Graz University of Technology

Institute for Computer Graphics and Vision

Master's Thesis

Exploiting 3D Information for Robust Real-Time Tracking of Multiple Objects in Complex Scenarios

Horst Possegger

Graz, Austria
January 2013

Thesis supervisor
Univ.-Prof. Dipl.-Ing. Dr.techn. Horst Bischof

Thesis advisors
Dipl.-Ing. Sabine Sternig
Dipl.-Ing. Thomas Mauthner
Dipl.-Ing. Dr.techn. Peter M. Roth

Vision is more than sight.

Matt Gilman

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am …………………………… ………………………………………………..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………… ………………………………………………..
date (signature)

Abstract

Obtaining location estimates for objects in a scene (*e.g.*, humans) is one of the most important steps in many real-world applications, such as video surveillance or sports analysis. Therefore, over the past decades, object tracking has received significant attention from the computer vision community.

Combining foreground images from multiple views by projecting them onto a common ground-plane has been recently applied within many multiple object, multiple camera tracking approaches. These planar projections introduce severe artifacts and constrain most approaches to objects moving on a common 2D ground-plane, *i.e.*, they cannot robustly handle out-of-plane motion (*e.g.*, jumping people).

To overcome these limitations, we introduce the concept of an *occupancy volume* – exploiting the full geometry and the objects' center of mass – and develop an efficient algorithm for 3D object tracking which is not restricted to the common ground-plane assumption. Individual objects are tracked using the local mass density scores within a particle filter based approach, constrained by a Voronoi partitioning between nearby trackers. Furthermore, we benefit from the geometric knowledge given by the *occupancy volume* to robustly extract features and train classifiers on-demand, when volumetric information becomes unreliable.

We evaluate our approach on several challenging real-world scenarios including the public APIDIS basketball dataset. Experimental evaluations demonstrate significant improvements compared to state-of-the-art methods in terms of precision and accuracy, while reaching near real-time performance.

Kurzfassung

In dieser Arbeit präsentieren wir einen effizienten Tracking Algorithmus zur robusten und präzisen Lokalisierung mehrerer Objekte mit Hilfe überlappender Kameraansichten. Primär verwenden wir dazu geometrische Informationen, basierend auf einer volumetrischen Rekonstruktion der Szene. Da die rein geometrische Information in manchen Situationen zu Mehrdeutigkeiten führen kann – z.B. wenn Objekte nah beieinander stehen – greifen wir in diesen Fällen zusätzlich auf Farbinformation zurück. Dadurch sind wir in der Lage, die individuellen Objekte selbst in komplexen Szenarien verlässlich zu identifizieren.

Im Gegensatz zu dem weit verbreiteten Ansatz, 2D Bildinformationen von mehreren Kameras mit Hilfe planarer Homographien auf eine gemeinsame Grundebene zu projizieren, sind wir durch die volumetrische Rekonstruktion nicht auf die Grundebene beschränkt – d.h. die Objekte können sich frei im 3D Raum bewegen. Weiters vermeiden wir dadurch Projektionsfehler, die bei solchen Ansätzen häufig auftreten und zu Problemen bei der Lokalisierung der Objekte führen. Um den Effekt von Rekonstruktionsfehlern zu reduzieren, analysieren wir die Massendichte in lokalen Nachbarschaften des rekonstruierten Volumens. Durch geeignete Anpassung der Nachbarschaftsrelation an die jeweilige Objektklasse (z.B. Personen) ist es weiters möglich, die Position des Massenschwerpunkts der Objekte zu ermitteln. Anschließend erfolgt die Schätzung der Objektpositionen unter Verwendung eines Partikelfilters, wobei wir zusätzlich eine Voronoi-Zerlegung des Suchraumes nutzen, um effizient mehrere Objekte gleichzeitig lokalisieren zu können.

Zur Evaluierung verwenden wir unterschiedliche Personentracking Szenarien, wie z.B. den öffentlichen APIDIS Basketball Datensatz. Wie aus den experimentellen Auswertungen ersichtlich ist, erzielt unser Ansatz bessere Ergebnisse im direkten Leistungsvergleich – sowohl in Bezug auf die erreichte Genauigkeit, als auch Geschwindigkeit – als vergleichbare Tracking Algorithmen.

Acknowledgments

First and foremost, I want to thank my parents, Margret and Josef, for giving me the opportunity to choose an educational career according to my liking and for their unconditional support during my studies. I would also like to thank my siblings Heidrun and Joe for their mental support and for sending me special “power-up” packages.

Furthermore, I am grateful to my supervisor Prof. Horst Bischof and my advisors Sabine Sternig, Thomas Mauthner, and Peter Roth for their technical support and guidance, as well as for providing me the facilities to work at the institute. I am also very thankful for their excellent and helpful comments while proofreading my thesis – over and over again – and for their assistance during conference submissions. Special thanks go to all members of the institute for the fruitful discussions, especially to Matthias Straka and Christian Reinbacher for providing me an in-depth insight into efficient parallel processing using the CUDA architecture.

From time to time, it is also necessary to step back from work in order to get new ideas and motivation. Thanks to all my friends who helped me out in this regard – especially Gabriel, Gerd, Martin, and Raphael for the “recreational” climbing sessions. I would also like to thank my girlfriend Marlene for all her support during the last years and for her patience when I was immersed in thoughts.

Thank you!

This work was supported by the Austrian Science Foundation (FWF) project Advanced Learning for Tracking and Detection in Medical Workflow Analysis (I535-N23).

