histogram-based appearance representations, while for contour-based representations object edges are usually used as features. In general, many tracking algorithms use a combination of features.

In this chapter, in Section 3.1, we give an overview of the Vicats project in which we carried out the research on single and multi-camera vehicle tracking. In Section 3.2 we give an overview of common visual features for tracking, their characteristics, and advantages and disadvantages in the context of tracking vehicles and people in real-world environments. We also explain desirable characteristics of features in such a context, and in Section 3.3 introduce Radon transform like image projection features that we use in our work. Similar features have been used for object detection [Betke 00], human gait [Lee 07] and handwritten text recognition [Rath 03]. However, we are to our knowledge the first who extensively use these features through the entire framework for people and vehicle tracking, multi-view appearance modelling and object recognition. In Section 3.3 we also present main advantages of these features that motivate us to use them in our work, and conclude the chapter in Section 3.4.

3.1 The Vicats project

Tunnel security and surveillance has become very important topic in traffic management. A situation where tunnel operators are particularly on the alert is when trucks carrying dangerous goods enter the tunnel. When such a vehicle catches fire inside the tunnel the consequences can be devastating. For a tunnel operator it is very important to detect when such a truck enters the tunnel and

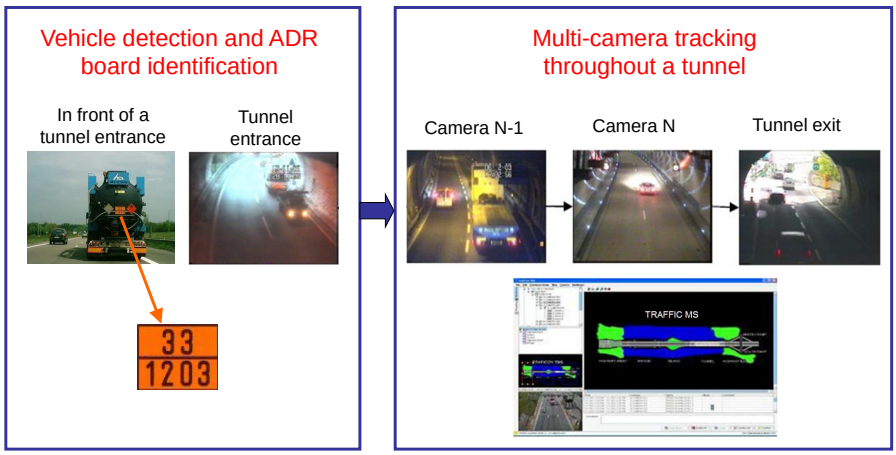


Figure 3.1: Illustration of two main components in the Vicats project. Left: At the entrance of the tunnel the vehicle (typically a truck) carrying dangerous goods is being detected and identified using ADR plates that indicate what substance the vehicle is carrying; Right: Throughout the tunnel the vehicle is being followed with the surveillance cameras in the tunnel. A traffic management software can then automatically visualize the position of the vehicle throughout the tunnel.

its exact location in the tunnel. This was the main research goal of the Vicats (Video Content Analysis for Tunnel Surveillance) project: detecting trucks carrying dangerous goods entering the tunnel and tracking them throughout the tunnel by the means of video cameras. This problem was split in two parts, namely the detection and identification of vehicles carrying dangerous goods at the entrance of tunnels on the one hand, and tracking of the vehicles throughout the tunnel on the other hand, see Figure 3.1.

At the entrance of the tunnel the truck carrying dangerous goods is being detected and identified with a dedicated license plate recognition (LPR) camera, which can also recognize ADR plates that indicate what substance the truck is carrying. The LPR cameras are high resolution cameras. Then the truck is being followed with the surveillance cameras in the tunnel. These are typically low resolution cameras operating in harsh lighting conditions so their videos often contain many artefacts and illumination problems. From a research point of view the core challenge is to correctly detect and track a certain vehicle in a camera image and over the different cameras. This was the research goal we carried out in this PhD thesis. The project was done in collaboration of academic and industrial partners in Flanders, Belgium, in the iMinds VICATS project [1], and together with academic and industrial partners from Serbia, through the Eureka VICATS project [2].

[1] More details can be found at `http://www.iminds.be/en/projects/2014/03/07/vicats`
[2] More details can be found at `http://www.eurekanetwork.org/project/-/id/4160`

In the project the cameras have been placed along the tunnel and pointed in the same direction (the traffic direction). The configuration utilizes the minimal number of cameras in order to visually cover as much as possible of the tunnel space. Therefore, there is no overlap between the camera views. Because of the layout of surveillance cameras, multi-camera tracking in this context is done in a scenario with non-overlapping camera views.

In the development of our multi-camera tracker we start from the single camera tracker and extend it later to a more complex multi-camera environment. In the single camera tracking task, the goal is to develop a tracking module that repeatedly registers positions of vehicles in each frame of a video sequence acquired by a single camera. Even in this single camera case many trackers fail in real environments, and especially in tunnels due to occlusions and poor lighting conditions. The solution for this non-trivial problem is divided in two parts:

- *Feature extraction*, appropriate choice of features that are extracted from video frames, and can be efficiently used for tracking;
- *Tracking algorithm* itself, which delivers the exact trajectory of the object of interest throughout the sequence.

In this project the focus is on robust and low complexity approaches that can work in real-time.

3.1.1 Vicats contributions and credits

The research contribution to the project VICATS in the area of multi-camera vehicle tracking was done by the Image Processing and Interpretation (IPI) research group at Ghent University and the Vision for Industry Communications and Services (VISICS) research group at the Katholieke Universiteit Leuven.

At the Image Processing and Interpretation (IPI) research group, Prof. dr. ir. Aleksandra Pižurica (project supervisor), Jorge Oswaldo Niño Castañeda, Andres Frías Velázquez and myself were involved in the project. Jorge Oswaldo Niño Castañeda was mainly focused on vehicle detection and single camera tracking using local binary patterns and optical flow. Andres Frías Velázquez was focused on vehicle detection using mathematical morphology, and vehicle identification by using orthonormal circus functions from the trace transform. At the VISICS research group at the Katholieke Universiteit Leuven, Reyes Rios Cabrera was focused on using rectangular Haar features and cascades of AdaBoost classifers to perform vehicle detection, identification and tracking. At the Faculty of Technical Sciences, University of Novi Sad, Serbia, dr. Borislav Antić, dr. Dubravko Ćulibrk and Prof. dr. Vladimir Crnojević were working on background modelling and segmentation, and detection of moving objects (vehicles) using wavelet transform and texture features. Overall industrial support and video materials for the experiments in this project were provided by the industrial partner Traficon N.V. under the supervision of dr. Wouter Favoreel.

The emphasis of my work in this context, presented in this thesis, lies on using image projection features and feature matching methods to robustly track vehicles in real-time under severe occlusions and in challenging lighting conditions. I focused on improving the tracking accuracy by adding image projection cues into the Kalman filter framework, and using these cues to create a multi-observation appearance model for more accurate vehicle recognition. Within this project, in close cooperation with Jorge Oswaldo Niño Castañeda and Andres Frías Velázquez, I designed and implemented a real-time demonstrator, which was presented at the end of the Vicats project. The demonstrator showed that it is possible to achieve high accuracy in tracking of vehicles that transport dangerous goods in tunnels.

3.2 Overview of features for tracking

Colour. Colour is one of the most used features for object tracking. The apparent colour of an object is influenced primarily by two physical factors: the spectral power distribution of the illuminant and the surface reflectance properties of the object. In image processing there are several common ways to represent colour. Usually, the *RGB* (red, green, blue) colour space is used. However, the *RGB* space is not a perceptually uniform colour space, that is, the differences between the colours in the *RGB* space do not correspond to the colour differences perceived by humans [Paschos 01]. Additionally, the *RGB* values are highly correlated. In contrast, *L-u-v* and *L-a-b* spaces are perceptually uniform colour spaces, while *HSV* (Hue, Saturation, Value) is an approximately uniform colour space. However, color values in these colour spaces are sensitive to noise [Song 96]. Therefore, there is no last word on which colour representation is the most suitable for real-world applications. Moreover, in many real-world situations the colour of objects is not a very discriminative and reliable feature, and some other features are better suited for tracking. For instance, in vehicle tracking the challenge is to exploit colour features in different weather conditions, low-light and artificially illuminated environments (e.g. in tunnels or at roads at night), see Figure 3.2. There are also many vehicles of the same colour. In video conferencing applications, it is difficult to distinguish different people based on colour, because in a business setting people's clothing typically includes very limited colour variations and colours from artificial lights in meeting rooms (the ambient or projector lighting) can create local colour variations or supersede true colours of the viewed objects, see Figure 3.3. These are the reasons why in our work we paid a significant attention to finding more robust features than colour.

Edges. Object boundaries usually generate strong changes in image intensities (brightness). This is true for boundaries between different parts of an object, as well as for patterns visible on the object's surface. Edge detection is typically used to identify these brightness changes. An important property of edges is that they are less sensitive to illumination changes compared to colour features. Algorithms that track the boundary of the objects

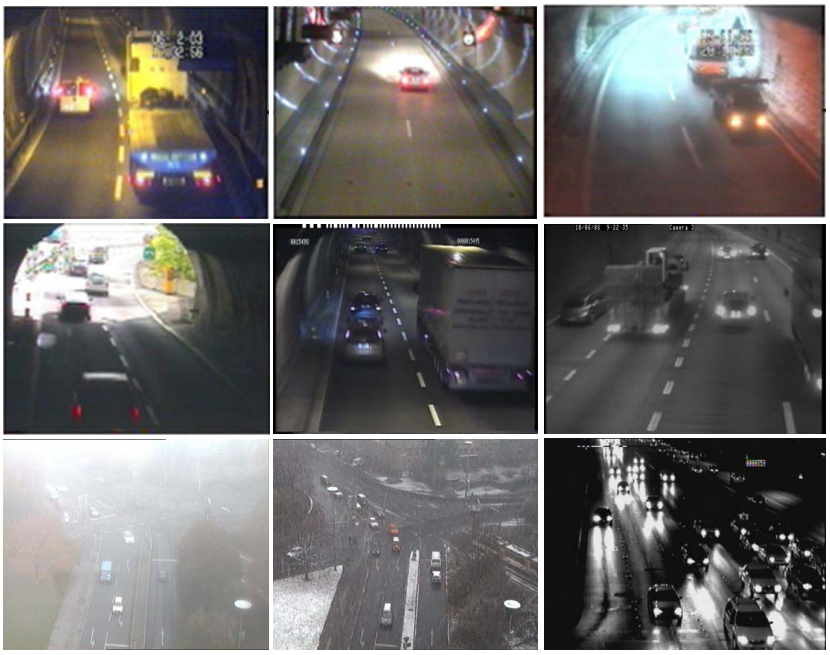


Figure 3.2: Examples of traffic surveillance videos. In many real-world situations the colour of objects is not a very discriminative and reliable feature. The colour of artificial lighting can supersede natural colours, in too bright or too dark environments colours are not visible, in low-light conditions grayscale cameras might be used instead of colour cameras (due to higher dynamic range), and in difficult weather conditions (fog, snow, rain) the colours are typically also less visible.

or use object's shape for tracking usually use edges as the representative feature. One of the most popular edge detection methods is the Canny edge detector [Canny 86], mainly due to its simplicity and accuracy. More detailed information about edge detectors and an evaluation of edge detection algorithms is made by Bowyer *et al.* [Bowyer 01]. Beside for tracking, edges can also be used for foreground/background segmentation [Grünwedel 14] and object recognition [Shan 08]. One of the biggest drawbacks in edge detection is the sensitivity to thresholding, which needs to be performed to extract edges from the image. This thresholding step strongly depends on the chosen parameters, which are difficult to generalize for different environments, different illumination conditions, image resolutions and other variable factors. Figure 3.4 shows an example of Canny edge detection results obtained using the same parameters for three images of the same vehicle. We see that the edge detection can vary significantly due to different illumination conditions. Therefore, in our work our intention has been to find features that have the advantages of edge features, but without the thresholding disadvantage.

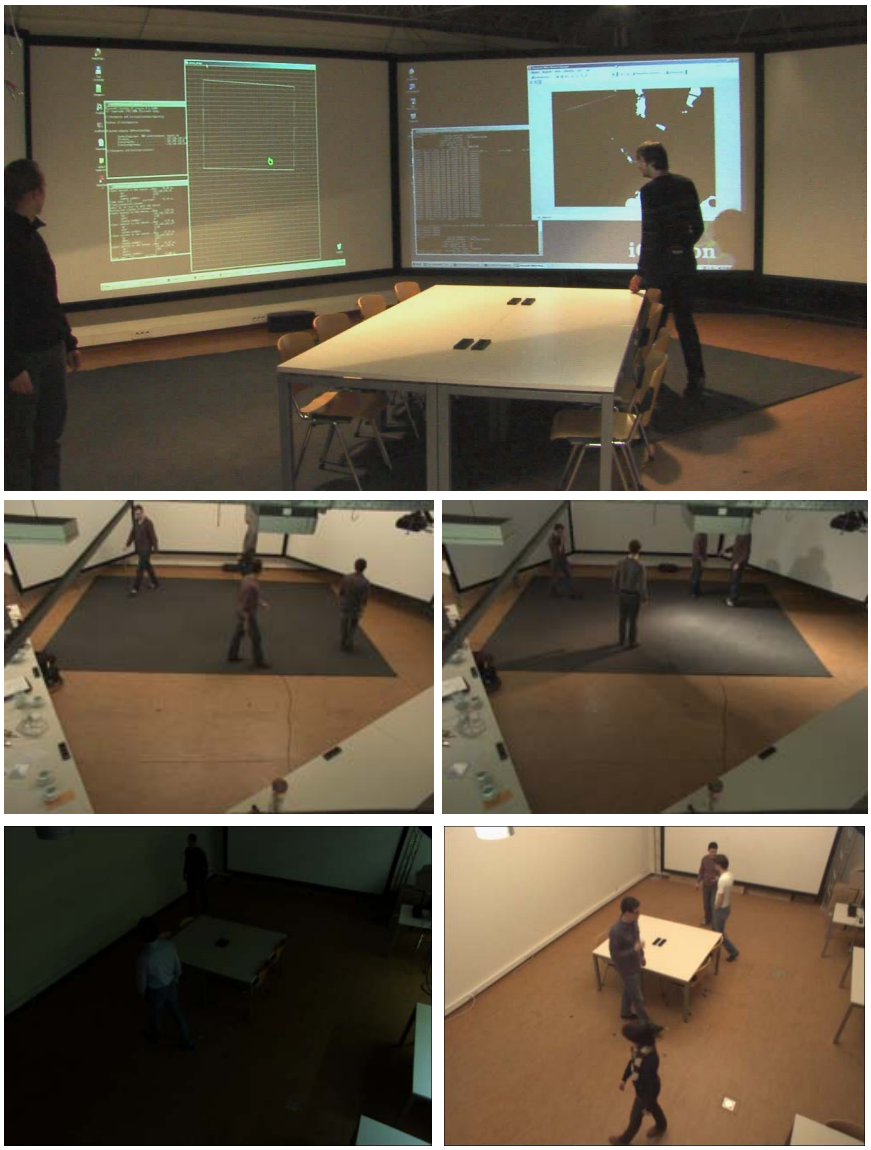


Figure 3.3: Examples of video conferencing videos. It is difficult to distinguish different people based on colour, because people's clothing typically includes very limited colour variations and colours from artificial lights in meeting rooms (the ambient or projector lighting) can create local colour variations or supersede true colours of the viewed objects.

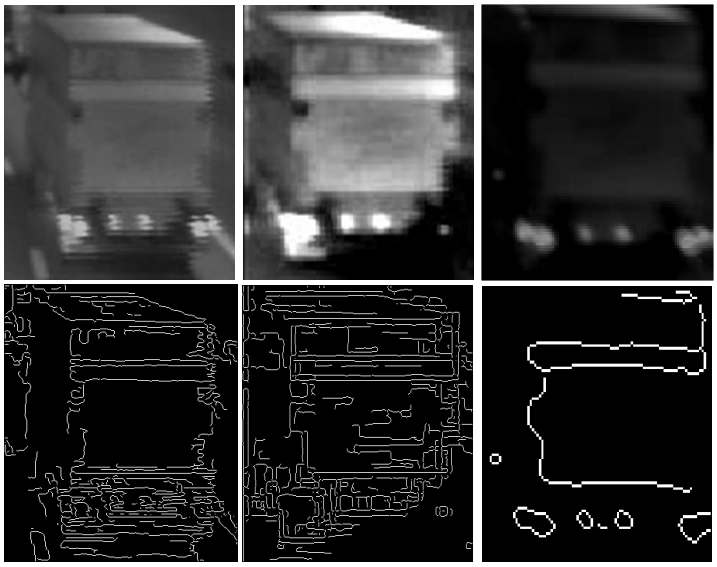


Figure 3.4: Canny edge detection response for the images of the same vehicle taken in different illumination conditions. The same parameters for edge detection were used in all cases. We see that the edge detection results can vary significantly due to different illumination conditions. This is manly due to thresholding parameters, which need to be adapted for different conditions, but in the case of dynamic conditions this becomes a difficult task.

Optical flow. Optical flow is a dense field of displacement vectors which defines the translation of each pixel in a region. It is computed using the brightness constraint, which assumes brightness constancy of corresponding pixels in consecutive frames [Horn 81]. Optical flow is commonly used as a feature in motion-based segmentation and tracking applications. Popular techniques for computing dense optical flow include methods by Horn and Schunck [Horn 81], Lucas and Kanade [Lucas 81], [Black 96], [Szeliski 97]. For the performance evaluation of the optical flow methods, we refer the interested reader to the survey by Barron *et al.* [Barron 94]. There are three essential conditions that need to be satisfied for a successful use of optical flow features.

- *Brightness constancy.* This means that image brightness needs to remain the same or very similar in a small region around the feature, even when the region moves within the image.
- *Spatial coherence.* This assumption means that neighbouring points in the scene typically belong to the same surface and have similar motions. Since neighbouring points in the scene project to the nearby points in the image, this results in the spatial coherence in the optical flow.
- *Temporal persistence.* This means there is a gradual change in motion of a surface patch (a region in the image) and not a sudden change.

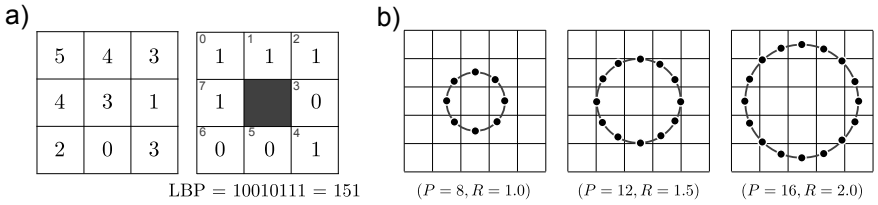


Figure 3.5: *Local binary patterns*: a) The local binary pattern (LBP) is computed by comparing the central pixel to each of its eight neighbours; the binary value associated with each position is set to one if that neighbour is greater than or equal to the central pixel; the eight binary values can be read out and combined to make a single 8-bit number; b) LBP can be computed over larger areas by comparing the current pixels to the (interpolated) image at positions on a circle. This type of LBP is characterized by the number of samples P and the radius of the circle R.

Due to these requirements the tracked objects need to appear neither too large nor too small in the image, and motions of these objects, as well as their observed brightness changes, have to be relatively smooth. These conditions are difficult to meet when the objects pass close to the camera and when illumination changes suddenly. Therefore, in these cases some other features need to be used for tracking, either solely or as additional cues next to optical flow.

Texture. Texture is a measure of the intensity variation in an image, and it quantifies properties such as smoothness and regularity. Compared to colour, texture requires a processing step to generate the descriptors. There are various texture descriptors: Gray-Level Co-occurrence Matrices (GLCM) [Haralick 73] (a 2D histogram which shows the co-occurrences of intensities in a specified direction and distance), Laws' texture measures [Laws 80] (2D filters generated from five 1D filters corresponding to level, edge, spot, wave, and ripple), wavelets [Mallat 89] (orthogonal bank of filters), steerable pyramids [Greenspan 94], local binary patterns [Heikkila 06], etc. Similar to edge features, texture features are less sensitive to illumination changes compared to colour, but it is necessary that cameras capture high level of details, for which typically high resolution cameras are needed (Figure 3.5 illustrates local binary patterns calculation and demonstrates that texture typically needs to be captured on a pixel or even sub-pixel level).

Beside these features, local interest points such as Harris corners, SIFT, SURF or FAST, explained in Chapter 2, Section 2.2.1, are also often used for tracking, especially in a tracking-by-detection framework. In this framework other often used features are Haar-like rectangular features, very popular for face detection and tracking [Viola 04], and histogram of oriented gradients (HoG) features often used for people detection [Dalal 05]. Recently, Haar-like features have been used for vehicle detection and tracking as well [Rios Cabrera 12]. Main drawbacks of local interest points are insufficient repeatability, quantity and uniqueness of these features in low resolution or noisy images,

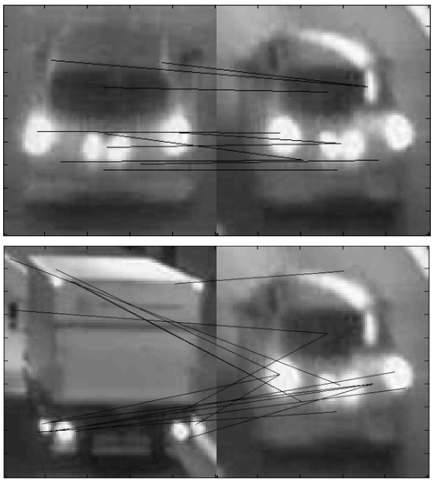


Figure 3.6: Example of SIFT response in a traffic surveillance scenario. Main drawbacks of local interest points are insufficient repeatability, quantity and uniqueness of these features in low resolution or noisy images common in video surveillance scenarios. We see that similar number of SIFT correspondences are found for two different vehicles.

which are common in video surveillance scenarios, as well when the objects appear relatively small in the image [Mikolajczyk 05b], see an example for SIFT features in Figure 3.6. Haar-like features perform better in these cases, but their performance depends strongly on the quality and amount of training data and the training process (classifier). Both local interest points and Haar-like features do not capture a semantic interpretation of an object so in cases when tracked objects have multiple similar surfaces, or there are other similar surfaces in the vicinity of the object, these surfaces can be confused.

As we see from this overview, there are many different features that can be used for object tracking. Features for tracking are mostly chosen manually by the user, depending on the application domain. Since this is not optimal, there is significant research dedicated to the problem of automatic feature selection. Automatic feature selection methods can be divided into filter methods and wrapper methods [Blum 97]. The filter methods try to select the features based on a general criteria. For example, one such a criterion is that the features should be uncorrelated. Principal Component Analysis (PCA) is an exemplary filter method for the feature reduction. PCA involves transformation of a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called the principal components. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. On the other hand, the wrapper methods select the features based on the usefulness of the features in a specific problem domain, for example, the

classification performance using a subset of features. One wrapper method to select the discriminatory features for tracking of a particular class of objects is the Adaboost method [Tieu 04]. Adaboost is a method for finding a strong classifier based on a combination of moderately accurate weak classifiers. Given a large set of features, one classifier can be trained for each feature. Adaboost, as discussed in Chapter 2, Section 2.2.4, will create a weighted combination of classifiers (representing features) that maximize the classification performance of the algorithm. The higher the weight of the feature, the more discriminatory it is. One can use the first n highest-weighted features for tracking.

In our work we focused on finding tracking features with the following characteristics.

- *Computational efficiency.* Ideally, the calculation of the features should be possible in one reading of the image and with basic arithmetic operations, preferably only addition and subtraction, like in the integral images [Viola 01].

- *Data efficiency.* It should be possible to create feature descriptors in few data bytes so they could be easily transmitted between the cameras with a negligible power consumption.

- *Robustness to illumination changes.* There should be a high correlation of features of the same object taken from different illumination conditions, both in cases of global and local illumination changes.

- *Robustness to object pose and camera viewpoint changes.* Changes in object pose and camera viewpoint should be either minimally reflected in the feature values or reflected in a predictable way.

- *No parameters that need to be adapted to different environments.* Ideally, the feature calculation should not depend on any parameter that needs to be set differently for different environments.

Inspired by the work of Betke *et al.* [Betke 00] and Lee *et al.* [Lee 07], we focused on adopting image projection features for tracking and building a necessary framework to incorporate these features in real-world tracking scenarios. Betke *et al.* used vertical and horizontal projections of a vehicle edge map for accurate positioning of the bounding box in tracking. Similar projections have been used by Lee *et al.* for human gait recognition. Compared to these works we go a step further, showing that it is possible to use image projections for object appearance representation and matching. We do not use projections of object edge maps, but of the object images directly.

In the remainder of this chapter we will define these features and show whether and how they satisfy the aforementioned characteristics we aim for. Furthermore, in Chapters 4, 5 and 6 we will explain how we use these features to build object appearance descriptors suitable for single and multi-camera tracking and object recognition.

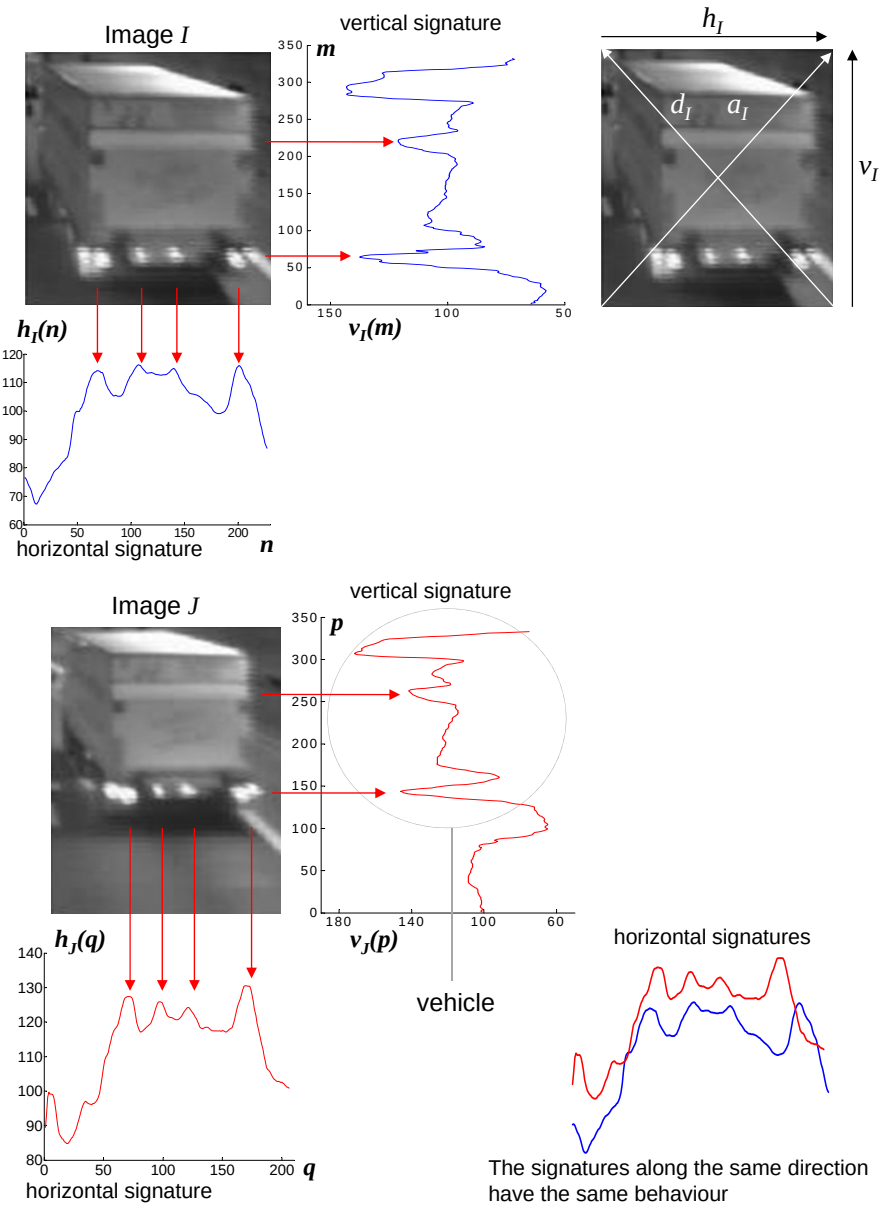


Figure 3.7: Two vehicle images captured by a surveillance camera in a tunnel, and corresponding horizontal and vertical signatures. We see that signature peaks capture the brightness changes of vehicle parts and patterns. There is a clear similarity in behaviour of the signature parts that represent the vehicle. We define the horizontal, vertical, and diagonal signatures as shown in the image in the upper right corner.

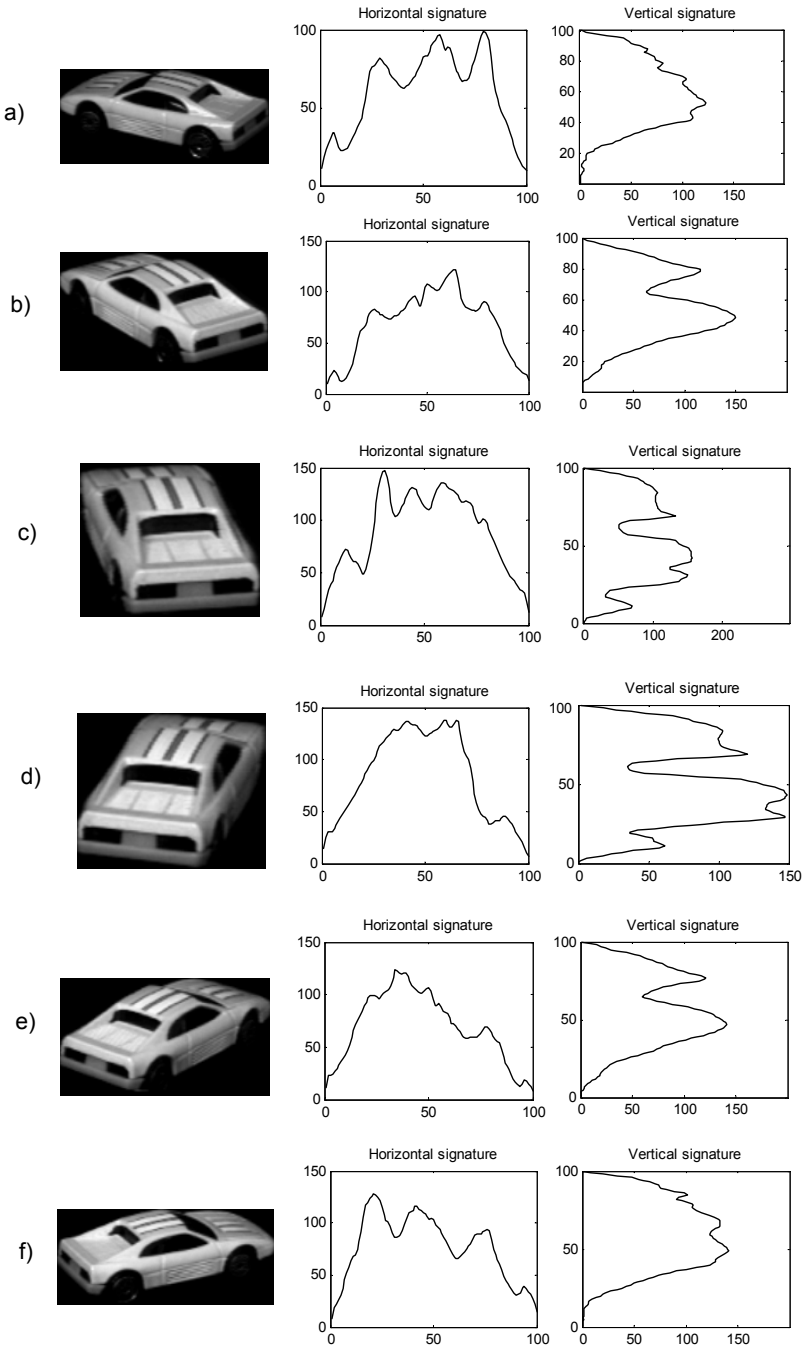


Figure 3.8: Illustration of behaviour of the signatures under significant pose changes of the observed object. The figure contains six images (a-d) of the same car captured from different viewing angles. The difference in the viewing angle between successive images is 25 degrees. These images are taken from the Columbia Object Image Library (COIL-20).

3.3 Image projection features

Let I be an image of size $M \times N$, as shown in Figure 3.7. We define the image projection features of the image I as Radon transform like projection profiles along a certain direction. We call these features *signatures*. The vertical signature $\mathbf{v}_I$ consists of the arithmetic means of the pixel intensities in each image row,

$$v_I(m) = \frac{1}{N} \sum_{n=1}^{N} I(m,n), \quad m = 1, ..., M, \tag{3.1}$$

where $I(m,n)$ is an intensity value of the image pixel at the position (m,n). Analogously, the components of the horizontal signature $\mathbf{h}_I$ are the arithmetic means of the pixel intensities in each image column,

$$h_I(n) = \frac{1}{M} \sum_{m=1}^{M} I(m,n), \quad n = 1, ..., N. \tag{3.2}$$

Next to the vertical and horizontal signatures, defined by Equations (3.1) and (3.2), the signatures in other directions can be defined analogously, for instance along image diagonals.

The signatures along different directions can be combined in a single image descriptor. In this sense, we define the n-dimensional signature vector $\mathbf{s}_I$ calculated from the image I as an n-tuple of n projections (signatures) on different lines. In our work we use 2-dimensional (2-D) and 4-dimensional (4-D) signature vectors. The 2-D signature vector is a pair of the vertical and horizontal signature,

$$\mathbf{s}_I = (\mathbf{v}_I, \mathbf{h}_I), \tag{3.3}$$

while the 4-D signature vector contains also two diagonal signatures (see Figure 3.7, bottom right),

$$\mathbf{s}_I = (\mathbf{v}_I, \mathbf{h}_I, \mathbf{d}_I, \mathbf{a}_I), \tag{3.4}$$

where $\mathbf{d}_I$ and $\mathbf{a}_I$ are signatures on the main-diagonal and anti-diagonal, respectively. We see that the signature vectors represent an image as multiple 1-D vectors, which significantly reduces the amount of data needed to represent the appearance of an imaged object. In the following sections we will analyse the characteristics of the proposed signature descriptors with respect to the requirements given in Section 3.2.

In Figure 3.7 we see two images of the same vehicle viewed by grayscale cameras in a tunnel. The images are represented by horizontal and vertical signatures. The vehicle's bright areas, captured by the signatures, are marked with arrows. The bright areas correspond to the local maxima in the signatures. Analogously, the dark patterns are represented by the local minima. If two horizontal signatures are plotted one over the other (the signatures at the bottom right of Figure 3.7), we see that they have similar behaviour (shape). The background, a road, has almost uniform brightness so it does not change significantly the behaviour of the signatures. Also in the vertical direction, the

signature parts that correspond to the vehicle are similar, both when the detection includes the background or only a part of the vehicle. Having noticed this, we propose using the signatures as descriptors of the appearance of observed objects.

An advantage of the proposed signature based representation compared to an edge based representation, which also has responses in the areas of brightness changes, is in the fact that there is no thresholding in the calculation of the signatures. The shape of the signatures is, thus, less influenced by the intensity gradient values within the image and more robust against lighting changes than edges. Since vehicles are rigid objects, the signatures are similar in different observations, mainly except for translation and scale. If the vehicle appears rotated between observations, parts of the corresponding signatures shrink or stretch, but overall shape of the signatures remains similar. We will demonstrate these characteristics in the following sections.

Note that the signatures could also be defined by using other pooling methods, e.g. maximum instead of average pooling. Furthermore, they could be calculated on different images, for instance, on a gradient image instead of an intensity image. In our work we opted for average pooling on intensity images to have higher robustness to illumination changes (slower saturation) and to capture the information both about image brightness changes and brightness intensities.

3.3.1 Signatures in pose/viewpoint changes

It is very important that object tracking is robust to apparent changes of object pose. These changes occur in almost all tracking scenarios. Therefore, in this section we analyse the behaviour of the signatures in cases of significant pose changes.

Figure 3.8 contains six images (a-d) of the same object (a car) captured from different viewing angles. The difference in the viewing angle between successive images is 25 degrees. These images are taken from the Columbia Object Image Library (COIL-20). They are captured from such an angle that the vehicle appears to be rotating mainly around the vertical image axis and only slightly around the horizontal axis. For each image the corresponding vertical and horizontal signatures are shown, scaled to the length of 100 points using cubic interpolation. We see there is a similarity in the shape of the vertical signatures when the vehicle is captured in poses that from the vertical axis perspective reveal similar parts of the vehicle (e.g. see the vertical signatures in poses c and d, b and e, and a and f). On the other hand, we also see that the vertical image axis is the axis of symmetry for the horizontal signatures like for the vehicle itself (again, it is visible the most between poses c and d, b and e, and a and f).

In our work we exploit the illustrated signature behaviour to create a multi-view appearance model that represents viewed objects from several representative viewpoints. In this way it is possible to make signature based descriptors

robust to object pose and viewpoint changes. We explain this in more detail in Section 5.3.

3.3.2 Signatures under illumination changes

In Figure 3.9 we illustrate the behaviour of the signatures under illumination changes. We show three observations of the same vehicle affected by different illumination (the same observations are used for edge calculation in Figure 3.4). We see that horizontal and vertical signatures preserve similar shape even under such severe illumination changes (for easier comparison, all signatures are shown scaled to 100 points in length, using cubic interpolation). In Chapter 5 we thoroughly demonstrate this robustness. We define a similarity measure between the signatures and compare a large database of images, showing that signatures of the same object are more similar than signatures of distinct objects, even under different illumination.

3.3.3 Computational and data efficiency

An important advantage of the signature features is their computational and data efficiency. The signatures for all objects in an image can be computed in a single reading of the image. In addition to this, when signatures are compared instead of the whole images, matching 2D data is computationaly simplified to matching multiple 1D data. Furthermore, since the most information about the objects is captured in the signature peaks (see Figure 3.7), it is possible to downscale the signatures still preserving most of the information, which even more increases computational and data efficiency. We demonstrate this in Chapter 5, Section 5.6.4. Note also that signature calculation does not contain thresholding or other parameters. This is one of important advantages that enables consistence of signatures in various illumination conditions.

3.4 Conclusion

In this chapter we presented an overview of features typically used for object tracking: colour, edges, texture features, optical flow and local invariant points. We also introduced popular methods for automatic feature selection: PCA and AdaBoost classification. We explained why the common features used for tracking do not perform well in real-world conditions and introduced image projection features (signatures) that we use in our tracking methods. We will use these features in the following chapters as a basis for the whole tracking framework, from low-level to high-level, to model object appearances in harsh lighting and weather conditions, deal with inaccurate detections and occlusions.

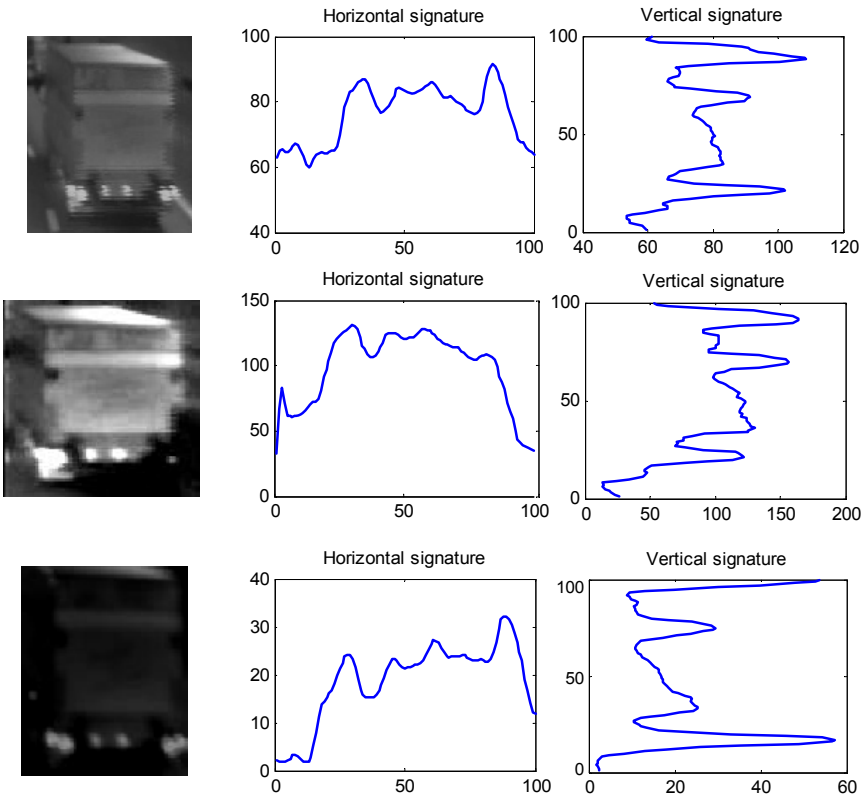


Figure 3.9: Illustration of behaviour (shape) of the signatures under significant illumination changes. The figure contains three images of the same vehicle captured under different lighting conditions. The signatures show a big robustness to the illumination changes, i.e. their shape remains very similar even in this case of a severe illumination change. For easier comparison, the signatures are shown scaled to the same length of 100 points.

4

Tracking in a Single Camera View

In this chapter our focus is on tracking objects in real-world environments observed by only one camera, i.e. object tracking from a single viewpoint. This is still a common scenario in many applications, such as perimeter and traffic surveillance, outdoor people tracking, and many others. In this chapter we address several significant challenges of this object tracking task.

- Big variations in illumination and weather conditions, as well as frequent occlusions of the viewed objects are a tremendous challenge for accurate tracking.
- Some of the typically used image features for tracking, such as colour, shape or texture, are often not discriminative enough in real-world conditions.
- Deployment of smart visual sensors, such as smart cameras, creates the need for low complexity and computationally efficient tracking methods.

In our work in this context, we use a multi-cue Kalman filter framework for tracking. Our focus is on the signatures defined in Chapter 3, as computationally efficient and illumination invariant cues. In this chapter we define a time warping technique for signature matching, which we use to find boundaries of tracked objects in each video frame. We demonstrate the advantages of such signature based cues versus foreground blobs and optical flow cues. We evaluate our approach on several traffic surveillance sequences recorded in different weather and illumination conditions.

The remainder of the chapter is organized as follows. Section 4.1 reviews briefly related work on single-camera tracking in the context of our work. In Section 4.2 we formulate the tracking problem we focus on. Section 4.3 explains our signature based cues that increase robustness and accuracy of single camera tracking. In Section 4.4 we explain the tracking algorithm. Experimental results are presented and discussed in Section 4.5. We present the results

from a real tunnel video sequence, several traffic surveillance sequences taken in different weather conditions and during night, from different viewpoints. Finally, we conclude this chapter in Section 4.6.

4.1 Related work

Monocular tracking approaches have been an active research topic in the past decades. They involve tracking of a single or multiple targets using only one camera view. Common features for tracking in these scenarios are colour based (mean, histogram or correlogram of the colour [Huang 97,Porikli 03,Rahimi 04, Javed 05, Choe 10]), gradient based (edges [Guo 07]), key points based (optical flow of local features like Shi-Tomasi [Shi 94], Harris corners [Harris 88], SIFT [Lowe 04], FAST [Rosten 05], etc.) or texture based (e.g. local binary patterns [Ojala 96]). More recent methods use also feature classification, with or without online adaptation, [Babenko 11], [Rios Cabrera 12]. *Edges* and *texture* features are less sensitive to the lighting conditions and more robust to illumination changes than colour, but in the case of sudden and intensive illumination changes between successive video frames the corresponding edge or texture maps can be significantly different and thus difficult to match. *Optical flow* of local features is very popular in real-time applications since it is not computationally demanding, neither for calculation of features nor for their matching (even for SIFT alike features there are many speed-up methods, e.g. SURF [Bay 08]). However, the accuracy of optical flow tracking depends on the number of detected features, their repeatability and uniqueness, so this technique also suffers from inconsistencies in the cases of sudden and intensive illumination changes.