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

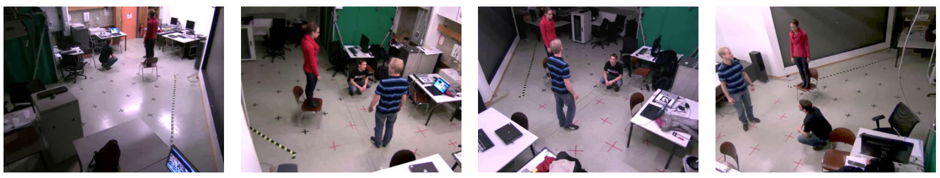
Introduction

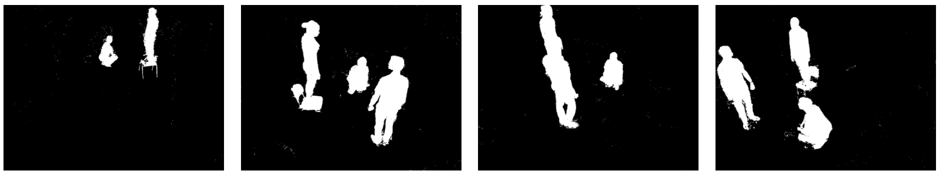
Contents

The robust localization of objects is one of the most important steps for video analysis in numerous applications, such as visual surveillance, sports analysis, or industrial applications. Therefore, considerable research activity has been made in the area of tracking objects from video sequences. For single object tracking, various successful approaches have been proposed, even for robustly handling heavy changes in appearance (*e.g.*, [5]) or geometry (*e.g.*, [43]). In contrast, multi-object tracking is still a challenging problem (*e.g.*, [15, 33, 41, 78]). As soon as the objects of interest are occluding each other, the positions of single instances cannot be estimated reliably.

One way to deal with this problem is to take advantage of multiple cameras. In general, these approaches (*e.g.*, [32, 38, 57, 67, 70]) assume overlapping views observing the same 3D scene by exploiting constraints like objects moving on a common ground-plane, a known number of objects, or that two objects cannot occupy the same position at the same time. These constraints are typically referred to as *closed-world assumptions* [58]. Very often, such methods apply change detection in a first step to estimate the foreground likelihood of each pixel (*e.g.*, [38, 67, 70]). Then, this information is fused exploiting the common ground-plane by either computing a score map [38, 67] or by estimating axes intersections [70].

One of the main problems of these methods, as illustrated in Figure 1.1, is that the usually applied planar projections (*i.e.*, homographies) are only valid for the ground-plane, resulting in unreliable projections for points not lying on the ground-plane. Thus, these projections generate significant ghosting artifacts which have to be handled, *e.g.*, by computing the transformations on several planes, such as [67].


(a) Input images.


(b) Foreground segmentations.

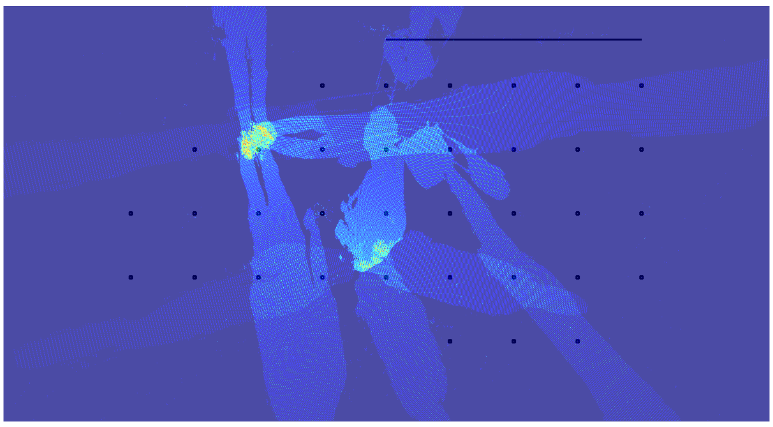
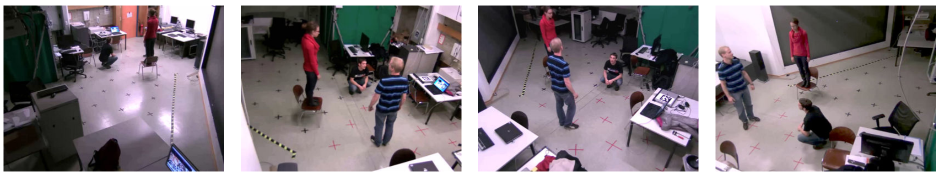
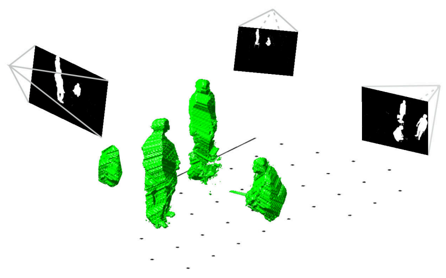

(c) Fused segmentations on the ground-plane.

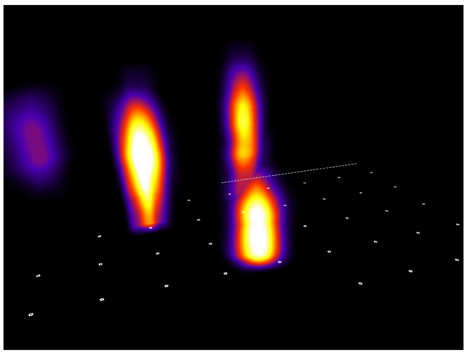
Figure 1.1: Commonly used transformations on input images (a), such as projecting the foreground segmentations (b) onto the ground-plane, cause severe ghosting artifacts (c) and cannot handle violations of the common ground-plane assumption, as illustrated by the person standing on the chair.

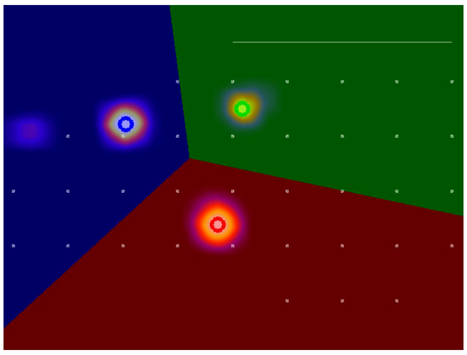
In order to overcome these limitations, a few approaches rely on exploiting the scene structure, such as using epipolar constraints (*e.g.*, [97, 98]) or volumetric 3D reconstructions (*e.g.*, [46, 82]). By additionally considering the 3D information, these approaches suffer significantly less from ghosting artifacts. Furthermore, the knowledge about the camera positions in combination with the estimated object locations allows to robustly handle dynamic occlusions caused by the objects of interest.

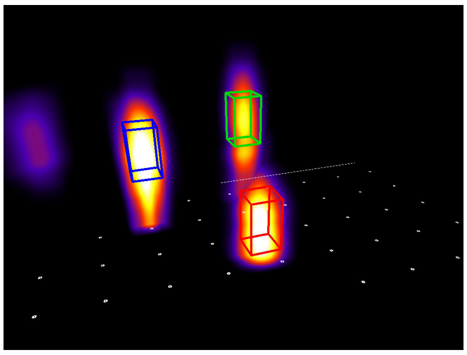
Inspired by the idea of incorporating 3D scene structure for multiple object tracking, we propose a robust and real-time capable approach relying on geometric information as a primary cue and resorting to appearance information solely on-demand, as opposed to [46, 82].


(a) Input images.


(b) Reconstructed visual hull.


(c) Occupancy volume.


(d) Estimating (x, y) coordinates.


(e) Estimating z coordinates.

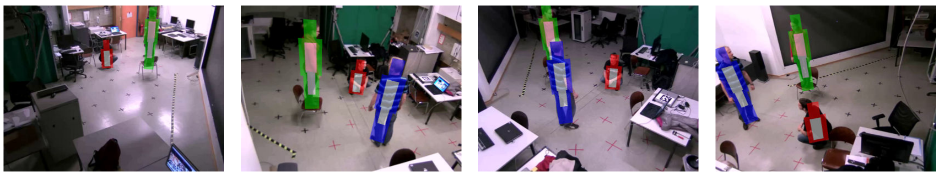

(f) Illustrative tracking results.

Figure 1.2: Using foreground segmentations of the input images (a) we reconstruct the visual hull (b), which is used to compute the occupancy volume based on local mass densities (c). The occupancy volume allows for deriving an occupancy map (d) used for robust estimation of the (x, y) coordinates using particle filtering in combination with Voronoi partitioning. In a final step, we estimate the vertical mass center (e), which allows to reason about the z coordinates (f).

1.1 Thesis Goals and Overview

In this thesis, we propose a novel multiple object tracking algorithm which robustly handles common real-world challenges, such as complex articulations of the individuals, crowded situations, and even out-of-plane motions. Furthermore, by exploiting the inherent parallelism of the underlying techniques, we aim for robust tracking performances at high frame rates.

Therefore, the primary cue for object localization in the proposed approach will be geometric information derived from visual hull reconstructions. For that purpose, we introduce the concept of an *occupancy volume*, which is based on local mass densities of a coarse 3D reconstruction of the objects' visual hull, as illustrated in Figures 1.2(a) – 1.2(c). As can be seen, the incorporation of the local mass density significantly reduces noise and artifacts of the visual hull reconstruction. Furthermore, this allows to derive an occupancy map, which represents the objects' mass center on the ground-plane for robustly estimating the objects' (x, y) coordinates using a particle filter approach in combination with Voronoi partitioning, see Figure 1.2(d). The corresponding z coordinate is then determined using the occupancy volume in a subsequent step, *i.e.*, by locating the vertical mass center within a local neighborhood of the estimated (x, y) coordinates, as shown in Figure 1.2(e). Therefore, in contrast to existing approaches, we are not limited to objects moving on a common ground-plane, which allows robust tracking of complex scenes (*e.g.*, people stepping on ladders, jumping, etc), as illustrated in Figure 1.2(f).

Although geometric information derived from visual hull reconstructions provides a valuable clue for tracking, consistent labeling of tracker hypotheses often cannot be achieved without considering appearance information of the tracked objects, *e.g.*, consider scenarios where people move very close to each other, such as sports games, or children's games[1] like *leapfrog* or *musical chairs*. To overcome these ambiguous situations, we additionally resort to appearance information to correctly re-identify the objects of interest. Therefore, we exploit the 3D scene structure in combination with the tracking results to on-line collect samples for each individual object. Whenever geometric information becomes unreliable, we use the on-line collected appearance information to train discriminative appearance models on-demand. Thus, we can correctly re-identify the corresponding objects and ensure a consistent labeling.

We evaluate our approach on several challenging real-world people tracking scenarios, including the public APIDIS basketball dataset. Experimental

[1]In fact, these games are designed for the young and the young-at-heart. Thus, even researchers enjoy playing these games, as can be seen by carefully analyzing the facial expressions of the participating people. We refer the interested reader to the evaluations of the Leaf 1 & 2 and Much scenarios in Chapter 5.

evaluations demonstrate significant improvements compared to state-of-the-art methods, while reaching near real-time performance.

1.2 Outline

The remainder of this thesis is organized as follows. First, we review related visual tracking approaches in Chapter 2. Second, we discuss the theoretical background of the applied techniques, *i.e.*, 3D reconstruction and recursive Bayesian filters, in Chapter 3. Third, we detail our multi-cam multi-object tracking approach in Chapter 4. Next, we present qualitative and quantitative tracking results of the proposed approach and compare it to the state-of-the-art using several challenging datasets, including the public APIDIS dataset, in Chapter 5. Finally, we conclude this thesis and provide an outlook on future work in Chapter 6.