In the computationally low-cost (real-time) tracking approaches, the features are typically used to form blobs, which are then used for tracking [Collins 03]. Blobs are usually detected on a frame-by-frame basis and are tracked by comparing their shape, location and appearance from one frame to another. The *BraMBLe* tracker [Isard 01], for example, is a Bayesian multi-blob tracker that computes the likelihood for each blob based on a known background model and the appearance model of tracked objects. It uses particle filtering to track an unknown number of objects. Problems arise when object blobs merge with blobs of other close-by objects or with blobs of occluding objects, degrading the performance of this tracker. However, this method can be improved by taking more cues into account, such as in the following works.

Giebel *et al.* [Giebel 04] use Bayesian tracking based on particle filters, combined with a detector using learned spatio-temporal shapes to perform multi-cue 3D object tracking in a single camera view. Their spatio-temporal object representation involves a set of distinct linear subspace models or Dynamic Point Distribution Models (DPDMs,) and it is learned fully automatically from training data. Furthermore, the representation is enriched with texture information by means of intensity histograms and 3D measurements provided by a

stereo system. The reported results are very good, but they have been generated on a relatively small data set. This method also requires shape, texture and image depth information to reliably track objects. Therefore, it requires a significant computation time.

Smith *et al.* use particle filtering based on Markov chain Monte Carlo optimization to track people and handle entrances and exits using a fixed camera [Smith 05]. Their framework uses a joint multi-object state-space formulation to recursively estimate the multi-object configuration and efficiently search the state-space by using particle filtering. As a global appearance model, binary images based on background subtraction, together with foreground and background colour statistics, are used to discriminate between different objects in the scene.

One recent approach is the method of Babenko *et al.* [Babenko 11]. This is a tracking technique based on the concept of "tracking by detection". It uses *multiple instance learning (MIL)* to train a discriminative classifier in an online manner to separate the object from the background, based on *histogram of oriented gradients (HOG)* features. Its disadvantage is also a relatively high computational load, which makes it too complex for usage on smart cameras.

In our work we combine blobs obtained as foreground regions, optical flow in these foreground regions, and the signatures (image projection profiles) that we defined in Chapter 3. When the lighting conditions do not change rapidly, foreground blobs and optical flow are suitable measurements for tracking. In the cases of sudden illumination changes foreground blobs suffer from gaps and ghosting effects, which leads to stretching or shrinking of foreground bounding boxes, while the optical flow vectors change rapidly, both in magnitude and direction, see Figure 4.1. However, the vehicle signatures are relatively invariant to such illumination changes so we use them to obtain more reliable measurements. We compare the signatures from two successive video frames to find their alignment and to correct the position and size of tracked objects. Using the signatures to improve robustness of the tracking is one of the main novelties and contributions explained in this chapter.

4.2 Problem formulation

We define the single camera tracking problem as the problem of localizing observed objects in each video frame captured by only one camera, and associating the localizations from successive frames. We assume that the initial detection of each object is known and provided when objects appear in the scene for the first time. In our work, depending on the application domain, we used vehicle and people detections obtained by the algorithms of [Rios Cabrera 12], [Frías Velázquez 11], [Jelača 08] or [Grünwedel 14].

The single camera tracking is performed in the 2D plane of camera images. A state of an object at time instance t is defined as a six-dimensional vector $\mathbf{x}_t = (x_t, y_t, \dot{x}_t, \dot{y}_t, w_t, h_t)$, which contains the object's position along x and y image axis (x_t and y_t), x and y components of the velocity ($\dot{x}_t$ and $\dot{y}_t$), and the object's

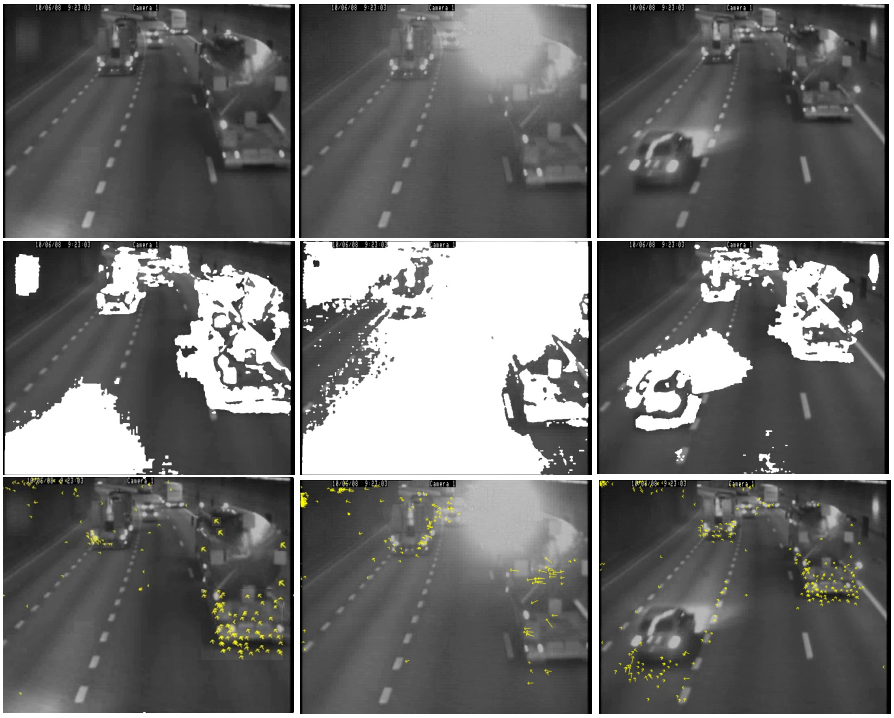


Figure 4.1: Examples of camera images from a tunnel. a) First row: We see that light reflections in tunnels can be very intensive; b) Second row: Foreground detection - incorrect in cases of illumination changes; c) Third row: Optical flow of Shi-Tomasi features [Shi 94] - in cases of illumination changes optical flow vectors change rapidly, both in magnitude and direction.

size, i.e. the width and height of the object bounding box (w_t and h_t). A linear Kalman filter is used for tracking, as a technique for filtering and prediction (see Chapter 2, Section 2.3.1.2). The Kalman filter averages a prediction of an object's state with a new measurement using a weighted average. The purpose of the weights is that values with smaller estimated uncertainty are "trusted" more. The filter represents beliefs by the moments, the mean μ_t and the covariance Σ_t. The weights are calculated from the covariance. The result of the weighted average is a new state estimate that lies in between the predicted and measured state, and has a better estimated uncertainty than either alone.

The state transition is modelled as follows:

$$\mathbf{x}_t = A_t\mathbf{x}_{t-1} + B_t\mathbf{u}_t + \epsilon_t, \tag{4.1}$$

where $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ are state vectors at time instances t and $t-1$ respectively, and $\mathbf{u}_t$ is the control vector, which is in our algorithm omitted since we do

not have control inputs in this system. A_t is the state-transition matrix and ϵ_t the process noise. The noise is modelled as a multivariate Gaussian random variable with zero mean and the covariance Q_t. Both A_t and Q_t have the size $n \times n$, where n is the dimension of the state vector $\mathbf{x}_t$. A constant velocity model is used in modelling the state transition A_t.

The state is updated according to the following equation:

$$\mathbf{z}_t = C_t \mathbf{x}_t + \delta_t, \tag{4.2}$$

where $\mathbf{z}_t$ is the vector of measurements at time t, C_t is a $k \times n$ dimensional matrix, with k being the dimension of the measurement vector, and δ_t is the measurement noise. The vector δ_t is modelled as a multivariate Gaussian variable with zero mean and covariance R_t.

In this context, one or multiple of the following measurements are used for single camera tracking: location and size of the foreground mask, the mean velocity obtained from the optical flow statistics and location and size of the object's bounding box calculated from the signatures we proposed in Chapter 3. In cases of sudden and intensive illumination changes and occlusions, the covariance values for foreground and optical flow measurements increase so they become less reliable. Therefore, it is essential that the signature based measurements are constructed in a way that enables robustness to illumination changes and occlusions. In the following section we explain the signature based measurement in detail. In Section 4.5.1 we present the results of vehicle tracking in tunnels, showing the benefit of adding the signature measurement next to the foreground and optical flow measurements. In Sections 4.5.2 and 4.5.3 we show the results of using only signature measurement in different weather and illumination conditions, demonstrating the robustness of the signature measurements.

4.3 Signatures based measurement for tracking

In Chapter 3 we illustrated the characteristics of Radon transform like image projection profiles of viewed objects. We call these projection profiles *signatures*. Based on the horizontal and vertical signatures we can calculate the motion of an object (along x and y image axis) and its approximate size (width and height). In this way we obtain a measurement (a bounding box), which is used in our Kalman filter based tracker to correct predicted position and size of the object.

In Chapter 3 we defined a *signature vector* of an observed object, $\mathbf{s}_I = (\mathbf{v}_I, \mathbf{h}_I, \mathbf{d}_I, \mathbf{a}_I)$. Such a vector can consist of an arbitrary number of signatures along different projection lines: vertical, horizontal, diagonal and others. However, for calculation of the object motion along x and y image axis, the horizontal and vertical signatures are the most informative ones so we use these two projections for tracking (see Chapter 5, Section 5.6.5 for more information about comparison of 2D and 4D signature vectors). In the remainder of

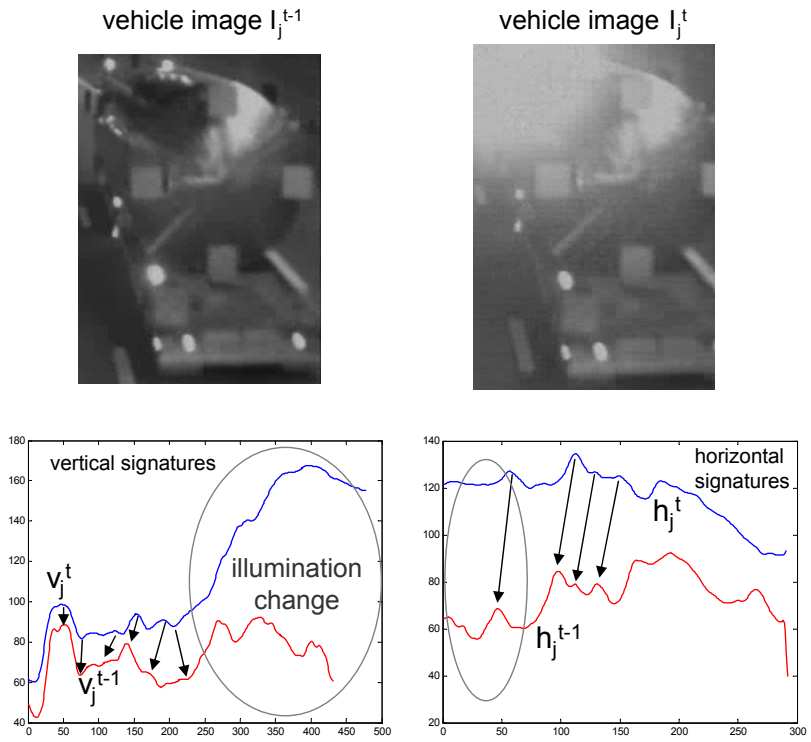


Figure 4.2: Images of the same vehicle from two successive time instances with the corresponding horizontal and vertical signatures. Vehicle's rotating light partially blinds the camera and the bottom right part of the vehicle is least affected by the illumination change. We see that even in such an intensive illumination change, the signatures preserve similar behaviour (shape).

this section we define the calculation of our signature based measurement and explain how we use it for tracking.

Let I_{t-1} be the image of an object defined by its bounding box BB_{t-1} of size $M \times N$, and obtained at the time instance $t-1$, see Figure 4.3. Let further R_t be a rectangular region of size $P \times Q$ ($P \geq M$, $Q \geq N$) around image I, defined in such a way that the viewed object between the time instance $t-1$ and the time instance t can move only within the region R. We then compare the horizontal and vertical signatures of the image I_{t-1} with the corresponding signatures of the region R_t to find their alignment. For this purpose we use a curve alignment technique based on dynamic time warping, as follows.

The challenges for robust signature matching come from scale, shift and rotation variations between the observations that are compared (see Figure 3.7). *Scale differences* result from different apparent size of objects at different distance from camera. *Shift* results from differences in the bounding box location

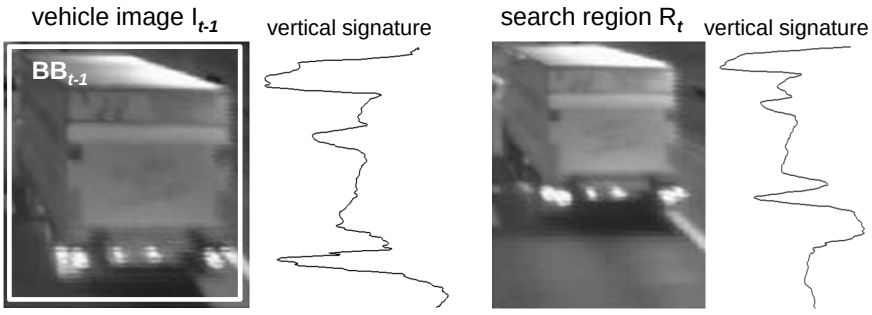


Figure 4.3: The image and bounding box of a vehicle at time $t-1$, the tracker's region of interest at time t, and their vertical signatures. We see the similarity of the signature parts that correspond to the vehicle.

with respect to the objects. *Rotation* is caused by pose changes of objects, together with camera viewing angle changes. Due to the scale difference the lengths of the corresponding parts of the signatures differ. A consequence of the bounding box shift is the signature shift along the shift direction, while the vehicle rotation results in shrinking and stretching of the signature parts. Note that this is only approximately true for small rotations, so to cope with this we use descriptors from multiple viewpoints with relatively small rotations. More detail about this is given in Chapter 5. For robust signature matching it is necessary to use a curve alignment method able to cope with these deformations.

As proposed by Sebastian *et al.* [Sebastian 03], we can use the alignment computed between two curves to define a distance measure between these curves, from the cost of deformations. There are two essential intrinsic properties of the curve that we use to define a similarity metric based on the alignment: length and curvature. The optimal correspondence of these properties can be found by an efficient dynamic-programming method called dynamic time warping. This method is also effective in the presence of a variety of curve transformations because it performs global and local curve alignment. An alternative to time warping can be a cross-correlation at multiple scales. In the following sections we define and explain these methods in detail.

4.3.1 Deformation based curve alignment

Let $c(a) = (x(a), y(a)), a \in [0, L]$ and $c^*(a^*) = (x^*(a^*), y^*(a^*)), a^* \in [0, L^*]$ be two curves of length L and L^* respectively. Let further $c|_{[A_1,A_2]}$ be the segment of the curve c between its points $A_1 = (x(A_1), y(A_1))$ and $A_2 = (x(A_2), y(A_2))$. Analogously, the curve segment between points $A_1^* = (x^*(A_1^*), y^*(A_1^*))$ and $A_2^* = (x^*(A_2^*), y^*(A_2^*))$ on the curve c^* we denote as $c^*|_{[A_1^*,A_2^*]}$.

We define a mapping (alignment) function of the two curves as $g : [0, L] \to [0, L^*]$, $g(a) = a^*$. Let $g|_{([A_1,A_2],[A_1^*,A_2^*])}$ denote the mapping of the curve

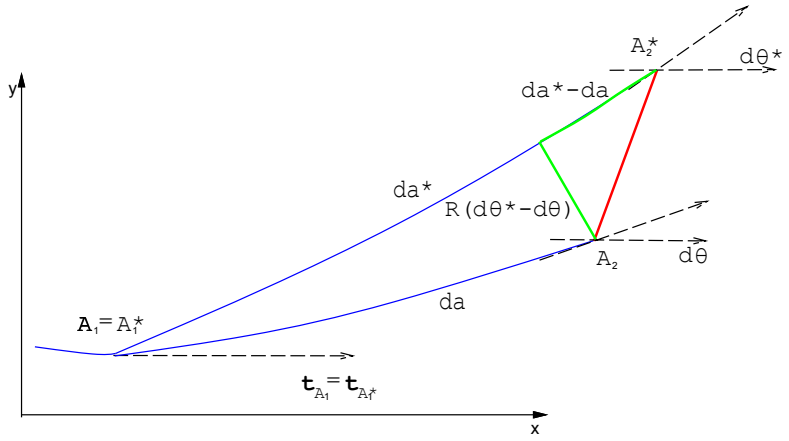


Figure 4.4: The cost of deforming a segment A_1A_2 to segment $A_1^*A_2^*$, when the start points and their tangents are aligned ($A_1 = A_1^*$, $\mathbf{t}_{A_1} = \mathbf{t}_{A_1^*}$), is related to the distance $A_2A_2^*$, and is defined by $|da^* - da| + R|d\theta^* - d\theta|$.

segment $c|_{[A_1,A_2]}$ to the curve segment $c^*|_{[A_1^*,A_2^*]}$, where $A_1^* = g(A_1)$ and $A_2^* = g(A_2)$. Further, let us define a measure η on this alignment function, $\eta[g]|_{([A_1,A_2],[A_1^*,A_2^*])} : g|_{([A_1,A_2],[A_1^*,A_2^*])} \rightarrow \mathbf{R}^+$ to denote the cost of deforming $c|_{[A_1,A_2]}$ to $c^*|_{[A_1^*,A_2^*]}$. To be able to decompose the curve matching process into a number of matches of the curve segments, we restrict the measure η to one that has an *additivity property*:

$$\begin{aligned}\eta[g]|_{([A_1,A_3],[A_1^*,A_3^*])} = \eta[g]|_{([A_1,A_2],[A_1^*,A_2^*])} + \eta[g]|_{([A_2,A_3],[A_2^*,A_3^*])},\\ \forall A_1 \leq A_2 \leq A_3 \in [0,L], \forall A_1^* \leq A_2^* \leq A_3^* \in [0,L^*].\end{aligned} \tag{4.3}$$

where $A_i^* = g(A_i), i = 1,2,3$. This additivity property enables us to write the matching measure as a functional

$$\eta[g]|_{([0,L],[0,L^*])} = \int_0^L \eta[g]|_{([a,a+da],[g(a),g(a+da)])} da. \tag{4.4}$$

The optimal alignment is then given by

$$g^* = \arg\min_g \eta[g]|_{([0,L],[0,L^*])}. \tag{4.5}$$

The cost measure η is in our approach defined as proposed by [Sebastian 03]. Let us consider two infinitesimal curve segments $c|_{[A_1,A_2]}$ and $c^*|_{[A_1^*,A_2^*]}$ of lengths da and da^*, and curvatures κ and κ^*, respectively. Since we compare the intrinsic aspects of these curve segments, we align their start points A_1 and A_1^*, and their respective tangents $\mathbf{t}_{A_1}$ and $\mathbf{t}_{A_1^*}$, see Figure 4.4. The cost

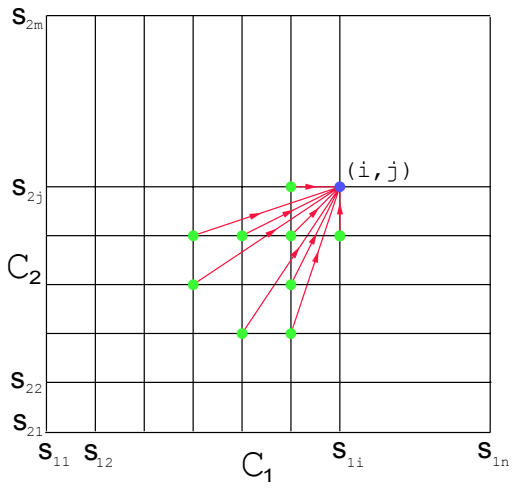


Figure 4.5: This figure illustrates the template that is used to find the edit distance of curve segments using dynamic programming. Discrete samples along the curves are the axes. The entry at (i, j) represents $d(i, j)$. To update the cost at (i, j) (blue dot) we limit the choices of the k and q in Equation (4.8), so that only the costs at a limited set of points (green dots) are considered.

of matching the curve segments is the degree by which the endpoints A_2 and A_2^* differ, which can be formulated as

$$\eta[g]|_{([A_1, A_1+da], [A_1^*, A_1^*+da^*])} = |da^* - da| + R|d\theta^* - d\theta|, \tag{4.6}$$

where R is a scale constant. The functional from Equation (4.4) is therefore

$$\begin{aligned} \eta[g] &= \int_c [|\frac{da^*}{da} - 1| + R|\frac{d\theta^*}{da^*}\frac{da^*}{da} - \frac{d\theta}{da}|]da \\ &= \int_c [|g^{'}(a) - 1| + R|\kappa^*(g(a))g^{'}(a) - \kappa|]da, \end{aligned} \tag{4.7}$$

where the first term penalizes "stretching" and the second term penalizes "bending". In the following section we explain how to calculate the alignment by means of dynamic programming.

4.3.1.1 Finding the optimal alignment curve

This section describes a dynamic programming algorithm, called *dynamic time warping* (DTW), for finding the optimal alignment curve α^* for two curves c_1 and c_2. Dynamic time warping (DTW) is a technique that finds the optimal alignment between two signals (time series) if one signal may be "warped" non-linearly by stretching or shrinking it along its time axis. The warping between

signals can be used to find their corresponding regions or to determine their overall similarity. DTW is often used in speech recognition to determine if two waveforms represent the same spoken phrase. In a speech waveform, the duration of each spoken sound and the interval between sounds are permitted to vary, but the overall speech waveforms must be similar. In addition to speech recognition, DTW has also been found useful in many other disciplines, including data mining, gesture recognition, robotics, manufacturing, medicine and others.

In the optimal alignment, the alignment of curve segments also has to be optimal. The alignment curve of curves c_1 and c_2 is a mapping curve, the axes of which are specified by the curve segments. We discretize c_1 and c_2 at samples $s_{11}, s_{12}, ..., s_{1n}$ and $s_{21}, s_{22}, ..., s_{2m}$, respectively (see Figure 4.5). The alignment curve is then represented by a sequence of N points $(\alpha_1, ..., \alpha_N)$, where $\alpha_k = (s_{1i_k}, s_{2j_k})$, $i_k \in 1, ..., n$, $j_k \in 1, ..., m$, $k = 1, ..., N$, and $\alpha_1 = (s_{11}, s_{21})$ and $\alpha_N = (s_{1n}, s_{2m})$.

Let $d(i, j)$ denote the cost of matching the discrete curve segments $c_1|_{[s_{11}, s_{1i}]}$ and $c_2|_{[s_{21}, s_{2j}]}$. Let $\delta|_{([k,i],[q,j])}$ denote the cost of matching subsegments $c_1|_{[s_{1k}, s_{1i}]}$ and $c_2|_{[s_{2q}, s_{2j}]}$. Due to the optimal substructure property of the distance function (in the optimal alignment, the alignment of curve segments also has to be optimal) we can write

$$d(i, j) = \min_{k,q}[d(i - k, j - q) + \delta|_{([k,i],[q,j])}], \tag{4.8}$$

which is a formula for computing the edit distance $d(c_1, c_2)$ via dynamic programming. The matching cost is found by sequentially updating the dynamic programming table, and the optimal alignment by tracing through the table (see Figure 4.5). We have to compute $d(i, j)$ at every point in the 2D grid. Therefore, the complexity of matching curve segments is $O(n^2)$, where n is the number of samples along the curve segments.

The optimal alignment between the curves is found in the position with the minimal edit distance $d(c_1, c_2)$. This optimal alignment shows the translation of the tracked object (motion along the corresponding direction) between the compared frames.

4.3.1.2 Matching at different scales

Due to the use of intrinsic properties of the curves, the length and the curvature, the curve alignment method we use is invariant to relative translations and rotations of the curves. However, the dissimilarity measure is not scale invariant because the stretching term in the functional is not scale invariant. Therefore, to match curves (signatures) at different scales it is necessary to find a global optimal scale parameter λ for one curve to match the other, and then minimize over several scales. To reduce the influence of scaling on the matching result, it can be done as well by upscaling one curve by $\sqrt{\lambda}$ and downscaling the other curve by the same factor to reach a common curve. In this way, the functional

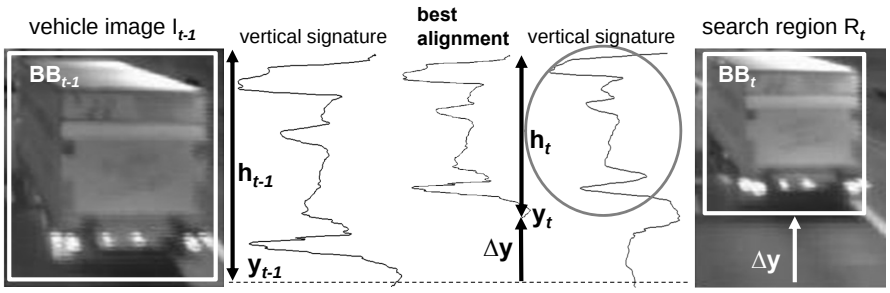


Figure 4.6: Illustration of our signature based measurement for tracking. The position and size (the bounding box) of a vehicle at time t are found by aligning the signatures of vehicle image at time $t-1$ with the signatures of selected region of interest at time t.

from Equation (4.7) becomes

$$\eta_\lambda[g] = \int_c |\sqrt{\lambda} da^* - \frac{1}{\sqrt{\lambda}} da| + R|d\theta^* - d\theta|. \tag{4.9}$$

The optimal scaling factor λ^* is then computed as $\arg\min_\lambda \eta_\lambda[g]$, which can be computed using gradient descent. This optimal scaling factor between horizontal and vertical signatures shows the change of the apparent object size (width and height of its bounding box) between the compared frames.

4.3.1.3 Signature based measurement

As defined in the beginning of this section, let I be the image of an object (a vehicle) at time $t-1$, defined by its bounding box BB_{t-1}, see Figure 4.3. Let further R be a rectangular region defined in such a way that the vehicle between the time instance $t-1$ and the time instance t can move only within the region R. The signature measurement for tracking is then computed by comparing (aligning) the horizontal and vertical signatures of the image I with the signatures of the region R along the same directions. The optimal alignment is obtained as previously explained. The alignments along vertical and horizontal directions define the position and size of the tracked object, i.e. its bounding box at time t, BB_t, within the search region R_t, see Figure 4.6.

4.4 Tracking algorithm

We perform vehicle tracking in the 2D plane of camera images. To infer the real-world positions from the obtained 2D positions, it is possible to use lane marks on the road since their distance in the real-world is known and standardized. The lane marks could be automatically detected or simply marked manually in one of the camera images.

As explained in Section 4.2, we define a state of a vehicle at time instance t as a six-dimensional vector $\mathbf{x}_t = (x_t, y_t, \dot{x}_t, \dot{y}_t, w_t, h_t)$, which contains the vehicle′s position (x_t, y_t) along the x and y image axes , its apparent velocity ($\dot{x}_t$ and $\dot{y}_t$), and apparent size, i.e. the width and height of the vehicle bounding box (w_t and h_t). For tracking we use a linear Kalman filter as a technique for filtering and prediction (see more details in Section 4.2).

As the measurements in the Kalman filter we use the location and size of the object, obtained by our proposed signature based measurement and the foreground mask, together with the mean velocity obtained from the optical flow statistics. We also design a tracker that uses only signature based measurement, without foreground and optical flow, to infer the position and size of the tracked objects as explained in Section 4.3.1.3. In the following section we present the results of these two tracking approaches.

4.5 Results

In Section 4.5.1 we present the results of vehicle tracking in tunnels, showing the benefit of adding the signature measurement next to the foreground and optical flow measurements. In Sections 4.5.2 and 4.5.3 we show the results of using only signature measurement (without foreground and optical flow) in different weather and illumination conditions, demonstrating the robustness of the signature measurement.

4.5.1 Vehicle tracking in a tunnel

Tunnels are environments prone to severe traffic accidents. To enable timely actions that can save lives and minimize the damage, it is important to track vehicles throughout a tunnel. For this purpose, multiple surveillance cameras are typically mounted along tunnels, often with non-overlapping fields of view. As an aid to human operators, computer vision algorithms can then be used for automatic tracking of vehicles. Such algorithms consist of three parts: vehicle detection, tracking of vehicles in a field of view of one camera and vehicle matching, which is used for “handover” of vehicles between the cameras. It is crucial that these algorithms are robust and perform in real-time.

In this context, in Chapter 5 we address the problem of real-time *matching* of vehicles observed by different cameras along tunnels. In this chapter, however, we focused on the problem of vehicle *tracking* in a field of view of one camera. This tracking is challenging due to poor lighting conditions in tunnels, low resolution of vehicle images and frequent light reflections from tunnel walls, road, traffic signalization and vehicles themselves. The most intensive and disturbing light reflections are caused by vehicles with rotating lights, which are also of high importance to be tracked correctly: vehicles that transport poisonous or explosive materials, emergency or law enforcement vehicles. Their rotating lights periodically blind the cameras causing significant artefacts in

Table 4.1: Experimental results - vehicle tracking in a tunnel

Method	Intensive illumination change	Without losses [%]
Without signatures	No	93
With signatures	No	96
Without signatures	Yes	68
With signatures	Yes	89

vehicle images (see Figure 4.2). In such cases it is difficult to extract informative and reliable features for vehicle tracking.

We tested our tracking approach on a real tunnel video sequence acquired by a security camera mounted roughly in the center of a tunnel pipe ceiling and oriented in the direction of the traffic flow. There were in total 150 vehicles recorded passing through the camera view. For the evaluation purpose we manually annotated the vehicles by creating a bounding box around them in each frame.

We compared the tracking results with and without using the proposed image projection clues (the signatures). The comparison was done both in a qualitative and quantitative way, for cases with and without lighting changes. The qualitative results are shown in Figure 4.7, for cases when vehicles are observed with and without severe lighting changes. For each case, the original camera images are given in the first row. The second row shows tracking results obtained using foreground and optical flow as measurements in the Kalman filter, while the results after adding the signatures to the measurement set are shown in the third row. We see that in the case when there are no strong lighting changes (like for the vehicle in the left lane), both trackers, with and without using the signatures, give good results. On the other hand, if lighting changes occur, the tracker that uses the signatures stays on the target, while the tracker without the signatures gets incorrectly updated by the foreground and optical flow measurements, which ultimately causes drift and the target loss (exemplified by the vehicle in the right lane). We see also that even in the case when the camera is partially blinded by the lights the signatures of the vehicle contain enough information to correctly estimate the new position and the size of the vehicle. This is because the vehicle parts that remain visible in the image remain also "visible" in the signatures and correlate well between frames (as shown in Figure 4.2).

Figure 4.7: The tracking results at six different time instances. We assume a vehicle is detected, i.e. the tracker is initialized when a vehicle enters the scene completely. Different bounding box colours represent different vehicles. a) First/fourth row: Original camera images; b) Second/fifth row: tracking results using a Kalman filter with foreground blobs and optical flow as measurements; c) Third/sixth row: tracking results when the signatures based measurement is added to the tracker in b). We see that our tracking approach using signature cues is able to cope with intensive illumination changes.

Figure 4.8: Qualitative results of vehicle tracking in different weather conditions: foggy and snowy. Two video frames and several vehicles are shown as a reference in both video sequences. We see that signature based tracking stays on the target even in these harsh weather conditions.

Quantitative results are given in Table 4.1 and expressed in the percentage of vehicles tracked without losses. We count a vehicle as lost if there is less than 25% overlap between its manually annotated bounding box and the bounding box estimated by the tracker. The results show there are less vehicle losses when the signatures based measurement is added as an additional tracking clue, especially when vehicles are observed under strong illumination changes. The remaining losses are mainly due to occlusions when smaller vehicles (cars) get occluded by big vehicles (e.g. trucks or buses) and stay in such occlusions so long that Kalman filter does not predict well their position. When this happens it is difficult to determine proper measurements for the correction step and often the tracker can not recover after the occlusion.

An additional strong point of the proposed signature based measurement is its computational efficiency. In our C++ implementation, computation of horizontal and vertical signatures and the similarity measure for two vehicle images was achieved in 9 ms on a single-core 1.86 GHz CPU. Such efficiency allowed us to incorporate this measurement into the tracker, still preserving a real-time performance of the tracking algorithm (25 fps).

Table 4.2: Experimental results - vehicle tracking in harsh weather conditions

Weather conditions	Average overlap with GT	Tracking without losses
Fog	71 %	88 %
Snow	76 %	92 %

4.5.2 Tracking in various weather conditions

In this section we present results of our proposed algorithm using two traffic surveillance sequences taken in harsh, foggy and snowy, weather conditions. The sequences are a part of a publicly available Karlsruhe traffic surveillance dataset and we here refer to them as the *Karlsruhe-fog* and *Karlsruhe-snow* sequences. The initial vehicle detections (the initial bounding boxes) we set manually for all vehicles. We also manually created the ground truth by drawing a bounding box around the vehicles in every fifth video frame. Note that in these sequences having the ground truth every fifth frame is sufficient because observed vehicle movements between frames are not abrupt as in tunnel sequences (the camera is positioned higher and vehicles and their motion appear smaller in the image).

Qualitative results are shown in Figure 4.8, with two exemplary video frames for each of the two sequences. These examples show tracking results (bounding boxes) for four vehicles. We see that even in the case of a heavy fog the vehicles can be accurately tracked, even when they are barely visible to the human eye. In the case of snowy conditions, road and vehicles are wet and the contrast between them is low, which together with the snowfall poses additional challenges for tracking. Nevertheless, our signature based tracking is able to perform well even in these conditions.

Quantitative results are given in Table 4.2 and expressed by two measures: the accuracy (the percentage of overlap between the tracker's bounding boxes and the ground truth bounding boxes), and the percentage of vehicles tracked without losses. We count a vehicle as lost in a particular video frame if there is less than 25% overlap between its manually annotated bounding box (ground truth) and the bounding box estimated by the tracker. The results are then averaged for all vehicles and all video frames. The results show that the tracking accuracy is acceptable, and more importantly that the number (percentage) of losses is low.

4.5.3 Tracking in low-light conditions

In this section we present results of our proposed algorithm using a traffic surveillance sequence recorded during night (see Figure 4.9. The initial vehicle detections (the initial bounding boxes) we set manually for all vehicles. We also manually created the ground truth for all vehicles, by drawing a bounding box around them in every fifth video frame. Here, like in the Karlsruhe sequences

Figure 4.9: Qualitative results of vehicle tracking in a night video. Two video frames and several vehicles are shown as a reference. We see that signature based tracking stays on the target even in low light conditions during night.

Table 4.3: Experimental results - vehicle tracking in low-light conditions

Traffic density	Average overlap with GT	Tracking without losses
High	42 %	72 %
Low	63 %	83 %

in Section 4.5.2, having the ground truth every fifth frame is sufficient because observed vehicle movements between frames are not abrupt.

Qualitative results are shown in Figure 4.9, with two exemplary video frames for each of the two sequences. These examples show tracking results (bounding boxes) for four vehicles. We see that even in these very low-light conditions the vehicles can be accurately tracked. This is mainly because the signatures capture their frontal or rear lights and our tracker tracks them. Therefore, we noticed that in this low-light scenario the estimation of a vehicle apparent size is much more challenging and the resulting vehicle bounding boxes are unstable (shrink or stretch relatively frequently).

Quantitative results are given in Table 4.3 and expressed by two measures: the accuracy (the percentage of overlap between the tracker's bounding boxes and the ground truth bounding boxes), and the percentage of vehicles tracked without losses. We count a vehicle as lost in a particular video frame if there is less than 25% overlap between its manually annotated bounding box (ground truth) and the bounding box estimated by the tracker. The results are then averaged for all vehicles and all video frames. The results show that even the tracking in very low-light conditions performs decently, but due to the light reflections, it is significantly less accurate when the traffic is dense.

4.6 Conclusion

In this chapter we demonstrated that using Radon-transform like signatures of viewed objects as an additional clue for object tracking improves the tracking performance. The tracking is robust to illumination changes, which is an important issue since illumination changes are frequent in real-world scenarios. The proposed signature measurements are also efficiently computed using dynamic time warping, so it is possible to include them into tracking algorithms preserving the real-time performance of the tracker. We also showed that the proposed single-camera tracking with signature clues performs well in low-light and different weather conditions. It also has consistent performance from different viewpoints, and signature measurements help to reduce the tracking drift in case of partial occlusions. In the work of Betke *et al.* [Betke 00] it has been shown that object signatures can also be used for object detection. This is especially beneficial for smart cameras where it is desirable to reuse same features for multiple tasks (e.g. object detection, tracking and recognition) to save processing power.

Additional improvement in the signatures based measurement can be done on the size estimation of objects, to avoid occasional shrinking or stretching of the bounding boxes. For this purpose object redetections during the course of tracking can be of great help. Redetections are especially beneficial when objects quickly change moving direction relative to the camera, because in these cases their appearance in consecutive observations can change significantly (e.g. from almost a frontal to almost a side view).

This research was published in the proceedings of several international conferences [Jelača 08], [Jelača 11b], [Frías Velázquez 11], [Jelača 12], and has been submitted to an international journal [Jelača 14].

5

Tracking in Non-overlapping Camera Views

In Chapter 4 our focus was on object tracking in a single camera view. In this chapter and Chapter 6 we go beyond single camera tracking and focus on object tracking in camera networks. From the point of observation completeness there are two types of camera networks: with or without overlapping camera views. In a network with overlapping camera views, each part of the area of interest is viewed by at least one camera and typically by several cameras. There are usually multiple views on an object, from different viewpoints. Tracking needs to fuse information from multiple views to enhance its performance. On the other hand, in a non-overlapping camera network there are "blind" areas where neither of the cameras has a view on the object. In these networks it is necessary to not only track objects in each camera view, but also to re-identify each object when it appears in the other views so that the trajectories in different views can be connected. In this chapter we focus on object tracking in non-overlapping camera views common in traffic surveillance.

For the purpose of traffic management and fast reaction in cases of traffic accidents it is important to timely detect potential incidents or disturbances in the traffic flow. Therefore, surveillance cameras are typically mounted along roads, but for commercial reasons often with non-overlapping fields of view. As an aid to human operators, computer vision algorithms can then be used to automatically detect and track vehicles in the video stream. Such algorithms consist of three parts: vehicle detection, tracking of vehicles in a field of view of one camera (single-camera tracking) and vehicle matching, which is used for a "handover" of vehicles between cameras, i.e. for multi-camera tracking. Typically, results of vehicle detections and single-camera tracking are bounding boxes with vehicle images being regions of interest inside the bounding boxes. Vehicle detections are input to the vehicle matching.

In traditional camera networks the cameras send all acquired data to the

central server that performs video analysis. However, networks of smart cameras open a possibility to process the acquired video data by the cameras themselves and transfer only the obtained metadata to the other cameras and to the central server. In this context, this chapter of the thesis addresses the problem of matching vehicles as they are imaged by a network of stationary smart cameras with non-overlapping views. We focus on the problem of finding a computationally and data efficient, but still discriminative and robust representation of vehicle appearances that can be computed by the cameras themselves and sent between the cameras without sending the whole images. We also focus on finding a computationally efficient algorithm for matching such vehicle representations, suitable for execution on the cameras themselves. Although the framework proposed in this chapter is developed in the context of a vehicle tracking application in tunnels, the basic idea and the associated techniques can be applied to vehicle tracking and matching in general, as well as to matching of other types of rigid objects when the cameras are placed to view them from relatively similar viewpoints.

In traffic surveillance such placement of cameras is achieved in many cases: in tunnels, along roads, or even at intersections of roads by placing more cameras at the crossroads. Still, vehicle matching remains challenging due to significant appearance changes in between cameras, camera view differences and inaccurate and false vehicle detections, which are all common in real-world applications. The vehicle *appearance changes* are due to various reasons: illumination changes in the environment (e.g. a different lighting in different areas of the environment, shadows, light reflections), changes of the vehicle pose as it moves through the multi-camera environment, turning vehicle lights on or off, etc. The *camera view changes* result from differences in camera settings or technical characteristics (e.g. a scale difference due to a different zoom). *Inaccurate vehicle detections* are detections of vehicles or their parts together with a part of the background. They cause misalignment of vehicle images. *False detections*, i.e. detections of the background as vehicle, detections of multiple vehicles as one and multiple detections of one vehicle can cause significant problems to matching algorithms, if not discarded.

In our test application (vehicle tracking in tunnels) there are also some challenges due to a tunnel environment. Firstly, tunnels are often partly dark, artificially illuminated, which makes colour information unreliable (the same holds for outdoor environments in general in cases of poor lighting conditions). Secondly, tunnels are tubular environments, so strong light reflections from the walls and ceiling can disturb cameras and "pollute" the images. Figure 5.1 shows images of six vehicles, acquired in a tunnel by three cameras and automatically extracted from the corresponding frames using the detector proposed by Rios Cabrera *et al.* [Rios Cabrera 12]. The images illustrate a variety of vehicle appearance and observation changes. Moreover, if vehicle images are of low to medium resolution, which is common in video surveillance, the motion blur and noise in the images are significant. This imposes an additional challenge for extraction of robust features from vehicle images.

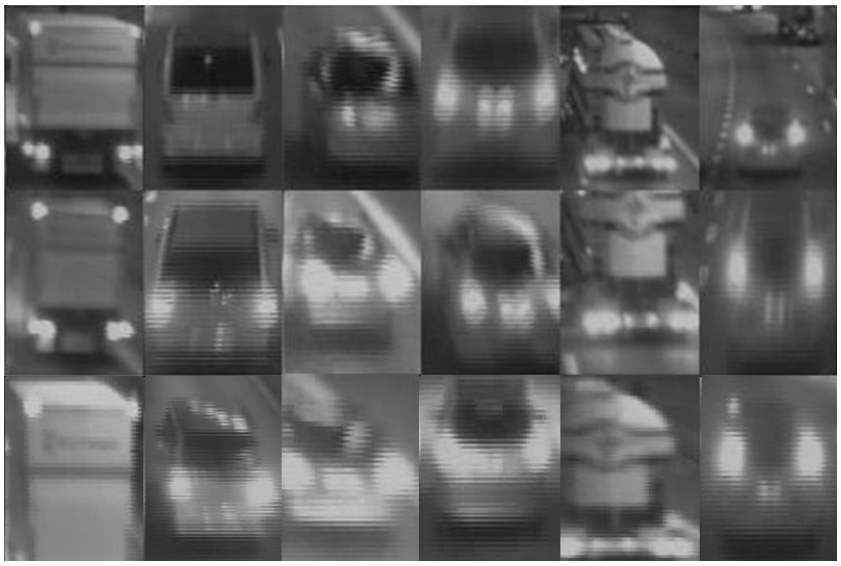


Figure 5.1: Each column contains vehicle images (detections) of the same vehicle observed by three cameras along a tunnel. The cameras are mounted roughly in the middle of the tunnel ceiling and oriented in the direction of the traffic flow. From left to right the images illustrate a vehicle appearance change due to different level of visible details when the detections are taken at different distance from the camera, turning on/off the rear lights, change of the scene illumination, change of pose as vehicle moves away from the camera or changes lane, and inaccurate detections.