Chapter 2

Related Work

Contents

Locating objects is an important step in many real-world applications. Therefore, object tracking has received significant attention from the computer vision community over the past decades. In the following, we provide an overview on related multiple object tracking algorithms using either monocular camera systems or multiple camera networks.

In particular, we discuss selected approaches which focus on tracking multiple humans. For a more detailed overview, we refer the interested reader to the numerous excellent surveys by Aggarwal and Cai [1], Gabriel et al. [39], Gavrila [42], Moeslund and Granum [99], Porikli [109], or Yilmaz et al. [137].

2.1 Single Camera Tracking

Various different multi-object tracking approaches have been proposed for applications which rely on a single camera. In general, a crucial step in object tracking is to obtain exact location estimates for the targets. For example, a commonly used concept is to segment moving objects via background modeling or frame differencing, *e.g.*, Han et al. [48], Haritaoglu et al. [49], or Intille and Bobick [58]. However, precise location estimates for all objects in densely crowded situations can hardly be obtained using pure binary information from a single view. In such situations, tracking algorithms often benefit from appearance information, *e.g.*, Comaniciu et al. [24], Nummiaro et al. [103], or Seo et al. [118]. Nevertheless, these approaches suffer from similar appearance of the objects, since color-based identification of single instances becomes significantly harder.

Another commonly used concept is tracking-by-detection, *i.e.*, object detectors are used to obtain location estimates. For example, Wu and Nevatia [135] represent people by an assembly of body parts. Therefore, they train their detectors by boosting several weak classifiers based on edgelet features. The tracking step is performed by a data association procedure in combination with mean shift [23].

Recently, tracking-by-detection has also shown to be a promising solution if the camera itself is allowed to move, *e.g.*, as demonstrated by Breitenstein et al. [15], Ess et al. [33], or Gall et al. [41]. Nevertheless, a major drawback of detection-based approaches is that they require large amounts of training data to obtain suitable object detectors. Furthermore, the detection step is still sensitive to severe occlusions.

Although there are several different ways to obtain location estimates, these are usually associated with uncertainties, *e.g.*, due to noisy measurements. A commonly used concept to handle such uncertainties is to formulate the tracking problem in the context of statistical estimation, *e.g.*, by using a Bayesian framework. Often, such tracking systems rely on particle filters [59] to estimate the posterior probability which describes the state (*e.g.*, position and speed) of an object. For example, Isard and MacCormick [60] explicitly extend the state-space to estimate the state of multiple objects by a single particle filter. However, this approach is sensitive to identity switches caused by the proposed foreground model and furthermore, poor estimates of a single target may severely degrade the entire estimation.

To overcome this problem, several efficient schemes based on Markov Chain Monte Carlo (MCMC) sampling have been proposed to replace the importance sampling of standard particle filters, *e.g.*, Khan et al. [69], Pérez et al. [108], Smith et al. [122], Yu et al. [138], or Zhao and Nevatia [141]. In contrast to these approaches, Vermaak et al. [132] introduce a mixture particle filter which is able to maintain the multi-modality of the estimated posterior distribution of the target states over time. Several multi-object tracking approaches, such as Cai et al. [19] or Okuma et al. [104], have applied this modified particle filter. However, these approaches are sensitive to occlusions and therefore, deteriorate quickly at densely crowded scenes.

Another interesting multi-object tracking system based on a Bayesian framework has been proposed by Kristan et al. [75]. In particular, they consider the problem of analyzing indoor sports games using a single top-view camera observing the whole court. Furthermore, they incorporate a-priori knowledge to reduce the computational complexity. For example, by using a top-view camera they can safely assume that multiple objects cannot occupy the same location on the observed image at the same time. This allows to track each object by an individual particle filter, exploiting a Voronoi partitioning of the input image. Nevertheless, a major drawback of this approach is that the single static top-view camera limits its field of application to indoor scenarios. Furthermore, since the required wide-angle lenses introduce significant distortion effects, the extracted features to distinguish people are usually less discriminative compared to features extracted from views at a lower elevation. Nevertheless, they presented promising experimental results which inspired us to adopt the concept of using Voronoi partitioning for efficient particle filtering.

To overcome the limitations of most single camera tracking algorithms *w.r.t.* occlusion handling in crowded scenarios, several other approaches also rely on mounting the camera at a high elevation, *e.g.*, using lampposts. On the one hand, this allows for easier modeling interactions between the objects, as demonstrated by Pellegrini et al. [106] or Scovanner and Tappen [115]. On the other hand, techniques from evacuation dynamics may be applied, as proposed by Ali and Shah [3]. Since these approaches usually assume strong motion priors for the individuals, they cannot robustly handle situations where the people violate the expected behavior, *e.g.*, if a person suddenly changes her direction.

In contrast to these approaches, other tracking algorithms rely on additional cameras to overcome the occlusion problem. Therefore, we will summarize state-of-the-art multiple object, multiple camera tracking approaches in the following section.

2.2 Multi-Camera Tracking

Traditionally, multiple cameras have mainly been used to extend the tracking region, *e.g.*, Cai and Aggarwal [18], Khan et al. [68], Kettnaker and Zabih [65], or Quaritsch et al. [113]. However, since these approaches use only a single camera for tracking at any time, the tracking performance cannot be improved compared to standard single camera approaches. Therefore, there is considerable interest in multi-camera networks with overlapping views, which provide additional information to overcome the occlusion problem.