Previous work on object appearance matching has mainly focused on extracting robust features from acquired images, so that those features remain invariant to appearance changes [Turk 91, Bischof 04, Sidla 04, Lowe 04, Bay 08, Yu 11, Shan 08, Guo 07, Porikli 03, Javed 05, Hou 09]. Many different features have been proposed, based on colour, local features, edges, image eigenvectors or entropy, all with limited success in achieving the goal of invariance. Calculation and matching of such features is also often computationally demanding, so object comparison in real-time is typically done using only one image per camera for each object. Therefore, the accuracy of these approaches strongly depends on the quality of observations and the matching is much more challenging if observations contain disturbances like light reflections, strong shadows or occlusions.

In our work we try to overcome these problems by a conceptually different approach, based on two novelties in vehicle matching. Firstly, we use simple descriptors of vehicle appearances that are easy to compute and compare, yet highly informative in low resolution images. For this purpose we model vehicle appearances using the signatures proposed in Chapter 3, that are Radon transform like projection profiles of the acquired vehicle images. Matching of the

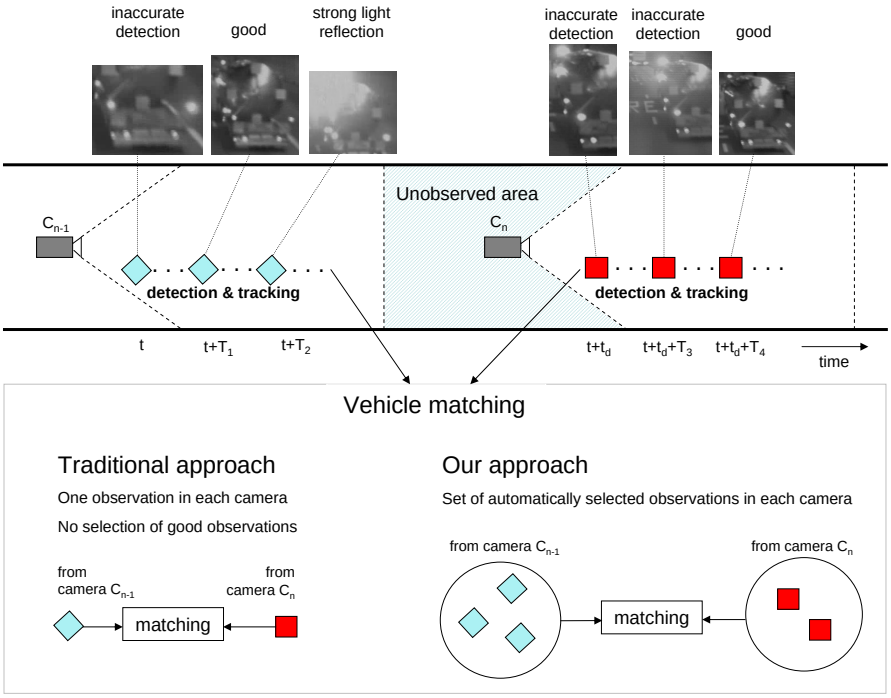


Figure 5.2: Observations of a vehicle along the trajectory viewed by two consecutive cameras. The images can be very different due to inaccurate detections and significant appearance changes. The key idea of our approach is to reduce this problem by preselecting good observations before performing the inter-camera matching itself. We use multiple good observations from each camera to allow for difference in appearance.

appearance models is then obtained by a simple combination of 1-D correlations in a coarse-to-fine procedure. The signatures are also used to learn scale differences between the observations from different cameras, which is important for their alignment. The second novelty is to use signatures from multiple images for creating a multiple observation appearance model and automatic selection of good observations for matching (i.e. informative observations with few disturbances), as shown in Figure 5.2. Such an appearance model enables representation of vehicles from multiple views, collected online as they move through the multi-camera environment. This is especially beneficial when vehicles change pose (e.g. by changing lane or moving away from the camera). Finally, since each vehicle has one and only one corresponding vehicle in other cameras, we employ the Hungarian algorithm to resolve ambiguities and to optimize the matching.

The remainder of this chapter is organized as follows. Section 5.1 gives related work on multi-camera tracking, object representation and matching.

Section 5.2 briefly formulates the problem of vehicle matching and tracking in non-overlapping views. In Section 5.3 we propose a novel appearance model based on the vehicle signatures, together with the procedure for collection of good observations along the vehicle trajectory. Matching of vehicle appearances using the proposed appearance model is explained in Section 5.4. The complete matching algorithm that optimizes the association of vehicle correspondences is given in Section 5.5. In Section 5.6 we present and discuss the experimental results and finally, we conclude in Section 5.7.

5.1 Related work

Most of the work on multi-camera tracking by cameras with non-overlapping views, e.g. [Huang 97, Kettnaker 99, Collins 01, Porikli 03, Javed 03, Javed 05, Guo 07, Choe 10], uses object appearance representations based on colour information (e.g. mean, histogram or correlogram of the colour). Colour alone is, however, not reliable as a feature in many traffic surveillance applications, especially in tunnels. To address such a problem [Porikli 03], [Javed 05] and [Hou 10] present a method for matching object appearances by calculation of a brightness transfer function for every pair of non-overlapping cameras. They map an observed colour value in one camera to the corresponding observation in the other camera. Once such a mapping is known, the correspondence problem is reduced to matching of the transformed appearance models. However, real illumination often varies between frames and scenes depending on a large number of parameters, which is very difficult to model. Moreover, colours of artificial lights in tunnels or in artificially illuminated environments in general can supersede vehicle colours, which makes the mapping of vehicle colours even more challenging (especially in presence of variable road signs, rotating and emergency lights, etc.).

Appearance representations that do not need colour information are often based on eigenimages (often used for face recognition) [Turk 91, Bischof 04], local invariant features (e.g. SIFT [Lowe 04], SURF [Bay 08] or ASIFT [Yu 11]) or edge maps [Shan 08, Shan 05, Guo 07].

Methods based on *eigenimages* require offline training and their accuracy highly depends on variations of objects and their appearances present in the training set. Therefore, adaptation of these methods to appearance changes is limited. These methods also require alignment of objects before matching, which is an additional challenge in real world scenarios.

The accuracy of methods based on *local features* depends on the number of corresponding key points found in images and on the dimension of the local descriptors calculated for each key point. In our experiments with vehicle images acquired by surveillance cameras, too few reliable and unique features were found and thus many features were wrongly matched. Similar findings have been reported in previous works [Guo 07, Shan 08]. Also, calculation of high dimensional local descriptors is computationally demanding, which creates additional difficulties to use local features for real-time vehicle matching.

In the context of *edge* based methods [Shan 08] has proposed a measurement vector and an unsupervised approach to learn edge measures for matching vehicle edge maps. The edge maps are compared after spatial alignment. A solution for the alignment has been proposed in [Guo 07]. The reported results show that the learned edge matching measures can be relatively invariant to changes in illumination and vehicle pose. This invariance is further increased by [Shan 05], by matching embedded vehicle descriptors instead of direct vehicle matching between different camera views. The embedded descriptors were obtained by matching vehicles with exemplar vehicles from the same camera view. However, automatic selection of good exemplars has remained a problem. Also, the learned edge measure weights indicate the illumination and aspect differences between two scenes, but not the quality of compared observations themselves (some observations can be influenced more than others by certain changes, especially if changes are temporary and only in some parts of a scene).

Beside the mentioned features and approaches, recently *Haar-like* features have been proposed for vehicle matching [Rios Cabrera 12]. These features are often successfully used for object detection [Viola 04, Rios Cabrera 12], so reusing the same features for matching reduces the computational cost of the matching itself. The work of [Rios Cabrera 12] shows that, indeed, reusing Haar-features for vehicle matching in tunnels is possible and can be highly accurate if the training set of vehicle images is acquired in the same tunnel and under similar environmental conditions as the testing vehicle images. This condition is, however, difficult to meet in real world applications, which is a limitation of this approach. The training set needs to be large enough to include vehicle images from various environments (e.g. different tunnels, different cameras) and various environmental conditions (e.g. different lighting, wet/dry road, different aspect of vehicles etc.). This further increases complexity of the training process. Therefore, in our work we are focused on finding a vehicle matching approach that does not require supervised training and has a built-in procedure for collecting various vehicle appearances to create more informative appearance models.

Our method for vehicle appearance matching is an extension of the signature based appearance modelling explained in previous chapters, mainly Chapters 3 and 4. In this chapter we extend the concept of signature based tracking to the environments observed by multiple cameras with non-overlapping views. In these environments it is necessary to track vehicles in each view separately and re-identify (match) them when they appear in new views. Due to temporal and spatial differences between different views there are significant changes in illumination and poses of vehicles, as well as in vehicle detection. Vehicles can be inaccurately detected so that only their parts are captured by detection, the background (a road, shadows or some other objects) can be wrongly detected as a vehicle, or some vehicles can pass through some camera views without being detected. Therefore, vehicle matching needs to be robust to all these situations. Using the signatures for automatic selection of good observations for matching is another step forward of our method compared to previous

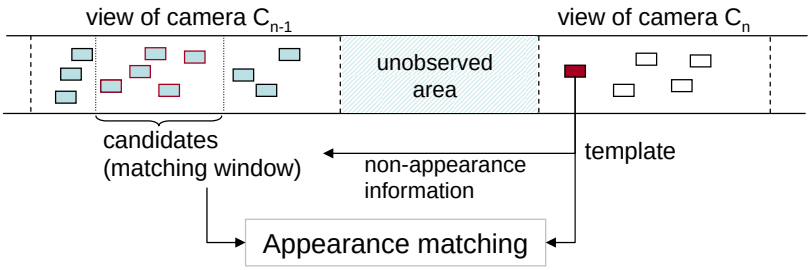


Figure 5.3: The problem illustration. For each vehicle (a template) from camera C_n we find a set of possible matching candidates from camera C_{n-1} using non-appearance information (road constraints, inter-camera distances and vehicle kinematics). A template-candidate matching is then obtained based on similarity of vehicle appearances.

works. Such an automatic selection could also be useful for selection of vehicle exemplars in [Shan 05] and [Rios Cabrera 12].

In addition to appearance matching, previous work, [Huang 97, Kettnaker 99, Collins 01, Porikli 03, Javed 03, Rahimi 04, Markis 04, Stauffer 05], has given significant attention to object matching using "extra" information that does not come from the tracked objects alone or is not based on their appearance (e.g. camera calibration, topology of the camera network and allowable movement paths, a site model, motion characteristics, transition probabilities, etc.). In our work we use the motion trajectories, vehicle velocity and estimated inter-camera travel time to constrain the number of matching candidates for each vehicle (all this information is automatically obtained from a single-camera tracking). However, since the focus of this chapter is on matching of vehicle appearances, in this chapter we present results obtained both with and without using this "extra" information, to show how discriminative the proposed appearance matching is by itself.

5.2 Problem formulation

We define the vehicle matching problem as the problem of classifying pairs of vehicles observed by cameras with non-overlapping views in the categories "the same vehicle" or "different vehicle". Without losing generality we can assume that these cameras are consecutive in a predefined sequence of cameras, so we denote two of them as C_n and C_{n-1} (camera C_{n-1} being the one that vehicles pass before reaching camera C_n). The match score between vehicle appearances, μ, is defined as

$$\mu = f(A_i^{n-1}, A_j^n), \tag{5.1}$$

where f is a similarity measure between two appearance models A_i^{n-1} and A_j^n corresponding to the i-th and j-th observations O_i^{n-1} and O_j^n in cameras C_{n-1} and C_n, respectively. We call the model A_j^n the template and the model A_i^{n-1} the candidate.

In the context of online (real-time) multi-camera tracking, vehicle observations are responses of vehicle detection and single-camera tracking. For each template a set of possible candidates (a temporal matching window) is defined according to road constrains, inter-camera distances and vehicle kinematics, see Figure 5.3. A template-candidate association is then obtained according to the matching score μ, assuming that each template inside its matching window has one and only one corresponding candidate.

5.3 Robust multi-observation appearance model

While moving through the multi-camera environment the appearance of vehicles between observations (detections) can change due to many reasons. For robust appearance matching it is essential to create an appearance model of each vehicle using a diversity of good observations. In this section we analyse observations in tunnels and explain how signatures can be used for automatic selection of good observations.

The images in Figures 5.4 and 5.5 are typical examples of vehicle observations in tunnels (the images are rescaled to the same size for easier visual comparison). Their signatures are also presented. Figure 5.4 (top) shows observations of the same vehicle, acquired by one camera. They illustrate the appearance change caused by a different camera viewing angle when the vehicle moves away from the camera. As the vehicle moves away from the camera it appears smaller and its lights and license plate appear bigger due to the light dissipation effect. Some parts of the vehicle are even not visible any more (e.g. the roof) and some other parts become more visible (e.g. the back window). As certain vehicle parts appear bigger or smaller, the corresponding signature parts stretch or shrink. Hence, these observations contain different, but complementary information and representing possible variations in appearance increases informativeness of the appearance model. This is especially important because vehicles are observed at various distances and from various angles.

A second example, in Figure 5.4 (bottom), shows vehicle appearance change due to the actions taken by the driver, in this case turning on the rear lights. If we compare the signatures that represent the cases when the lights are off and on, we see that there is a corresponding change in values and behaviour of the signatures. As in the first example, having the observations with lights off and on in the appearance model increases its robustness, because the vehicle can be captured in both states in other cameras.

Conversely, some observations should not be included in the appearance model, particularly false and inaccurate detections, clutters and occlusions. For detecting them we exploit the fact that the signatures in such observations change differently than in the two aforementioned situations. When occlusions

occur, a new object in the image causes a significant change of the observations (see Figure 5.5). This change is gradual as the vehicle gets more or less occluded, until it reaches the unoccluded state again or the state in which the occlusion is constant. This behaviour is present in the signatures as well. Also, strong illumination sources can blind the cameras or significantly disturb observations, e.g. when vehicles with rotating lights enter the scene those lights are periodically disturbing the camera and "polluting" the observations. In consecutive frames this effect is again visible as gradual change in vehicle images. Analogous phenomena outside of tunnels can be observed due to reflection of sunlight from vehicles or cast shadows from objects alongside the road.

False and inaccurate detections can also be detected by analysing signatures. False and inaccurate detections occur in real scenarios regardless of the vehicle detector that is used, mostly due to lack of visible features, intensive illumination changes in some parts of the scene or light reflections on the road. Our experiments showed that such detections are typically unstable, i.e. they change quickly, capturing different vehicle parts in consecutive frames (see Figure 5.5 the bottom row). As a consequence, the signatures of false and inaccurate detections also change quickly and not gradually.

The stated signature characteristics allow us to use them for selection and representation of good appearance states that should be included in the appearance model. The selection procedure is presented in Figure 5.6. Let A be the appearance model and $\mathbf{s}_t$ the signature vector of the appearance state observed at the time instance t, defined by Equations (3.3) and (3.4). We consider that the appearance state is stable if the appearance remains similar enough in a predefined number of successive frames T. In terms of signatures this condition is satisfied if a similarity measure μ_s between the signature vectors of these T successive observations remains above some predefined similarity value M_{st},

$$\mu_s(\mathbf{s}_t, \mathbf{s}_{t+\tau}) > M_{st}, \forall \tau \in [1, T]. \tag{5.2}$$

We call this condition the stability criterion, with M_{st} and T being the stability parameters. A method for calculating the similarity μ_s between the signature vectors is given in detail in Section 5.4. If the appearance state at the time instant t is stable, its signature vector should be included in the appearance model $A \equiv \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_N\}$ only if it brings additional information to the model, i.e. if it is different enough from other, previously observed states already included in the model,

$$\mu_s(\mathbf{s}_t, \mathbf{s}_n) < M_{var}, \forall n \in [1, N]. \tag{5.3}$$

This condition represents variability criterion and M_{var} is the variability threshold.

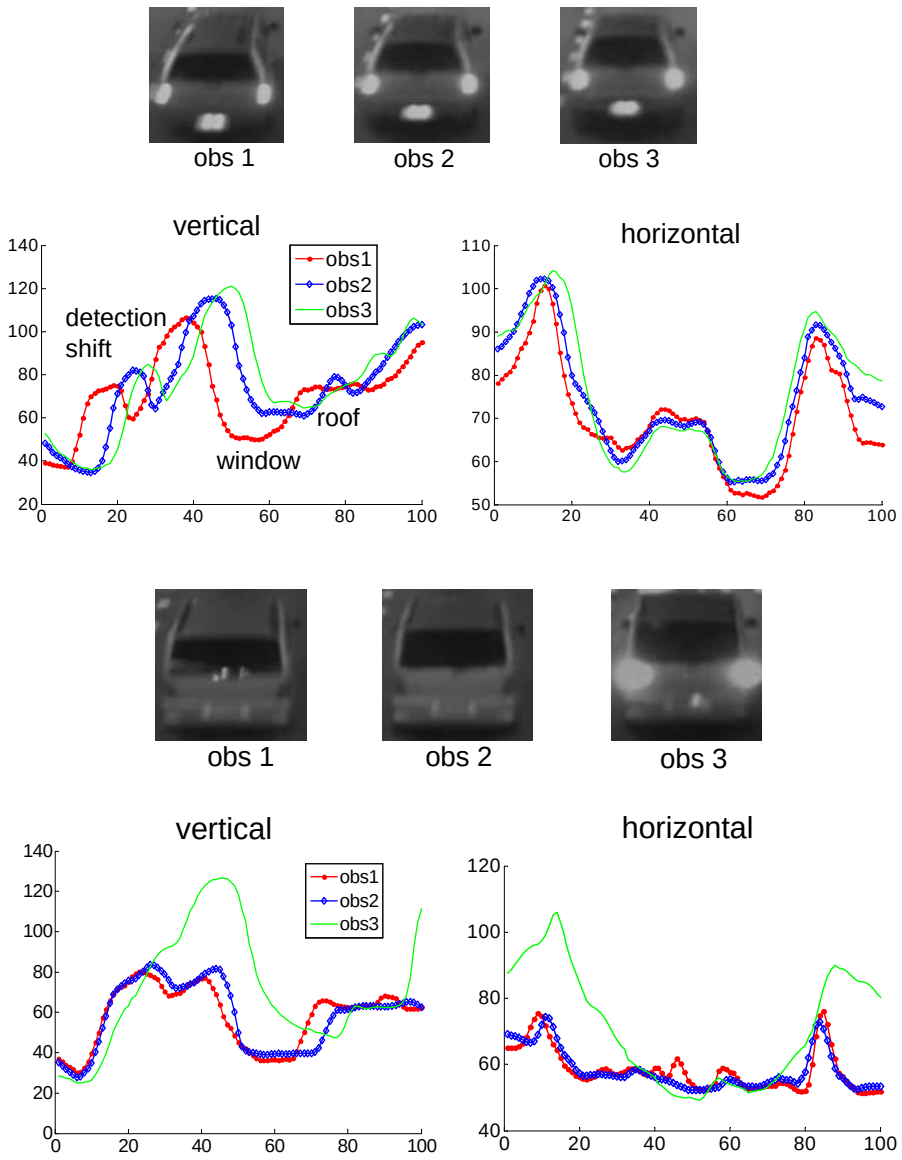


Figure 5.4: *Vehicle appearance changes.* Images of vehicles observed by one camera and their corresponding vertical and horizontal signatures; Top: A typical case of the vehicle appearance change when vehicles are observed by a camera mounted on a tunnel ceiling: first, the vehicle is observed from above and then, as it moves away from the camera, it is viewed from behind, so some of its parts become more and some less visible; Bottom: A vehicle appearance change when its lights turn on: signature parts that correspond to the lights gain higher value and the number of implied pixels when the lights are on.

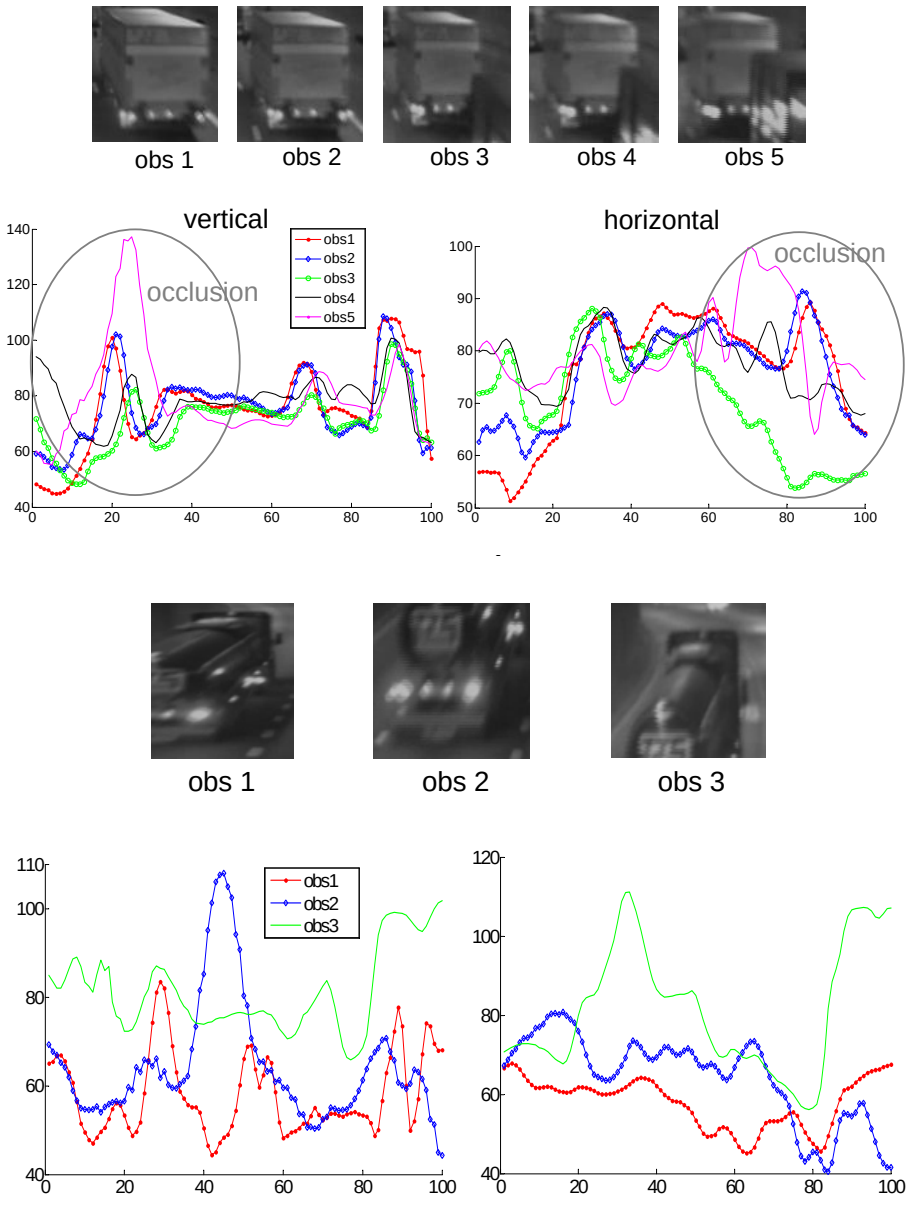


Figure 5.5: *Vehicle occlusion and inaccurate detection.* Images of vehicles observed by one camera and their corresponding vertical and horizontal signatures; Top: An occlusion from another object: there is a significant gradual change of the vehicle signatures in the parts affected by occlusion; Bottom: Inaccurate detections; the signatures change quickly, not gradually between consecutive frames.

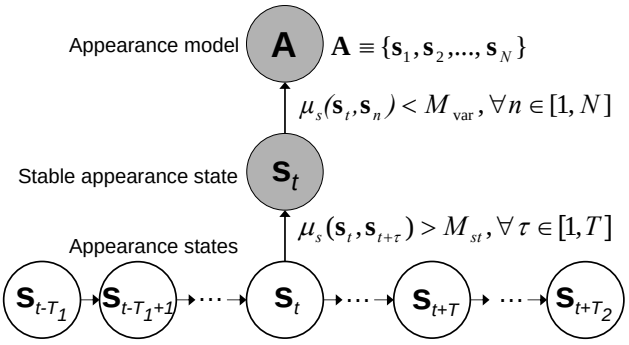


Figure 5.6: A procedure for collection of good observations for modelling the appearance of vehicles as they move in a multi-camera environment; the stability and variability conditions for including appearance states into an appearance model are given. All appearances are represented by signature vectors.

By using the proposed appearance collection procedure, most occlusions, clutters and inaccurate and false detections can be excluded from the appearance model (due to the stability condition). However, the model can still include persistent inaccurate detections for which both a vehicle detector and a single-camera tracker constantly return a stable, positive response. Such persistent inaccurate detections typically occur when bigger vehicles (e.g. trucks or buses) are partly detected or when the detections of smaller vehicles (e.g. cars) contain parts of the background or other vehicles. We noticed also that for some vehicles all stable detections were inaccurate, so our intention in this work was to develop a signature matching procedure robust to such detection inaccuracies.

5.4 Vehicle appearance matching

5.4.1 Signature matching

The challenges for robust matching of the signatures come from scale, shift and rotation variations between the vehicle observations that are compared. *Scale differences* result from different camera zoom settings or different distances between the observed vehicles and the camera. *Shift* results from differences in the bounding box location with respect to vehicles. *Rotation* is caused by vehicle pose changes together with camera viewing angle changes. All these effects are present in the example of Figure 5.7. Due to the scale difference the lengths of the corresponding parts of the signatures differ. A consequence of the bounding box shift is the signature shift along the shift direction while the vehicle rotation results in shrinking and stretching of the signature parts. Thus, we propose a coarse-to-fine signature matching procedure composed of

four parts: signature rescaling, and global and local alignment, followed by calculation of the final similarity measure.

5.4.1.1 Learning of rescaling factors

To achieve the scale invariance necessary for signature matching, we rescale the signatures by estimated factors using cubic interpolation. In the following we present a method to estimate these rescaling factors. They depend on the camera zoom settings and the position of the vehicle in the scene (further from the camera smaller the vehicle image, i.e. shorter the vehicle signatures and vice versa). We represent the vehicle position by the y-coordinate of its bounding box bottom line (see Figure 5.7a). Suppose we want to determine the rescaling factors between the vertical signatures of two vehicle images O_i^{n-1} and O_j^n, extracted at positions y_i^{n-1} and y_j^n . We define the rescaling factor for vertical signatures as following:

$$r_v^{ji}(y_j^n, y_i^{n-1}) = \frac{l_{v_j}^n}{l_{v_i}^{n-1}}, \tag{5.4}$$

where $l_{v_j}^n$ and $l_{v_i}^{n-1}$ are the lengths of the vertical signatures of images O_j^n and O_i^{n-1}, respectively. We rescale the signatures to the same reference length l_r and compare using 1-D correlation. If the obtained correlation coefficient is high enough, i.e. above a predefined threshold (0.9 in our experiments), the reference rescaling factors $r_v^j = l_{v_j}^n / l_r$ and $r_v^i = l_{v_i}^{n-1} / l_r$ properly estimate the scale difference between the vehicles at positions y_i^{n-1} and y_j^n. Then, the rescaling factor r_v^{ji} is calculated as $r_v^{ji} = r_v^j / r_v^i$ and we use it as an estimate of the scale difference between observations at positions y_i^{n-1} and y_j^n. If the obtained correlation coefficient is bellow the threshold, such a signature pair is considered unreliable, so it is not used for the rescaling factor learning.

In this way we estimate the rescaling factors for different pairs of positions in two fields of view. Taking into account that vehicles get detected at multiple positions along their trajectory in each camera view, the estimation of the rescaling factors can be done fairly quickly for many position pairs. If there is no rescaling factor for a certain position pair, we use the factor for the nearest position pair. If there are multiple factors for the same position pair, the arithmetic mean of the latest n is used ($n = 3$ in our experiments). This enables automatic adaptation to the change of the camera zoom parameters. The rescaling factors for horizontal and diagonal signatures are estimated analogously.

Note that by dividing the images in regions, see Figure 5.7a, it is possible to group vehicle positions and to learn rescaling factors for pairs of image regions instead for position pairs. This enables faster learning, but the estimated rescaling factors are less precise. However, if the scale differences within the same regions are relatively small (in our experiments 15%), the global and local alignment can still be properly obtained.

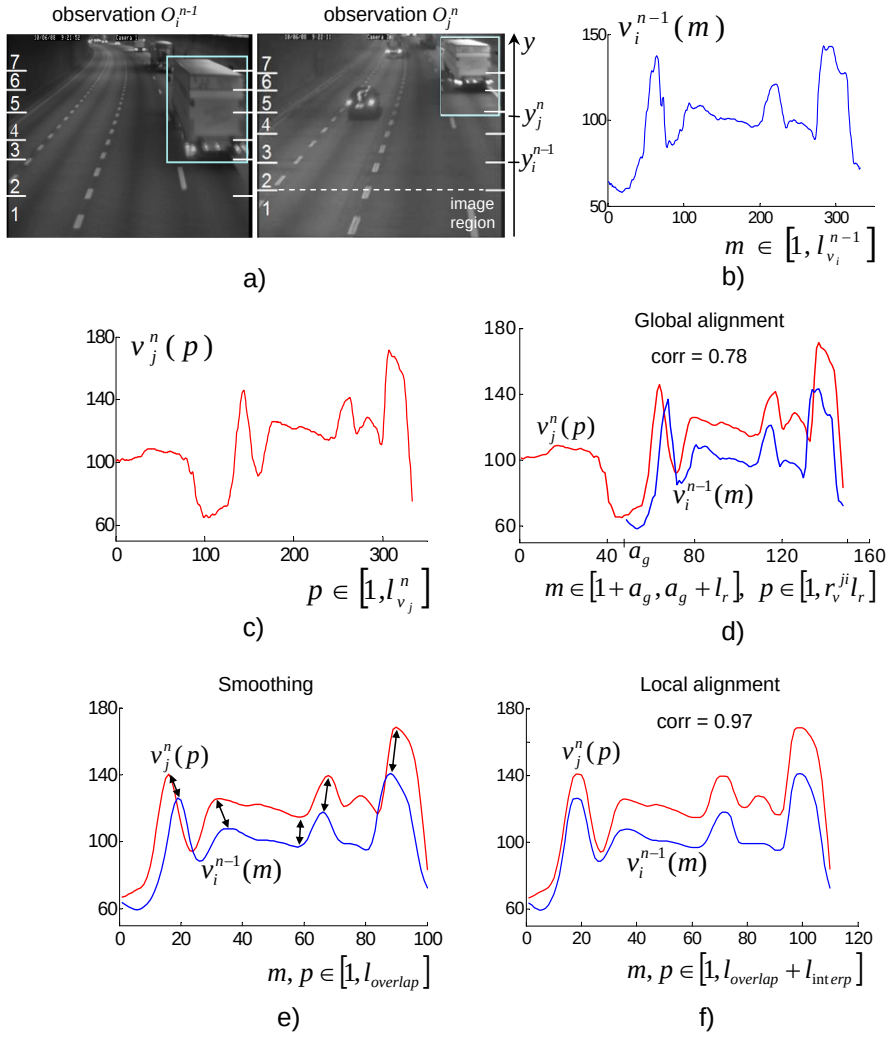


Figure 5.7: The signature matching procedure: a) Two observations of the same vehicle captured by two cameras (the same as in Figure 3.7); The y-coordinates of the bounding box bottom lines represent the positions of vehicles in the scenes; The images are divided in regions (marked with numbers) so that vehicle scale difference within each region remains less than 15%; b) Vertical signature of the observation O_i^{n-1} (good detection); c) Vertical signature of the observation O_j^n (inaccurate detection); d) Signatures are rescaled using the reference length $l_r = 100$ using the rescaling factor $r_v^{ji} = 1.48$, estimated between the positions y_j^n and y_i^{n-1} in images from cameras C_n and C_{n-1}, respectively; after rescaling, the global alignment is found in the position a_g; e-f) The local alignment: unstable extrema are removed by smoothing the signatures; the remaining local extrema (some of which are marked with arrows) are aligned by interpolating the signatures; after local alignment the final matching measure between the signatures is calculated.

5.4.1.2 Global alignment by correlation with shifting

Due to the possible signature shift, it is necessary to align the signatures before comparing them (see Figures 3.7 and 5.7d). The alignment we perform is twofold. First, after rescaling the signatures by the estimated factor we align them globally. Then, a finer, local alignment is obtained. The global alignment is done by shifting one signature along the other one, finding the position with the highest correlation coefficient between the two signatures. Suppose $\mathbf{x}$ is the signature with M elements and $\mathbf{y}$ the signature with $N > M$ elements. The signature $\mathbf{x}$ is then shifted along $\mathbf{y}$ and the correlation coefficient ρ_s, obtained in each shift position $s \in [0, N-M]$ is defined as

$$\rho_s = \frac{\sum_{i=1}^{M}(x(i)-\bar{\mathbf{x}})(y(i+s)-\bar{\mathbf{y}}_s)}{\sqrt{\sum_{i=1}^{M}(x(i)-\bar{\mathbf{x}})^2(y(i+s)-\bar{\mathbf{y}}_s)^2}}, \tag{5.5}$$

where $\mathbf{y}_s$ is the part of the signature $\mathbf{y}$, which in shift position s overlaps with the signature $\mathbf{x}$. The signatures are aligned in a position a_g, in which the correlation coefficient ρ_s has maximal value ρ_g,

$$\rho_g = \max_s \rho_s. \tag{5.6}$$

This is a coarse, global matching measure of two signatures.

5.4.1.3 Local alignment and signature matching measure

Perspective changes of the vehicle observation, subtle appearance changes and imprecise rescaling cause shrinking and stretching of signature parts (see Figures 3.7 and 5.7d). Hence, a local alignment of signatures is needed before calculating their correlation. For that purpose we propose a method similar to Iterative Closest Point (ICP) [Rusinkiewicz 01]. Our method aligns corresponding local extrema. Local extrema are robust features of the signatures, preserved even if vehicles change pose or if they are observed in different illumination conditions. This is because they correspond to different parts/ patterns of the vehicle. As long as those parts/patterns remain visible in two observations, the local extrema remain present in the signatures (see Figure 3.7). Therefore, we propose the following local alignment method.

Step 1. The signatures are iteratively smoothed until the same number of extrema is found in two consecutive iterations, or until the number of signatures' extrema remains above a predefined limit (in our experiments we set this limit to 10). Smoothing removes most of the extrema that originate from noise and camera interlacing. Fine appearance details can also be lost, but due to the low resolution of vehicle images they are mostly not present.

Step 2. The signatures are iteratively interpolated to align the closest local extrema of the same kind (maximum or minimum), see Figure 5.7e-f. Suppose $\mathbf{x}$ and $\mathbf{y}$ are two signatures. For each local maximum $x(m)$ we find its closest maximum $y(n)$, i.e. the one for which the absolute difference in their position

$|m - n|$ is minimal. In the same way the closest minimum is found for each local minimum of the signal $\mathbf{x}$.

Ambiguities occur if multiple extrema from the signature $\mathbf{y}$ have the same closest extrema in the signature $\mathbf{x}$. Therefore, the local alignment is performed in iterations. Firstly, the signatures are interpolated so all extrema with a unique correspondence are aligned. We used cubic interpolation for this purpose. After interpolation some of the ambiguities might be resolved. Then, the whole procedure of finding the closest extrema and aligning them repeats until all extrema with unique correspondence are aligned.

Note also that other possible curve alignment approach is dynamic time warping (DTW), e.g. the method of [Sebastian 03]. DTW automatically handles both scale and translation effects globally and locally. It can be implemented in dynamic programming so it is also efficient. However, in real scenarios the signatures can be significantly misaligned so for a proper initialization of DTW (selection of the starting and ending point) a coarse global alignment is still an advantage. Moreover, the vehicle signatures taken at different lighting conditions can vary significantly in intensities, which further can lead to inaccuracies of DTW. Therefore, we found that using the proposed iterative ICP-alike approach is a better option.

Finally, after the local alignment, 1-D correlation coefficient ρ_l between the signatures is calculated. We define the final matching measure between the signatures $\mathbf{x}$ and $\mathbf{y}$ as

$$\rho(\mathbf{x}, \mathbf{y}) = \begin{cases} \rho_l(\mathbf{x}, \mathbf{y}), & \rho_l(\mathbf{x}, \mathbf{y}) > 0 \\ 0, & \rho_l(\mathbf{x}, \mathbf{y}) \leq 0. \end{cases} \tag{5.7}$$

Negative values of the correlation coefficient ρ_l are set to zero in the signature similarity measure ρ. This is because we do not make a difference between no correlation and negative correlation since these are both cases when the compared vehicles are different.

5.4.2 Matching of the appearance models

As explained in Section 5.3, the appearance of each vehicle is modelled by multiple appearance states of which each is represented by a signature vector. In this work we used signature vectors that consist of two and four signatures. Therefore, if A is the vehicle appearance model, it is a set of N signature vectors, $A \equiv \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_N\}$, of which each is represented by one vertical and one horizontal signature, $\mathbf{s}_n = (\mathbf{v}_n, \mathbf{h}_n)$ or with diagonal signatures added, $\mathbf{s}_n = (\mathbf{v}_n, \mathbf{h}_n, \mathbf{d}_n, \mathbf{a}_n)$, $n \in [1, N]$. The correlation between signatures is calculated as presented in Section 5.4.1. Comparison of the appearance states requires then combining the correlation coefficients between signatures into one matching measure between signature vectors.

Figure 5.8 shows 2-D and 3-D scatter plots of the correlation coefficients ρ_l for the pairs of vertical, horizontal and main diagonal signatures of the 300 vehicles in our database (each compared with 21 candidates). Red circles

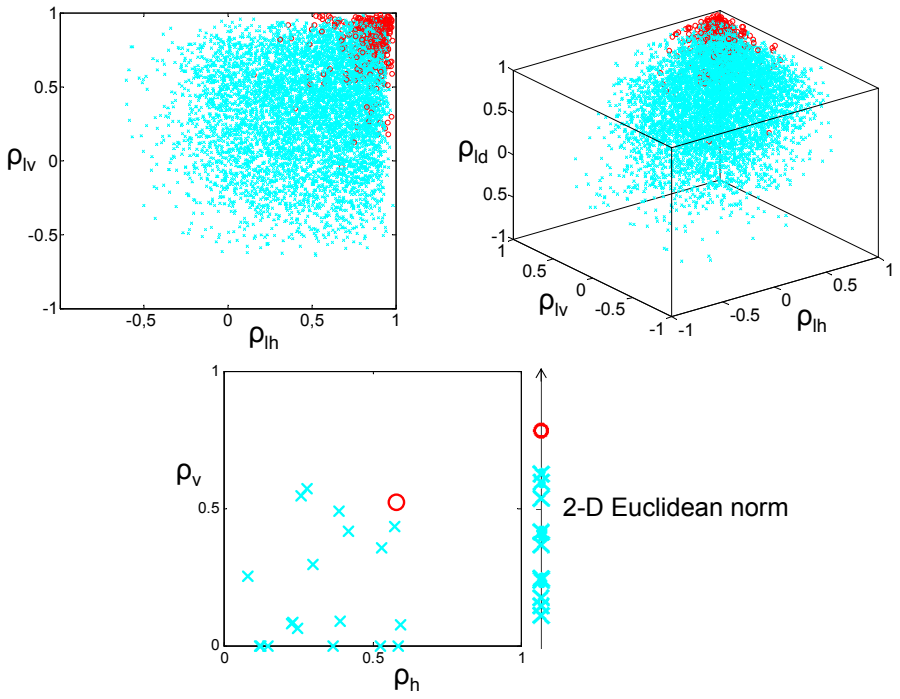


Figure 5.8: Top: 2-D and 3-D scatter plots of the correlation coefficients ρ_l between the signatures of the vehicle images from our database; each vehicle from one camera (template) is compared with the same vehicle and 20 different vehicles viewed by the other camera; on the x, y and z axes are correlation coefficients between pairs of horizontal, vertical and main diagonal signatures, ρ_{lh}, ρ_{lv} and ρ_{ld} respectively. Red circles represent values for the same vehicles while cyan crosses represent values for different vehicles. The correlation values for the same vehicles are clustered in the area with high values for each of the signature pairs. Bottom: example for one template and its candidates; 2-D scatter plot of the signature similarity measures ρ. Even when the similarity measures ρ_h and ρ_v are relatively low for a true match, a true match is still further from the zero correlation point than false matches.

represent the correlation values between the signatures of the same vehicles (according to the manually annotated ground truth) and cyan crosses represent the values for different vehicles. As expected, the values for the same vehicles are clustered in the area with high correlation coefficients for the each signature pair, i.e. the area furthest from the zero correlation point (point $(0, 0)$ for 2-D plot or $(0, 0, 0)$ for 3-D plot). Also, even if in some cases the correlation values of true matches are not in the cluster of red circles their distance from the zero correlation point is mostly still higher than the distance for false matches, as in the example at the bottom in Figure 5.8. Therefore, we define a similarity measure μ_s between two signature vectors as the Euclidean norm of an n-D

vector, where each dimension represents the similarity measure ρ between the signatures along the same projection direction, i.e.

$$\mu_s(\mathbf{s}_n, \mathbf{s}_m) = \|(\rho(\mathbf{v}_n, \mathbf{v}_m), \rho(\mathbf{h}_n, \mathbf{h}_m))\| \tag{5.8}$$

for the appearance representation by 2-D signature vectors, or analogously the Euclidean norm of a 4-D vector when the appearance states are represented by 4-D signature vectors.

Finally, the matching measure μ between two appearance models $A_p \equiv \{\mathbf{s}_1^p, \mathbf{s}_2^p, ..., \mathbf{s}_M^p\}$ and
$A_q \equiv \{\mathbf{s}_1^q, \mathbf{s}_2^q, ..., \mathbf{s}_N^q\}$ is the maximal similarity measure obtained when comparing all their states,

$$\mu(A_p, A_q) = \max_{m,n} \mu_s(\mathbf{s}_m^p, \mathbf{s}_n^q), \quad m \in [1, M], n \in [1, N]. \tag{5.9}$$

This means that the vehicle matching is done according to the most similar appearances in the vehicle appearance models.

5.4.3 Template-candidate association

Each template is compared with all its candidates according to the procedure given in Section 5.4.2. Since every template has one and only one corresponding candidate in its matching window, we optimize the template-candidate association by finding a solution with a maximal sum of all individual similarity measures μ (see Figure 5.9). We use the Hungarian algorithm with voting [Rios Cabrera 12] to solve this maximum assignment problem. Since the Hungarian method has been initially developed to calculate an assignment of jobs to workers that has minimal total cost, we convert the maximum assignment into a minimum assignment problem by multiplying all similarity measures by factor -1. Note that when it is necessary to establish template-candidate associations as quickly as possible, the assignments could be made based only on matching measures μ and optimized with a delay, when all templates that could be matched to the same candidates pass through the camera view.

5.5 Vehicle matching algorithm

Given the problem of matching vehicles observed by two cameras with non-overlapping views, C_n and C_{n-1}, formulated in Section 5.2, our matching algorithm consists of the following four steps.

Step 1. During the movement of the j-th vehicle through the field of view of camera C_n its appearance model (template T_j) is created using the procedure explained in Section 5.3. The vehicle observations are responses of vehicle detection and single-camera tracking.