In general, such approaches consider static camera networks where moving objects may easily be detected via standard background subtraction techniques. For example, Berclaz et al. [8] use a generative model where people are represented by rectangles. The models over all views are iteratively combined to estimate the probabilities of occupancy at the common ground-plane. Hence, this approach is also well-known as probabilistic occupancy map, *i.e.*, POM [38]. In the tracking step, the estimated locations are linked into consistent trajectories using the Viterbi algorithm in a greedy way. As shown by Berclaz et al. [9], this approach can be improved by using the K-shortest path (KSP) algorithm. However, as these approaches do not incorporate appearance information, they are very sensitive to identity switches, especially in crowded scenes.

Therefore, Mandeljc et al. [88] recently proposed a multi-modal tracking approach where they combine visual cues obtained from the POM detector with ultra-wide band radio information. The combined occupancy probabilities are then linked into target trajectories using the KSP tracker [9]. As shown by their experimental evaluations on a challenging indoor dataset, this fusion of vision-based and radio-based localization significantly improves the tracking performance. However, in most real-world scenarios only visual input data may be available for object tracking and hence, this improvement is limited to few scenarios.

Another widely used cue to locate people in a scene is to compute the intersection points of the principal axes of people on the ground-plane. For example, Kim and Davis [70] apply viewpoint-independent appearance models to segment the people and use the intersection points of their principal axes within a particle filtering framework to estimate the target positions. Similarly, Du and Piater [30] also rely on intersecting the principal axes to locate the people. In particular, they use separate particle filters to detect people within each view and use the axes intersection points to combine the results on the common ground-plane. Furthermore, they apply additional particle filtering on the ground-plane to obtain the location estimates. However, a major drawback of this approach is the limited ability of the individual particle filters to handle occlusions.

In contrast to intersecting the principal axes, Sternig et al. [124] modify the voting scheme of the standard Hough forest [40] such that codebook entries vote for the foot point location of the object. The detection results from the individual camera views are then combined by projecting them onto a common ground-plane using planar homographies. The actual tracking step is then performed using particle filters. Although achieving good tracking performance, their approach suffers from the high computational complexity of the Hough forest itself.

Previously, the location of people's feet has also been used by Khan and Shah [66] for robust multi-object tracking. In particular, they apply a Gaussian mixture model to obtain foreground likelihood maps for each camera view which are subsequently projected onto the ground-plane. To locate the people, they fuse the projected likelihood maps by applying the constraint that the people's feet are visible in each view. This approach has later been extended by the same authors in [67] by considering multiple planes parallel to the ground-plane to overcome inaccurate projections. However, feet may often be occluded in crowded scenes and thus, the tracking performance deteriorates quickly in such situations.

Therefore, Eshel and Moses [32] propose to track the heads of people, observed from multiple cameras at high elevations. In particular, they use homographies to project the input images onto multiple planes parallel to the ground, approximately at head level. The head positions are then found by evaluating the intensity variances. However, similar hair color of people in combination with projection artifacts cause several false positive estimates.

Furthermore, as they lack additional appearance features to describe the individuals, their approach also suffers from identity switches.

Another valuable cue for object tracking with multi-camera networks is geometric information. For example, Krumm et al. [77] use stereo cameras to observe people within a living room. In contrast to this approach, Mittal and Davis [97] exploit the epipolar geometry of standard cameras within a surveillance network. In particular, they match foreground blobs corresponding to the objects along the epipolar lines. Originally, they use standard color segmentation to separate the people. In [98], the same authors introduced a much more discriminative color model to overcome limitations of the simple segmentation. Nevertheless, this approach is still sensitive to similar appearance of the objects. Therefore, Gupta et al. [47] propose a method to perform visibility analysis for each person which allows for automatic camera selection. In their experimental evaluations, they show to significantly improve the results of [98] by additionally incorporating their proposed view selection.

Other multiple object tracking approaches focus the target localization on the geometric information derived from viewing cones spanned by foreground segmented blobs. For example, Otsuka and Mukawa [105] formulate a probabilistic framework to analyze occlusions based on the intersection of viewing cones on the ground-plane. Similarly, Yang et al. [136] use the intersection of 3D viewing cones projected onto the 2D ground-plane for counting and tracking people within a room. Nevertheless, the intersections used by these approaches are sensitive to noisy foreground segmentation and may cause both false positive and false negative location estimates. One possibility to reduce this noise sensitivity is to analyze volumetric reconstructions, which also builds the base of our proposed tracking algorithm.

Although volumetric reconstruction methods have widely been used in the field of marker-less human motion capture (*e.g.*, Bottino et al. [13], Caillette and Howard [20], Mikić et al. [95], or Theobalt et al. [130]), only few approaches exploit 3D reconstructions for multiple object tracking. For example, Guan et al. [45] improve the visual hull reconstruction by incorporating appearance information in a Bayesian framework. In [46], the same authors show that this improved reconstruction can be used to robustly track multiple people, even under heavy occlusions. However, this approach is significantly limited by the computational complexity of the proposed Bayesian inference which requires approximately two minutes per frame. Furthermore, the color-based inference deteriorates quickly in situations where the objects are vested similarly.

We consider the people tracking approach of Liem and Gavrila [82] to be most closely related to ours. In particular, they rely on a simple top-view projection of the standard visual hull reconstruction to obtain location estimates at the ground-plane. If a projected blob is larger than an average person, *e.g.*, if multiple people are standing close to each other, they apply expectation-maximization (EM) to split the blob into several location hypotheses. Subsequently, they extract RGB histograms for each candidate blob and compare it to appearance models of the targets. The actual tracking step consists of Kalman filtering [62] using the location estimates in combination with the color information. Nevertheless, their approach suffers from several drawbacks: By using the standard visual hull reconstruction, they share its noise sensitivity. Furthermore, the blob-splitting approach may produce wrong location hypotheses and additionally, they need to extract appearance features at any time to robustly locate the targets.

Chapter 3

Geometry and Recursive Bayesian Filtering

Contents

In the following chapter, we summarize the mathematical background of the applied techniques for a better understanding of the proposed tracking approach.

3.1 Image Formation

In order to reconstruct 3D scene structure from 2D images knowledge about the image formation process must be applied. Therefore, in Section 3.1.1, we briefly summarize the concepts of the standard pinhole model, as it is most frequently used in machine vision to mathematically describe the image formation of standard consumer cameras, *i.e.*, perspective cameras. Next, we discuss common lens distortion phenomena which occur in practice, along with their corrections in Section 3.1.2. Finally, we discuss geometric camera calibration techniques needed to apply the pinhole model in practice in Section 3.1.3.

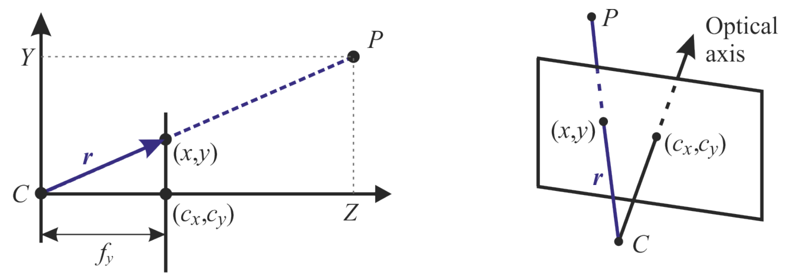


Figure 3.1: Projection of the 3D real-world point $P = (X, Y, Z)^\top$ onto the image plane using the standard pinhole camera model. The corresponding pixel coordinates $(x, y)^\top$ can be obtained from similar triangles, *e.g.*, the vertical coordinate is computed as $y = f_y \frac{Y}{Z} + c_y$. Note that for clarity, we assume that the camera's projection center is located at the real-world point $(0, 0, 0)^\top$ and the axes of the world coordinate system and the camera reference frame are aligned.

Note that a detailed introduction on camera geometry is clearly beyond the scope of this thesis. Therefore, we refer the interested reader to the excellent text books of Faugeras [35], Szeliski [129], and especially Hartley and Zisserman [54] for a detailed discussion on the mathematical principles.

3.1.1 Standard Pinhole Camera Model

The standard model to describe the projection of 3D real-world points onto a 2D image plane in computer vision is the *pinhole camera* or *central perspective projection*. This model assumes that no lenses are used, *i.e.*, the camera aperture is a single point, the pinhole.

The pinhole model follows the principle of *collinearity*, *i.e.*, each real-world point is projected by a straight line through the projection center onto the image plane, as illustrated in Figure 3.1. This means that all three points, *i.e.*, the real-world point P, the center of projection C, and the imaged point $(x, y)^\top$, are located on the same line.

The origin of the camera coordinate system coincides with the projection center, also referred to as camera center, at the position $C = (X_0, Y_0, Z_0)^\top$ *w.r.t.* the world coordinate system. In general, the axes of the coordinate systems do not coincide, *i.e.*, the camera is rotated, as the z-axis of the camera frame is defined to be perpendicular to the image plane. This rotation is usually expressed by the Euler angles ω, φ, and ψ, which define the rotations around the x, y, and z-axis, respectively. As the rotation and translation define the camera's orientation in space, *i.e.*, the *camera pose*, these parameters are commonly referred to as the *extrinsic camera parameters*.

Projecting a real-world point onto the image plane can now be done as follows. First, the point's coordinates $P = (X, Y, Z)^\top$ are transformed from the world coordinate system to the camera coordinate system, *i.e.*, $\tilde{P} = (\tilde{X}, \tilde{Y}, \tilde{Z})^\top$. Therefore, the point needs to be translated and rotated *w.r.t.* to the camera pose. This transformation is defined as

$$\tilde{P} = \mathbf{R}P + \mathbf{t}, \tag{3.1}$$

where $\mathbf{t}$ is the coordinate vector of the origin of the world coordinate system in the camera reference frame, *i.e.*, $\mathbf{t} = -\mathbf{R}C$, and $\mathbf{R}$ denotes the rotation matrix obtained from the orthogonal matrices which define the rotation around the individual axes:

$$\mathbf{R} = \mathbf{R}_z \mathbf{R}_y \mathbf{R}_x, \tag{3.2}$$

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\omega & -\sin\omega \\ 0 & \sin\omega & \cos\omega \end{pmatrix}, \tag{3.3}$$

$$\mathbf{R}_y = \begin{pmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{pmatrix}, \tag{3.4}$$

$$\mathbf{R}_z = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.5}$$

Note that the rotations are performed clockwise, *i.e.*, first around the x-axis, next around the y-axis, and finally around the z-axis.

In a subsequent step, the point is projected onto the image plane of the camera. The corresponding transformation is defined by the camera's *intrinsic parameters*, namely:

- the focal lengths for the x and y dimensions, *i.e.*, f_x and f_y, which define the magnification in the corresponding direction,

- the factor γ to account for a possible skew between the sensor axes - in case the sensor is not mounted perpendicular to the optical axis,

- the location of the optical center, also called *principal point*, *i.e.*, the pixel coordinates $(c_x, c_y)^\top$ where the optical axis intersects the image plane - this position is used as a translation vector, since in general, the origin of coordinates in the image plane (mostly top-left) is not at the principal point.