Step 2. The matching window (the set of candidates) for the template T_j is determined. It is done according to the distance $D_{n,n-1}$ between the cameras

C_n and C_{n-1}, together with the estimated velocity and the trajectories of the vehicles as observed by camera C_{n-1}. We estimate the velocity according to the lane marks on the road, using responses of the single-camera tracking. The distance between the lane marks is known (complies with the known standards) so the velocity is estimated by measuring the time vehicles need to move between the lane marks. The lane marks in the images could be automatically detected or marked manually. Suppose that ν_j^n is an estimated velocity of the template vehicle T_j in the field of view of camera C_n at the time instance t. Let further ν_{min} and ν_{max} be the minimal and maximal allowable vehicle velocities (taking into account possible over-speeding and down-speeding, ν_{min} and ν_{max} can be determined relative to the velocity ν_j^n or in absolute values). Then, all vehicles that disappeared from the field of view of camera C_{n-1} between time instances $t - \frac{D_{n,n-1}}{\nu_{min}}$ and $t - \frac{D_{n,n-1}}{\nu_{max}}$ are considered as matching candidates for the template T_j, if being in the same or adjacent lane as the template. Note that steps 1 and 2 can be performed simultaneously.

Step 3. A template-candidate association is computed using the Hungarian algorithm with voting, as proposed in [Rios Cabrera 12].

Step 4. After the template-candidate assignments, we update all candidate appearance models with new states. These are the appearance states that are collected in the field of view of camera C_n and are different enough from the states collected in previous cameras, $C_1, ..., C_{n-1}$. The appearance states are different enough if they fulfil the condition in Equation (5.3). This updating procedure enables learning of vehicle appearances online, along the multi-camera track.

5.6 Experimental evaluation

We composed two databases of vehicle images from three security cameras with non-overlapping views, mounted roughly in the center of a tunnel pipe ceiling and oriented in the direction of the traffic flow. The databases contain 300 different vehicles, manually annotated for evaluation purposes. Each vehicle is represented by 20 images per camera, extracted from successive video frames along their tracks, starting from the frame in which the vehicles are observed completely. For the first database, denoted as DBM, vehicle images were manually extracted from the videos resulting in similar detections between the frames and the cameras. The second database, denoted as DBA, contains real (automatic) vehicle detections, which are less accurate, thus less stable along the single-camera tracks and different between the cameras. Figures 5.1, 5.2, 5.4 and 5.5 show some examples of vehicle images in DBA database. The automatic detections are obtained using the detector of Rios Cabrera *et al.* [Rios Cabrera 12].

Five major results are presented in this section. Firstly, we give the results of our matching method for two camera pairs (C_2,C_1) and (C_3,C_2), with and without using multiple templates (observations) of each vehicle. We demonstrate that our proposed method yields a better matching accuracy than the

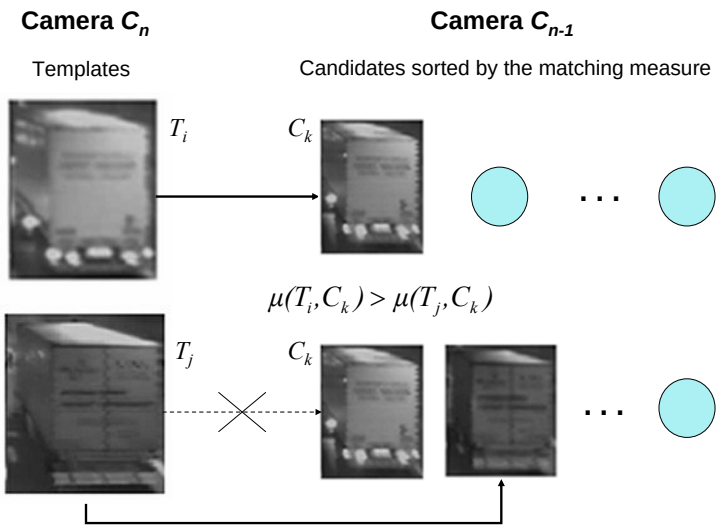


Figure 5.9: Match association. Example for two vehicles. Using the Hungarian algorithm an optimal assignment with minimal total cost (maximal sum of individual similarity measures) is found. The matching optimization does not allow multiple matches with one vehicle. In this example, template T_i is matched to candidate C_k due to a higher matching measure between T_i and C_k than between template T_j and C_k. This leads to a correct matching of template T_j, too.

reference matching techniques (2-D image correlation, SIFT, eigenimages, and Haar features based matching). Secondly, we prove that our method performs well if vehicles are visually distinctive, i.e if there is enough information in the images based on which they can be recognized. For that purpose we present separately the matching results for big vehicles (trucks, buses, etc.) and cars. Our third result shows that the signatures can be downscaled without losing the essential information, which significantly increases the computational efficiency of our method. The fourth result illustrates the gain from adding two diagonal signatures to the appearance representation based on only horizontal and vertical signatures. Finally, the fifth result demonstrates the performance of the whole matching algorithm in a tunnel application, including non-visual information derived from physical constraints of vehicle motion.

5.6.1 Results for different camera pairs

We have compared the matching score for two different camera pairs, (C_2,C_1) and (C_3,C_2) using our method with and without collection of multiple observations (templates) along the vehicle trajectories, see Figure 5.10. The matching rate is significantly higher when multiple templates are used for matching. Note that having multiple templates also reduces the difference in performance be-

tween different environments (if a single template is used, the matching rate drops in a more challenging environment of the camera pair (C_3,C_2) while this is not the case when multiple templates are used).

5.6.2 Comparison with other methods

We have compared the matching score computed between the vehicles in DBM and DBA databases using our matching algorithm with four other appearance matching methods based on 2-D image correlation, SIFT [Lowe 04], eigenimages [Turk 91] and Haar features [Rios Cabrera 12]. In our method we used 2-D signature vectors, which contain horizontal and vertical signatures. 2-D image correlation was obtained using vehicle images normalized to the same size. In the SIFT-based method, vehicle matching was done using the kd-tree and nearest neighbour search between SIFT features found in vehicle images (as in [Lowe 04]). For the eigenimages method the datasets were divided in two disjunct parts, the training and testing subset (in both databases 100 images were taken for training and tests were then performed on the other 200 images). Finally, for the comparison with the matching method of Rios Cabrera *et al.* we refer to their results reported in [Rios Cabrera 12], since those results were obtained using the images from the same tunnel recordings we used in our experiment (the only difference is that Rios Cabrera *et al.* provide results for matching each vehicle with up to 50 vehicles, while we do it up to 100 vehicles). To evaluate how discriminative the signature based appearance model is, we did multiple experiments with different numbers of candidates in the matching window. The Hungarian algorithm with voting, as proposed by Rios Cabrera *et al.* [Rios Cabrera 12], was used to optimize the assignment in all methods.

The results are given in Figure 5.11, separately for DBM and DBA dataset and the matching window size in the range from 3 to 101 with the step of 2, taken to include the corresponding vehicle and 2 to 100 other vehicles. The graphs show percentages of correct matches obtained using different methods. We see that selecting good observations suitable for matching increases the matching accuracy, especially when vehicle detections are done automatically. In our experiments the stability and variability parameters for our method (defined in Section 5.3) have been set to values $M_{st} = 0.9$, $T = 4$ and $M_{var} = 0.75$, selecting on average 1.7 good observations per single-camera vehicle track in DBM set and 2.4 in DBA set. These numbers mean that the majority of vehicles change in appearance along the track and that many disturbing observations in DBA set get disqualified. This preselection of observations good for matching is a key advantage comparing to the other methods, which creates the resulting difference in the matching accuracy. The methods without this functionality fail when the input from the detector and/or tracker is not accurate enough or the observations used for matching contain some disturbances. The methods that require registration of images before matching, like 2-D correlation and eigenimages based methods, are especially sensitive to inaccurate detections, which explains the rapid drop of their performance on DBA images (see Figure 5.11 bottom graph). In this sense, the results of 2-D correlation

and eigenimages based methods are given here also to illustrate the difference between detection accuracies in DBM and DBA set.

5.6.3 Results for different vehicle categories

On inspection we found that many of the wrong matches could be attributed to a visual similarity of vehicles. This especially holds for smaller vehicles (cars), see Figure 5.1. On the other hand, big vehicles like trucks, buses and vans usually have characteristic patterns (different design, company logos, commercials and so on), which are visually distinctive and big enough to be visible in low resolution videos. To evaluate the influence of this constraint on performance of our matching method, we have divided our database of vehicles in two categories, denoted as *cars* and *trucks*. Category *cars* contains 185 vehicles, while other 115 vehicles in the database are categorized as *trucks*. The matching results, obtained using our signature based method with collection of good observations, are in Figure 5.12 presented separately per each category.

These results clearly show that the proposed method is highly accurate for matching vehicles from the *trucks* category. Even when the templates are compared with as much as 101 candidates, more than 96% of *trucks* in DBM set and above 70% in DBA set are correctly matched. This is beneficial for applications where tracking trucks is more important, e.g. for tracking of vehicles that transport dangerous goods. The accuracy in the *cars* category is much lower and it shows that the success of appearance matching is highly limited by quality of images from surveillance cameras and distinctiveness of vehicles themselves. One way to increase vehicle distinctiveness could be using colour information when it is available.

5.6.4 Results for different reference lengths

As explained in Section 5.4.1, the signatures are rescaled using the reference length l_r before performing the matching operations. Thus, the reference length has a major impact on the number of computations needed for signature matching. In Figure 5.13 we present the matching results obtained using different reference lengths, to evaluate their influence on the performance of the method.

We see that similar results are obtained for reference lengths in the range from 30 to 150 and that the performance drop is noticeable when the lengths are below 30 points (the curves for 20 and 10 are shown). This shows that the signatures can be highly downscaled between local extrema, without affecting the performance significantly. It is due to the fact that the local extrema of the signatures capture most of the information, so most of the points between the extrema can be discarded. However, discarding all points except the local extrema leads to a performance drop because the shape of the signature between the extrema captures subtle appearance differences, which are important to distinguish similar vehicles (signatures of the vehicles in our database have 12 local extrema on average while the performance drop is noticeable when the signatures contain less than 30 points).

The possibility of downscaling the signatures for the reference length $l_r = 30$ before their matching, enables very efficient performance of vehicle matching, both in terms of data and computations. In our implementation of the proposed algorithm, matching of two appearance states was achieved in 1.02 ms on a single-core 1.86 GHz CPU. Such efficiency allowed us to compare in 11.5 seconds all 300 vehicles viewed by two cameras in a period of 8 minutes. Also, each signature vector was computed in less then 1 ms, which enabled calculation of signatures and collection of good observations online, during tracking of vehicles in a single camera view.

5.6.5 Comparison of 2-D and 4-D signature vectors

In Section 3.3 we defined the appearance representation using two and four signatures. The previous results are all obtained using the appearance model based on two signatures (vertical and horizontal), while here we analyse the gain from adding diagonal signatures. A comparison of the results obtained using the two appearance models is given in Figure 5.14. We see that for manual detections there is a slight increase of accuracy (aprox. 5 to 10 percent) when the diagonal signatures are added, but it is negligible for automatic detections. This suggests that diagonal signatures add some information, but they are sensitive to the detection misalignment and the vehicle pose change. Moreover, taking into account that the diagonal signatures triple the amount of data needed for appearance representation and matching, in our work we mostly use the appearance model based on only vertical and horizontal signatures.

5.6.6 Results in a tunnel application

In the previous sections the results of the vehicle appearance matching have been shown for different sizes of the matching window. However, in most traffic environments it is possible to reduce the number of candidates for each template. For this purpose we use the information based on space-time consistency of vehicle motion, as explained in Section 5.5. In this way, taking 30 kmph for the minimal and 160 kmph for the maximal velocity of vehicles in a tunnel application with medium traffic density, we have been able to reduce the matching window size to on average 9 candidates for each template (each vehicle was compared with 9 other vehicles). In this setting the accuracy of our 2-signatures based vehicle matching between two successive cameras along the tunnel was 97% on DBM set and 95% on DBA set, see Figure 5.11. Classified per vehicle categories, as shown in Figure 5.12, correct matching was achieved for 100% of trucks and 95% of cars on DBM set, while 98% of trucks and 94% of cars on DBA set. In Figure 5.11 we also see that our method outperforms the state-of-the-art vehicle matching method of Rios Cabrera *et al.* [Rios Cabrera 12] by 9% on DBA set, while they perform similarly on DBM set. In our method there is also no need for supervised training, which is an additional advantage compared to the method of [Rios Cabrera 12].

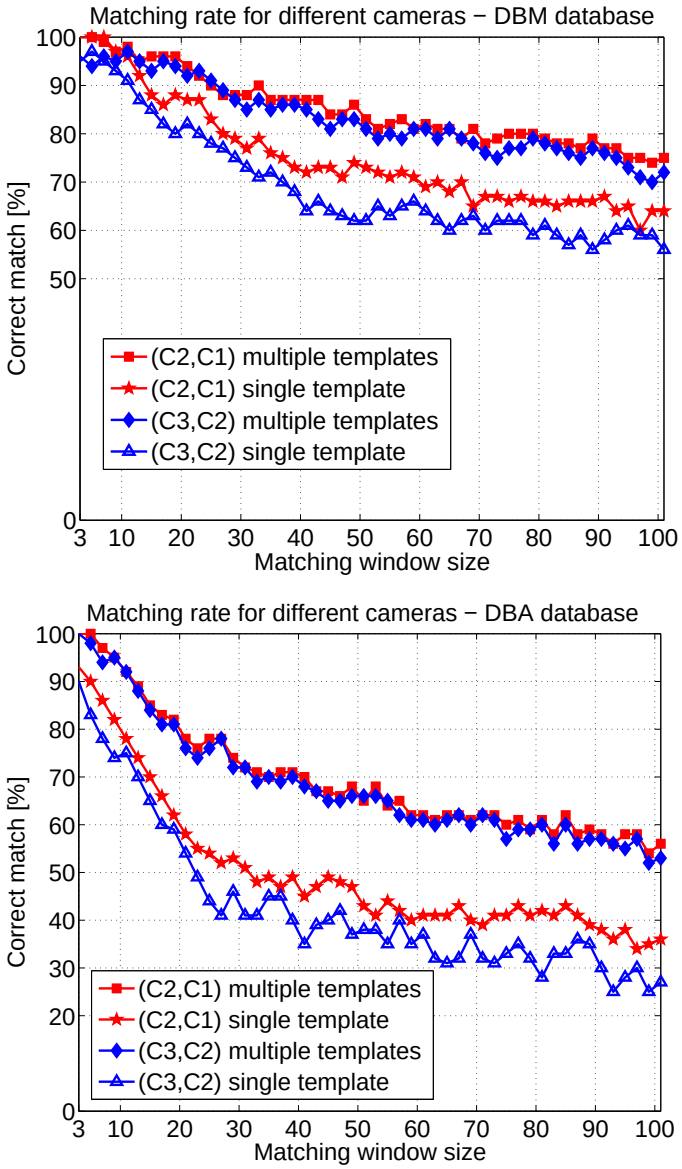


Figure 5.10: Comparison of the results of our matching method with a single and multiple templates, for two camera pairs (C_2,C_1) and (C_3,C_2) and datasets of manual (DBM, left column) and automatic (DBA, right column) vehicle detections. Each curve depicts the percentage of correct matches on the y-axis in function of the number of matching candidates (the matching window size) on the x-axis.

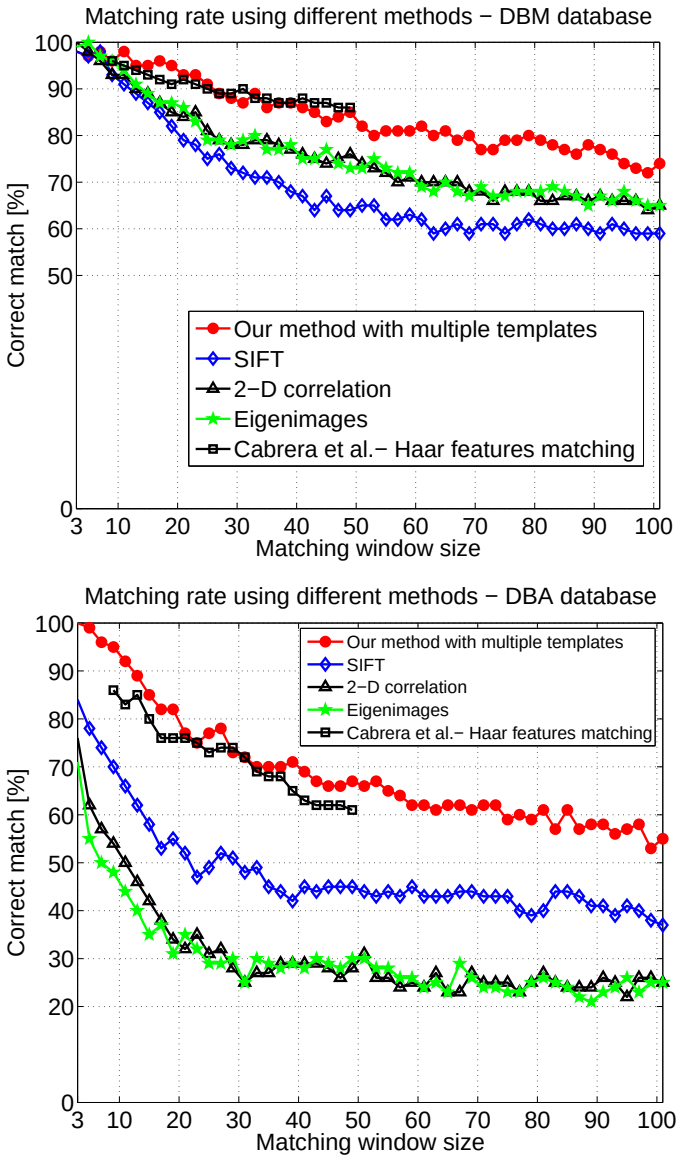


Figure 5.11: Comparison of matching results of our method and other methods, averaged over two camera pairs (C_2,C_1) and (C_3,C_2); Top: manual vehicle detections (DBM); Bottom: automatic vehicle detections (DBA). Each curve depicts the percentage of correct matches on the y-axis in function of the number of matching candidates (the matching window size) on the x-axis.

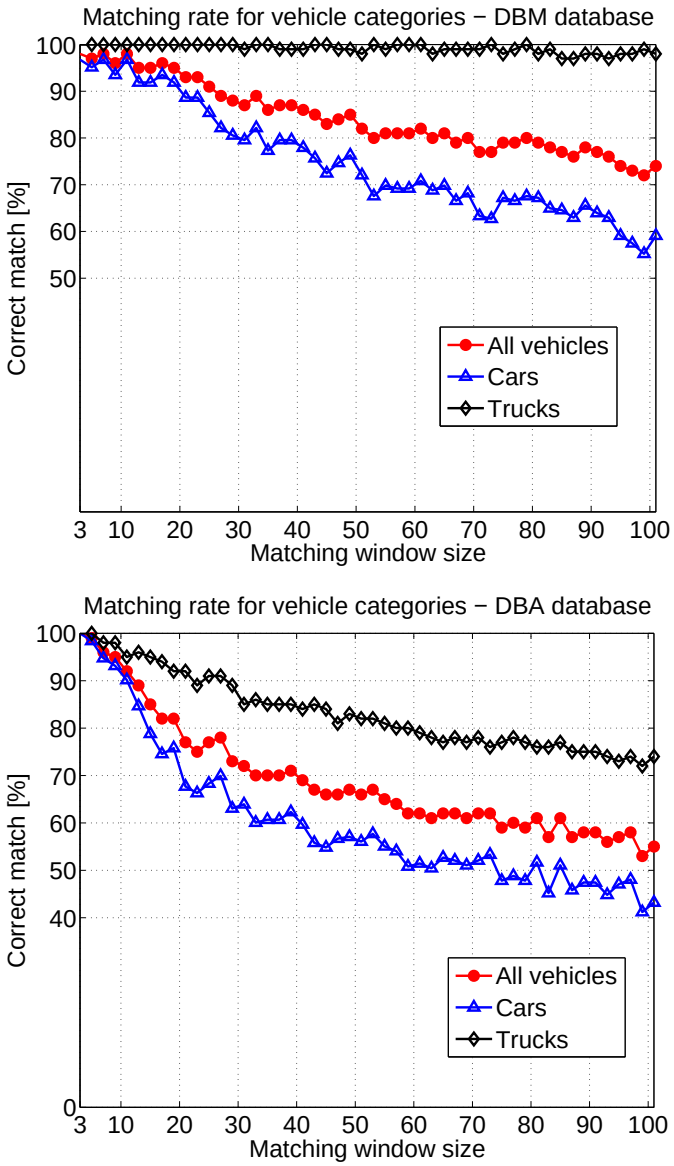


Figure 5.12: Matching results averaged over two camera pairs (C_2,C_1) and (C_3,C_2), shown for different categories of vehicles; Top: manual vehicle detections (DBM); Bottom: automatic vehicle detections (DBA). Each curve depicts the percentage of correct matches on the y-axis in function of the number of matching candidates (the matching window size) on the x-axis.

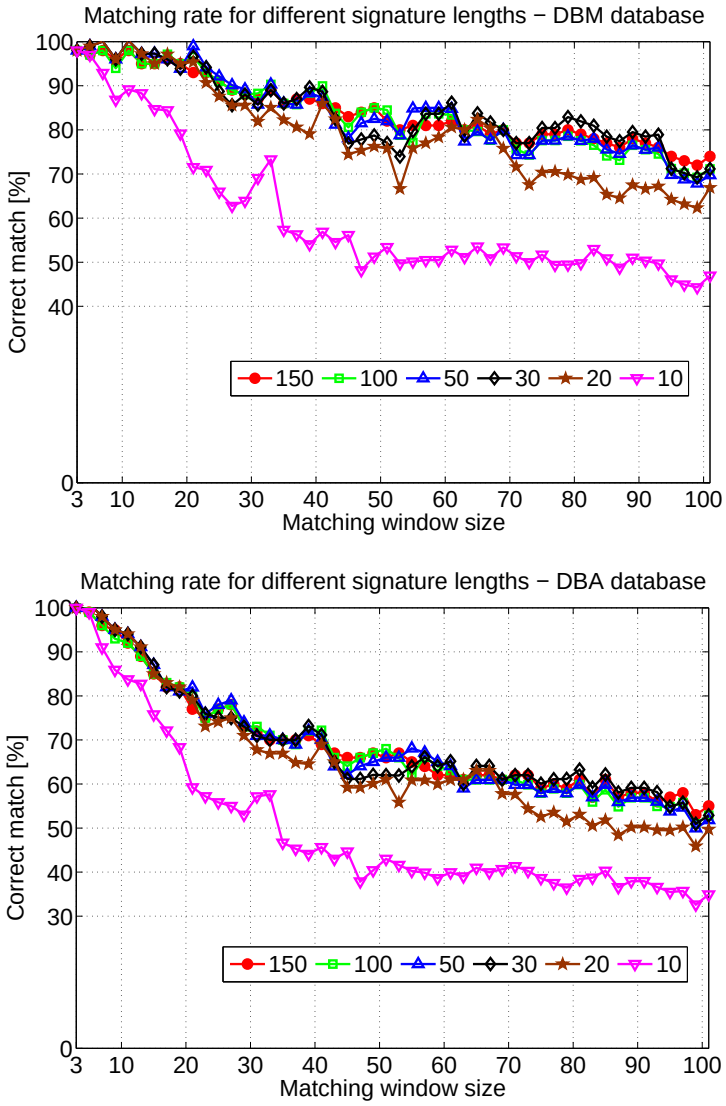


Figure 5.13: Matching results averaged over two camera pairs (C_2,C_1) and (C_3,C_2), obtained using the signatures of different length; Top: manual vehicle detections (DBM); Bottom: automatic vehicle detections (DBA). Each curve depicts the percentage of correct matches on the y-axis in function of the number of matching candidates (the matching window size) on the x-axis.

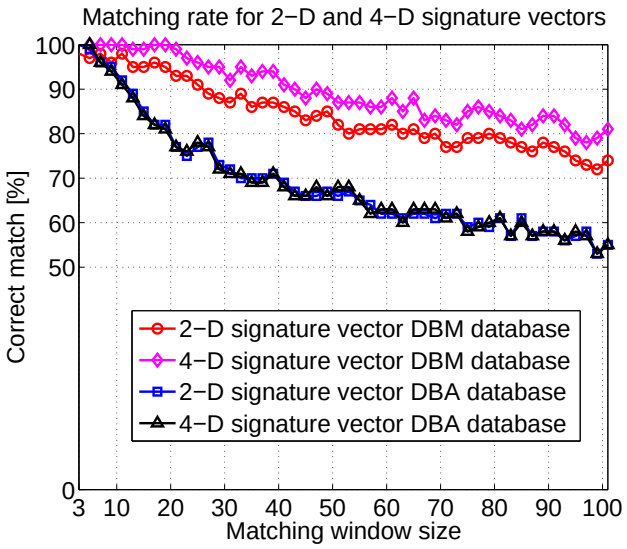


Figure 5.14: Comparison of the results using the appearance models based on two and four signatures, averaged over two camera pairs (C_2,C_1) and (C_3,C_2). Each curve depicts the percentage of correct matches on the y-axis in function of the number of matching candidates (the matching window size) on the x-axis.

5.6.7 Discussion

In this Section we discuss reasons for possible failures of vehicle matching using the proposed framework and we propose some solutions to prevent these failures.

1. If multiple vehicles from the same matching group (vehicles that appear at approximately the same time in the scene) appear in poses significantly different from the poses observed in previous cameras. When a vehicle appears in a camera in a pose significantly different from the poses observed in previous cameras, the multi-observation appearance model of the vehicle would not contain a descriptor of the vehicle in such a pose, and this could likely lead to a failure in vehicle matching. To reduce the chance of such a failure, the proposed framework has a matching optimization step, which results in the optimal assignment for each vehicle in the corresponding matching group. Consequently, this means that the failure would possibly happen if multiple vehicles, not only one, from the same matching group change pose to a previously not observed pose, which is less likely. By extending the proposed framework with a multi-hypotheses alike approach it would be further possible to recover from incorrect assignments over time, as there are more observations of each vehicle from multiple cameras.

2. If a vehicle is very similar or the same in appearance with some

other vehicle or vehicles from the same matching group. There are a lot of vehicles, especially cars, of the same make or with similar appearance, which is a significant challenge for any appearance based vehicle matching method. It is often not possible to re-identify vehicles based on their appearance only. In the proposed framework it is, therefore, possible to add additional information to select matching candidates for each vehicle based on vehicle kinematics, and this is what we exploit in our work presented in this chapter. The proposed framework also supports more precise selection of the matching candidates by using the contextual information such as vehicle constellations (see Figure 5.15), and probabilities and evidences of changes in these constellations (e.g. probabilities or evidences that a vehicle overtook another vehicle, changed the lane, etc.), but in this chapter we did not use this additional information.

3. If a vehicle is occluded or inaccurately detected so that significant or distinctive parts of the vehicle are not captured. To correctly match two vehicles using any appearance based matching method it is important there is enough visual information to do the matching. If some vehicles are occluded or inaccurately detected, a significant amount of information might be lost. Therefore, in the proposed framework we introduced a method to automatically detect such cases and select good observations to perform matching. As shown in our experiments (see Figure 5.11) there is a significant improvement in the matching performance due to selecting of good observations. However, if a vehicle is significantly occluded or detected with high inaccuracy in a whole scene, its re-identification would likely fail due to absence of good observations. This is in some cases solved by matching optimization in the proposed framework, but it could be further improved by the previously mentioned approaches of multi-hypotheses and contextual alike matching.

5.7 Conclusions

In this chapter we proposed a novel method for vehicle appearance modelling and matching. We proposed using image projection profiles to obtain vehicle signatures that significantly reduce the amount of data needed for vehicle matching. We showed that in low resolution images such signatures capture well the spatial distribution of vehicle parts and patterns, which was used for their matching. We showed also that by selecting a set of good observations along the multi-camera track, it was possible to overcome many matching problems that occur due to inaccurate detections, intensive illumination changes, clutters and occlusions, as well as changes of the vehicle appearance. As a consequence, object matching itself was significantly simplified and yet outperformed more complex methods. The presented results also show that it is possible to highly downscale the signatures without affecting the performance significantly, which further reduces the amount of data and computations needed for vehicle matching. Thus, the proposed appearance matching method can be used to perform vehicle matching on embedded systems (e.g. smart cameras) or by a

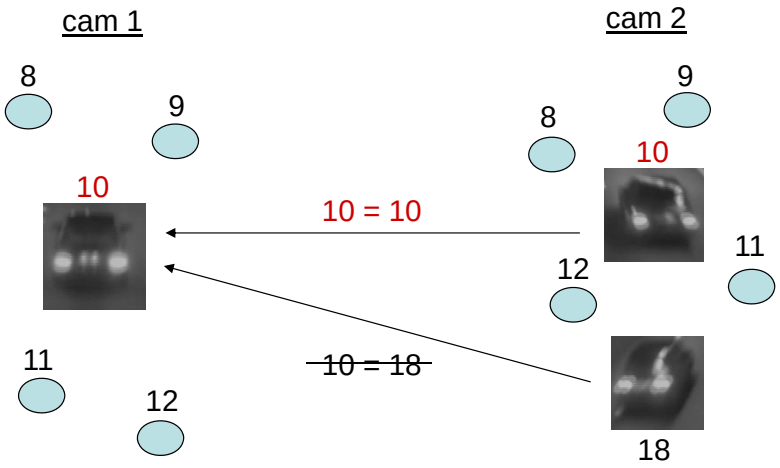


Figure 5.15: The matching of similarly looking vehicles could be improved by using contextual information. In two consecutive scenes, especially if the distance between these scenes is not big, vehicles are likely to belong to the same constellation.

low-complexity central server without a need for sending the images between the cameras or to the server.

An interesting future direction is to extend this approach towards matching of vehicles in traffic environments in which camera views are significantly different, e.g. along city roads or crossroads. In this case it is important to add automatic detection of good appearance states for matching depending on the cameras' view. Also, in the environments where colour information is available, it can be incorporated to further increase matching accuracy.

The work presented in this chapter has been published in the journal *Image and Vision Computing* [Jelača 13], and in the proceedings of two international conferences [Jelača 12], [Niño Castañeda 11].

6

Tracking in Overlapping Camera Views

In this chapter we focus on object tracking in overlapping camera views. This is one of the often present setups in video surveillance of indoor environments. Real-time tracking of people is usually an essential task in these setups, of which security and surveillance for path-retracing is among the widely spread applications [Pflugfelder 10], [Morris 08]. Many other applications are constantly emerging. For instance, in telecommunications (more specifically videoconferencing), positional data of each meeting attendant can be used to define regions of interest containing people, to limit more detailed processing to these areas. It can be helpful to focus pan-tilt-zoom (PTZ) cameras on specific people [Aghajan 09], to determine when they enter and leave the room, their identity (even when they do not face a camera), and infer their activities [Fathi 11] such as presenting, looking at the presenter, and others. In these applications, in which the goal is to determine individual trajectories of each person, avoiding tracking losses is essential. Trajectories of individuals should not be lost due to occlusions and individuals should not be mixed up when they get close together or when their paths intersect.

One way to avoid this problem is to rely on high-level feature analysis [Babenko 11] and, for example, to periodically re-identify people. Such algorithms are computationally intensive and it is often better to restrict their usage. For instance, they can be activated when there is a doubt about the current tracks or they can be run every few seconds only. Alternatively, high-level algorithms can be used to correct tracking losses when they are already executed for other purposes, as demonstrated in our experimental setup. Here, we analyse people's faces when they are entering the room or when they are seated in front of a web camera [Deboeverie 11]. This information can correct some tracking loss problems, but often with a large delay. In conclusion, while algorithms relying on high-level analysis are certainly valuable, it is still very important for any tracking algorithm to minimize tracking losses in the first place.

It is also essential to have high tracking accuracy. On the one hand, a

higher tracking accuracy helps to reduce tracking losses, because a reduced accuracy is an indicator of near losses [Aghajan 09]. On the other hand, a high accuracy is needed for detailed behaviour analysis, e.g. to detect interactions between people (people who know each other well tend to stand closer to each other, shake hands, possibly hug or kiss, etc.) [Fathi 11]. These situations are especially common in video-conferencing scenarios. In such an application, to avoid mixing up people when they come close to each other, the distance between the tracking response and the true position of a person needs to be smaller than the width of a person.

There are also applications where accurate tracking statistics are more important than the accuracy of individual trajectories. For instance, in elderly care, behaviour analysis based on trajectory statistics such as walking speed, path smoothness and average activity levels can yield important information about physical and mental health of the observed persons [Kröse 11]. In such applications it is necessary that the tracking accuracy is high enough to position a person in the right area (e.g. the kitchen or the living room). This also holds for applications in marketing and in-store promotions (e.g. to know whether visitors stand in front of the billboards or shop's windows). Furthermore, in a work environment, statistics about the time workers spend sitting, walking or standing, can help to pinpoint productivity problems or potential health hazards, such as taking too few breaks. This information can also be used to optimize the energy consumption in the observed spaces, but since the data are used in a statistical manner it is not necessary to have as high accuracy as in the video-conferencing scenario. From these examples we see that the application scenarios determine how big the tracking accuracy needs to be. Nevertheless, a general tracking principle is to strive to minimize tracking losses and maximize tracking accuracy.

Reliable accurate tracking of multiple people in crowded or highly cluttered scenes is still a very challenging task, mainly due to frequent occlusions and environmental changes. Even detecting and tracking a single non-occluded person sometimes poses problems for state-of-the-art single camera tracking algorithms, e.g. due to poor illumination, lighting changes or occlusions. Tracking multiple people in the presence of furniture, and other occluding objects, poses additional problems. In this case, the problems can be significantly reduced by using a top view rather than a side view camera. While tracking may be possible with top view cameras only [Ozturk 09], there is limited amount of information that can come from top views. There are also significant distortions in top views and in some environments it is difficult to obtain a good scene coverage with top views. Therefore, most applications need side view cameras to cover the whole scene and enable more detailed analysis or visualization (e.g. of people's faces), so it is a good choice to use these side view cameras for people tracking as well. Joint analysis of multiple side view camera streams increases the robustness in most applications [Aghajan 09], especially in highly cluttered indoor scenes. The triangulation principles can help to estimate the positions of people with high accuracy. If enough cameras are available, problems due to

occlusions can also be avoided so top view cameras may not even be needed. In our work we experimented with two setups: one with only side views and another with combined side and top views.

In our solution, presented in this chapter, beside on the tracking accuracy in real-world conditions, we focus also on real-time, low-latency and scalability of people tracking. This adds another level of complexity compared to state-of-the-art methods as published by [Berclaz 11]. *Real-time* and *low-latency* operation is needed in many indoor trackers since they need to react quickly to changes in people's positions. The trackers also often need to deliver real-time response to select appropriate high-resolution views for content displaying and running detailed analysis algorithms. Examples of such applications include surveillance, occupancy monitoring, telepresence and teleclassing, where it is necessary to focus one or more cameras on the moving presenter. These applications typically involve long-term monitoring, so robustness and tracking accuracy are also critical. The problem of *scalability* is often overlooked in multi-camera research: centralized processing of multiple video streams creates not only a computing but also a communication bottleneck. This means that adding a camera to the centralized network can significantly impact the network's ability to distribute and process the acquired data. From this point of view, decentralized and distributed tracking approaches that group cameras into clusters which communicate with a local fusion centre (decentralized) or with each other (distributed) are much more scalable than centralized systems [Taj 11].

Combining real-time, low-latency, scalability, accuracy and robustness requirements is highly non-trivial. Nevertheless, in recent years it has become possible with the deployment of smart camera networks, which shift the computation load towards the cameras [Hengstler 06], [Hengstler 07], [Soro 09b], and increase robustness and accuracy of the system by combining the information from individual cameras. Therefore, in this thesis our focus is on finding suitable features, extraction, distribution and fusion of the information gathered in a smart camera network to create a tracker that is up to the stated challenges.

The work presented in this chapter has been done in close collaboration with my colleagues Sebastian Grünwedel and Jorge Oswaldo Niño Castañeda. Sebastian Grünwedel presented parts of this work in his PhD thesis, too. His focus was on foreground/background segmentation of images using edges, occupancy mapping using Dempster-Shafer theory of evidence, and a conceptual design and comparison of performances of centralized and decentralized tracking methods. The part of conceptual design of a decentralized multi-camera system with several different approaches will also be elaborated in this thesis. However, my focus in this PhD thesis is on using illumination invariant image features that we introduced in Chapter 3 (as computationally efficient tracking cues), creating a multi-view appearance model using these features, matching of these features to build a tracker in tracking-by-recognition fashion, and using contextual reasoning to increase the tracking accuracy.

In our approach, each camera performs low-complexity tracking. Each camera represents objects (people) as bounding boxes (cuboids) with respect to a world coordinate system. Since we assume calibrated cameras, we are estimating the person's cuboid in world coordinates rather than in image coordinates. This allows a physical motion model to be expressed more easily than a model in the image domain where apparent speeds depend on the position of the person with respect to the camera. Furthermore, the estimates of a cuboid in world coordinates from different cameras can be directly fused in a fusion centre or even in other cameras without knowing the relationship between a camera image domain and the world coordinate system.

Some of the possible tracking approaches for this purpose are based on tracking the motion of each blob using advanced motion estimation techniques in a smart camera, such as optical flow, as in [Grünwedel 12], or by tracking SIFT, SURF, FAST or other local features within the blobs [Anjum 09]. In this work we rather aim to show that tracking is possible using simple image projection features in combination with extremely simple blob analysis. For blob tracking, we rely on feedback from the fusion centre on the most recent positions, speeds and geometries of individuals in the scene. Based on this feedback, we perform probabilistic occlusion reasoning in the camera to identify which blobs belong to which cuboid. The analysis also yields updated cuboid parameters (e.g. positions). Our method does not involve sophisticated motion estimation in each camera, although we demonstrate the benefits of having the context aware motion models.

We developed and compared two approaches for on-camera tracking: histogram filtering (HF) with local state estimation and an approach without local state estimation (NLE). Both approaches obtain estimates for each person based on FG/BG segmentation, signatures defined in Chapter 3 and the feedback from the fusion centre. Since these estimates are in world coordinates, uncertainties are introduced by the back-projection from the image domain to the world coordinate system. Nevertheless, the fusion centre uses a linear Kalman Filter (KF) to obtain a joint decision of all cameras for each person. The final estimates are fed back to each camera to minimize the uncertainties of the camera estimates.

By extensive experiments of people tracking in meeting rooms, we demonstrate the advantages of our approach, i.e. that a relatively simple analysis of changes in images can be reliable and accurate for tracking multiple objects in a multi-camera network. It is achieved when information from multiple cameras is fused and communicated back to each camera, creating a powerful mechanism to overcome many issues with occlusions. More specifically, we demonstrate that the number of tracking losses in such a feedback based framework is low, even in sequences with abundant occlusion. The average tracking accuracy in these sequences is around 10 cm. These results show an improvement on state of the art algorithms as reported by [Berclaz 11] and [Fleuret 08]. They also show that adding image projection features to FG/BG cues proposed by [Grünwedel 14] improves tracking accuracy, without affecting the real-time

performance and scalability.

Our system has a very low communication overhead: a frequency of 10 updates per second alone is sufficient for each camera to transmit the parameters (position, speed, width and height) together with a reliability measure of each tracked cuboid to the fusion centre. These geometrical descriptors are integrated by the fusion centre, which then returns fused descriptors for all tracked objects to all smart cameras. The resulting transmission bandwidths from the camera to the fusion centre and back are in the order of 1 kilobyte/second per object for each camera. The low communication overhead results in a highly scalable system. Moreover, it is an asset in battery operated smart cameras, where a balance between computation and communication load is critical for reducing power consumption. The low communication overhead is also an asset in wireless camera setups (these setups are often preferred to avoid building works on cable installations).

The remainder of this chapter is structured as follows. In Section 6.1 we give an overview of the iCocoon project in which this research was carried out. Section 6.2 contains an overview of related work. In Section 6.3 we formulate the problem of multi-camera tracking with overlapping views, and give an overview of our solution in Section 6.4. Section 6.5 describes the video processing and local tracking within each camera, while the consensus tracking that fuses data from all cameras and returns global states of tracked objects is explained in Section 6.6. In Section 6.7 we present experimental results to demonstrate the tracking performance: accuracy, precision, computational efficiency, scalability and communication overhead. We compare our methods with the methods of [Grünwedel 14] and [Berclaz 11], and show improvements in the accuracy and precision. The video data we use are acquired in two real-world setups. We also show results on a publicly available data set of [Berclaz 11], [Fleuret 08]. The results show that our system outperforms the other methods in terms of accuracy and precision, having also the advantage of being real-time and scalable, with low communication overhead. Section 6.8 concludes this chapter.

6.1 The iCocoon project

The purpose of the project "iCocoon" (Immersive Communication by means of Computer Vision) is to drastically change the way people communicate remotely. This is realized by creating third-generation video conferencing applications based on video technologies such as computer vision, scene understanding, 3D reconstruction and others. The targeted solution provides an immersive sensation to the communicating parties, with good understanding of the environment and the context in which the communication is taking place. The project was carried out by academic and industrial partners in Flanders, Belgium [1]. The scope of the project was applied scientific research with a proof of concept of a new generation video communication system, built with innovative 2D and 3D computer vision technologies.

The main characteristics of such a system are an affordable price, flexible communication and easy set-up of smart multi-camera networks, intelligent context understanding of the scene, 3D capturing, efficient information encoding and transport, innovative rendering and displaying techniques (see Figure 6.1). From this perspective, the solution is different from state-of-the-art high-end video conferencing systems, which are expensive, targeted at formal communications between a relatively small number of people and require dedicated communication rooms.

The project addresses the aforementioned issues, and moreover, includes the following ideas.

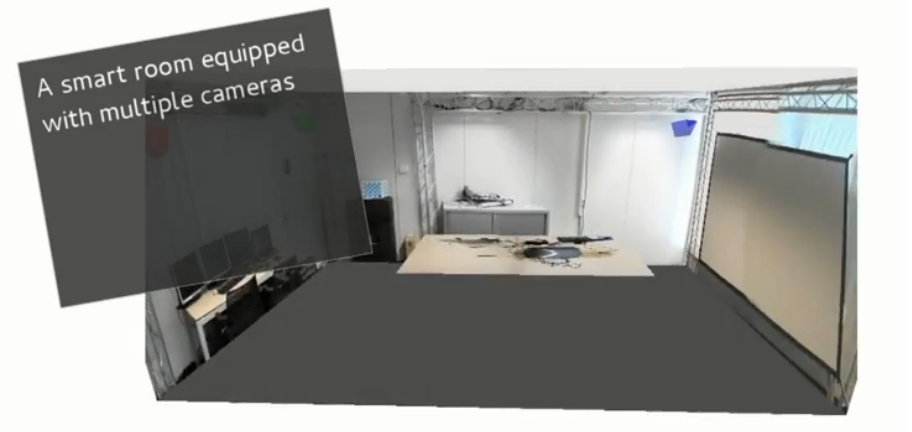
1. Computer vision technologies are used to analyse the scene and human behaviour in each meeting room. The data is captured from a distributed smart camera network which registers, in an inexpensive way, every ongoing activity in the room. The smart camera network is plug-and-play with automatic camera calibration. The visual cues important for communication control are detected and automatically annotated. This enables the system to take decisions over which events to visualize to an end-group/user on the remote side.

2. On the display side, the goal is to visualize the captured information in the most effective way. This is done by automatically showing the most relevant actions and automatically editing the appropriate camera shots from all available 2D video data. A consistent and ongoing overview of the virtual communication environment is produced via a combination of live and synthetic video streams. The latter takes the form of a dynamic 3D rendering of a virtual meeting room in which the participants are displayed as moving avatars (see Figures 5.2 and 5.3).

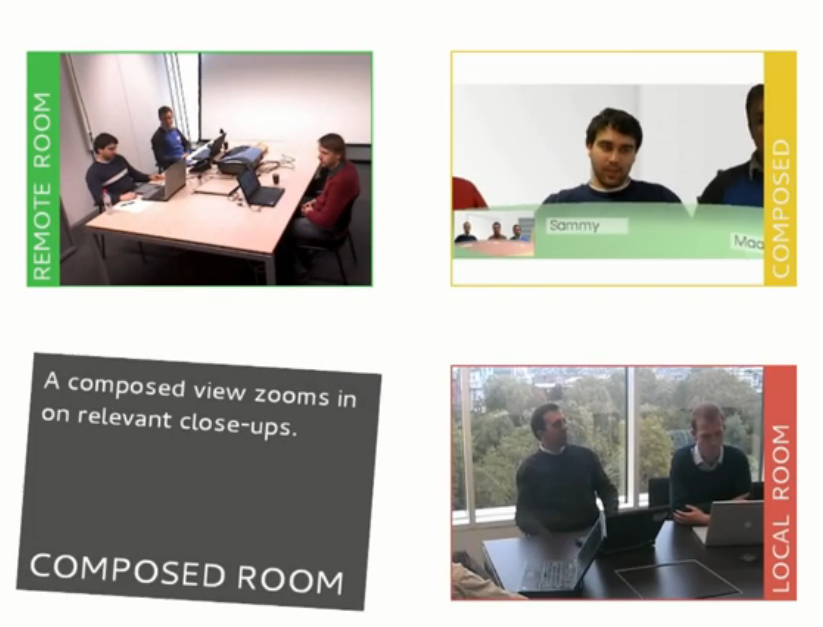
3. In the long term, the realism of this teleconferencing experience could be enhanced using techniques from augmented reality. For this reason,

[1] More details can be found at `http://www.iminds.be/en/research/overview-projects/p/detail/icocoon-2`

Figure 6.1: *iCocoon - meeting room setup.* These figures show the way people could communicate remotely. A meeting room is equipped with fixed and portable cameras observing the on-going meeting using video technologies (such as computer vision, scene understanding and 3D reconstruction), and meeting participants are rendered using their avatars.



(a) Smart meeting room



(b) Composed virtual view

Figure 6.2: *iCocoon - system overview.* In (a), the goal is to visualize the captured information using a multi-camera network in the most effective way. The system produces a consistent and ongoing overview of the virtual communication environment via a combination of live and synthetic video streams in the form of a 3D dynamic rendering of a virtual meeting room in which the participants are displayed as moving avatars (b). These pictures are taken from the demo video produced by Alcatel-Lucent [Alcatel-Lucent 12].

(a) Symbolic room

(b) Presenting event

Figure 6.3: *iCocoon - system overview.* In a) a virtual meeting room provides an overview of the participants and events happening in a meeting. A real-time tracking system, using a multi-camera network, is used to estimate the whereabouts of the participants and to detect events such as an ongoing presentation (b). For such a system, accuracy is very important to prevent the tracking system mistakenly following another participant or non-human objects. The representative pictures are taken from the demo video produced by Alcatel-Lucent [Alcatel-Lucent 12].

the project investigated capturing and compression of 3D models of people, meeting rooms, furniture and other objects, to compose a virtual meeting room. It also investigated the difficulties involved in the 3D capture of persons in real-time and provide a non-real-time, semi-interactive simulation meant to probe users on how they would experience such a system.

The project aimed to deliver a real-time demonstration of the people tracking, video selection, displaying and the symbolic overview features. For that purpose, two tracking environments based on smart camera networks were set up at Hogeschool Gent and at Alcatel-Lucent in Antwerp.

6.1.1 iCocoon contributions and credits

The research contribution to the project "iCocoon" in the area of multi-camera people tracking was done by the Image Processing and Interpretation (IPI) research group at Ghent University and the Vision Systems (VIS) research group at the Hogeschool Gent.

For a video-conference meeting it is necessary to determine positions of people within a meeting room in real-time. The research goal of the project was to determine the location, pose, body motion and head orientation of people in meeting rooms, using low-latency multi-camera video processing for a video conferencing application. A smart camera solution was aimed at, in which most of the processing is done within each smart camera. In this context, the smart cameras output meta-data rather than video data, whenever possible. Therefore, the data exchange and the network load is limited. This approach differs radically from the traditional approaches, which jointly analyse all video streams in a central location.

However, such a task is challenging due to frequent occlusions of people by furniture and other people in a meeting room. An additional challenge is imposed by changing lighting conditions (e.g. due to turning lights on or off, or changing presentation slides).

To deal with the occlusions, we use a multi-camera network, viewing a room from different viewpoints, to monitor participants of a meeting. However, this creates new challenges, especially in terms of real-time processing and fusion of the acquired data from different cameras. Our proposed system is scalable because it requires a very small communication bandwidth and only light-weight processing on a "fusion centre" that produces final tracking results. Thus, the fusion centre does not need to be sophisticated and expensive, and can also be duplicated to increase reliability. All low-level video processing is performed on smart cameras. The smart cameras transmit a compact high-level description of moving people to the fusion centre, which fuses this data using a Bayesian approach. In our system, the camera-based processing takes feedback from the fusion centre about the most recent locations and motion states of tracked people into account. Based on this feedback and background subtraction results, the smart cameras generate a best hypothesis for the location of each person.

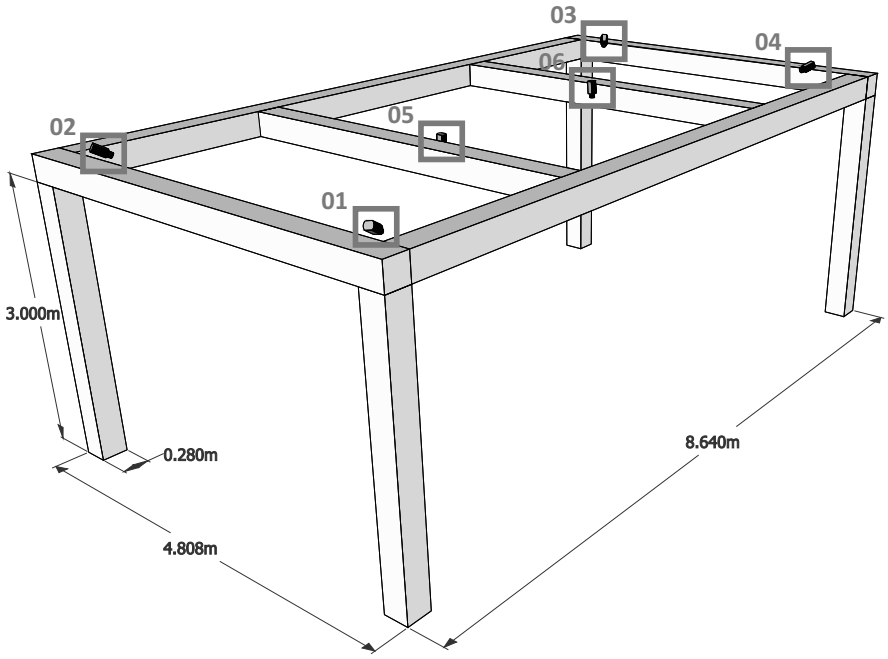


Figure 6.4: *HoGent - setup overview.* An experiment room was set up for the project "iCocoon" at the Hogeschool Gent. The room is equipped with six cameras, four side-view and two top-view cameras, operating at a frame rate of 20 FPS. The cameras were mounted at ceiling height ($3m$ approximately), and extrinsically calibrated and synchronized up to frame accuracy.

Furthermore, to deal with lighting changes, we use our illumination robust signature features proposed in Section 3.3 and an edge-based foreground/background segmentation method proposed by [Grünwedel 11b]. These methods are used on each smart camera to determine regions of interest that contain motion and objects of interest, which in turn is used to track objects.

At the Image Processing and Interpretation (IPI) research group, Peter Van Hese, who was supervising the project, Dirk Van Haerenborgh, Jorge Oswaldo Niño Castañeda, Sebastian Grünwedel and myself were involved in the project. At the Hogeschool Gent, the team consisted of Dimitri Van Cauwelaert and Francis Deboeverie.

An experiment room was set up for this project at the Hogeschool Gent with help of Dimitri Van Cauwelaert and Dirk Van Haerenborgh and under the supervision of Prof. dr. ir. Peter Veelaert. The room was equipped with six cameras, four side-view and two top-view cameras, operating at a frame rate of 20 fps (Figure 6.4). The cameras were mounted at ceiling height ($3m$ approximately), and extrinsically calibrated and synchronized up to frame accuracy. Dimitri Van Cauwelaert and Dirk Van Haerenborgh took care of this part of

the project, as well as running experiments and testing the system.

Francis Deboeverie worked on the face analysis part of the project. His goal was to obtain a best view selection based on face recognition with geometric features. In his research, faces are represented as Curve Edge Maps (CEMs), which are collections of polynomial curves with a convex region. Face recognition is performed by matching face CEMs driven by histograms of image intensities and histograms of relative positions. Moreover, the face recognition was also used to recognize the attendees of a meeting in the distributed real-time tracking approach.

The design of the tracking approaches in the iCocoon project were planned and implemented by Sebastian Grünwedel, Jorge Oswaldo Niño Castañeda and myself.

The research work of Jorge Oswaldo Niño Castañeda is concentrating on person detection and appearance modelling using machine learning techniques, which is an ongoing research. Moreover, he was working on behaviour analysis, which involved the detection of events for each meeting attendee, such as "standing", "sitting", "walking" or "gesturing".

Sebastian Grünwedel was responsible for conceptual design, research and implementation of occupancy mapping and decentralized tracking approaches. He focused on the aspects of scalability, accuracy and precision, as well as limitations of these approaches. Furthermore, he developed an edge-based FG/BG segmentation method which he used in both tracking systems and demonstrated that this FG/BG segmentation is more robust to illumination changes than state-of-the-art approaches.

The emphasis of my work in this context, presented in this thesis, lies on using image projection features and feature matching methods to robustly track people in real-time under severe occlusions and in challenging lighting conditions. I focused on improving the tracking accuracy by adding image projection clues into the Bayesian filtering framework. Within this project, I designed, in close cooperation with Sebastian Grünwedel and Jorge Oswaldo Niño Castañeda, both tracking approaches: a multi-camera tracking approach based on occupancy maps and a distributed multi-camera tracking approach with a feedback loop. Furthermore, I together with Sebastian Grünwedel implemented the system architecture. The implementation of such a system resulted in a real-time demonstrator at Hogeschool Gent and Alcatel-Lucent Bell Labs in Antwerp. The multi-camera tracking approach based on occupancy maps is thoroughly explained in the PhD thesis of Sebastian Grünwedel so it is not a part of this thesis.

6.2 Related work

In this section, we provide an overview of state-of-the-art approaches for multi-camera tracking. There are several interesting surveys about multi-camera approaches and architectures to which the reader is referred to for more details: Smith *et al.* [Smith 06], Liu *et al.* [Liu 07], Aghajan *et al.* [Aghajan 09] and

Taj *et al.* [Taj 10], [Taj 11].

Detection and tracking of multiple, possible occluded, people in complex environments is a challenging task which makes multiple cameras indispensable. The different viewpoints offered by multiple cameras decrease the number and size of occluded regions. Also, multiple cameras simplify 3D analysis of a scene and provide redundant information which can help to improve robustness. Numerous papers addressed multi-target tracking using the principle of associating objects in multiple views. For example, in [Nakazawa 98], human tracking is performed using template matching to track moving people. A state transition map together with action rules is used to coordinate between cameras. The state transition map consists of three types of areas according to the camera coverage: areas visible to only one camera, areas visible to multiple cameras and areas not visible to any camera. This state transition map stores the view parameters of all cameras, while the action rules define the operations performed by each camera.

Cai *et al.* [Cai 98] adopted a view selection approach: their tracker is a single-camera one, based on a Bayesian classification scheme. As soon as the current camera has no longer a good view of the tracking target, the tracker switches to another camera. This switch is predicted by the tracking system using the position of an object along a spatio-temporal domain. The tracker gathers sample pictures of the upper part of human bodies seen from various viewing angles, and uses them for internal state. The principal component analysis (PCA) is used to exclude non-human moving objects.

Bayesian networks are another popular approach to address the problem of multi-camera tracking. Chang *et al.* [Chang 01] used a Bayesian network approach to combine geometry (epipolar geometry, homographies, and landmarks) and recognition (height and appearance) based features for matching objects between consecutive image frames and multiple camera views. In the work of [Dockstader 01b] Bayesian networks were used to track objects and resolve occlusions in multiple calibrated cameras. On the other hand, Nillius *et al.* [Nillius 06] focused more on the high-level tracking and assumed the existence of an isolated track graph. In their approach they associated the identities of those graph tracks. The problem was formulated as a Bayesian network inference which uses standard message propagation to find the most probable set of paths in an efficient way. Their multi-object tracking is applied to the problem of soccer player analysis.

Stereo vision is used in [Krumm 00], [Darrell 01], [Mittal 03]. For instance, Krumm *et al.* use a stereo camera approach wherein depth information from multiple stereo cameras is combined in 3D space. Firstly, they perform background subtraction and then detect human-shaped blobs in 3D space. Afterwards, people are identified and tracked using a probability distribution based on colour histograms. Nevertheless, the underlying problem of any feature type, such as appearance, colour, blob shapes, etc., remains also in their approach: the features are easily corrupted due to occlusions or environmental and lighting changes.

Khan *et al.* [Khan 06], [Khan 09] use a homographic occupancy constraint to fuse foreground evidence retrieved by a background subtraction method. They make it from multiple cameras using geometrical constructs. The homographic occupancy constraint interprets foreground as scene occupancy by non-background objects. Pixels that correspond to occupancies on a reference plane are consistently warped to foreground regions in every view. Their method resolves occlusions by localizing people on multiple reference planes, and attempts to find image locations of scene points that are occupied by people. Similar work was presented in [Mittal 03], [Franco 05], [Berclaz 06] and [Fleuret 08]. Fleuret *et al.* estimate probabilities of occupancy on the ground plane given binary images obtained by background subtraction. They use a generative model representing humans as simple rectangles. The probabilities of occupancy at every location are approximated as the marginals of a product law, minimizing the Kullback-Leibler divergence from the conditional posterior distribution. Optimal tracks are computed from the raw observations by a greedy search strategy based on dynamic programming.

In a later extension, [Berclaz 11], the trajectory estimation is treated as a constrained flow problem. This results in a convex optimization problem, which is solved using the k-shortest paths algorithm. The results show a very good performance on difficult real-word applications. In many of the aforementioned papers, the accumulated knowledge about the location of people is represented by occupancy maps. This is inspired by the research on robot navigation using range-sensor based sensors [Elfes 89], [Thrun 03]. However, methods based on occupancy maps usually perform poorly when humans are partially hidden (e.g. by furniture), or if the input data (result of background subtraction) is noisy or even not existent due to environmental changes in at least one of the cameras. The main reason is the assumption that pixels corresponding to occupancies will warp to foreground regions in every camera view. This assumption is not always valid and can therefore lead to errors in some of the occupancy maps, eventually resulting in tracking errors. The major difference in our approach is that we calculate a local estimate of individuals in each camera directly. Our approach obtains global estimates of individuals using a Bayesian estimator in a continuous state space with respect to a world coordinate system, rather than a discretization of the ground plane into grid cells. Each camera makes its own estimates for each person and a global (final) estimate is obtained by the fusion of all camera estimates at the fusion centre side. This has the advantage that cameras itself can make mistakes as long as the global estimates are correct.

Taj *et al.* [Taj 09] used a centralized tracking approach where the input data from each camera view is projected on a top-view through the multilevel homographic transformation of [Delannay 09]. This homographic transformation projects foreground evidence to planes parallel to the ground plane. The projected planes are added up to generate a detection volume. The method adopts a track-before-detect (TBD) approach to keep track of possible humans in the scene. In the TBD approach the entire image is considered as a measurement, which is a highly non-linear function of the target state. The target state

consists of the position and speed of an object and the intensity of the image. In their approach tracking is solved by employing non-linear state estimation techniques such as particle filtering.

Anjum *et al.* [Anjum 09] used an unsupervised inter-camera trajectory correspondence algorithm to link objects across a multi-camera network. Object association is implemented as a hybrid approach using local trajectory pairs estimated by multiple spatio-temporal features. Then image plane reprojections of the matched trajectories are employed to resolve conflicting situations. The latter approaches have high-data transfer rates due to the nature of centralized processing and therefore a lack of scalability and energy efficiency. However, our proposed approach is scalable, efficient with respect to the communication bandwidth, and operates in real time due to the limited data exchange between cameras and the fusion centre.

Other approaches for visual tracking range from Bayesian filtering algorithms to Probabilistic Graphical Models (PGMs). Dore *et al.* [Dore 10] made a state-of-the-art review of Bayesian state estimation and PGMs, with respect to tracking applications. In particular, in computer vision and video processing, algorithms have been proposed based on different types of PGMs such as Hidden Markov Models (HMMs) or Kalman filter. In their review, they describe PGMs as a statistical framework suitable for handling complex object representations. This framework enables consistent formalization and handling of uncertainties of visual observations. Moreover, this framework allows efficient solutions for complex problems, meeting real-time requirements. One important step in these approaches is data association modelling. Here, one commonly uses the Joint Probabilistic Data Association Filter (JPDAF) [Bar-Shalom 87], [Kirubarajan 04].

Rasmussen *et al.* [Rasmussen 01] use a constrained JPDAF filter for their randomized tracking algorithm which oversees correspondence choices between the tracker and image features. The algorithm is applied to three different kinds of tracking features, namely homogeneous regions, textured regions, and contours described as snakes. In addition, they consider depth ordering of tracked objects relative to the camera, resulting in the ability to predict occlusions between objects and allowing likelihoods coming from different cues. Maggio *et al.* [Maggio 08] propose a filtering framework for multi-target tracking based on particle filtering and data association, using graph matching. Their tracker is able to compensate for missing detections and to remove noise and clutter produced by the detector. In their approach, a novel particle resampling strategy is proposed, and, moreover, the dynamic and observation models are adapted to cope with various object scales. There are some shortcomings of the JPDAF: the JPDAF does not consider situations in which multiple measurements can be assigned to one object or where the same measurement is represented by two objects. Moreover, by using a JPDAF for data association, the processing time increases significantly with the number of objects since all possible hypotheses need to be calculated. It also cannot handle objects entering, or already tracked objects exiting, the field of view.

Another approach for data association is Multiple Hypothesis Tracking (MHT) [Reid 79]. In the MHT algorithm, several correspondence hypotheses for each object at each time are maintained and assessed. Using this approach, the correspondence decision is deferred until the most likely set of observation correspondences is found. In MHT, the probability of each potential track is calculated, and typically only the most probable of all the tracks is reported. The algorithm has the ability to create or terminate tracks of objects, entering or exiting the field of view. Moreover, it can also handle occlusions in a way that the continuation of a track is found even if some of the measurements from the object are missing. MHT makes associations in a deterministic sense and exhaustively enumerates all possible associations. A particle filtering approach that handles multiple measurements to track multiple objects has been proposed by [Hue 02]. In their method, data association is handled in a similar way as in MHT, however, the state estimation is achieved through particle filters. In comparison with the latter approaches, we use a Bayesian filtering approach. These approaches provide efficient solutions [Dore 10], useful to achieve a scalable and efficient communication load due to limited data exchange between cameras and the fusion centre. However, in our proposed method each camera sends a local estimate per person to the fusion centre, together with a reliability measure. The fusion centre hereby fuses these local estimates for a specific person of all cameras to one global estimate. The data association strategy is described in a bottom-up manner, meaning that evidence for a specific person is gathered locally in a camera. In the special case of occlusions, where a concrete data association strategy is needed since it could introduce ambiguities, we use probabilistic foreground modelling (see Section 6.5.1) to perform occlusion reasoning. Furthermore, we use an approach with no local state estimation, like in MHT, to explore the posterior probability, taking all current measurements into account. However, we do not yet keep track of different hypothesis over time. This particular idea and the exchange of several possible hypothesis of a single object can lead to further improvements of our method.

Relevant previous work at the IPI research group includes occupancy mapping [Tessens 10], [Morbee 11], [Grünwedel 12]. In [Tessens 10], a technique was developed to calculate ground occupancy maps with a set of calibrated and synchronized cameras. In particular, a fusion method of the ground occupancies based on Dempster-Shafer theory of evidence was proposed. The method yields very accurate occupancy detection results and it outperforms probabilistic occupancy mapping methods and fusion by summing in terms of concentration of the occupancy evidence. In [Morbee 11], building on the work of [Tessens 10], line sensors were used to calculate an accurate occupancy map. The emphasis of this work is the study of the usage of different sensors (cameras, line sensors), and their data output types (full images, scan lines from full images, scan lines from light-integrating line sensors). An overall comparison between the different systems was presented covering the obtained occupancy map quality, the memory and computational requirements, the price of the system, its

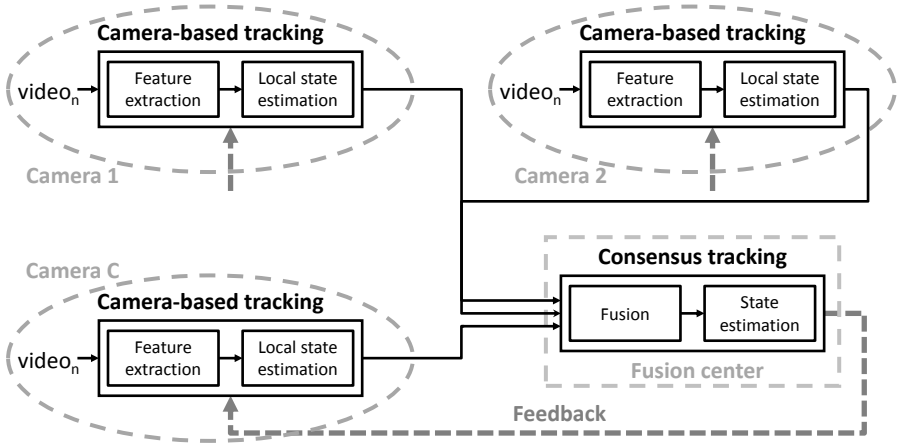


Figure 6.5: *Decentralized system architecture.* In the camera-based tracking block, executed on a smart camera, features (foreground blobs) are extracted from the input video. Then the local states of all people (the position on the ground plane, speed, width and height of each person) are estimated. Afterwards, a compact representation is sent to the fusion centre, which fuses the individual estimates into a global estimate, resulting in the best possible global state for each person. These states are fed back to each camera to correct possible mistakes.

power consumption and its privacy-friendliness. The work of [Grünwedel 12] further improves the occupancy mapping by introducing additional edge based foreground measurements for better evidence of people's positions.

6.3 Problem formulation

Our goal is to perform tracking of an unknown number of objects (people) observed by multiple cameras from different viewpoints. We formulate this problem as a *consensus tracking* that estimates the most probable *global state* of a hidden Markov process, given a set of independent *local estimates* of each camera obtained at each time instance t (see Figure 6.5).

In this context, we model a person as a cuboid of width w_{w} and height h_{w} at the location $(x_{\mathrm{w}}, y_{\mathrm{w}})$ on the ground plane, moving at velocity $\mathbf{v} = (\dot{x}_{\mathrm{w}}, \dot{y}_{\mathrm{w}})$, as shown in Figure 6.6. All of the values are expressed with respect to a world coordinate system as indicated by the subscript w. They are different for each person $m = 1, \ldots, M$ and vary over time as the person moves. Together they constitute an unknown *global state vector* $\mathbf{x}_t^m$ per person:

$$\mathbf{x}_t^m = (x_{\mathrm{w}}, \dot{x}_{\mathrm{w}}, y_{\mathrm{w}}, \dot{y}_{\mathrm{w}}, w_{\mathrm{w}}, h_{\mathrm{w}})^T . \tag{6.1}$$

One of the main challenges come from frequent occlusions of observed people, especially in meeting rooms where furniture occludes people in side views.

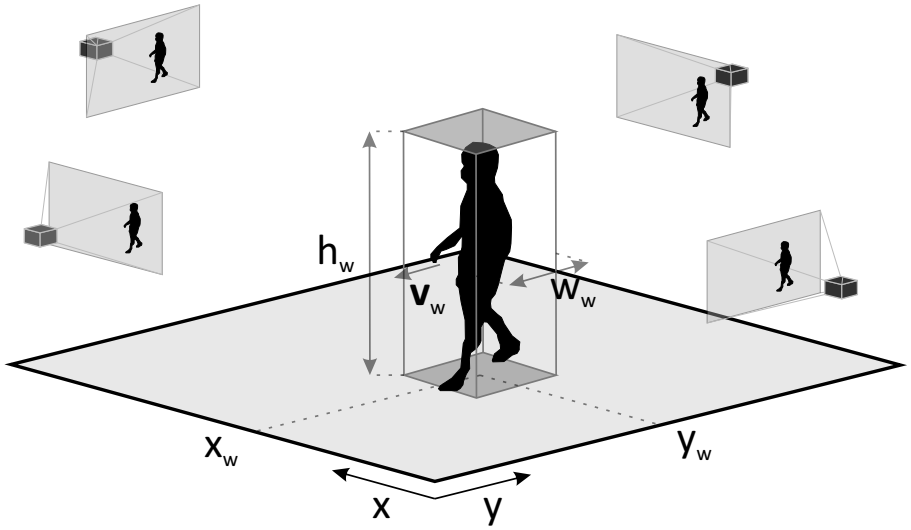
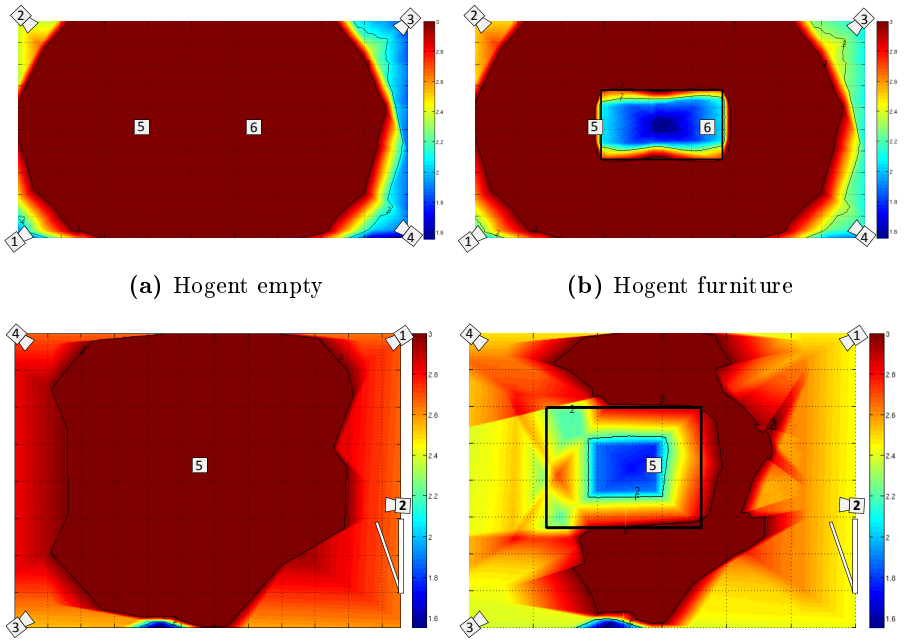


Figure 6.6: *Person model.* A person is modelled as a "cuboid" with an attached speed vector. The cuboid model is described by its state, composed of the location (x, y) on the ground plane, the speed $\mathbf{v} = (\dot{x}, \dot{y})$, the width and height (w, h). All of these variables are expressed in a real-world coordinate system.

Figure 6.7 shows camera coverage maps in our two demonstration environments, Hogent (six cameras in total, four side and two top views) and Alcatel-Lucent Bell Labs (five cameras in total, four side and one top view). We see that in furnished rooms tables and chairs occlude objects in majority of side views. Most areas are fully observed (without occlusions) by only one or two side view cameras. Occlusions in top views are less severe, but these views have higher distortion and contain less information for tracking (e.g. object appearance information). The occlusion problem is reduced by having multiple overlapping views, but as the coverage maps show, occlusion is still a very significant problem that needs to be addressed in our tracking approach.

6.4 Smart multi-camera system: our approach

In our approach every camera C calculates a local estimate of each person m, denoted as $\widehat{\mathbf{x}}_t^{C,m}$, and sends to the fusion centre the estimates together with the corresponding reliability measure $\mathbf{P}_t^{C,m}$. This compact representation reduces the communication load between the cameras and the fusion centre, which contributes to scalable and real-time tracking. Since every camera calculates the estimates $\widehat{\mathbf{x}}_t^{C,m}$ of each person locally and sends a compact representation of each person m to the fusion centre, it is appropriate to assume that the estimation vectors $\widehat{\mathbf{x}}_t^{C,m}$ are independent random variables.



(a) Hogent empty **(b)** Hogent furniture

(c) ALU empty **(d)** ALU furniture

Figure 6.7: *Camera coverage maps.* Camera coverage maps in our two demonstration environments: the one at Hogent (six cameras in total, four side and two top views) and the one at Alcatel-Lucent Bell Labs (five cameras in total, four side and one top view). We see that in furnished rooms tables and chairs occlude objects in majority of side views. Most areas are fully observed (without occlusions) by only one or two side view cameras.

There is a direct data association of local estimates $\mathbf{Z}_t^m$ to the global state $\mathbf{x}_t^m$. In Section 6.5 we explain why we can assume such a data association.

Given the vector $\mathbf{Z}_t^m = \left(\widehat{\mathbf{x}}_t^{1,m}, \ldots, \widehat{\mathbf{x}}_t^{C,m}\right)$, i.e., the set of local estimates of the m-th individual in C cameras at time t, our task is to find the global state, $\mathbf{x}_t^m$, as the posterior joint probability

$$p\left(\mathbf{x}_t^m | \mathbf{Z}_{1:t}^m\right).$$

Using Bayes' theorem and the Markov assumption as shown in [Thrun 05], we obtain for all individuals that

$$p\left(\mathbf{x}_t^m | \mathbf{Z}_{1:t}^m\right) = \eta \cdot p\left(\mathbf{Z}_t^m | \mathbf{x}_t^m\right) p\left(\mathbf{x}_t^m | \mathbf{Z}_{1:t-1}^m\right), \tag{6.2}$$

where $\eta = p\left(\mathbf{Z}_t^m | \mathbf{Z}_{1:t-1}^m\right)^{-1}$. The distribution $p\left(\mathbf{Z}_t^m | \mathbf{x}_t^m\right)$ is the likelihood of observing $\mathbf{Z}_t^m$ given the global state at time instance t, whereas $p\left(\mathbf{x}_t^m | \mathbf{Z}_{1:t-1}^m\right)$ is the predicted posterior probability computed at time $t-1$. The likelihood

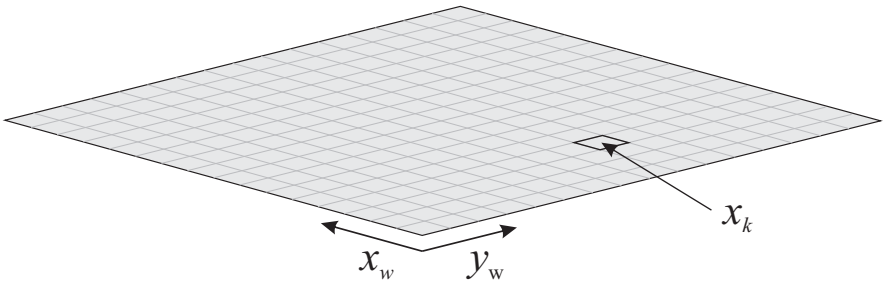


Figure 6.8: *Decomposition of* $\mathbf{x}_t^m$. We partition the ground plane into an evenly-sized grid with fixed resolution, namely a histogram filter [Thrun 05]. Histogram filters decompose the continuous state $\mathbf{x}_t^m$ into possible values x_k.

$p\left(\mathbf{Z}_t^m|\mathbf{x}_t^m\right)$ specifies the probabilistic law according to which the local estimates $\mathbf{Z}_t^m$ are generated from the global state $\mathbf{x}_t^m$.

The predicted posterior probability $p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t-1}^m\right)$ is given as follows

$$p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t-1}^m\right) = \int p\left(\mathbf{x}_t^m|\mathbf{x}_{t-1}^m\right) p\left(\mathbf{x}_{t-1}^m|\mathbf{Z}_{1:t-1}^m\right) d\mathbf{x}_{t-1}^m. \tag{6.3}$$

We track each person separately, i.e., the fusion centre uses a global Bayesian estimator for each person m, which results in an efficient solution suited for real-time applications with limited data exchange. The following two sections will explain in detail the estimation of the local estimates, $\widehat{\mathbf{x}}_t^{C,1}, \ldots, \widehat{\mathbf{x}}_t^{C,M}$, namely the *camera-based tracking* on the camera side (Section 6.5), and the estimation of $p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t}\right)$, i.e. the global Bayesian estimator per person m, denoted as *consensus tracking* on the fusion centre (Section 6.6).

6.5 Object tracking in smart cameras

In this section we outline the calculation of the single camera (local) estimates $\widehat{\mathbf{x}}_t^{C,1}, \ldots, \widehat{\mathbf{x}}_t^{C,M}$ in each camera C, using two approaches: *histogram filtering (HF)* or a method with *no local state estimation (NLE)* (each camera performs only assignment of observations to objects given by the fusion centre).

In the both methods we use not only the current camera's estimations, but also feedback from the fusion centre, which consists of the global posterior distributions of the global people's state $\mathbf{x}_{t-1}^m$ at the earlier time instance $t-1$, calculated by a Bayesian estimator on the fusion centre. The distribution for each person m is modelled as a Gaussian with the mean μ_{t-1}^m (the most likely global state of person m) and covariance matrix $\mathbf{K}_{t-1}^m$. Section 6.6 explains in detail how these distributions are estimated.

The local estimate of the people at time t in each camera C is computed from the posterior joint probability of the local states $\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M}$, given the

images of camera C acquired until time t

$$p\left(\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M} | I_{1:t}^C\right).$$

Note that $\mathbf{x}_t^{C,m}$ represents the *local* state vector of person m, estimated by camera C, whereas $\mathbf{x}_t^m$ is the *global* state vector of person m, estimated by all cameras.

Applying Bayes' theorem and the Markov assumption as shown in [Thrun 05], for all local states we obtain that

$$p\left(\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M} | I_{1:t}^C\right) = \eta \cdot p\left(I_t^C | \mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M}\right) p\left(\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M} | I_{1:t-1}^C\right), \tag{6.4}$$

where $\eta = p\left(I_t^C | I_{1:t-1}^C\right)^{-1}$. The distribution $p\left(I_t^C | \mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M}\right)$ is the likelihood of observing I_t^C given all local state vectors at time t, whereas $p\left(\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M} | I_{1:t-1}^C\right)$ is the predicted posterior probability from time $t-1$. The likelihood $p\left(I_t^C | \mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M}\right)$ specifies the observation model, i.e., the probabilistic law according to which the images I_t^C are generated from the local state vectors $\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M}$ for all persons in camera C.

We simplify Equation (6.4) and assume that all $\mathbf{x}_t^{C,m}$ are independent random variables, i.e. that people's trajectories are independent of trajectories of other people, and moreover, that individuals are not totally occluded by other people in the camera images $\mathbf{I}_{1:t}$:

$$p\left(\mathbf{x}_t^{C,1}, \ldots, \mathbf{x}_t^{C,M} | I_{1:t}^C\right) = \prod_{m=1}^{M} p\left(\mathbf{x}_t^{C,m} | I_{1:t}^C\right).$$

Taking into account the assumed conditional independence of the local state vectors, and assuming that the projection of a person does not overlap with projections of other persons, the posterior probability can be represented as

$$p\left(\mathbf{x}_t^{C,m} | I_{1:t}^C\right) = \eta \cdot \underbrace{p\left(I_t^C | \mathbf{x}_t^{C,m}\right)}_{\text{Likelihood}} \int \underbrace{p\left(\mathbf{x}_t^{C,m} | \mathbf{x}_{t-1}^{C,m}\right)}_{\text{Motion model}} p\left(\mathbf{x}_{t-1}^{C,m} | I_{1:t-1}^C\right) d\mathbf{x}_{t-1}^{C,m}. \tag{6.5}$$

While the assumption of independent random variables is not very restrictive in practice, the assumption of conditional independence is violated when one person occludes another. Therefore, we treat the occlusion problem separately and explain it in more detail in Section 6.5.1.

To simplify the calculations, we partition the state space into a set of discrete samples. For example, we partition the object's position state on the ground plane into a uniform grid, as shown in Figure 6.8. We approximate the continuous state $\mathbf{x}_t^{C,m}$, which describes the position of the tracked object m along the x ground plane axis, by a discrete state $\mathbf{x}_t^{C,m} = x_k$ representing the centre of the corresponding cell k. All cells x_k are pairwise disjoint, i.e.

$x_k \cap x_l = \varnothing$ for each $k \neq l$, and all cells x_k together constitute the whole ground plane. Therefore, in the remainder of this chapter, $\mathbf{x}_t^{C,m}$ refers to a discrete random variable describing a set of possible values x_k.

Consequently, the posterior probability $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$ becomes a discrete probability distribution. Moreover, as shown in [Thrun 05], Equation (6.2) can be approximated for discrete values x_k of $\mathbf{x}_t^{C,m}$ as

$$p\left(\mathbf{x}_t^{C,m} = x_k|I_{1:t}^C\right) = \eta \cdot \underbrace{p\left(I_t^C|\mathbf{x}_t^{C,m} = x_k\right)}_{\text{Likelihood}} p\left(\mathbf{x}_t^{C,m} = x_k|I_{1:t-1}^C\right), \qquad (6.6)$$

where

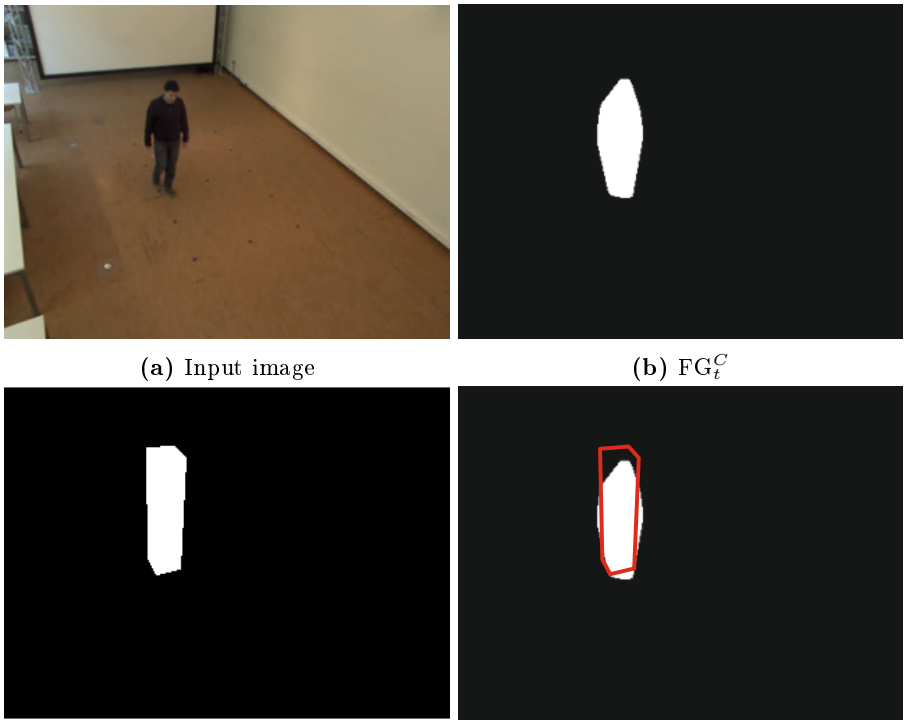
$$p\left(\mathbf{x}_t^{C,m} = x_k|I_{1:t-1}^C\right) = \sum_i \underbrace{p\left(\mathbf{x}_t^{C,m} = x_k|\mathbf{x}_{t-1}^{C,m} = x_i\right)}_{\text{Motion model}} p\left(\mathbf{x}_{t-1}^{C,m} = x_i|I_{1:t-1}^C\right)$$

In the following sections we explain the estimation of the likelihood (the observation model) and two on-camera tracking approaches based on Equation (6.6).

6.5.1 Foreground/background measurement

Whether or not a particular part of a person's silhouette is detected as foreground in the camera depends not only on geometrical considerations, but also on other factors, mainly the speed of the person (immobile persons are often invisible in FG/BG segmentation), the appearance similarity with the background, and occlusions. Since the method of Grünwedel *et al.* [Grünwedel 11b], which we use for FG/BG segmentation (explained in Section 2.2.2), successfully copes with the problem of the movement speed and appearance, in this section we specifically address the problem of FG/BG segmentation in the presence of occlusions. In case of no occlusion, a foreground blob can be considered to belong to a single person and the likelihood estimation is relatively straightforward. In the case of occlusions, this representation cannot be guaranteed and the FG/BG segmentation can result in foreground blobs that represent the presence of multiple persons in a single blob. Hence, it is necessary to introduce additional reasoning to treat this problem.

In this context, we use *probabilistic foreground modelling* to perform occlusion reasoning, i.e. we relate the projection of $\mathbf{x}_t^{C,m} = x_k$ to the image FG_t^C produced by FG/BG segmentation. We calculate the likelihood $p\left(I_t^C|\mathbf{x}_t^{C,m} = x_k\right)$ using probabilistic foreground modelling as follows. Let $\Omega_k^{C,m}$ be the image area obtained by the projection of the cuboid model associated with the local state $\mathbf{x}_t^{C,m} = x_k$ (shown in Figure 6.6) onto the image of camera C. We define $\Omega_t^{C,m}(i)$ as a binary image where the pixel i is described as being part of this projection, with the value "1" if part, or "0" if not. Such an image is a function of $\mathbf{x}_t^{C,m} = x_k$. Furthermore, let $\mathrm{FG}_t^C(i)$ be a binary image that represents the result of the FG/BG segmentation on camera C at time t.



(a) Input image **(b)** FG_t^C

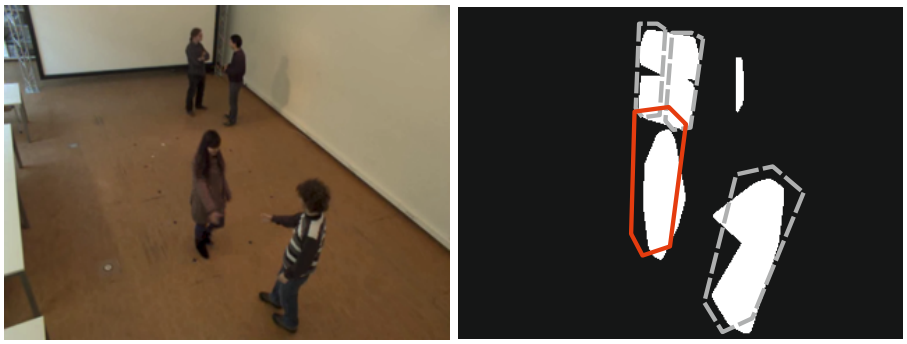
(c) $\Omega_k^{C,m}$ **(d)** FG_t^C with $\Omega_k^{C,m}$ overlay

Figure 6.9: *Occlusion reasoning.* The principle of occlusion reasoning is to relate the projection $\Omega_k^{C,m}$ of the local state $\mathbf{x}_t^{C,m} = x_k$ to the image produced by FG/BG segmentation FG_t^C. From the input image (a), the foreground mask (b) is calculated. The projection $\Omega_k^{C,m}$ for a particular representation x_k of the local state $\mathbf{x}_t^{C,m}$ is shown in (c) and compared to the foreground mask (d).