Note that in the literature there are several different ways to represent the intrinsic parameters. However, all these representations can be easily converted back and forth, *e.g.*, some authors prefer to state a single focal length value f in combination with the aspect ratio α_f, which in fact is equivalent to $f_x = f$ and $f_y = \alpha_f f$ *w.r.t.* above notation.

Using the intrinsic camera parameters, $\tilde{P}$ can now be projected onto the image plane, *i.e.*, pixel coordinates $(x, y)^\top$, as:

$$x = \frac{f_x \tilde{X} + \gamma \tilde{Y}}{\tilde{Z}} + c_x, \tag{3.6}$$

$$y = \frac{f_y \tilde{Y}}{\tilde{Z}} + c_y. \tag{3.7}$$

Note that if the focal lengths are measured in meters, an additional scaling must be applied to obtain the imaged point coordinates in pixels. For more details, we refer to the literature, *e.g.*, [54, 55].

A major advantage of the pinhole model is that the image formation can be easily represented by a matrix multiplication. Therefore, the point P is first represented by *homogeneous* coordinates (*e.g.*, [12, 54]), *i.e.*, its coordinate vector is augmented such that P becomes a point of the projective space $\mathcal{P}^3$. Then, the projection onto the image plane can be computed from multiplication with the *projection matrix* $\mathbf{P}$, as

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{pmatrix} = \mathbf{P} \begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix}, \tag{3.8}$$

where $(\tilde{x}, \tilde{y}, \tilde{w})^\top$ is the coordinate vector of the imaged point in the projective space $\mathcal{P}^2$. In order to obtain the pixel coordinates $(x, y)^\top \in \mathbb{R}^2$, one simply divides the projective coordinates by the scaling factor $\tilde{w}$:

$$x = \frac{\tilde{x}}{\tilde{w}}, \tag{3.9}$$

$$y = \frac{\tilde{y}}{\tilde{w}}. \tag{3.10}$$

The camera's projection matrix, which contains the intrinsic and extrinsic parameters, is defined as

$$\mathbf{P} = \mathbf{K}\mathbf{R}\,[\,\mathbf{I}_{3\times3} \,|\, -C\,] = \mathbf{K}\,[\,\mathbf{R} \,|\, -\mathbf{R}C\,], \tag{3.11}$$

where $\mathbf{K}$ is the *calibration matrix* which holds the intrinsic parameters. Following the previously defined notation of these parameters, the calibration matrix may be defined[2] as the upper-triangular[3] matrix:

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.12}$$

[2]Similar to the intrinsic parameters, there are several equivalent representations of the calibration matrix too.

[3]Note that this upper-triangular representation allows to recover the intrinsic parameters from the camera's projection matrix using QR decomposition [129].

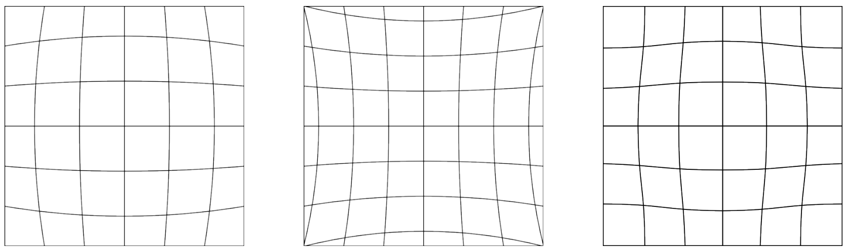

(a) Barrel. (b) Pincushion. (c) Mustache.

Figure 3.2: Radial distortions cause the image coordinates to be displaced away, *i.e.*, *barrel* distortion (a), or towards, *i.e.*, *pincushion* distortion (b), the image center by an amount proportional to the corresponding radial distance. Furthermore, complex lenses may also exhibit a mixture of both, commonly known as *mustache* distortion (c).

3.1.2 Geometric Lens Distortions

The pinhole model assumes a linear projection, *i.e.*, straight lines in the world project to straight lines in the image. Therefore, it is only an approximation of the real camera projection, since in general, standard lenses usually suffer from distortion and thus, image coordinates are displaced. Hence, in order to apply the pinhole model, the distortion needs to be corrected in a pre-processing step, *i.e.*, the images need to be rectified.

According to [54, 129], lens distortion occurs during the initial projection of the point $\tilde{P} = (\tilde{X}, \tilde{Y}, \tilde{Z})^\top$ onto the image plane. Thus, image correction needs to be applied at this place. In the following, we briefly demonstrate the correction of the most frequently occurring *radial* distortion, which manifests itself as a visible curvature in the projection of straight lines, see Figure 3.2. For many applications, a sufficiently accurate correction can be provided by simple radial distortion models using low-order polynomials.

Therefore, let $(\hat{x}, \hat{y})^\top$ denote the *normalized* image coordinates, obtained *after* the perspective division, and *before* scaling by the focal length and shifting by the optical center:

$$\hat{x} = \frac{\tilde{X}}{\tilde{Z}}, \qquad \hat{y} = \frac{\tilde{Y}}{\tilde{Z}}. \tag{3.13}$$

Following [54], the correction can now be applied by using a Taylor expansion to approximate the true radial distortion. Thus, we obtain the normalized image coordinates $(\hat{x}_d, \hat{y}_d)^\top$ after accounting for the distortion as

$$\hat{x}_d = \hat{x}\left(1 + \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3 + \dots\right), \tag{3.14}$$

and

$$\hat{y}_d = \hat{y}\left(1 + \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3 + \dots\right), \tag{3.15}$$

where $r = \sqrt{\hat{x}^2 + \hat{y}^2}$ is the radial distance from the image center. After this correction of the radial distortion, the rectified pixel coordinates can be computed as:

$$x = f_x \hat{x}_d + \gamma \hat{y}_d + c_x, \qquad y = f_y \hat{y}_d + c_y. \tag{3.16}$$

Note that the choice of the order of the polynomial actually depends on the camera optics, *i.e.*, distortion models for wide-angle lenses require higher-order polynomials to provide an accurate correction, in contrast to distortion models for standard lenses. Furthermore, the coefficients κ_i need to be estimated during the camera calibration step and are usually considered to be part of the intrinsic calibration.