In cases with *no occlusions*, to calculate the likelihood $p\left(\mathrm{FG}_t^C | \mathbf{x}_t^{C,m} = x_k\right)$ for $\mathbf{x}_t^{C,m} = x_k$, we only take into account the projection $\Omega_k^{C,m}$ within the image of that camera C (Figure 6.9). We model the image FG_t^C produced by FG/BG segmentation as the ideal FG/BG image with random noise. Since the likelihood estimation error increases empirically with the area of $\Omega_k^{C,m}$, we introduce a normalized pseudometric d to account for this. For the binary image $\Omega_k^{C,m}$, we denote by $\left|\Omega_k^{C,m}\right|$ the number of pixels with value "1", and by $\otimes$ the product per pixel of two images. Then, we define d as

$$d\left(\Omega_k^{C,m}, \mathrm{FG}_t^C\right) = 1 - \left(\frac{\left|\Omega_k^{C,m} \otimes \mathrm{FG}_t^C\right|}{\left|\Omega_k^{C,m}\right|}\right)\left(1 - \frac{\left|\left(1 - \Omega_k^{C,m}\right) \otimes \mathrm{FG}_t^C\right|}{\left|\mathrm{FG}_t^C\right|}\right), \quad (6.7)$$

where $\left(1 - \Omega_k^{C,m}\right)$ denotes the complement of $\Omega_k^{C,m}$. The first factor of the



(a) Input image **(b)** FG_t^C

Figure 6.10: *Probabilistic foreground modeling.* From the input image (a), the foreground mask (b) is calculated. The projection of the remaining local state vectors (gray dashed line) and the current local state vector (red line) is shown. Using this likelihood function it is possible to find an estimate of the current local state $\mathbf{x}_t^{C,m}$ despite presence of occlusions in the imageby other people.

pseudometric d describes how well the cuboid person model matches the given foreground mask, whereas the latter specifies how much foreground is represented outside the person model. The model measures the agreement between our assumed person model (Figure 6.6) and the observed foreground mask.

We use the following model for $p\left(\mathrm{FG}_t^C|\mathbf{x}_t^{C,m}=x_k\right)$:

$$p\left(\mathrm{FG}_t^C|\mathbf{x}_t^{C,m}=x_k\right)=\frac{1}{\sigma}e^{-\frac{1}{\sigma}d\left(\Omega_k^{C,m},\mathrm{FG}_t^C\right)}.$$

The parameter σ accounts for the quality of the FG/BG segmentation. The smaller the σ is, the more the FG_t^C is picked around its ideal value $\Omega_k^{C,m}$. In our experiments we empirically set the value of σ to 0.01.

However, in the presence of occlusions let us assume that at each time t a person may be only partially visible in the camera of interest due to occlusions by other people. In these cases, the foreground mask FG_t^C may contain blobs with silhouettes of several people in a single blob. Therefore, in the estimation of the likelihood $p\left(\mathrm{FG}_t^C|\mathbf{x}_t^{C,m}=x_k\right)$ we take into account the local state vectors of the remaining persons in the scene, $\mathbf{x}_t^{C,1},\ldots,\mathbf{x}_t^{C,m-1},\mathbf{x}_t^{C,m+1},\ldots,\mathbf{x}_t^{C,M}$. For a particular state x_k of a person m, we use the estimates of the remaining local state vectors obtained at time $t-1$, $\widehat{\mathbf{x}}_t^{C,1},\ldots,\widehat{\mathbf{x}}_t^{C,m-1},\widehat{\mathbf{x}}_t^{C,m+1},\ldots,\widehat{\mathbf{x}}_t^{C,M}$.

Let $\mathrm{OM}_t^{C,m}$ be an occlusion map for an individual m in camera C, which combines all projections of the remaining individuals $\mathbf{x}_t^{C,1},\ldots,\mathbf{x}_t^{C,m-1},\mathbf{x}_t^{C,m+1},\ldots,\mathbf{x}_t^{C,M}$ as

$$\mathrm{OM}^{C,m}=\oplus_{\substack{n=1\\n\neq m}}^{M}\Omega_{t-1}^{C,n}, \tag{6.8}$$

where $\oplus$ denotes the "union" between binary images, and $\Omega_{t-1}^{C,m}$ is the projection of the cuboid model at time $t-1$ (as shown in Figure 6.6). In other words, the occlusion map $\mathrm{OM}_t^{C,m}$ for an individual m characterizes the cuboid model projections for all individuals excluding the individual m. Note that the occlusion map $\mathrm{OM}_t^{C,m}$ is still a binary image.

The new pseudometric d can be calculated as

$$d_{\mathrm{occ}}\left(\Omega_k^{C,m}, \mathrm{FG}_t^C\right) = 1 - \left(\frac{\left|\Omega_k^{C,m} \otimes \mathrm{FG}_t^C\right|}{\left|\Omega_k^{C,m}\right|}\right) \left(1 - \frac{\left|\left(1-\Omega_k^{C,m}\right) \otimes \left(\mathrm{FG}_t^C \otimes \mathrm{OM}_t^{C,m}\right)\right|}{\left|\mathrm{FG}_t^C\right|}\right). \quad (6.9)$$

The first factor is the same as in Equation (6.7), and describes how well the cuboid person model matches the given foreground mask. The latter specifies how much foreground is represented outside the person model.

Therefore, the conditional distribution $p\left(\mathrm{FG}_t^C | \mathbf{x}_t^{C,m} = x_k\right)$ in the case of occlusion is defined as follows:

$$p\left(\mathrm{FG}_t^C | \mathbf{x}_t^{C,m} = x_k\right) = \frac{1}{\sigma} e^{-\frac{1}{\sigma} d_{\mathrm{occ}}\left(\Omega_k^{C,m}, \mathrm{FG}_t^C\right)}. \quad (6.10)$$

In this way, using this likelihood function we make an estimate of the current local state of each person, $\mathbf{x}_t^{C,m}$, despite occlusions with other people.

6.5.2 Signature based appearance modelling

In Chapter 4 we showed how it is possible to use signatures of observed objects to track them in a single camera view. In a multi-camera network with overlapping views it is possible to go a step further and use the signatures to create a multi-view appearance model. In this section we explain how we create such appearance models for each object, and how we use them to obtain an additional robust measurement for the on-camera tracking.

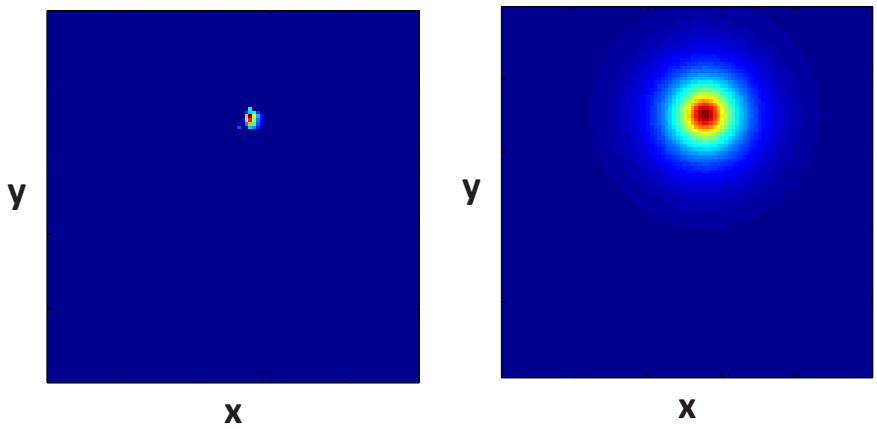
Let us assume we have a single object (a human) viewed by multiple cameras from different viewpoints (see Figure 6.11). This enables capturing the object appearance from multiple sides simultaneously. Moreover, since we express the location of such an object in real-world coordinates on the ground plane (as shown in Figure 6.6), and since we assume that the camera network is extrinsically calibrated, we can derive a movement direction of objects relative to each camera (e.g. towards the camera, from the left to the right side, away from the camera, etc.). In this way, based on the object's movement, each camera can expect which appearance it would see (frontal, side-view, backside, etc.; we assume the objects move typically in their frontal direction). Once the appearance model is constructed, these camera expectations helps during the

Figure 6.11: As a person moves through the multi-camera environment it is possible to create a multi-view appearance model of that person. Moreover, based on the person's movement direction relative to each camera, the cameras can infer what viewpoint they have at the person and take the appropriate appearance template for the signature measurement.

tracking to speed up the selection of the right appearance template from the model.

For construction of the multi-view appearance model we use the method explained in detail in Section 5.3. Let us assume that over time the cameras capture N views of the tracked object. In each view we calculate the horizontal and vertical signature, $\mathbf{h}_I$ and $\mathbf{v}_I$, of the captured image I, as defined in Section 3.3. These signatures calculated from N views construct N signature vectors $\mathbf{s}_I^k, k = 1 \ldots N$. Using our method for computing a multi-view appearance model, out of those N vectors we find a predefined number M of the most uncorrelated (the most informative) ones and include these into the appearance model. In our camera setups these were typically signature vectors from relatively uniformly distributed views (frontal, 2 side-views, backside, and views in between), as shown in Figure 6.11.

Once the appearance model is constructed, at each time instance t we obtain a signature based measurement for tracking, in the way explained in Section 4.3. By comparing the signatures of the observation at time t with the most appropriate template from the appearance model (the template that corresponds to the same moving direction) we calculate the measurement to update the



(a) $p\left(\mathbf{x}_{t-1}^{m}|\mathbf{I}_{t-1}\right)$ (b) $p\left(\mathbf{x}_{t}^{m}|\mathbf{I}_{t-1}\right)$

Figure 6.12: *Example distribution describing the motion model.* We use a very simple and unconstrained motion model. A human can move in any direction with an average speed of 1.4 m/s, but is limited by a maximal speed of 4.0 m/s. In (a), the posterior probability $p\left(\mathbf{x}_{t-1}^{m}|\mathbf{I}_{t-1}\right)$ at time $t-1$ is shown. Now, we apply the described motion model which result in the distribution shown in (b). Note, the posterior is discretized in an evenly-spaced grid and the corresponding probabilities are not normalized for a better graphical representation. As can be seen, the model is isotropic and describes how likely it is for every cell that a person has moved there.

object's position and the apparent size. This measurement is used together with the FG/BG measurement in our on-camera tracking approaches.

6.5.3 Context aware motion model

In many people tracking algorithms motion of a person is described using a very simple and unconstrained motion model [Fleuret 08]. It can be assumed that a human can move for a distance d_a in any direction with an average speed of 1.4 m/s at each time instance t, but this movement is limited by a maximal distance $d_{\max}$ according to a maximal speed of 4.0 m/s (Figure 6.12):

$$p\left(\mathbf{x}_{t}^{m}=x_k|\mathbf{x}_{t-1}^{m}=x_i\right)=\begin{cases}\eta\cdot e^{-\frac{1}{d_a}\|x_k-x_i\|}, & \text{if } \|x_k-x_i\|<d_{\max}\\ 0, & \text{otherwise.}\end{cases} \tag{6.11}$$

Here, η represents a normalization factor.This model is isotropic and the function of (6.11) decreases with the distance from location x_k to zero, if the distance is greater than the maximal distance $d_{\max}$. In the special case of $d_a \to \infty$, the model becomes a uniform distribution within the boundaries of the maximal allowed distance $d_{\max}$.

However, this simple and unconstrained motion model does not take into account contextual information, which is its significant drawback. For instance,

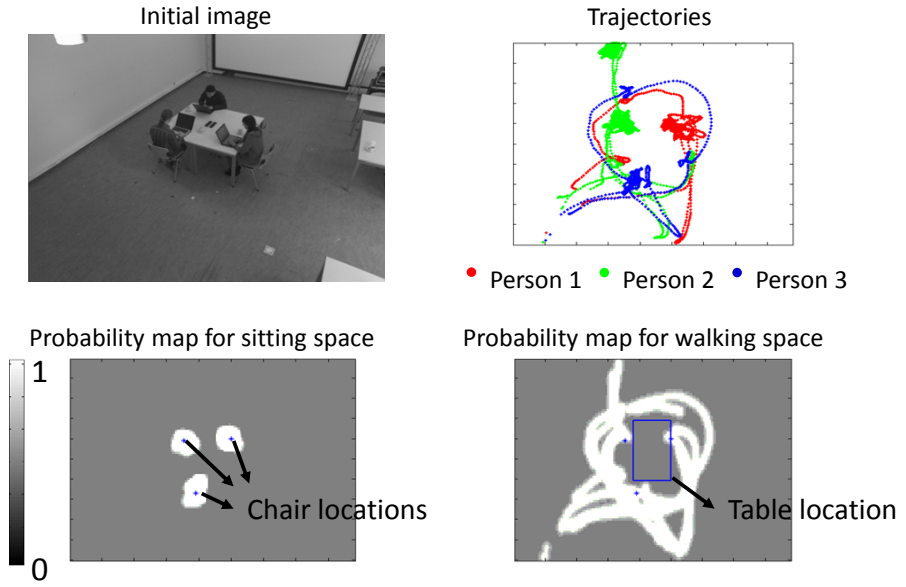


Figure 6.13: Based on trajectories of people we construct two probability maps in the meeting room, which represent the sitting and the walking space. The regions with high probability in the sitting space map are recognized as chairs, while the table positions are estimated in the non-walking areas between the chairs.

in the environments such as meeting rooms, the motion of people is constrained by furniture. Furthermore, when people sit down their motion is limited until they stand up again. We use this information to create a contextually aware motion model. We estimate position of furniture in the meeting room and reduce probability of people's trajectories at these positions. We also estimate sitting and standing states of tracked people to adapt the motion prediction accordingly.

Initially we start from the aforementioned unconstrained motion model and over time learn position of tables and chairs to adapt the model. We use trajectories of people to recognize tables and chairs in a meeting room. Here we give an overview of this method, and refer the interested reader to the work of [Xie 12] for more details.

The trajectories used for furniture estimation include ground locations of people and their approximate height, as a function of time. Here, the height refers to the top of person's head, and can change as the user performs different activities. We assume it to be around 1.8m when the person is standing and 1.2m when sitting. We define three kinds of human activities: sitting, standing and walking, out of which sitting and walking are important for detection of furniture. Using *sitting* activity we are able to detect chairs in the room, while by *walking* activity and the detected chairs it is feasible to detect tables. The

standing activity is not involved in object recognition directly, but we use it as a transitional state between sitting and walking.

Over time, we construct two probability maps of the meeting room, which represent the sitting and the walking space. The sitting space map contains probabilities of predefined ground plane cells being occupied by a chair. Regions with high probability in the sitting space map are recognized as candidate chairs. Analogously, the walking space map represents probabilities of ground plane cells being free for walking (not occupied by furniture). This map is calculated using trajectories of tracked people and learning where they can walk. The locations of tables we derive using both probability maps. We take the assumption that a ground plane area is occupied by a table if people do not walk in that area, and if chairs are detected around that area (see Figure 6.13). With these assumptions, the largest rectangle surrounded by walking space and chairs is considered as the table, and it is adjusted according to the locations of the chairs.

Suppose that W_t is the walking space map obtained by the time t. If the map value for the ground plane cell k is zero, it means that this cell is certainly occupied by furniture, i.e. it is not possible to walk at this ground plane location. Using this approach we modify the motion model given by Equation (6.11) to include this contextual information about the environment. The factor $p(x_k|W_{t-1})$, which is introduced into the motion model, is the probability of the motion at the ground plane location k at time t, given the walking space map constructed by the time $t-1$. Using this factor, the information about the furniture locations is propagated into the enriched motion model:

$$p\left(\mathbf{x}_t^m = x_k | \mathbf{x}_{t-1}^m = x_i\right) = \begin{cases} \xi \cdot e^{-\frac{1}{d_a}\|x_k - x_i\|} \cdot p(x_k|W_{t-1}), & \text{if } \|x_k - x_i\| < d_{\max} \\ 0, & \text{otherwise.} \end{cases} \tag{6.12}$$

Here, ξ represents a normalization factor, like in Equation (6.11). If the location k is certainly free for walking, the motion model at that location is the same isotropic one as in Equation (6.11), illustrated in Figure 6.12. Conversely, if the cell is certainly occupied by furniture, $p(x_k|W_{t-1})$ is zero, meaning that this enriched motion model bends around furniture-occupied cells and becomes anisotropic. In this way, the motion model adapts to the environmental context.

6.5.4 Histogram filtering tracking

In this section, we describe the estimation of the local state $\mathbf{x}_t^{C,m}$ in each camera based on a histogram filtering approach. The goal is to estimate the local state $\mathbf{x}_t^{C,m}$ of each person in camera C on a frame-by-frame basis. This has the advantage that a smart camera can track people independently for a certain number of frames, operating on the frame rate of the smart camera itself. This is an advantage over the later-explained approach with no local state estimation (Section 6.5.5), which depends on the feedback frequency from the fusion centre.

Furthermore, the feedback from the fusion centre ensures that possible failures in the local state estimation are corrected. We compare the local states $\mathbf{x}_t^{C,m}$ to the feedback from the fusion centre and re-initialize the filter from the current feedback if we encounter a probable failure (i.e. the distance is bigger than the assumed width of a person). The decentralized system architecture could take advantage of this design and further reduce the communication load. For instance, a smart camera could only be corrected when it is definitely wrong, which saves bandwidth, and hence, energy.

As already mentioned at the beginning of Section 6.5, we use a discrete representation of the posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$. The continuous state $\mathbf{x}_t^m$ is now approximated by a discrete one, i.e., the centre of the cell to which $\mathbf{x}_t^m = x_k$ belongs. All cells x_k are pairwise disjoint, i.e., $x_k \cap x_l = \varnothing$ for each $k \neq l$, and describes an area (cell) on the ground plane. All x_k positions together constitute the whole ground plane.

We recursively estimate the local state $\mathbf{x}_t^{C,m}$, according to Equation (6.6). Here, the motion model is the context-aware model explained in Section 6.5.3. The calculation of the likelihood $p\left(I_t^C|\mathbf{x}_t^{C,m} = x_k\right)$ for the local state $\mathbf{x}_t^{C,m}$ was explained in Sections 6.5.1 and 6.5.2, using FG/BG and signature based measurements.

The resulting posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$ is approximated by a Gaussian, $N_{\mathbf{x}_t^{C,m}}\left(\widehat{\mathbf{x}}_t^{C,m}, \mathbf{P}_t^{C,m}\right)$, with the mean $\widehat{\mathbf{x}}_t^{C,m}$, and a corresponding covariance matrix $\mathbf{P}_t^{C,m}$. This limits the data exchange between cameras and the fusion centre to a few parameters per person, and is important for a real-time, low-latency and scalable multi-camera system. Here, the mean $\widehat{\mathbf{x}}_t^{C,m}$ of the Gaussian is obtained by:

$$\widehat{\mathbf{x}}_t^{C,m} = \eta \sum_{k=1}^{M} p\left(\mathbf{x}_t^{C,m} = x_k|I_{1:t}^C\right) x_k,$$

where $\eta = \left(\sum_{k=1}^{M} p\left(\mathbf{x}_t^{C,m} = x_k|I_{1:t}^C\right)\right)^{-1}$ and M is the number of cells.

Furthermore, the covariance matrix $\mathbf{P}_t^{C,m}$ is estimated over all M cells using the mean $\widehat{\mathbf{x}}_t^{C,m}$:

$$\mathbf{P}_t^{C,m} = \eta \sum_{k=1}^{M} \left(x_k - \widehat{\mathbf{x}}_t^{C,m}\right)\left(x_k - \widehat{\mathbf{x}}_t^{C,m}\right)^T,$$

where $\eta = \left(\sum_{k=1}^{M} p\left(\mathbf{x}_t^{C,m} = x_k|I_{1:t}^C\right)\right)^{-1}$. The covariance matrix $\mathbf{P}_t^{C,m}$ is of vital importance since it is a reliability measure of the local estimate which is sent to the fusion centre.

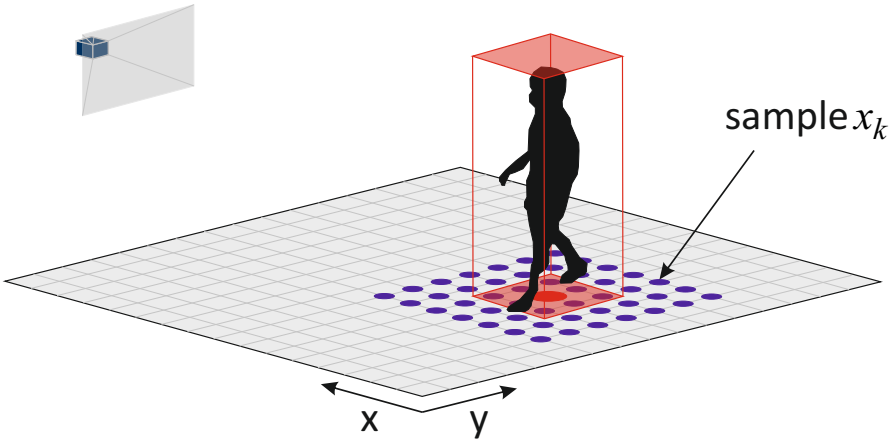


Figure 6.14: *Tracking with no local state estimation (NLE).* Using the uncertainties described in the covariance matrix $\mathbf{K}_{t-1}^{m}$ of the feedback, we can create an uncertainty area W_t^m around the last known position of the person m. Within this area we distribute possible samples $\mathbf{x}_t^{C,m} = x_k \in W_t^m$.

6.5.5 Tracking without local state estimation

In this section, we describe the estimation of the local state $\mathbf{x}_t^{C,m}$ in each camera based on an approach with no local state estimation. In this approach, the cameras do not estimate the local state recursively, as described in the previous approach. The predicted local posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t-1}^{C}\right)$ is approximated using the feedback of the fusion centre at time $t-1$

$$p\left(\mathbf{x}_t^{C,m}|I_{1:t-1}^{C}\right) \approx p\left(\mathbf{x}_{t-1}^{m}|\mathbf{Z}_{1:t-1}\right),$$

described as a normal distribution $N_{\mathbf{x}_{t-1}^{m}}\left(\mu_{t-1}^{m}, \mathbf{K}_{t-1}^{m}\right)$. Here, $p\left(\mathbf{x}_{t-1}^{m}|\mathbf{Z}_{1:t-1}\right)$ is the predicted global state $\mathbf{x}_{t-1}^{m}$. Using the uncertainties described in the covariance matrix $\mathbf{K}_{t-1}^{m}$ of the feedback, we can create an uncertainty area W_t^m around the last-known state $\mathbf{x}_{t-1}^{m}$ of the person m at time instance $t-1$. For this uncertainty area W_t^m, we only consider possible locations of person m on the ground plane in this uncertainty area and treat the remaining state variables of $\mathbf{x}_{t-1}^{m}$ as constant. The assumption is that person m can physically only move within the area W_t^m by time t.

Within this area, we distribute K possible samples, denoted as $\mathbf{x}_t^{C,m} = x_k \in W_t^m$ with $k = 1, \ldots, K$, according to the predicted global posterior distribution $p\left(\mathbf{x}_{t-1}^{m}|\mathbf{Z}_{1:t-1}\right)$, which is defined as a normal distribution. Then, the likelihood $p\left(I_t^C|\mathbf{x}_t^{C,m} = x_k\right)$ for each sample x_k is calculated. The result is a good approximation of the posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^{C}\right)$ of each person m in camera C around the last-known global state $\mathbf{x}_{t-1}^{m}$. The calculation of the

likelihood $p\left(I_t^C|\mathbf{x}_t^{C,m}=x_k\right)$ for the local state $\mathbf{x}_t^{C,m}$ is explained in Sections 6.5.1 and 6.5.2, using FG/BG and signature based measurements.

For a scalable and efficient communication load, limited data exchange between cameras and the fusion centre is essential. Therefore, the resulting posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$ is approximated by a Gaussian, $N_{\mathbf{x}_t^{C,m}}\left(\widehat{\mathbf{x}}_t^{C,m},\mathbf{P}_t^{C,m}\right)$, with the mean $\widehat{\mathbf{x}}_t^{C,m}$, and a corresponding covariance matrix $\mathbf{P}_t^{C,m}$. The mean $\widehat{\mathbf{x}}_t^{C,m}$ of the Gaussian is chosen as the mean of the posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$,

$$\widehat{\mathbf{x}}_t^{C,m}=\eta\sum_{k=1}^{K}p\left(\mathbf{x}_t^{C,m}=x_k|I_{1:t}^C\right)x_k,$$

where $\eta=\left(\sum_{k=1}^{K}p\left(\mathbf{x}_t^{C,m}=x_k|I_{1:t}^C\right)\right)^{-1}$. The covariance matrix $\mathbf{P}_t^{C,m}$ is estimated from all K samples and the mean $\widehat{\mathbf{x}}_t^{C,m}$. Moreover, the covariance matrix $\mathbf{P}_t^{C,m}$ is of vital importance since it is a reliability measure of the local estimate that is sent to the fusion centre.

6.6 Consensus tracking

In this section, we describe the estimation of the global state $\mathbf{x}_t^m$. The goal is to fuse the likelihood distributions $p\left(\widehat{\mathbf{x}}_t^{1,m},\ldots,\widehat{\mathbf{x}}_t^{C,m}|\mathbf{x}_t^m\right)$ of each camera C to a final decision $p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t}^m\right)$ for each person m (Equation (6.2)). Consensus tracking uses a *global Bayesian estimator* per person (as shown in Figure 6.5). In the camera-based tracking of each smart camera C, the local state posterior distribution $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$ of each person m is approximated by a Gaussian distribution $N_{\mathbf{x}_t^{C,m}}\left(\widehat{\mathbf{x}}_t^{C,m},\mathbf{P}_t^{C,m}\right)$ and the parameters are sent to the fusion centre. In this way we ensure that the communication is data efficient and fast, saving bandwidth and energy.

To estimate the global state $\mathbf{x}_t^m$ of each person m, we use a Bayesian filter which calculates the posterior probability based on a motion model and the acquired local estimates $\widehat{\mathbf{x}}_t^{1,m},\ldots,\widehat{\mathbf{x}}_t^{C,m}$ from every smart camera (Figure 6.15). The Bayesian filter algorithm is a recursive estimator and splits Equation (6.2) into a *prediction* and a *correction* step. As defined in Equation (6.3), the prediction is computed as follows:

$$p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t-1}^m\right)=\int p\left(\mathbf{x}_t^m|\mathbf{x}_{t-1}^m\right)p\left(\mathbf{x}_{t-1}^m|\mathbf{Z}_{1:t-1}^m\right)d\mathbf{x}_{t-1}^m.$$

Here, the state transition probability $p\left(\mathbf{x}_t^m|\mathbf{x}_{t-1}^m\right)$ describes the motion model for person m and $p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t-1}^m\right)$ is the predicted posterior probability.

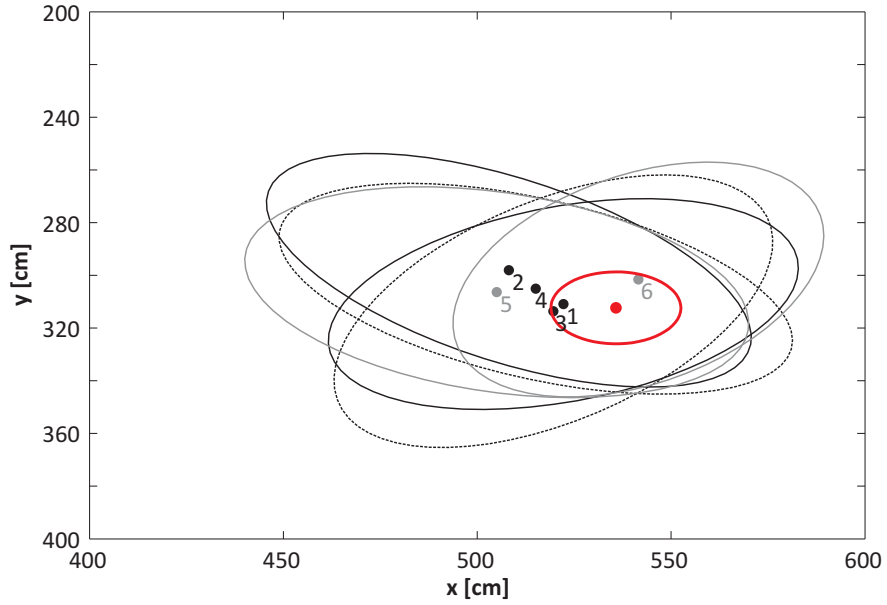


Figure 6.15: *Fusion of the local estimates from each camera.* The local posterior $p\left(\mathbf{x}_t^{C,m}|I_{1:t}^C\right)$ of person 1, sent by each smart camera C, is shown at time t. Here, the positional component of the mean $\widehat{\mathbf{x}}_t^{C,1}$ and the covariance matrix $\mathbf{P}_t^{C,m}$ are visualized for each camera (1 to 6). Since the posterior is approximated by a Gaussian distribution, it can be depicted as an ellipse with the semi-major and semi-minor axis, derived from the diagonal of the covariance matrix $\mathbf{P}_t^{C,m}$. For a better display, the semi-major and semi-minor axis are four times the standard deviation of the covariance matrix. The final fusion result is illustrated as a red ellipse.

The correction step takes the local estimates $\widehat{\mathbf{x}}_t^{1,m}, \ldots, \widehat{\mathbf{x}}_t^{C,m}$ from each smart camera into account and is defined as (Equation (6.2))

$$p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t}^m\right) = \eta p\left(\widehat{\mathbf{x}}_t^{1,m}, \ldots, \widehat{\mathbf{x}}_t^{C,m}|\mathbf{x}_t^m\right) p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t-1}^m\right),$$

where $\eta = p\left(\mathbf{Z}_t^m|\mathbf{Z}_{1:t-1}^m\right)^{-1}$. The likelihood distribution $p\left(\mathbf{Z}_t^m|\mathbf{x}_t^m\right)$ is the fusion of all local estimates $\mathbf{Z}_t^m = \left(\widehat{\mathbf{x}}_t^{1,m}, \ldots, \widehat{\mathbf{x}}_t^{C,m}\right)$ from every smart camera (see Figure 6.16).

As a particular implementation of the Bayesian filter in our proposed tracker, we use a linear Kalman filter [Kalman 60], denoted as *global Kalman filter*. The Kalman filter represents the posterior probability $p\left(\mathbf{x}_t^m|\mathbf{Z}_{1:t}^m\right)$ by the moments, the mean μ_t^m (the most likely global state of person m) and the corresponding covariance matrix $\mathbf{K}_t^m$. In other words, the Kalman filter is a parametric filtering approach and estimates the parameters of the following

normal distribution

$$N_{\mathbf{x}_t^m}\left(\mu_t^m, \mathbf{K}_t^m\right).$$

A linear Kalman filter assumes that the evolution of a person's state over time is described by the following state transition equation:

$$\mathbf{x}_t^m = \mathbf{A}_t \mathbf{x}_{t-1}^m + \mathbf{B}_t \mathbf{u}_t + \boldsymbol{\epsilon}_t \tag{6.13}$$

in which $\boldsymbol{\epsilon}_t$ is a multivariate Gaussian random variable. Here, $\mathbf{x}_t^m$ and $\mathbf{x}_{t-1}^m$ are state vectors, and $\mathbf{u}_t$ is the control vector at time t. Since we do not have control data in our system, we can omit the term $\mathbf{B}_t\mathbf{u}_t$. $\mathbf{A}_t$ is thereby the state-transition matrix and $\boldsymbol{\epsilon}_t$ the process noise. This is referred to as the prediction step. We use the constant velocity model in the state-transition modelling $\mathbf{A}_t$, which is defined as

$$\mathbf{A}_t = \begin{bmatrix} \mathbf{F} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & \mathbf{F} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & \operatorname{diag}(1,1) \end{bmatrix}, \mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}. \tag{6.14}$$

The vector $\boldsymbol{\epsilon}_t$ is modelled as a multivariate Gaussian random variable with zero mean and the covariance $\mathbf{Q}_t$. As shown in [Thrun 05], the covariance $\mathbf{Q}_t$ can be expressed as

$$\mathbf{Q}_t = \begin{bmatrix} \mathbf{D}(\sigma_{\dot{x}}) & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & \mathbf{D}(\sigma_{\dot{y}}) & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & \operatorname{diag}\left(\sigma_w^2, \sigma_h^2\right) \end{bmatrix}, \mathbf{D}(\sigma) = \begin{bmatrix} \frac{\sigma^2}{3}\Delta t^3 & \frac{\sigma^2}{2}\Delta t^2 \\ \frac{\sigma^2}{2}\Delta t^2 & \sigma^2 \Delta t \end{bmatrix}. \tag{6.15}$$

where $\left(\sigma_{\dot{x}}^2, \sigma_{\dot{y}}^2\right)$ are variances for velocity noise, and $\left(\sigma_w^2, \sigma_h^2\right)$ the variances describing the noise for width and height of a person; Δt refers to the time difference between two time instances.

The Kalman theory assumes that states cannot be observed directly. Therefore, in the correction step, the available inputs $\mathbf{Z}_t^m$, which are a linear function of the unknown global state $\mathbf{x}_t^m$, are incorporated into the Kalman filter as follows

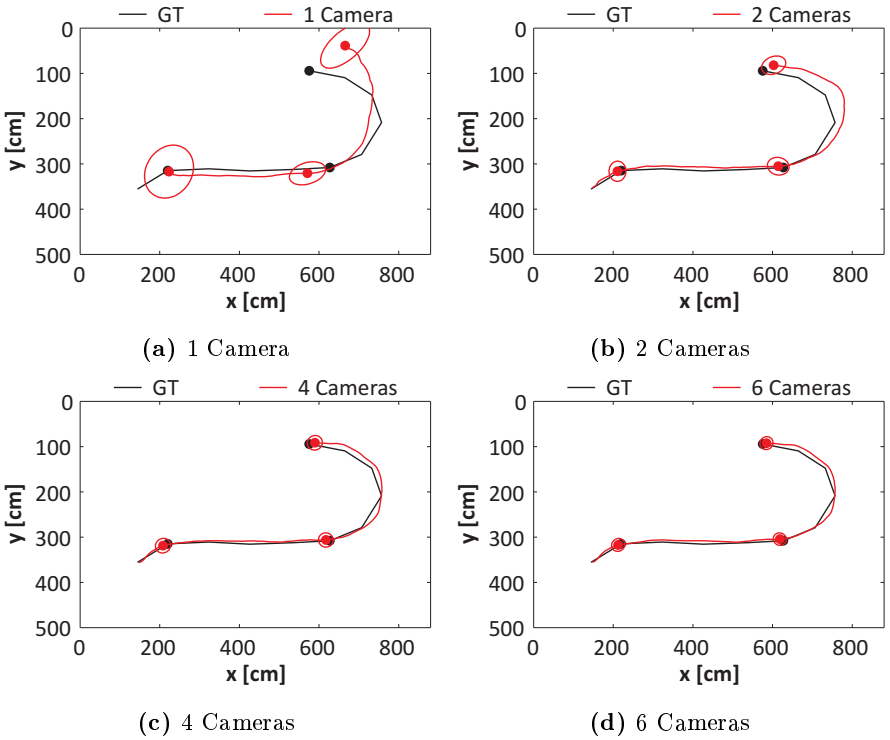
$$\mathbf{Z}_t^m = \mathbf{C}_t \mathbf{x}_t^m + \boldsymbol{\delta}_t. \tag{6.16}$$

Here, $\mathbf{C}_t$ corresponds to the measurement update matrix. The distribution of $\boldsymbol{\delta}_t$ is a multivariate Gaussian with zero mean and covariance $\mathbf{R}_t$. As already mentioned, $\mathbf{Z}_t^m$ describes the local estimates $\widehat{\mathbf{x}}_t^{1,m}, \ldots, \widehat{\mathbf{x}}_t^{C,m}$ from every smart camera.

The measurement update matrix $\mathbf{C}_t$ and the measurement noise matrix $\mathbf{R}_t$ are described as follows ([Thrun 05]):

$$\mathbf{C}_t = \begin{bmatrix} \mathbf{C}_t^{1,m} \\ \vdots \\ \mathbf{C}_t^{C,m} \end{bmatrix}, \mathbf{R}_t = \begin{bmatrix} \mathbf{P}_t^{1,m} & \cdots & 0_{6\times6} \\ \vdots & \ddots & \vdots \\ 0_{6\times6} & \cdots & \mathbf{P}_t^{C,m} \end{bmatrix}. \tag{6.17}$$

Here, the measurement update matrix $\mathbf{C}_t^{C,m}$ for each person m and a specific camera C are given by an identity matrix. The covariance matrices $\mathbf{P}_t^{C,m}$ are the covariances of the local estimates from the smart cameras.



(a) 1 Camera (b) 2 Cameras

(c) 4 Cameras (d) 6 Cameras

Figure 6.16: *Fusion example for different numbers of cameras over time.* The fusion result at the fusion centre for different numbers of cameras is shown (in red), together with the ground truth (in black). As explained in Figure 6.15, a reliability measure given by the covariance matrix $\mathbf{P}_t^{C,m}$ is shown as an ellipse around the fusion result. As we see in (a), the fusion centre is rather unsure about the position of the person if only one camera is used for fusion. The accuracy and precision increase with the number of cameras.

The equation above states that all local estimates for each person m from every camera are taken into account. Note that it is possible that a camera does not have any information about a specific person due to the fact that the person is not seen by this camera or is occluded by other persons or furniture. In this case, the measurement update matrix $\mathbf{C}_t^{C,m}$ of this camera C is a zero matrix, and this local estimate is not taken into account for the joint decision. Finally, the global estimates $\mathbf{x}_t^m$ of each person m are fed back to every camera to correct possible mistakes. This feedback is essential since the tracking of each individual camera can be inaccurate, e.g., in situations where a camera cannot contribute or gather any information.

For example, if a person is completely occluded for one camera, this camera does not contribute any information about this person, but also cannot estimate any further local state of this person. In this particular case, the only way for

this camera to keep track of the person is by using the feedback of the fusion centre, which is based on the input of other cameras. This is why the feedback is very important for the overall performance of the system.

6.7 Results

In order to evaluate our approach, we conducted several experiments using our collected video data for two different scenarios: people tracking in an indoor space without furniture, and people tracking in a meeting room (furnished with tables and chairs). These scenarios fall into the domain of surveillance and behaviour analysis during meetings (smart meetings). We evaluated the overall performance of our proposed multi-camera tracking system in the following aspects:

1. Accuracy (distance from the manually annotated positions of objects (people) on the real-world ground plane),
2. Precision (the number of object losses and switches between objects);
3. Execution speed (faster or slower than real-time);
4. Scalability (influence of adding more cameras to the system).

Furthermore, we provide qualitative and quantitative results and analyse two different camera-based tracking approaches: *histogram filtering (HF)* and the approach with *no local state estimation (NLE)*.

In the *indoor* scenario, people were observed while walking in a room without furniture. In the *meeting* scenario the room was equipped with furniture (tables and chairs) and people were observed while having a meeting: entering the room, shaking hands with each other, walking around the table to find a place to sit, sitting, moving chairs to sit at another position, standing (to give a presentation), and leaving the room. In total, we have collected more than 120 minutes of data. All videos were recorded using a six-camera setup, consisting of four side-view and two top-view cameras, operating on a frame rate of 20 fps. The cameras were mounted at ceiling height ($3m$ approximately), and extrinsically calibrated and synchronized up to frame accuracy. Furthermore, the width and height of a person was assumed to be on average 40 and 180 cm, respectively.

Our proposed framework was implemented in C++, in a client-server fashion. In the experiments we performed, each camera was connected to a PC (a client), with a single-core 2.8 GHz processor to simulate a "smart camera". All smart cameras were connected to another single PC, with a single-core 2.8 GHz processor which was functioning as the fusion centre.

The purpose of the experiments was to test several important attributes of the framework: the *calibration accuracy*, the *real-time performance* and *scalability* and the *overall performance of the proposed system* in terms of accuracy

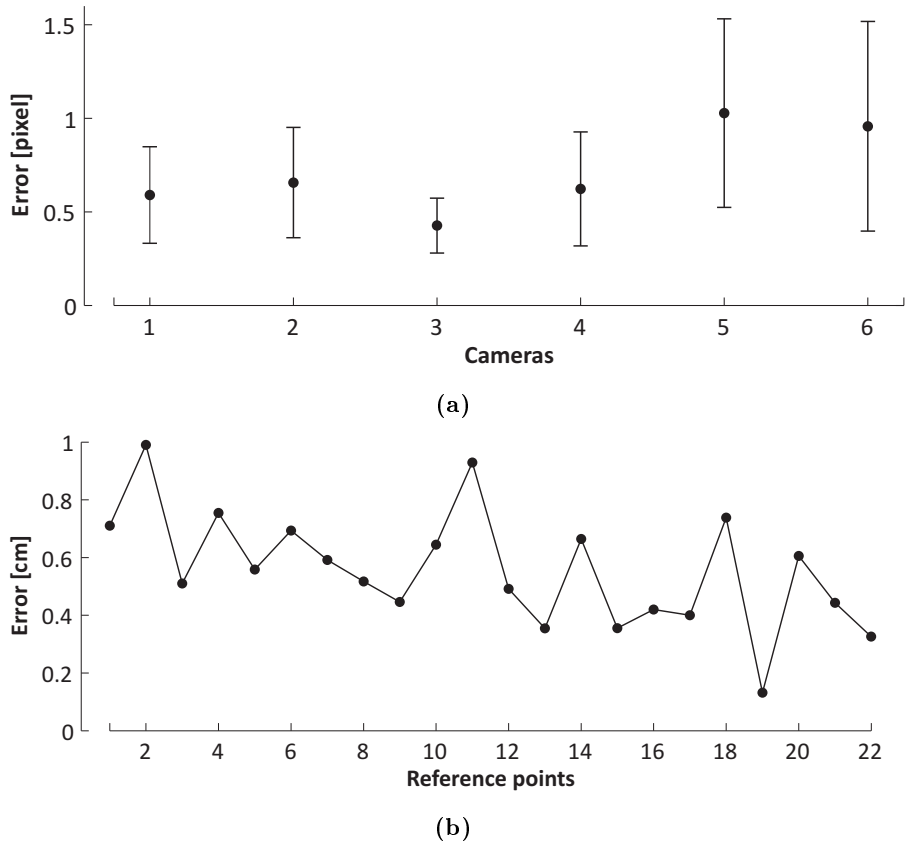


Figure 6.17: *Calibration accuracy.* In (a), the intrinsic calibration error of each camera is very small and therefore the calibration of an individual camera very accurate. To evaluate the overall accuracy of the calibration, we compared estimated 3D coordinates to manually measured reference points. In (b), the error to each reference point is shown. The average accuracy is 0.56 cm.

and precision. Furthermore, we compared our results with the *state-of-the-art methods* of [Berclaz 11] and [Grünwedel 14]. In this section, we present experimental results for each of these attributes separately.

6.7.1 Calibration accuracy

We calibrated the cameras using the calibration method of [Bouguet 99] for the side view cameras, and the method of [Kannala 06] for the top view cameras. We used a checkerboard pattern for intrinsic calibration and manually measured reference points in the scene for extrinsic calibration.

As shown in Figure 6.17a, the intrinsic calibration we obtained for each camera is very accurate, with an average error per camera below one pixel. To

measure the overall accuracy of all cameras, we used the obtained calibration results to compute the 3D coordinate of each reference point. Furthermore, we compared the results to our manually-measured reference points and obtained an overall accuracy of 0.56 cm (Figure 6.17b), which is the mean of the distance between the obtained and measured reference points.

The results show that our calibration procedure is precise and that 3D coordinates of objects in the scene can be obtained from 2D image points with sufficient accuracy. This is of vital importance for the accuracy of our multi-camera tracking approach. Therefore, taking into account the calibration accuracy of our multi-camera setup, our proposed tracking approach is also highly accurate, which we will outline in the following sections.

6.7.2 Data sets

All our experiments were conducted in a room of the size of 8.8×9.2 m, equipped with a network of smart calibrated cameras (780×580 pixels at 20 FPS) with overlapping views. The data sets we collected in this environment for evaluation purposes contain people walking around in the room or having meetings.

Recordings were taken for several minutes each, and ground truth positions of each person were manually annotated in the image plane at one second intervals. This manual annotation process was conducted by finding the feet positions of individual (usually on the ground plane) in each camera image. Using the camera calibration parameters, we then calculated the ground truth positions of each individual on the ground plane in real world coordinates (x_{w} and y_{w} coordinates, $z_{\mathrm{w}} = 0$).

The recorded sequences describe different aspects of a tracking system. Sequence 01 shows with one single person for one minute. In sequences 02 and 03, three and four people, respectively, are walking around in the room for several minutes. Sequences 04 to 10 contain meetings up to four participants for about twenty minutes. In sequences 11 to 13 up to three people are walking around for about 10 minutes to test the reliability of our proposed tracking system. Finally, sequences 14 to 21 were conducted under changing lighting conditions with up to four people walking around in the room equipped with furniture, such as tables and chairs. Each of them is about five minutes long.

6.7.3 Real-time performance and scalability

We tested two important aspects: the real-time performance of the whole system, which is limited by the tracking time on the camera side, and the scalability, limited by the tracking time on the fusion centre. We conducted all experiments with the HF and NLE approaches at the camera side (these two approaches are explained in detail in Section 6.5).

To test the real-time performance we measured the tracking execution time per frame on each camera for a different number of observed people (one to four people). To obtain an average execution time for one camera, we averaged

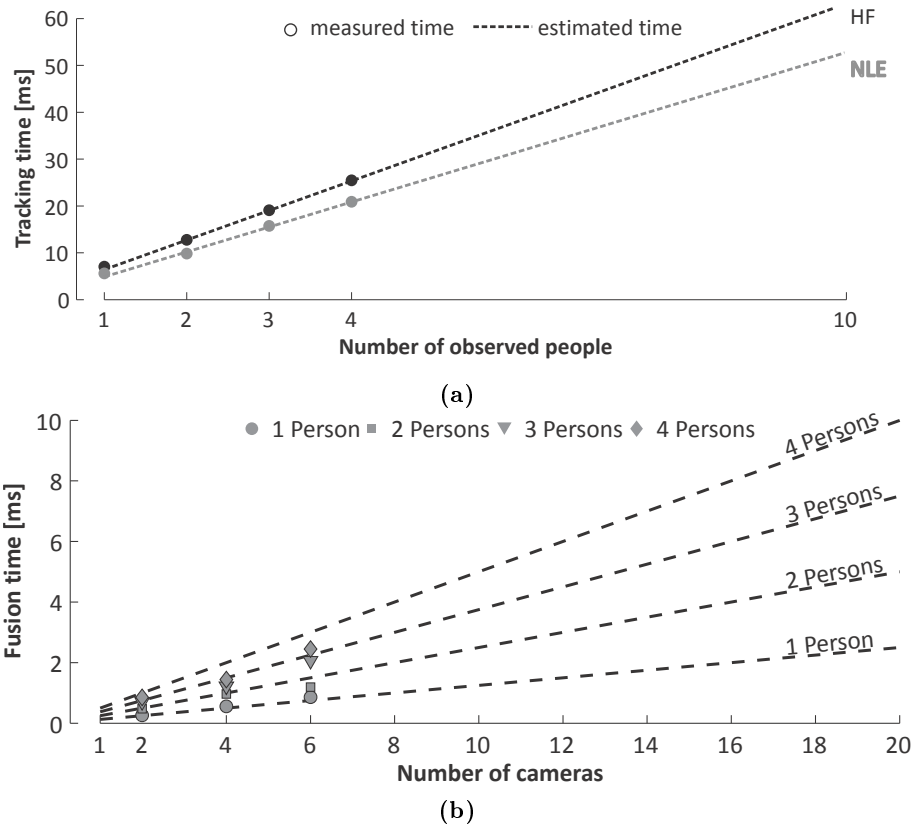


Figure 6.18: *Real-time performance and scalability.* (a) The dotted line shows the measured data and the dashed line the estimated timing depending on the number of people (up to 10 people) and the used approach (HF or NLE). Due to the nature of the approach at the camera side, the computation time increases linearly with the number of people. (b) The measured data with the dashed lines which show the estimated timing for up to 20 cameras and up to four persons. The estimated time is calculated assuming that a person is always seen by all cameras, which is not the case in practice. This is why there is a difference between the measured points and the estimated lines.

execution times over all test sequences. The results are shown in Figure 6.18a. We see that tracking time on the camera is less than 25 ms per frame, which is faster than real time (50 ms - 20 fps with a resolution of 780×580 pixels). As expected, the camera-based tracking time depends linearly on the number of viewed people due to the filtering approach on the camera side, where a separate filter is assigned to each person visible in the camera view. This means that an increase in the number of viewed people increases just the number of used filters, like the global Kalman filters in the fusion centre. Accordingly, the lines in Figure 6.18a represent the camera tracking time of each approach, estimated

for more than four people. We see that up to approximately 10 people can be tracked on a camera side at 20 fps. Such a performance is suitable for most applications since only in crowded environments one camera will have more than 10 people in its field of view. Note also that in our experiments we used relatively low processing power in the cameras (approximately a single-core 2.8 GHz processor), so by upgrading the processors the proposed tracking methods can easily run in real-time even in very crowded environments.

We also compared the averaged execution times of the two camera-based approaches (HF and NLE). As expected, the NLE approach is faster since there is no local filtering (tracking on the camera side). However, the accuracy of this approach strongly depends on the feedback frequency of the fusion centre, which is its significant drawback in the case of lower feedback frequency.

To test the scalability of the system (possibility of adding more cameras without affecting the real-time performance), we varied both the number of tracked people and the number of cameras connected to the fusion centre. The obtained results are shown in Figure 6.18b. Since each person is usually not visible in all cameras connected to the fusion centre, the fusion time is shorter than the one given by a linear function, because some cameras do not contribute with a measurement for each person. Therefore, the lines in the graph of Figure 6.18b represent the maximal fusion time as a function of the number of cameras connected to the fusion centre and the number of tracked people.

We see that it is possible to fuse information from many cameras in real-time, i.e. at 20 fps (e.g. from 20 cameras for 10 tracked people). Also, the fusion time and the number of cameras in which the person is observed correlate in a linear fashion. Such an efficiency enables highly scalable tracking systems that could deploy sufficient number of cameras for any area and any amount of people that need to be observed. Note that, like in the case of on-camera tracking, the estimated scalability could be further increased by using multi-core processing units or by optimizing the implementation for hardware accelerated processing (e.g. GPU processing).

6.7.4 Performance of the proposed system architecture

We express the performance of our proposed tracker in two ways: as *precision*, i.e. the total number of object losses (NoOL) and the total number of object switches (NoOS), and as *accuracy*, i.e. the Euclidean distance between the ground truth positions of people and the positions estimated by the tracker (the total average tracking error (TATE)). Note that in the case of a loss or a switch of the object, we manually correct its position and start tracking from the ground truth position in the following frame.

To calculate the number of object losses, we consider that people are lost by the tracker if the Euclidean distance between their estimated position and the ground truth position is bigger than 80 cm (twice the assumed width of a person). In contrast to an object loss, an object switch occurs when two objects

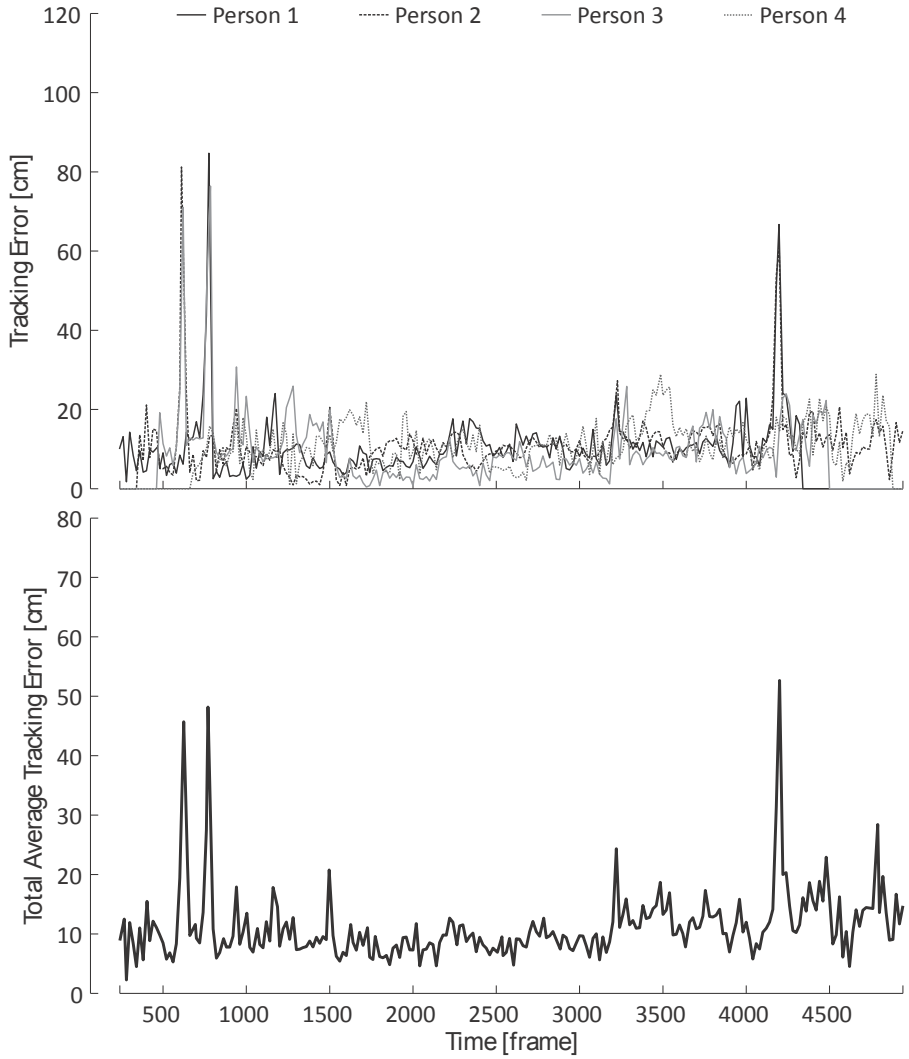


Figure 6.19: *Example sequence of a meeting scenario.* At the top, the tracking error for each individual is reported as the distance to the ground truth. At the bottom, the total average tracking error (TATE) on a frame-by-frame basis for a sequence with up to four people is shown. Note that there are no object losses. However, there are three object switches (the peaks) in this example sequence. The total average tracking error (TATE) over the whole sequence is about 7.3 cm.

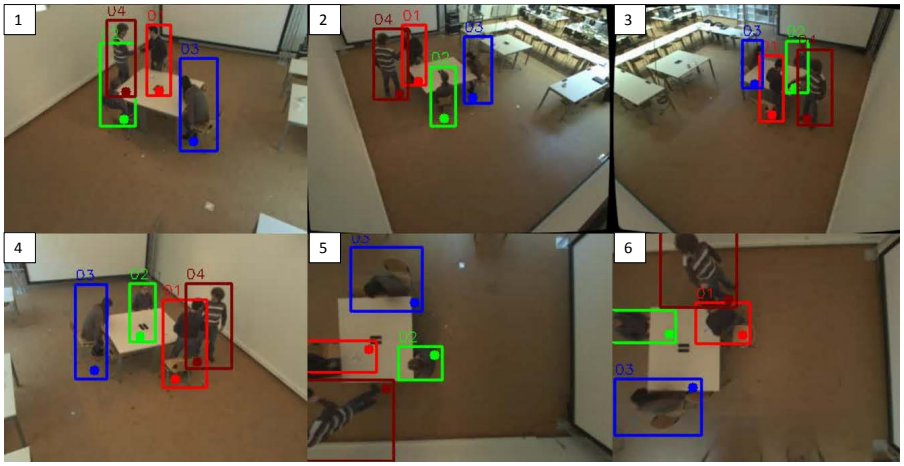


Figure 6.20: *Example frame of a meeting scenario.* This is an example frame of a meeting sequence (the same as in Figure 6.19) from all six cameras. As we see, our proposed multi-camera tracking is highly accurate for each individual.

Table 6.1: *Performance of the on-camera histogram filtering tracking (HF).*

Sequence	Accuracy	Precision		
	TATE [cm]	NoOL	NoOS	OL/min
01	8.1	0	0	0.0
02	6.3	0	0	0.0
03	7.7	0	0	0.0
04	9.8	0	0	0.0
05	10.1	0	0	0.0
06	9.4	0	0	0.0
07	11.3	2	0	0.2
08	10.2	1	1	0.1
09	9.9	3	1	0.1
10	10.8	0	0	0.0
11	9.1	0	0	0.0
12	8.7	0	0	0.0
13	10.9	0	0	0.0

are switching their positions with respect to the ground truth positions. The switches can happen if two individuals are in close proximity.

We conducted several experiments under different circumstances: several indoor scenarios of people walking in a room without furniture, and in a room equipped with furniture, in multiple meeting scenarios. Both cases include changing environmental conditions (lighting changes). In total, we have collected more than 120 minutes of data.

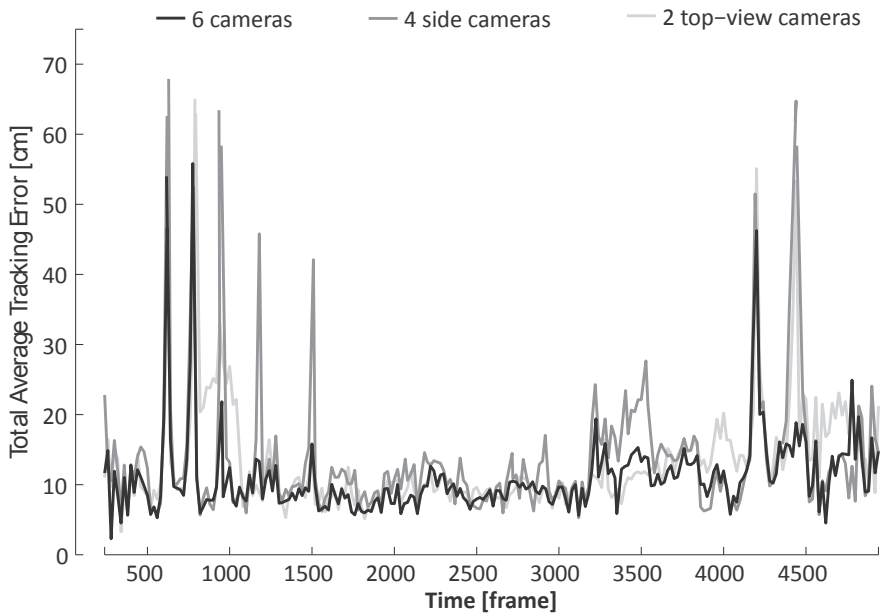


Figure 6.21: *Performance for different numbers of cameras.* The comparison was done between four side cameras (gray line), two top-view cameras (dashed line) and the complete setup, which includes all six views (black line). The best results are achieved by using top and side cameras together.

6.7.4.1 Overall performance of the proposed multi-camera system

The average accuracy and precision for our proposed smart multi-camera tracker over the 120 minutes of data are 9.8 cm total average tracking error and 0.1 object losses per minute. The results are very promising and show the robustness of our multi-camera system. The evaluation also includes some very difficult cases, i.e. we tested our tracking system under severe occlusions and lighting changes. In Section 6.7.4.3 we give a detailed evaluation of our two on-camera tracking approaches (see Table 6.1 and 6.2). Additionally, Table 6.4 and 6.5 show the performance under illumination changes.

An example of our tracking results is shown in Figures 6.19 and 6.20. We see that the tracking results are very accurate (TATE of 9.8 cm).

6.7.4.2 Performance for different number of cameras

In this section we show the influence of the number of cameras and their viewpoints on the performance of our proposed multi-camera system. For this experiment we used an indoor meeting sequence 06 from our data set (see Section 6.7.2) in which four people are walking around, shaking hands or giving a presentation. We conducted the experiment using the HF approach for camera-based tracking. Figure 6.21 shows the results for different number of

cameras and different viewpoints. We compare four side cameras, two top-view cameras, and the complete setup, consisting of all six cameras.

As expected, the results show that by using only side-view cameras it is more difficult to accurately locate people's ground position than when using top views. This is mainly due to occlusions in side views. Nevertheless, the highest accuracy has been achieved when both side views and top views were used.

Table 6.2: *Performance of the on-camera NLE tracking.*

Sequence	Accuracy	Precision		
	TATE [cm]	NoOL	NoOS	OL/min
01	11.8	0	0	0.0
02	10.2	0	0	0.0
03	10.7	0	0	0.0
04	9.5	0	0	0.0
05	13.8	1	1	0.4
06	10.1	0	0	0.0
07	12.7	4	3	0.5
08	10.8	3	1	0.2
09	13.2	5	0	0.2
10	11.6	3	0	0.2
11	12.3	0	0	0.0
12	11.1	0	0	0.0
13	10.3	0	0	0.0

Table 6.3: *Comparison between different feedback frequencies.*

	HF			NLE		
Freq. (Hz)	TATE (cm)	NoOL	NoOS	TATE (cm)	NoOL	NoOS
20	8.4	0	1	9.8	0	0
10	9.3	0	1	11.0	1	1
5	10.1	0	1	13.2	4	2
2	10.4	0	2	15.3	9	4
1	11.0	1	2	18.7	19	4

Table 6.4: *Performance of the HF approach for challenging test cases (with occlusions and lighting changes).*

Sequence	Accuracy	Precision		
	TATE [cm]	NoOL	NoOS	OL/min
14	10.1	0	0	0.0
15	12.0	1	0	0.8
16	10.7	1	0	0.8
17	9.8	0	0	0.0
18	10.1	0	0	0.0
19	13.1	8	2	1.4
20	10.6	0	1	0.0
21	9.9	0	0	0.0

Table 6.5: *Performance of the NLE approach for challenging test cases (with occlusions and lighting changes).*

Sequence	Accuracy	Precision		
	TATE [cm]	NoOL	NoOS	OL/min
14	10.9	0	0	0.0
15	12.5	3	0	2.4
16	14.1	4	0	3.2
17	9.7	0	0	0.0
18	11.1	0	0	0.0
19	15.2	17	1	3.1
20	13.8	6	0	1.1
21	11.9	0	0	0.0

Table 6.6: *Comparison of our method with the method of [Grünwedel 14].*

	[Grünwedel 14]			Ours		
Freq. (Hz)	TATE (cm)	NoOL	NoOS	TATE (cm)	NoOL	NoOS
20	11.1	0	6	8.4	0	1
10	11.7	0	6	9.3	0	1
5	11.9	2	4	10.1	0	1
2	12.5	1	10	10.4	0	2
1	14.9	4	12	11.0	1	2

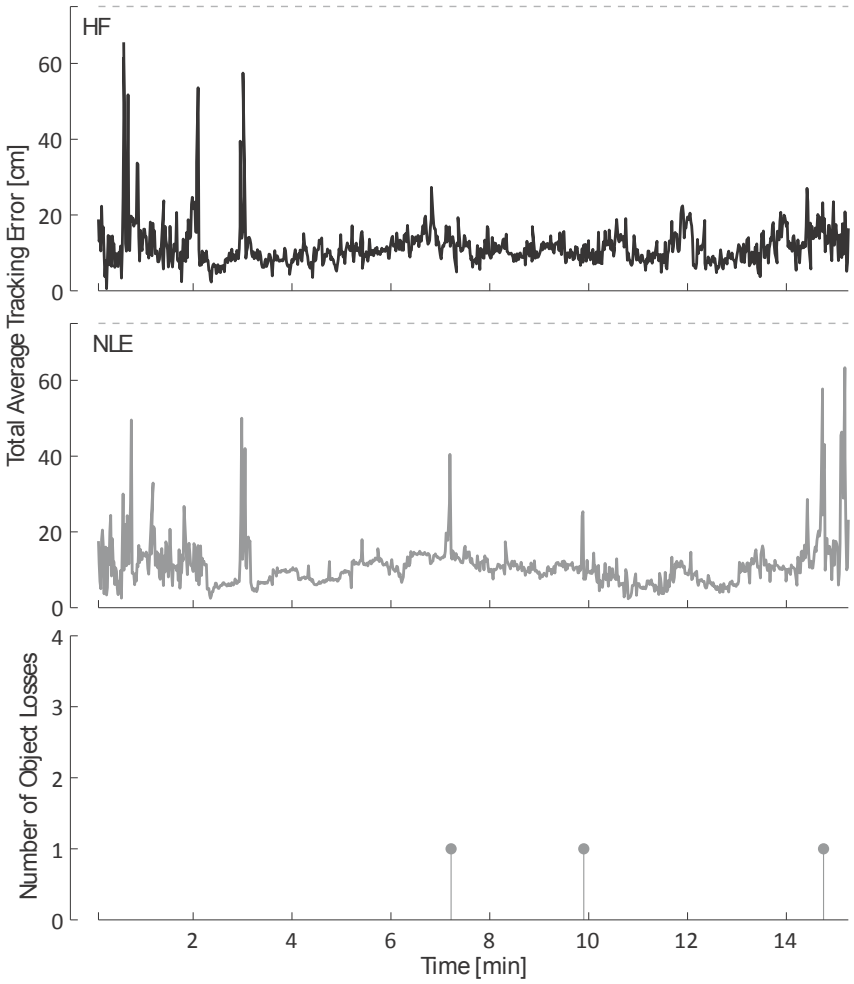


Figure 6.22: *Comparison of the HF and NLE camera-based approaches.* This comparison shows the performance of the camera-based tracking approaches (HF and NLE). We see that the accuracy is nearly the same for these approaches. We see that there are more object losses in the NLE approach. This is caused by the fact that this approach depends on the feedback frequency.

6.7.4.3 Comparison of the on-camera trackers

In this section we give a comparison between the two proposed camera-based tracking approaches: the histogram filtering (HF) (explained in Section 6.5.4), and tracking without the local state estimation (NLE) (explained in Section 6.5.5). For this comparison we used several different sequences and measured the performance of both methods (see Tables 6.1 and 6.2). All sequences were processed with a ground plane resolution of 10 cm.

Figure 6.22 shows the results for a meeting scenario with four attendees, and Tables 6.4 and 6.5 for challenging sequences with occlusions (furniture) and global and local lighting changes. We see that the accuracy of the trackers is comparable. The main difference is the number of object losses, which is higher for the NLE tracker. This is caused by the fact that the HF approach does not depend on the feedback frequency of the fusion centre and operates on the camera frame rate. In general, it is an advantage to keep a local estimate of each person on the camera side.

6.7.4.4 Influence of feedback and feedback frequency

The feedback loop is an essential part in the proposed system architecture. To demonstrate the use of feedback from the fusion centre, we tested the influence of feedback by comparing the performance of the system with and without feedback (see Figure 6.23). For this experiment we used an indoor scenario with up to four people walking around (Sequence 03).

In Figure 6.23, the results demonstrate clearly that the use of a feedback loop between the fusion centre and the smart cameras is beneficial. There is a big difference in accuracy and precision. This is mainly because a single smart camera often cannot recover from its own wrong estimates and often even cannot know that it made wrong estimates. In these cases the camera keeps sending wrong information to the fusion centre, and if these errors propagate to the other cameras, the whole system can fail and have difficulties to recover from such mistakes. Therefore, the feedback from the fusion centre is of vital importance for a robust multi-camera system as a prevention of the error propagation between the cameras. A possible improvement could be that the fusion centre detects mistakes made by the camera and only sends feedback to the cameras which are starting to fail. This could further improve the communication load and therefore be more energy efficient.

Table 6.3 shows the influence of different feedback frequencies on the on-camera trackers (HF and NLE). The results show that even with a feedback frequency of 5 times per second (a feedback every 200 ms) the HF tracking has good results. As expected, HF tracking is less sensitive to low feedback frequency than the NLE method.

6.7.5 Comparison with state-of-the-art methods

We compared our proposed tracker with the state-of-the-art multi-camera tracking approaches of [Berclaz 11] and [Grünwedel 14]. The latter is the method presented in the PhD thesis of my colleague Sebastian Grünwedel. In the method presented in this thesis, on the smart camera side we added the signature based appearance tracking and context aware motion modelling. In Table 6.6 we see the result of the comparison using a representative video sequence 03 and different feedback frequencies. The video sequence 03 captures four people moving in a room without furniture. We see that our method is

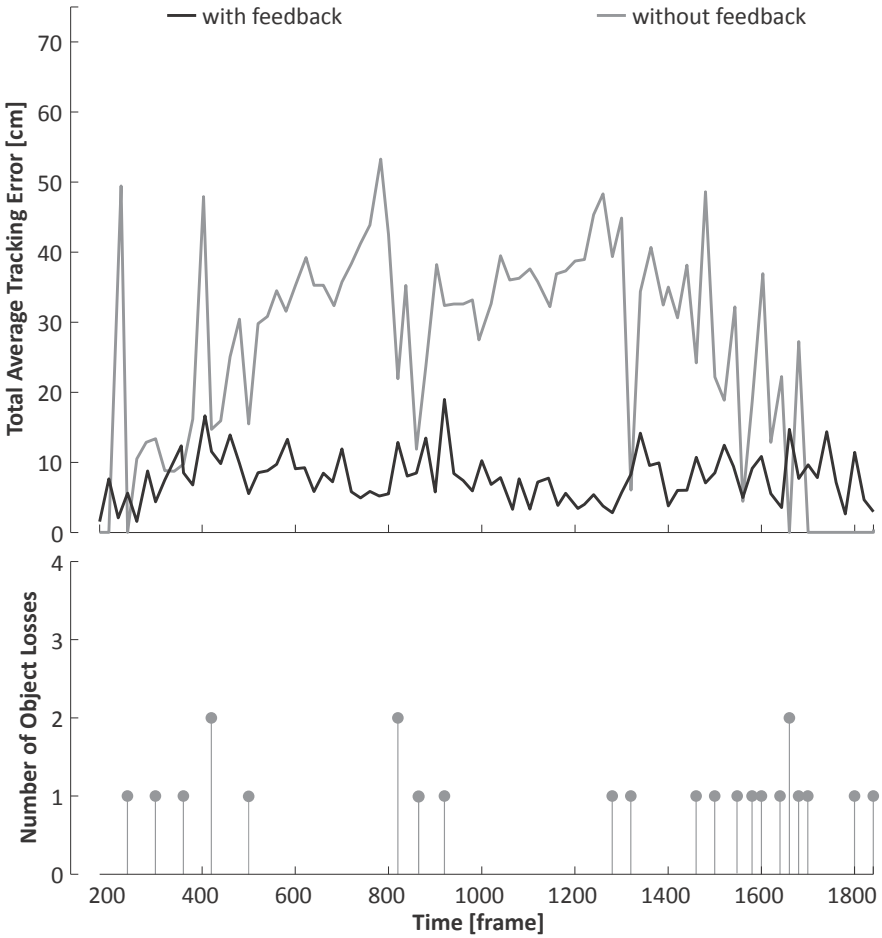


Figure 6.23: *Influence of feedback from the fusion centre.* To demonstrate the use of feedback from the fusion centre we explored the case with and without feedback. The results clearly show that accuracy and precision are much higher when feedback is used in the system.

more accurate and has fewer object losses and switches. It is also less sensitive to the feedback frequency. This is mainly because we improved on-camera tracking by introducing additional cues next to the cues used by [Grünwedel 14]. In fact, since the video sequence used in this experiment is captured in a room without furniture, our context-aware motion modelling has a negligible impact on the performance. Therefore, this experiment demonstrates mainly the benefit of adding signature based cues for tracking.

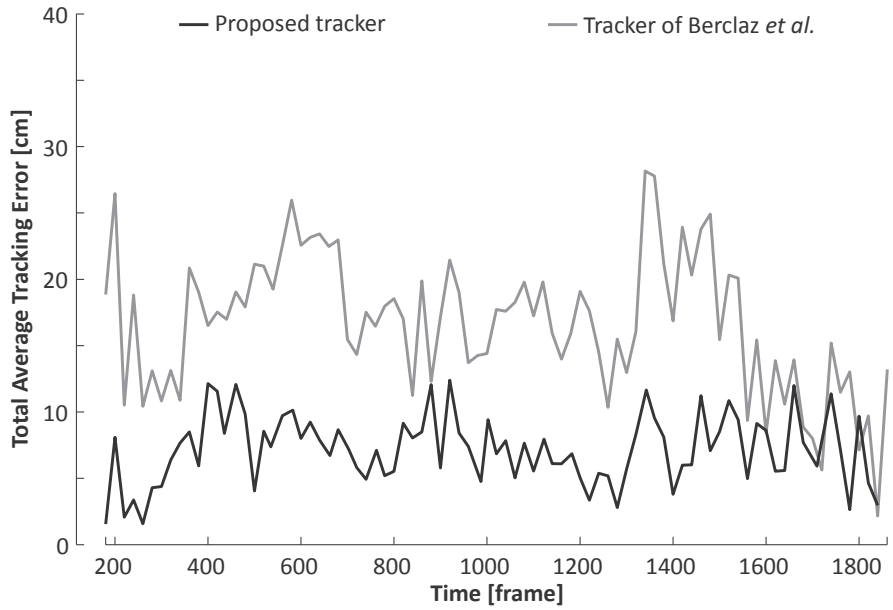


Figure 6.24: *Comparison on an indoor sequence.* We compare our proposed tracker with the tracker of [Berclaz 11]. The total average tracking error (TATE) is smaller for our tracker than for [Berclaz 11] (our proposed tracker: 6.9 cm; tracker of Berclaz *et al.*: 16.6 cm). We believe this is because we improved on-camera tracking using additional cues. There were no object losses in this sequence by neither of the trackers.

The comparison with the method of [Berclaz 11] we made using two of our sequences and publicly available sequences. an indoor sequence (without furniture) and a meeting sequence with up to four people. The tracker of Berclaz *et al.* was configured with a grid cell size of 10 by 10 cm, as well as ours. We tested different grid cell sizes (10, 20 and 30 cm), whereby 10 by 10 cm achieved the best results. Our multi-camera tracker was configured to use the HF tracking on the camera side, with a ground plane resolution of 10 cm. Furthermore, the width and height of a person was assumed to be on average 40 and 180 cm, respectively. In the following two sections we present in more detail the comparison with the method of [Berclaz 11].

6.7.5.1 Comparison on our data sets

Figure 6.24 shows the performance of our and [Berclaz 11] method on an indoor sequence with up to four people. We see that the performance of both trackers is comparable, although our tracker is more accurate. With our proposed tracker we achieve the accuracy of 6.9 cm compared to 16.6 cm of Berclaz *et al.* There were no object losses in this sequence for neither of the trackers.

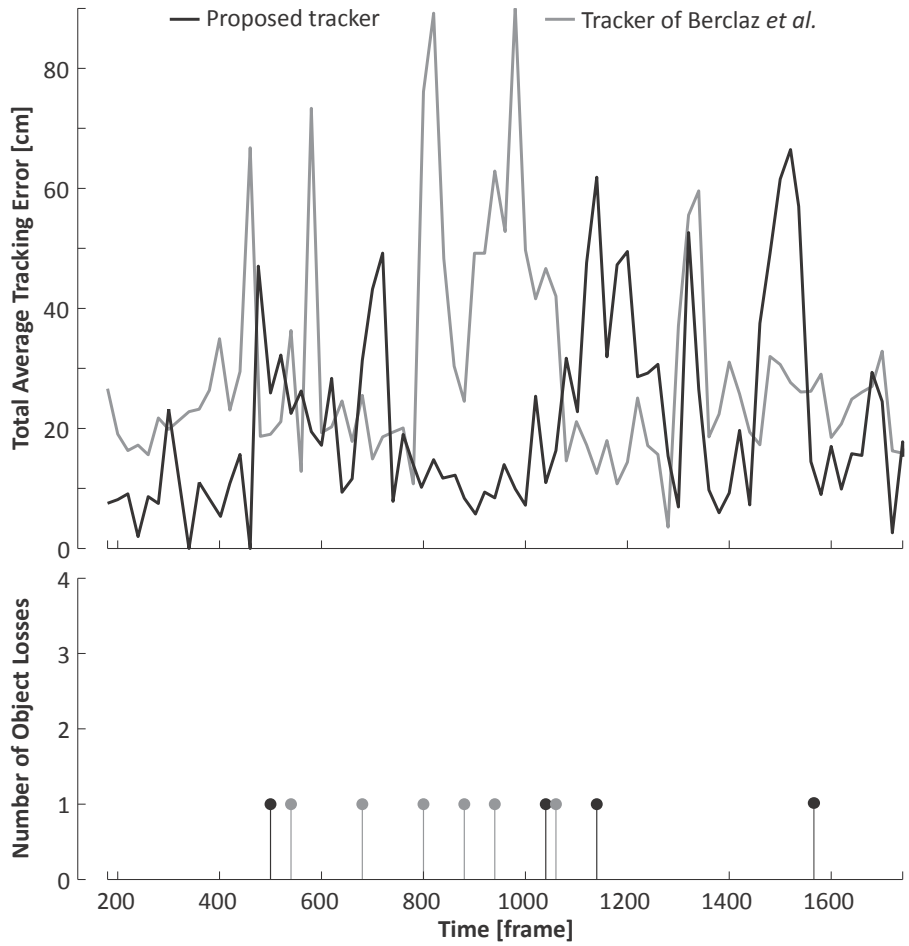


Figure 6.25: *Performance on an outdoor data set [Berclaz 11].* For a comparison using a public data set, we chose an outdoor data set provided by the lab of CVLAB at EPFL [Berclaz 11]. Note that this data set is acquired using only three cameras with small overlapping area. The accuracy of both methods is very good, but they differ from some object losses. We see that our method has fewer losses, which is mainly because it includes people's appearance as a tracking cue.

In the second experiment we compared the methods using a meeting sequence. Our tracking results for this meeting sequence are shown in Figure 6.22. The tracker of Berclaz *et al.* performed poorly on this sequence.This is mainly because when people are seated FG/BG blobs do not capture well their silhouettes and the tracker of [Berclaz 11] is not suited for these circumstances.

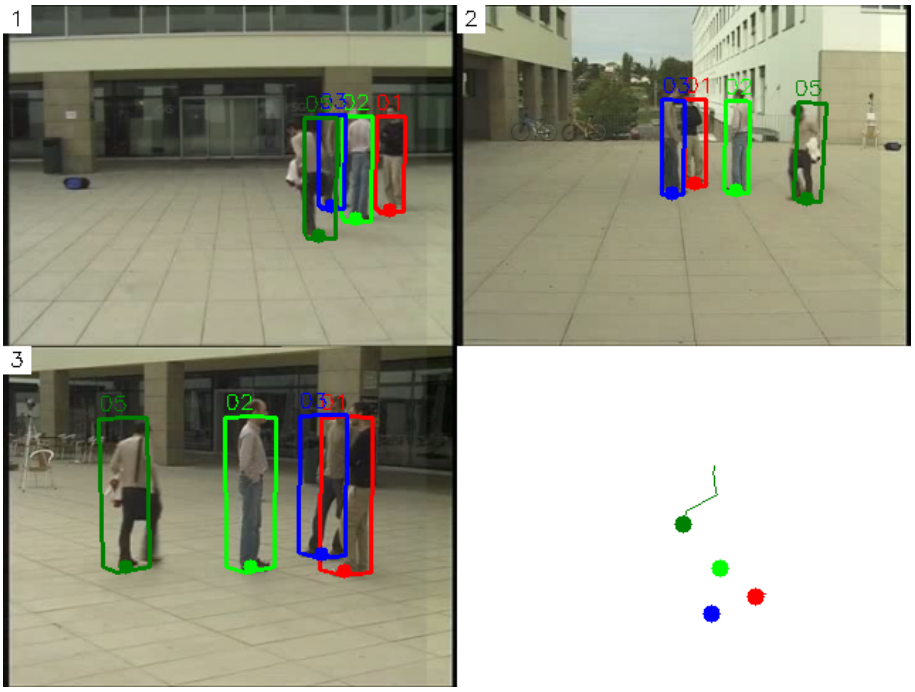


Figure 6.26: *Performance on an outdoor data set [Berclaz 11].* The sample frame from the public data set, provided by the lab of CVLAB at EPFL [Berclaz 11], is shown. The overall accuracy of our tracker for this sequence is 21.6 cm, which is acceptable considering the use of only three cameras.

6.7.5.2 Comparison on public data sets

To compare our proposed tracker with a public data set, we chose a data set provided by the CVLAB[2] at EPFL [Berclaz 11]. They used three cameras with a small overlapping area in an outdoor scenario. Up to five people can be seen in this sequence. We made ground truth at one-second intervals for one of the sequences we took for comparison. The results are shown in Figure 6.25. The overall accuracy of our tracker for this sequence is 21.6 cm, which is acceptable considering the use of only three cameras (Figure 6.26). The tracker of [Berclaz 11] shows an overall accuracy of 25.4 cm on this sequence, which is a comparable result to our tracker. However, the tracker of [Berclaz 11] losses tracked objects more often.

In summary, our tracking approach and approach of [Berclaz 11] have comparable, very good performance in terms of accuracy and precision. Both trackers have similar accuracy and comparable number of object losses, although our method is better in both of these parameters. Moreover, the implementation of [Berclaz 11] tracker optimizes trajectories over the whole sequence

[2]Data available at `http://cvlab.epfl.ch/data/pom`

to deal with object losses, while in our approach we prevent object losses by using robust multi-view appearance models and context-aware motion models. Our method is designed to work online, in real-time, while the method of [Berclaz 11] has delayed reasoning to optimize the trajectory assignment (in their implementation they use a five seconds delay). Therefore, in time critical applications, such as video conferencing, our method would outperform the method of [Berclaz 11].

6.7.6 Real-time demonstrator

As one of the results of this research, we have implemented a real-time smart multi-camera system using a camera network as explained in this chapter. The system was installed at Hogeschool Gent to track multiple individuals in real time. The network consists of six colour cameras (four side and two top-view cameras) with a resolution of 780×580 pixels, each connected to an Intel Core 2 Duo 2.8GHz processor. Each colour camera and the attached computer simulate a smart camera. A fusion centre with the same processor completes the system architecture. The cameras observe a scene of approx. 9×5 m. Each smart camera performs foreground/background segmentation and appearance modelling, as explained in Sections 6.5.1 and 6.5.2. Each camera also calculates a six dimensional state estimate for each individual based on the HF approach (Section 6.5.4). Therefore, instead of camera images, only a compact representation of each individual, represented as a Gaussian distribution, is sent to the fusion centre. The fusion centre calculates a final estimate based on the input of the smart cameras, using a Bayesian estimator (as explained in Section 6.6). After a final estimate of each individual is calculated, the fusion centre sends the results back to each smart camera to correct potential mistakes. The implemented demonstrator operates at 10 fps. It is a very good basis for further research on real-world multi-camera systems.

6.8 Conclusions

In this chapter, we presented a novel decentralized multi-camera system architecture for real-time tracking. The tracking is performed by distributing tasks between cameras and a fusion centre to obtain computational and data efficiency important for smart camera networks. We showed the advantages of distributing video processing and tracking tasks on each camera, and sending only high-level compact information to the fusion centre instead of the whole images. We also demonstrated the benefit of fusing the information from all cameras and communicating the fusion result back to each camera to correct their own estimations.

We compared our approach with the state-of-the-art methods of [Berclaz 11] and [Grünwedel 14], and demonstrated the improvements both in accuracy and precision. These improvements come from using our proposed signature measurement as an additional tracking cue, and improving the motion model of

tracked objects by higher level contextual reasoning. The results were obtained using multiple video sequences in two indoor scenarios, with and without furniture. The sequences contain cases of severe occlusions and lighting changes. Experimental results show high accuracy and precision, sufficient for many applications, such as surveillance or behaviour analysis of people in meetings, even in cases of occlusions.

There are several possible extensions to this work. One possibility is to incorporate more advanced methods to model the appearance of a person. Another extension could be a more detailed study of the fusion methods and the feedback loop. Adding the input from other sensors in the meeting rooms, such as laptop cameras and microphones, would also be beneficial for tracking improvement.

This research resulted in one publication in the international journal *ACM Transactions on Sensor Networks* [Grünwedel 14]. Furthermore, several papers have been published in the proceedings of international conferences [Grünwedel 12], [Jelača 11a], [Demeulemeester 11], [Xie 12].

7

Conclusions

Significant progress has been made in object tracking during the last few years. Several robust trackers have been developed which can track objects in real time in simple scenarios. However, the assumptions used to make the tracking problem tractable, for example, smoothness of motion, minimal amount of occlusion, illumination constancy, high contrast with respect to background, etc., are violated in many realistic scenarios and therefore limit tracker's usefulness in applications like automated surveillance, human computer interaction, video retrieval, traffic monitoring, and vehicle navigation. Thus, tracking and associated problems of feature selection, object representation, dynamic shape and motion estimation are very active areas of research and new solutions are continuously being proposed.

In this thesis we researched and developed object tracking techniques specifically designed for smart multi-camera networks. We proposed and developed algorithms for single and multi-camera tracking, focusing on robust real-time, low-latency and scalable tracking of vehicles and people, in which the most computationally intensive video processing is performed within smart cameras. We paid significant attention to performance in real-world conditions, where illumination changes, occlusions, inaccurate and false detections are common.

7.1 Overview of contributions

In this thesis we proposed a comprehensive framework for object tracking in smart camera networks. We addressed the tracking problems from low-level to high-level, i.e. from object detection and feature extraction to the high level contextual reasoning, information selection and fusion. We deeply integrated different tracking levels into a single framework.

7.1.1 Low-level tracking

At the low level we proposed using Radon-transform like image projection profiles, which we call signatures, as features for object detection, tracking and recognition. These features are computationally very efficient (calculated as

averaged pixel values along image axes). They can be calculated for all objects in a single reading of the image, and result in a compact image representation (an image of size 256×256 pixels is represented by 512 values instead of 65536).

We demonstrated the robustness of the signature features to illumination changes, viewpoint differences and occlusions. We showed also the advantages of signatures compared to edge based features, foreground blobs and optical flow. Therefore, using signatures of viewed objects as an additional cue for object tracking improves the tracking performance. The proposed signature cues are efficiently computed using dynamic time warping, so it is possible to include them into tracking algorithms preserving the real-time performance of the tracker. The presented preliminary results on vehicle detection indicate that signatures can also be used for object detection. This is especially beneficial for smart cameras where it is desirable to reuse same features for multiple tasks (e.g. object detection, tracking and recognition) to save processing power.

7.1.2 Mid-level tracking

A significant part of this dissertation focuses on tracking in decentralized/ distributed multi-camera networks with overlapping views. In this context, we researched and developed methods for tracking people in environments with frequent occlusions and lighting changes, for applications in surveillance, retail, video conferencing or similar.

In such applications, accurate tracking, real-time performance, scalability and flexibility of the system are important. Therefore, we focused on distributing tasks between cameras to obtain computational and data efficiency. We showed the advantages of distributing video processing and tracking tasks on each camera, and sending only high-level compact information between the cameras or to the central station (fusion centre). We demonstrated the benefit of communicating the fusion result back to each camera to correct their own estimations.

We compared our approach with the state-of-the-art methods and demonstrated the improvements both in accuracy and precision. These improvements come from using our proposed signature measurement as an additional tracking cue at the low level, and improving the motion model of tracked objects by higher level contextual reasoning. Experimental evaluation shows that such an approach has high accuracy and precision (average distance to the ground truth position is about 10 cm).

7.1.3 High-level tracking

At the high level we combine several low-level and mid-level cues, such as object detection, object recognition, object pose estimation, and others. We also incorporate contextual information into tracking. For environments like tunnels or meeting rooms, we model the scene structure, including scene entries and exits, occluders and relationships between the objects. We use kinematics of objects to constrain trajectory associations. All this high-level information

is then used to assign the long range trajectories and reduce trajectory fragmentation and possible identity switches.

7.1.4 Summary of contributions

To summarize, the main contributions of this thesis are:

- computationally and data efficient descriptors of the object appearance, based on 1-D Radon-transform like image projections (signatures) [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13];
- object appearance modelling based on the object's signatures, robust to pose and illumination changes, and occlusions [Jelača 11b], [Jelača 11a], [Jelača 12], [Jelača 13];
- computationally efficient appearance matching for object recognition using deformable curve alignment, dynamic time warping, and global and local 1D correlation [Jelača 13];
- distributed multi-view appearance modelling with automatic selection of informative observations for tracking [Jelača 13];
- robust real-time single camera tracking using image projection features in a Kalman filter framework [Jelača 12], [Jelača 14];
- a decentralized multi-camera framework for tracking with a feedback loop from the fusion centre [Jelača 11a], [Grünwedel 14], [Grünwedel 12], [Xie 12].

In total, the research from this PhD resulted in three publications in international peer-reviewed journals: two published [Jelača 13], [Grünwedel 14], and one article under review [Jelača 14]. Furthermore, thirteen papers have been published in the proceedings of international conferences [Jelača 08], [Despotović 10], [Jelača 11b], [Jelača 11a], [Grünwedel 11a], [Van Hese 11], [Demeulemeester 11], [Niño Castañeda 11], [Frías Velázquez 11], [Jelača 12], [Grünwedel 12], [Maćešić 12], [Xie 12].

7.2 Future research

A big challenge in tracking is to develop algorithms that perform well in unconstrained real-world environments. In this thesis we made an important step towards increased robustness against illumination changes, different weather conditions, camera viewpoints and resolutions, but further research is necessary to enable tracking in less structured environments or environments with non-stationary cameras.

In general, an important issue that has not been sufficiently explored is integration of contextual information into tracking algorithms. For example, in a vehicle tracking application, the location of vehicles relative to each other

(vehicle constellations) can be a valuable source of additional information for evaluating different hypotheses. In addition, advances in classifiers have made accurate detection of scene context possible, for example, man made structures, paths of movement, class of objects, etc. A tracker that takes advantage of contextual information to incorporate general constraints on the shape and motion of objects will usually perform better than one that does not exploit this information. This is because a tracker designed to give the best average performance in a variety of scenarios can be less accurate for a particular scene than a tracker that is tuned (by exploiting context) to the characteristics of that scene. Exploiting contextual information should be a very important direction in the future research.

Another important direction that we did not explore in this thesis is an automatic selection of tracking features. The use of a particular feature set for tracking can greatly affect the performance. Generally, the features that best discriminate between multiple objects and between the object and background are also best for tracking the object. Many tracking algorithms use a weighted combination of multiple features assuming that a combination of preselected features will be discriminative. A wide range of feature selection algorithms have been investigated in the machine learning and pattern recognition communities. However, most of these algorithms require offline training information about the target and/or the background. Such information is not always available. Moreover, as the object appearance or background varies, the discriminative features also vary. Thus, online selection and learning of discriminative features would greatly increase the applicability of our tracker.

The same applies for using additional sensors. For instance, in meeting scenarios audio sensors could provide valuable additional information to improve tracking accuracy and eliminate false object detections or associations. In traffic surveillance additional information could come from radar sensors. Overall, additional sources of information, in particular prior and contextual information, should be exploited whenever possible to adapt the tracker to the particular scenario in which it is used. A principled approach to integrate these disparate sources of information will hopefully result in a general tracker that can be employed with success in a variety of applications.

Bibliography

[Aghajan 09] H. Aghajan & A. Cavallaro. Multi-camera networks: principles and applications. Academic Press, 2009.

[Alcatel-Lucent 12] Alcatel-Lucent. *iCocoon: Immersive Video Conferencing System.* http://www.youtube.com/watch?v=TWUWJFVxvS8, 2012.

[Ali 01] A. Ali & J. Aggarwal. *Segmentation and recognition of continuous human activity.* In IEEE Workshop on Detection and Recognition of Events in Video, pages 28–35, 2001.

[Anjum 09] N. Anjum & A. Cavallaro. *Trajectory association and fusion across partially overlapping cameras.* In Proc. of the Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, pages 201–206, 2009.

[Avidan 01] S. Avidan. *Support vector tracking.* In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pages 184–191. IEEE, 2001.

[Babenko 11] B. Babenko, M.-H. Yang & S. Belongie. *Robust Object Tracking with Online Multiple Instance Learning.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 33, pages 1619–1632, 2011.

[Ballard 82] D. Ballard & C. Brown. Computer vision. Prentice-Hall, 1982.

[Bar-Shalom 87] Y. Bar-Shalom. Tracking and data association. Academic Press Professional, Inc., 1987.

[Barnich 09] O. Barnich & M. Van Droogenbroeck. *ViBe: a powerful random technique to estimate the background in video sequences.* In Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing, pages 945–948, 2009.

[Barnich 11] O. Barnich & M. Van Droogenbroeck. *ViBe: A universal background subtraction algorithm for video sequences.*

Image Processing, IEEE Transactions on, vol. 20, no. 6, pages 1709–1724, 2011.

[Barron 94] J. Barron, D. Fleet & S. Beauchemin. *Performance of optical flow techniques.* International Journal of Computer Vision, vol. 17, no. 12, pages 43–77, 1994.

[Bay 08] H. Bay, A. Ess, T. Tuytelaars & L. Van Gool. *Speeded-up robust features (SURF).* International Journal on Computer Vision and Image Understanding, vol. 110, no. 3, pages 346–359, 2008.

[Berclaz 06] J. Berclaz, F. Fleuret & P. Fua. *Robust People Tracking with Global Trajectory Optimization.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, volume 1, pages 744–750, 2006.

[Berclaz 11] J. Berclaz, F. Fleuret, E. Turetken & P. Fua. *Multiple Object Tracking using K-Shortest Paths Optimization.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 33, no. 9, pages 1806–1819, 2011.

[Betke 00] M. Betke, E. Haritaoglu & L. S. Davis. *Real-time multiple vehicle detection and tracking from a moving vehicle.* Machine Vision and Applications, vol. 12, pages 69–83, 2000.

[Birchfield 05] S. T. Birchfield & S. Rangarajan. *Spatiograms versus histograms for region-based tracking.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1158–1163, 2005.

[Bischof 04] H. Bischof, H. Wildenauer & A. Leonardis. *Illumination insensitive recognition using eigenspaces.* Computer Vision and Image Understanding, vol. 95, pages 86–104, 2004.

[Black 96] M. Black & P. Anandan. *The robust estimation of multiple motions: Parametric and piecewisesmooth flow fields.* Computer Vision and Image Understanding, vol. 63, no. 1, pages 75–104, 1996.

[Black 98] M. Black & A. Jepson. *Eigentracking: Robust matching and tracking of articulated objects using a view-based representation.* International Journal of Computer Vision, vol. 26, no. 1, pages 63–84, 1998.

[Blum 97] A.L. Blum & P. Langley. *Selection of relevant features and examples in machine learning.* Artificial Intelligence, vol. 97, no. 1–2, pages 245–271, 1997.

[Blum 98] A. Blum & T. Mitchell. *Combining labeled and unlabeled data with co-training.* In Proc. of the 11th Annual Conference on Computational Learning Theory, pages 92–100. ACM, 1998.

[Boser 92] B. Boser, I. Guyon & V. Vapnik. *A training algorithm for optimal margin classifiers.* In Proc. of the ACM Workshop on Conference on Computational Learning Theory, pages 142–152. ACM, 1992.

[Bouguet 99] J.-Y. Bouguet. *Visual methods for three-dimensional modeling.* PhD thesis, California Institute of Technology, Pasadena, CA, USA, 1999.

[Bowyer 01] K. Bowyer, C. Kranenburg & S. Dougherty. *Edge detector evaluation using empirical ROC curve.* Computer Vision and Image Understanding, vol. 10, pages 77–103, 2001.

[Cai 98] Q. Cai & J.K. Aggarwal. *Automatic tracking of human motion in indoor scenes across multiple synchronized video streams.* In Proc. of the Sixth IEEE European Conference on Computer Vision, pages 356–362, 1998.

[Cai 99] Q. Cai & J. Aggarwal. *Tracking human motion in structured environments using a distributed camera system.* Pattern Analysis and Machine Intelligence, IEEE Transaction On, vol. 2, no. 11, pages 1241–1247, 1999.

[Canny 86] J. Canny. *A computational approach to edge detection.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 8, no. 6, pages 679–698, 1986.

[Chang 01] T.-H. Chang & S. Gong. *Tracking Multiple People with a Multi-Camera System.* In Proc. of the IEEE Workshop on Multi-Object Tracking, pages 19–26, 2001.

[Choe 10] T. Choe, Lee M. & N. Hearing. *Traffic analysis with low frame rate camera networks.* In Proc. of the First IEEE Workshop on Camera Networks, 2010.

[Collins 01] R. Collins, A. Lipton, H. Fujiyoshi & T. Kanade. *Algorithms for cooperative multi-sensor surveillance.* In Proc. of the IEEE, volume 89, pages 7–12. IEEE, 2001.

[Collins 03] R.T. Collins. *Mean-shift blob tracking through scale space.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 234–240, 2003.

[Comaniciu 02] D. Comaniciu & P. Meer. *Mean shift: A robust approach toward feature space analysis.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 24, no. 5, pages 603–619, 2002.

[Comaniciu 03] D. Comaniciu, V. Ramesh & P. Andmeer. *Kernel-based object tracking.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 25, pages 564–575, 2003.

[Cremers 02] D. Cremers, T. Kohlberger & C. Schnorr. *Non-linear shape statistics in mumford-shah based segmentation.* In Proceedings of the European Conference on Computer Vision (ECCV). IEEE, 2002.

[Dalal 05] N. Dalal & B. Triggs. *Histograms of oriented gradients for human detection.* In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), volume 1, pages 886–893. IEEE, 2005.

[Darrell 01] T. Darrell, D. Demirdjian, N. Checka & P. Felzenszwalb. *Plan-view trajectory estimation with dense stereo background models.* In Proc. of the Eighth IEEE International Conference on Computer Vision, volume 2, pages 628–635, 2001.

[Deboeverie 11] F. Deboeverie, P. Veelaert & W. Philips. *Face analysis using curve edge maps.* In Lecture Notes in Computer Science, volume 6979, pages 109–118, 2011.

[Delannay 09] D. Delannay, N. Danhier & C. De Vleeschouwer. *Detection and recognition of sports (wo)men from multiple views.* In Proc. of the Third ACM/IEEE International Conference on Distributed Smart Cameras, pages 1–7, 2009.

[Demeulemeester 11] A. Demeulemeester, C. Hollemeersch, P. Lambert, R. Van de Walle, V. Jelača, S. Grünwedel, J.O. Niño-Castañeda, D. Van Cauwelaert, P. Veelaert, P. Van Hese & W. Philips. *Demo : real-time 3D visualization of multi-camera room occupancy monitoring for immersive communication systems.* In Proc. of the Fifth ACM/IEEE International Conference on Distributed Smart Cameras, pages 1–2, 2011.

[Despotović 10] I. Despotović, V. Jelača, E. Vansteenkiste & Philips W. *Noise-robust method for image segmentation.* In Advanced Concepts for Intelligent Vision Systems, Lecture Notes in Computer Science, volume 6474, pages 153–162, 2010.

[Dockstader 01a] S. Dockstader & A.M. Tekalp. *Multiple camera tracking of interacting and occluded human motion.* In Proceedings of the IEEE, volume 89, pages 1441–1455. IEEE, 2001.

[Dockstader 01b] S.L. Dockstader & A.M. Tekalp. *Multiple Camera Fusion for Multi-Object Tracking.* In Proc. of the IEEE Workshop on Multi-Object Tracking, pages 95–102, 2001.

[Dore 10] A. Dore, M. Soto & C.S. Regazzoni. *Bayesian tracking for video analytics.* IEEE Signal Processing Magazine, vol. 27, no. 5, pages 46–55, 2010.

[Edwards 98] G. Edwards, C. Taylor & T. Cootes. *Interpreting face images using active appearance models.* In International Conference on Face and Gesture Recognition, pages 300–305, 1998.

[Elfes 89] A. Elfes. *Occupancy grids: a probabilistic framework for robot perception and navigation.* PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1989.

[Elgammal 00] A. Elgammal, D. Harwood & L. Davis. *Non-parametric model for background subtraction.* In Proc. of the Sixth IEEE European Conference on Computer Vision, Lecture Notes in Computer Science, pages 751–767. Springer, 2000.

[Elgammal 02] A. Elgammal, R. Duraiswami, D. Harwood & L. Davis. *Background and foreground modeling using nonparametric kernel density estimation for visual surveillance.* In Proceedings of IEEE, pages 1151âĂŞ–1163, 2002.

[Fathi 11] A. Fathi, X. Ren & J.M. Rehg. *Learning to recognize objects in egocentric activities.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3281–3288, 2011.

[Felzenszwalb 06] P. Felzenszwalb & D.P. Huttenlocher. *Efficient Belief Propagation for Early Vision.* International Journal of Computer Vision, vol. 70, no. 1, 2006.

[Fieguth 97] P. Fieguth & D. Terzopoulos. *Color-based tracking of heads and other mobile objects at video frame rates.* In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pages 21–27. IEEE, 1997.

[Fleuret 08] F. Fleuret, J. Berclaz, R. Lengagne & P. Fua. *Multi-camera People Tracking with a Probabilistic Occupancy*

Map. IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 30, pages 267–282, 2008.

[Franco 05] J.-S. Franco & E. Boyer. *Fusion of multiview silhouette cues using a space occupancy grid*. In Proc. of the Tenth IEEE International Conference on Computer Vision, volume 2, pages 1747–1753, 2005.

[Freund 95] Y. Freund & R. Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. In Proc. of the ACM Workshop on Conference on Computational Learning Theory, pages 23–37. ACM, 1995.

[Frías Velázquez 11] A. Frías Velázquez, J. Niño Castañeda, V. Jelača, A. Pižurica & W. Philips. *A mathematical morphology based approach for vehicle detection in road tunnels*. In Proceedings of SPIE, the International Society for Optical Engineering, volume 8135, 2011.

[Giebel 04] J. Giebel, D. Gavrila & C. Schnörr. *A bayesian framework for multi-cue 3d object tracking*. In Proc. of the Eighth IEEE European Conference on Computer Vision, volume 3024 of *Lecture Notes in Computer Science*, pages 241–252, 2004.

[Grady 06] L. Grady. *Random Walks for Image Segmentation*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 28, no. 11, pages 1768 – 1783, 2006.

[Greenspan 94] H. Greenspan, S. Belongie, R. Goodman, P. Perona, S. Rakshit & C. Anderson. *Overcomplete steerable pyramid filters and rotation invariance*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 222–228, 1994.

[Grewe 95] L. Grewe & A. Kak. *Interactive learning of a multi-attribute hash table classifier for fast object recognition*. Computer Vision and Image Understanding, vol. 61, no. 3, pages 387–416, 1995.

[Grünwedel 11a] S. Grünwedel, V. Jelača, P. Van Hese, R. Kleihorst & W. Philips. *PhD forum: Multi-view occupancy maps using a network of low resolution visual sensors*. In Proc. of the Fifth ACM/IEEE International Conference on Distributed Smart Cameras, pages 1–2, 2011.

[Grünwedel 11b] S. Grünwedel, P. Van Hese & W. Philips. *An Edge-Based Approach for Robust Foreground Detection*. In Advanced Concepts for Intelligent Vision Systems, volume 6915 of *Lecture Notes in Computer Science*, pages 554–565, 2011.

[Grünwedel 12] S. Grünwedel, V. Jelača, J.O. Niño-Castañeda, P. Van Hese, D. Van Cauwelaert, P. Veelaert & W. Philips. *Decentralized tracking of humans using a camera network.* In Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, volume 8301, pages 1–9, 2012.

[Grünwedel 14] S. Grünwedel, V. Jelača, J.O. Niño Castañeda, P. Van Hese, D. Van Cauwelaert, D. Van Haerenborgh, P. Veelaert & W. Philips. *Low-Complexity Scalable Distributed Multi-Camera Tracking of Humans.* ACM Transactions on Sensor Networks, vol. 10, no. 2, May 2014.

[Guo 07] Y. Guo, S. Hsu, H. S. Sawhney, R. Kumar & Y. Shan. *Robust object matching for persistent tracking with heterogeneous features.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 29, no. 5, pages 824–839, 2007.

[Haralick 73] R. Haralick, B. Shanmugam & I. Dinstein. *Textural features for image classification.* Systems, Man and Cybernetics, IEEE Transactions on, vol. 33, no. 3, pages 610–622, 1973.

[Haritaoglu 00] I. Haritaoglu, D. Harwood & L. Davis. *W4: real-time surveillance of people and their activities.* Pattern Analysis and Machine Intelligence, IEEE Transaction On, vol. 22, no. 8, pages 809–830, 2000.

[Harris 88] C. Harris & M. Stephens. *A combined corner and edge detector.* In Proceedings of the 4th Alvey Vision Conference, pages 147–151, 1988.

[Heikkila 06] M. Heikkila & M. Pietikainen. *A texture-based method for modeling the background and detecting moving objects.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 28, no. 4, pages 657–662, 2006.

[Hengstler 06] S. Hengstler & H. Aghajan. *A smart camera mote architecture for distributed intelligent surveillance.* In Proc. of the ASME Dynamic Systems and Control Conference, Boulder, USA, 2006.

[Hengstler 07] S. Hengstler, D. Prashanth, S. Fong & H. Aghajan. *MeshEye: A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance.* In Proc. of the ACM/IEEE Conference on Information Processing in Sensor Networks, pages 360–369, Cambridge, MA, USA, 2007.

[Hofmann 12] M. Hofmann, P. Tiefenbacher & G. Rigoll. *Background segmentation with feedback: The pixel-based adaptive segmenter.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 38–43. IEEE, 2012.

[Horn 81] B. Horn & B. Schunk. *Determining optical flow.* Artificial Intelligence, vol. 17, pages 185–203, 1981.

[Hou 09] T. Hou, S. Wang & H. Qin. *Vehicle matching and recognition under large variations of pose and illumination.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, pages 290–295, 2009.

[Hou 10] T. Hou, S. Wang & H. Qin. *Active lighting learning for 3D model based vehicle tracking.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, 2010.

[Huang 97] T. Huang & S. Russell. *Object identification in a Bayesian context.* In Proc. of International Joint Conferences on Artificial Intelligence, 1997.

[Hue 02] C. Hue, J.P. Le Cadre & P. Perez. *Sequential Monte Carlo methods for multiple target tracking and data fusion.* IEEE Trans. on Signal Processing, vol. 50, no. 2, pages 309–325, 2002.

[Intille 97] S. Intille, J. Davis & A. Bobick. *Real-time closed-world tracking.* In Proc. of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pages 697–703. IEEE, 1997.

[Isard 98] M. Isard & A. Blake. *Condensation - conditional density propagation for visual tracking.* International Journal of Computer Vision, vol. 19, no. 1, pages 5–28, 1998.

[Isard 01] M. Isard & J. MacCormick. *BraMBLe: A Bayesian multiple-blob tracker.* In Proc. of the Eighth IEEE International Conference on Computer Vision, volume 2, pages 34–41, 2001.

[Javed 03] O. Javed, K. Rasheed, M. Shafique & M. Shah. *Tracking across multiple cameras with disjoint views.* In Proc. of the International Conference on Computer Vision, 2003.

[Javed 05] O. Javed, Shafique K. & M. Shah. *Appearance modelling for tracking in multiple non-overlapping cameras.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, 2005.

[Jelača 08] V. Jelača, A. Pižurica & Philips W. *Computationally efficient algorithm for tracking of vehicles in tunnels.* In Proc. of the 19th Annual Workshop on Circuits, Systems and Signal Processing, pages 335–338, 2008.

[Jelača 11a] V. Jelača, S. Grünwedel, J. Niño Castañeda, P. Van Hese, D. Van Cauwelaert, P. Veelaert & W. Philips. *Demo: Real-time indoors people tracking in scalable camera networks.* In Proc. of the Fifth ACM/IEEE International Conference on Distributed Smart Cameras, 2011.

[Jelača 11b] V. Jelača, J. Niño Castañeda, A. Frías Velázquez, A. Pižurica & W. Philips. *Real-time vehicle matching for multi-camera tunnel surveillance.* In Proceedings of SPIE, the Society of Photo-Optical Instrumentation Engineers, volume 7871, 2011.

[Jelača 12] V. Jelača, J. Niño Castañeda, A. Pižurica & W. Philips. *Image projection clues for improved real-time vehicle tracking in tunnels.* In Proceedings of SPIE, the International Society for Optical Engineering, volume 8301, 2012.

[Jelača 13] V. Jelača, A. Pižurica, J. Niño Castañeda, A. Frías-Velázquez & Philips W. *Vehicle matching in smart camera networks using image projection profiles at multiple instances.* Image and Vision Computing, vol. 31, no. 9, pages 673–685, 2013.

[Jelača 14] V. Jelača, J. Niño Castañeda, A. Pižurica & Philips W. *Robust real-time vehicle tracking using image projection profiles.* submitted to Electronics Letters, 2014.

[Jepson 03] A. Jepson, D. Fleet & T. Elmaraght. *Robust online appearance models for visual tracking.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 25, no. 10, pages 1296–1311, 2003.

[Kalman 60] R.E. Kalman*et al. A new approach to linear filtering and prediction problems.* Journal of Basic Engineering, vol. 82, no. 1, pages 35–45, 1960.

[Kannala 06] J. Kannala & S.S. Brandt. *A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 28, no. 8, pages 1335–1340, 2006.

[Kettnaker 99] V. Kettnaker & R. Zabih. *Bayesian multi-camera surveillance.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, 1999.

[Khan 03] S. Khan & M. Shah. *Consistent labeling of tracked objects in multiple cameras with overlapping fields of view.* Pattern Analysis and Machine Intelligence, IEEE Transaction On, vol. 25, no. 10, pages 1355–1360, 2003.

[Khan 06] S.M. Khan & M. Shah. *A Multiview Approach to Tracking People in Crowded Scenes using a Planar Homography Constraint.* In Proc. of the Ninth IEEE European Conference on Computer Vision, volume 3954 of *Lecture Notes in Computer Science*, pages 133–146, 2006.

[Khan 09] S.M. Khan & M. Shah. *Tracking Multiple Occluding People by Localizing on Multiple Scene Planes.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 31, pages 505–519, 2009.

[Kim 06] K. Kim & L.S. Davis. *Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering.* In Proc. of the Ninth IEEE European Conference on Computer Vision, pages 98–109. Springer, 2006.

[Kirubarajan 04] T. Kirubarajan & Y. Bar-Shalom. *Probabilistic data association techniques for target tracking in clutter.* Proceedings of the IEEE, vol. 92, no. 3, pages 536–557, 2004.

[Kockelkorn 03] M. Kockelkorn, A. Luneburg & T. Scheffer. *Using transduction and multiview learning to answer emails.* In Proc. of the European Conference on Principle and Practice of Knowledge Discovery in Databases, pages 266–277, 2003.

[Kröse 11] B. Kröse, T. Oosterhout & T. Kasteren. *Activity Monitoring Systems in Health Care.* Computer Analysis of Human Behavior, pages 325–346, 2011.

[Krumm 00] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale & S. Shafer. *Multi-Camera Multi-Person Tracking for EasyLiving.* In Proc. of the IEEE Workshop on Visual Surveillance, pages 3–10, 2000.

[Kuhn 55] H. Kuhn. *The Hungarian Method for the assignment problem.* Naval Research Logistics Quarterly, vol. 2, pages 83–97, 1955.

[Laws 80] K. Laws. *Textured image segmentation.* PhD thesis, University of Southern California, Electrical Engineering, 1980.

[Lee 00] L. Lee, R. Romano & G. Stein. *Monitoring activities from multiple video streams: Establishing a common coordinate frame.* Pattern Analysis and Machine Intelligence, IEEE Transaction On, vol. 22, no. 8, pages 758–768, 2000.

[Lee 07] S. Lee, Y. Liu & R. Collins. *Shape variation-based frieze pattern for robust gait recognition.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, 2007.

[Levin 03] A. Levin, P. Viola & Y. Freund. *Unsupervised improvement of visual detectors using co-training.* In Proc. of the IEEE International Conference on Computer Vision (ICCV), pages 626–633. IEEE, 2003.

[Liu 07] J. Liu, M. Chu & J.E. Reich. *Multitarget tracking in distributed sensor networks.* IEEE Signal Processing Magazine, vol. 24, no. 3, pages 36–46, 2007.

[Liyuan 02] L. Liyuan & L. Maylor. *Integrating intensity and texture differences for robust change detection.* Image Processing, IEEE Transactions On, vol. 11, no. 2, pages 105–112, 2002.

[Lowe 04] D. Lowe. *Distinctive image features from scale-invariant keypoints.* International Journal of Computer Vision, vol. 60, No. 2, pages 91–110, 2004.

[Lucas 81] B.D. Lucas & T. Kanade. *An iterative image registration technique with an application to stereo vision.* In Proceedings of Imaging Understanding Workshop, pages 121–130, 1981.

[Maccormick 00] J. Maccormick & A. Blake. *Probabilistic exclusion and partitioned sampling for multiple object tracking.* International Journal of Computer Vision, vol. 39, no. 1, pages 57–71, 2000.

[Maćešić 12] M. Maćešić, V. Jelača, J. Niño Castañeda, N. Prodanović, M. Panić, A. Pižurica, V. Crnojević & W. Philips. *Real-time detection of traffic events using smart cameras.* In Proceedings of SPIE, the International Society for Optical Engineering, volume 8301, 2012.

[Maggio 08] E. Maggio, M. Taj & A. Cavallaro. *Efficient multitarget visual tracking using random finite sets.* IEEE Trans. on Circuits and Systems for Video Technology, vol. 18, no. 8, pages 1016–1027, 2008.

[Mallat 89] S. Mallat. *A theory for multiresolution signal decomposition: The wavelet representation.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 11, no. 7, pages 674–693, 1989.

[Markis 04] D. Markis, T Ellis & J. Black. *Bridging the gaps between cameras.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, 2004.

[Matthies 89] L. Matthies, R. Szeliski & T. Kanade. *Kalman filter-based algorithms for estimating depth from image sequences.* International Journal of Computer Vision, vol. 3, no. 3, pages 209–238, 1989.

[Mikolajczyk 05a] K. Mikolajczyk & C. Schmid. *A performance evaluation of local descriptors.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 27, no. 10, pages 1615–1630, 2005.

[Mikolajczyk 05b] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir & L. Van Gool. *A comparison of affine region detectors.* International Journal of Computer Vision, vol. 65, no. 1/2, pages 43–72, 2005.

[Mittal 03] A. Mittal & L.S. Davis. *M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene.* International Journal of Computer Vision, vol. 51, pages 189–203, 2003.

[Monnet 03] A. Monnet, A. Mittal, N. Paragios & Ramesh V. *Background modeling and subtraction of dynamic scenes.* In IEEE International Conference on Computer Vision (ICCV), pages 1305–1312, 2003.

[Morbee 11] M. Morbee. *Optimized information processing in resource-constrained vision systems. From low-complexity coding to smart sensor networks.* PhD thesis, Ghent University, 2011.

[Morris 08] B. Morris & M. Trivedi. *A survey of vision-based trajectory learning and analysis for surveillance.* IEEE Trans. on Circuits and Systems for Video Technology, vol. 18, no. 8, pages 1114–1127, 2008.

[Munkres 57] J. Munkres. *Algorithms for the Assignment and Transportation Problems.* Journal of the Society for Industrial and Applied Mathematics, vol. 5, no. 1, pages 32–38, 1957.

[Munkres 01] J. Munkres. *On the tracking of articulated and occluded video object motion.* Real Time Image, vol. 7, no. 5, pages 415–432, 2001.

[Nakazawa 98] A. Nakazawa, H. Kato & S. Inokuchi. *Human tracking using distributed vision systems.* In Proc. of the Fourteenth International Conference on Pattern Recognition, volume 1, pages 593–596, 1998.

[Niño Castañeda 11] J. Niño Castañeda, V. Jelača, R. Rios Cabrera, A. Frías Velázquez, A. Pižurica, T. Tuytelaars & W. Philips. *Non-overlapping multi-camera detection and tracking of vehicles in tunnel surveillance.* In Proc. of the International Conference on Digital Image Computing Techniques and Applications, pages 591–596, 2011.

[Nillius 06] P. Nillius, J. Sullivan & S. Carlsson. *Multi-target tracking-linking identities using bayesian network inference.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 2187–2194, 2006.

[Ojala 96] T. Ojala, M. Pietikainen & D. Harwood. *A comparative study of texture measures with classification based on feature distributions.* Pattern Recognition, vol. 19, no. 3, pages 51–59, 1996.

[Oliver 00] N. Oliver, B. Rosario & A. Pentland. *A bayesian computer vision system for modeling human interactions.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 22, no. 8, pages 831–843, 2000.

[Ozturk 09] O. Ozturk, T. Yamasaki & K. Aizawa. *Tracking of humans and estimation of body/head orientation from top-view single camera for visual focus of attention analysis.* In Proc. of the IEEE 12th International Conference on Computer Vision Workshops, pages 1020–1027, 2009.

[Papageorgiou 98] C.P. Papageorgiou, M. Oren & T. Poggio. *A general framework for object detection.* In Proc. of the Sixth IEEE European Conference on Computer Vision, pages 555–562. IEEE, 1998.

[Paragios 02] N. Paragios & R. Deriche. *Geodesic active regions and level set methods for supervised texture segmentation.* International Journal of Computer Vision, vol. 46, no. 3, pages 223–247, 2002.

[Park 04] S. Park & J. K. Aggarwal. *A hierarchical Bayesian network for event recognition of human actions and interactions.* Multimedia Systems, vol. 10, no. 2, pages 164–179, 2004.

[Paschos 01] G. Paschos. *Perceptually uniform color spaces for color texture analysis: an empirical evaluation.* Image Processing, IEEE Transactions on, vol. 2001, no. 10, pages 932–937, 2001.

[Pflugfelder 10] R. Pflugfelder & H. Bischof. *Localization and trajectory reconstruction in surveillance cameras with nonoverlapping views.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 32, no. 4, pages 709–721, 2010.

[Porikli 03] F. Porikli. *Inter-camera color calibration using cross-correlation model function.* In Proc. of the International Conference on Image Processing, 2003.

[Quinlan 86] J. R. Quinlan. *Induction of decision trees.* In Machine Learning, pages 81–106, 1986.

[Rahimi 04] A. Rahimi, B. Dunagan & T. Darrell. *Simultaneous calibration and tracking with a network of non-overlapping sensors.* In Proc. of the International Conference on Computer Vision and Pattern Recognition, 2004.

[Rangarajan 91] K. Rangarajan & M. Shah. *Establishing motion correspondence.* In Proc. of the International Conference on Computer Vision, Graphics and Image Processing (CVGIP), volume 54, pages 56–73, 1991.

[Rasmussen 01] C. Rasmussen & G.D. Hager. *Probabilistic data association methods for tracking complex visual objects.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 23, no. 6, pages 560–576, 2001.

[Rath 03] T.M. Rath & R. Manmatha. *Word image matching using dynamic time warping.* In Proc. of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), volume 2, pages 521–527. IEEE, 2003.

[Reid 79] D. Reid. *An algorithm for tracking multiple targets.* IEEE Trans. on Automatic Control, vol. 24, no. 6, pages 843–854, 1979.

[Rios Cabrera 12] R. Rios Cabrera, T. Tuytelaars & L. Van Gool. *Efficient multi-camera vehicle detection, tracking, and identification in a tunnel surveillance application.* Computer Vision and Image Understanding, vol. 116, pages 742–753, 2012.

[Rittscher 00] J. Rittscher, J. Kato, S. Joga & A. Blake. *A probabilistic background model for tracking.* In European Conference on Computer Vision (ECCV), pages 336–350, 2000.

[Rosten 05] E. Rosten & T. Drummond. *Fusing points and lines for high performance tracking.* In Proceedings of the International Conference on Computer Vision (ICCV), pages 1508–1511, 2005.

[Rosten 06] E. Rosten & T. Drummond. *Machine learning for high-speed corner detection.* In Proceedings of the European Conference on Computer Vision (ECCV), pages 430–443, 2006.

[Rowley 98] H. Rowley, S. Baluja & T. Kanade. *Neural network-based face detection.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pages 23–38, 1998.

[Rusinkiewicz 01] S. Rusinkiewicz & M. Levoy. *Efficient Variants of the ICP Algorithm.* In Proc. of the International Conference on 3D Digital Imaging and Modeling, pages 1–8, 2001.

[Salari 90] V. Salari & I.K. Sethi. *Feature point correspondence in the presence of occlusion.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 12, no. 1, pages 87–91, 1990.

[Schunk 86] B. Schunk. *The image flow constraint equation.* In Proceedings of the International Conference on Computer Vision Graphics and Image Processing, volume 35, pages 20–46. IEEE, 1986.

[Schweitzer 02] H. Schweitzer, J.W. Bell & F. Wu. *Very fast template matching.* In Proceedings of the European Conference on Computer Vision (ECCV), pages 358–372. IEEE, 2002.

[Sebastian 03] T. Sebastian, P. Klein & B. Kimia. *On aligning curves.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 25, no. 1, pages 116–125, 2003.

[Serby 04] D. Serby, S. Koller-Meier & L. Van Gool. *Probabilistic object tracking using multiple features.* In IEEE International Conference of Pattern Recognition (ICPR), pages 184–187, 2004.

[Sethi 87] I. Sethi & R. Jain. *Finding trajectories of feature points in a monocular image sequence.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 9, no. 1, pages 56–73, 1987.

[Shafique 03] K. Shafique & M. Shah. *A non-iterative greedy algorithm for multi-frame point correspondence.* In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 110–115. IEEE, 2003.

[Shan 05] Y. Shan, H. Sawhney & R. T. Kumar. *Vehicle identification between non-overlapping cameras without direct feature matching.* In Proc. of the IEEE International Conference on Computer Vision, 2005.

[Shan 08] Y. Shan, H. Sawhney & R. T. Kumar. *Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 30, No. 4, pages 700–711, 2008.

[Shi 94] J. Shi & C. Tomasi. *Good features to track.* In Proc. of the IEEE Conference Computer Vision and Pattern Recognition, pages 593–600, 1994.

[Shi 00] J. Shi & J. Malik. *Normalized cuts and image segmentation.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 22, no. 8, pages 888–905, 2000.

[Sidla 04] O. Sidla, L. Paletta, Y. Lypetskyy & C. Janner. *Vehicle recognition for highway lane survey.* In Proc. of the International Conference on Intelligent Transportation Systems, pages 531–536, 2004.

[Smith 05] K. Smith, D. Gatica-Perez & J.M. Odobez. *Using particles to track varying numbers of interacting people.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, volume 1, pages 962–969, 2005.

[Smith 06] D. Smith & S. Singh. *Approaches to multisensor data fusion in target tracking: A survey.* IEEE Trans. on Knowledge and Data Engineering, vol. 18, no. 12, pages 1696–1710, 2006.

[Song 96] K.Y. Song, J. Kittler & M. Petrou. *Defect detection in random color textures.* Image and Vision Computing, vol. 14, no. 9, pages 667–683, 1996.

[Soro 09a] S. Soro & W. Heinzelman. *A survey of visual sensor networks.* Advances in Multimedia, vol. 2009, 2009.

[Soro 09b] S. Soro & W. Heinzelman. *Visual learning and recognition of 3D objects from appearance.* Advances in Multimedia, vol. 2009, 2009.

[Stauffer 00] C. Stauffer & W. Eric L. Grimson. *Learning Patterns of Activity Using Real-Time Tracking.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 22, pages 747–757, Aug. 2000.

[Stauffer 05] C. Stauffer. *Learning to track objects through unobserved regions.* In Proc. of the IEEE Workshop on Motion, 2005.

[Stenger 01] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee & J. Buhmann. *Topology free hidden Markov models: Application to background modeling.* In IEEE International Conference on Computer Vision (ICCV), pages 294–301, 2001.

[Streit 94] R.L. Streit & T.E. Luginbuhl. *Maximum likelihood method for probabilistic multi-hypothesis tracking.* In Proceedings of the International Society for Optical Engineering (SPIE), volume 2235, pages 394–405. IEEE, 1994.

[Szeliski 97] R. Szeliski & J. Coughlan. *Spline-based image registration.* International Journal of Computer Vision, vol. 16, no. 1–3, pages 185–203, 1997.

[Taj 09] M. Taj & A. Cavallaro. *Multi-camera track-before-detect.* In Proc. of the Third ACM/IEEE International Conference on Distributed Smart Cameras, pages 1–6, 2009.

[Taj 10] M. Taj & A. Cavallaro. *Multi-view Multi-object Detection and Tracking.* International Journal of Computer Vision, pages 263–280, 2010.

[Taj 11] M. Taj & A. Cavallaro. *Distributed and Decentralized Multicamera Tracking.* IEEE Signal Processing Magazine, vol. 28, no. 3, pages 46–58, 2011.

[Tessens 10] Linda Tessens. *Information selection and fusion in vision systems.* PhD thesis, Ghent University, 2010.

[Thrun 03] S. Thrun. *Learning Occupancy Grid Maps with Forward Sensor Models.* Autonomous Robots, vol. 15, no. 2, pages 111–127, 2003.

[Thrun 05] S. Thrun, W. Burgard & D. Fox. Probabilistic robotics (intelligent robotics and autonomous agents). The MIT Press, 2005.

[Tieu 04] K. Tieu & P. Viola. *Boosting image retrieval.* International Journal of Computer Vision, vol. 56, no. 1, pages 17–36, 2004.

[Turk 91] M. A. Turk & A. P. Pentland. *Eigenfaces for face recognition.* In Proceedings of the Conference on Computer Vision and Pattern Recognition, pages 586–591, 1991.

[Tuytelaars 08] T. Tuytelaars & K. Mikolajczyk. *Local Invariant Feature Detectors: A Survey.* Foundation and Trends in Computer Graphics and Vision, vol. 3, no. 3, pages 177–280, 2008.

[Van Droogenbroeck 12] M. Van Droogenbroeck & O. Paquot. *Background subtraction: experiments and improvements for vibe.* In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 32–37. IEEE, 2012.

[Van Hese 11] P. Van Hese, S. Grünwedel, J.O. Niño-Castañeda, V. Jelača & W. Philips. *Evaluation of background/ foreground segmentation methods for multi-view occupancy maps.* In Proc. of the 2nd International Conference on Positioning and Context-Awareness, pages 37–42, 2011.

[Vaswani 03] N. Vaswani, A. Roychowdhury & R. Chellapa. *Activity recognition using the dynamics of the configuration of interacting objects.* In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pages 633–640. IEEE, 2003.

[Veenman 01] C. Veenman, M. Reinders & E. Backer. *Resolving motion correspondence for densely moving points.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 23, no. 1, pages 54–72, 2001.

[Viola 01] P. Viola & M. Jones. *Rapid object detection using a boosted cascade of simple features.* In Proceedings of the Conference on Computer Vision and Pattern Recognition, pages 511–518, 2001.

[Viola 03] P. Viola, M. Jones & D. Snow. *Detecting pedestrians using patterns of motion and appearance.* In Proc. of the IEEE International Conference on Computer Vision (ICCV), pages 734–741. IEEE, 2003.

[Viola 04] P. Viola & M. J. Jones. *Robust real-time face detection.* International Journal on Computer Vision, vol. 57, no. 2, pages 137–154, 2004.

[Wren 97] C. Wren, A. Azarbayejani, T. Darrel & A. Pentland. *Pfinder: Real Time Tracking of the Human Body.* IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pages 780–785, Jul. 1997.

[Wu 93] Z. Wu & R. Leahy. *An optimal graph theoretic approach to data clustering: Theory and its applications to image segmentation.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 11, pages 1101–1113, 1993.

[Xie 12] X. Xie, S. Grünwedel, V. Jelača, J.O. Niño-Castañeda, D. Van Haerenborgh, D. Van Cauwelaert, P. Veelaert, W. Philips & H. Aghajan. *Learning about Objects in the Meeting Rooms from People Trajectories.* In Proc. of the 6th ACM/IEEE International Conference on Distributed Smart Cameras, page 6, 2012.

[Yilmaz 03] A. Yilmaz, K. Shafique & M. Shah. *Target tracking in airborne forward looking imagery.* International Journal of Computer Vision, vol. 21, no. 7, pages 623–635, 2003.

[Yilmaz 04] A. Yilmaz, X. Li & M. Shah. *Contour based object tracking with occlusion handling in video acquired using mobile cameras.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 26, no. 11, pages 1531–1536, 2004.

[Yilmaz 06] A. Yilmaz, O. Javed & M. Shah. *Object tracking: A survey.* ACM Computing Surveys, vol. 38, no. 4, page 13, 2006.

[Yu 11] G. Yu & JM. Morel. *ASIFT: An Algorithm for Fully Affine Invariant Comparison.* Image Processing On Line, 2011.

[Zhong 03] J. Zhong & S. Sclaroff. *Segmenting foreground objects from a dynamic textured background via a robust Kalman filter.* In IEEE International Conference on Computer Vision (ICCV), pages 44–50, 2003.

[Zhou 03] S. Zhou, R. Chellapa & B. Moghadam. *Adaptive visual tracking and recognition using particle filters.* In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), pages 349–352. IEEE, 2003.

[Zhu 96] S. Zhu & A. Yuille. *Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation.* Pattern Analysis and Machine Intelligence, IEEE Transactions On, vol. 18, no. 9, pages 884–900, 1996.

[Zhu 12] F. Zhu, P. Jiang & Z. Wang. *ViBeExt: The extension of the universal background subtraction algorithm for distributed smart camera.* In Proc. of the International Sym-

posium on Instrumentation & Measurement, Sensor Network and Automation, volume 1, pages 164–168. IEEE, 2012.

[Zivkovic 06] Z. Zivkovic & F. van der Heijden. *Efficient adaptive density estimation per image pixel for the task of background subtraction.* Pattern Recognition Letters, vol. 27, no. 7, pages 773–780, May 2006.