Another frequently occurring distortion is caused by a *decentering* of the lens, *i.e.*, lens elements are not strictly symmetric about the optical axis. In general, this is due to a misalignment during camera assembly and causes a *tangential* and *asymmetric radial* distortion of the images. In practice, this distortion is corrected by applying the Brown-Conrady model [16], or the model of Brown [17] which is able to handle both, radial and tangential distortion. For a more detailed overview on lens distortion, we refer the interested reader to the text books of Hartley and Zisserman [54] and Szeliski [129].

3.1.3 Geometric Camera Calibration

The goal of geometric camera calibration methods is to estimate the image formation parameters, *i.e.*, intrinsic and extrinsic parameters. Over the past decades, many different approaches have been proposed, mainly focusing on the calibration of the intrinsic camera parameters. Several approaches make simplifying assumptions to solve the intrinsic calibration, *e.g.*, the camera motion is restricted to be pure translational (*e.g.*, [28]), or to a known translation and rotation (*e.g.*, [6, 29]). Other calibration techniques do not impose such simplifications, *i.e.*, the camera motion may be arbitrary and is unknown, and estimate the intrinsic camera parameters from matching feature points, *e.g.*, [91]. Usually, these methods exploit the epipolar geometry between the camera views. On the contrary, if a single camera is to be calibrated which is mounted stationary but may be rotated, the epipolar geometry cannot be exploited to obtain the calibration. Therefore, alternative methods, such as [51], are required to handle purely rotational camera motion. Note that there have been many more camera calibration methods proposed over the years. Therefore, we refer the interested reader to the exhaustive survey on camera calibration methods by Clarke and Fryer [22].

Since our proposed tracking approach is based on visual hull reconstruction, we require fully calibrated cameras, *i.e.*, the camera pose, as well as the intrinsic parameters need to be known. For such applications, the standard calibration approach [131] is to simultaneously estimate the intrinsic

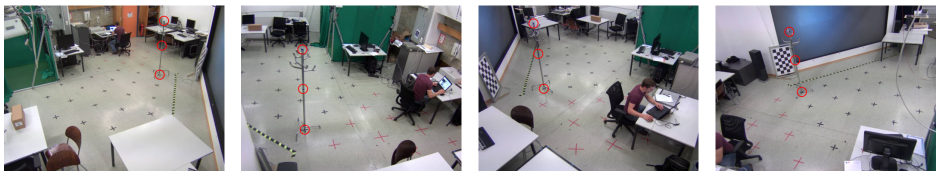
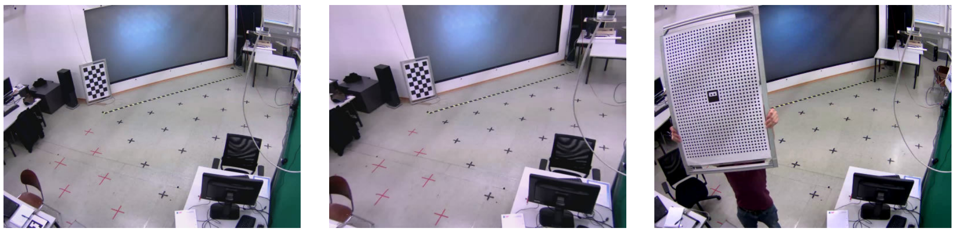


Figure 3.3: The infamous hat stand calibration target used to estimate the extrinsic parameters of the camera network at the ICG *Deskotheque* laboratory. Note that the marker positions are highlighted by red circles.

and extrinsic parameters *w.r.t.* a known calibration target. Usually, a 3D calibration rig with known geometry is used to calibrate the cameras.

Note that there are several alternative calibration methods which may be appropriate to calibrate such camera networks. For example, [126] proposes a multi-camera calibration method based on correspondences obtained from a laser pointer. Other approaches use correspondences by observing walking people to calibrate camera networks, *e.g.*, [73, 110] match detected head and foot positions, whereas [120] uses extracted contours of the people. However, these automatic calibration methods require additional modifications to explicitly define the position and alignment of the world reference frame, since they do not provide the opportunity to include known 3D points in the calibration process.

To calibrate the camera network used for recording the ICG laboratory datasets, we preferred the standard approach as it allows to easily align the world reference frame using the calibration target. However, the available 3D calibration rig was too small and thus, could not be captured at a sufficiently high resolution from all cameras simultaneously. Therefore, we decided to come up with a simple, yet effective, method to replace the 3D calibration rig. In particular, we used a standard *hat stand* with markers at different height levels (*i.e.*, at $z = 0\,\text{m}$, $z = 1\,\text{m}$, and $z = 1.7\,\text{m}$), as shown in Figure 3.3. By placing the hat stand at known ground-plane positions and manually extracting the imaged marker points, we obtained the point correspondences required for extrinsic calibration. Furthermore, we used the publicly available calibration toolbox of Bouguet [14] which proves very versatile, as it combines standard calibration methods (*e.g.*, [140]) with sophisticated distortion models (*e.g.*, [17, 55]). The intrinsic camera parameters along with the distortion coefficients are estimated using a planar calibration target, as shown in Figure 3.4.



(a) Original frame. (b) Rectified frame. (c) Calibration target.

Figure 3.4: Intrinsic geometric calibration. In practice, real camera images are distorted, *e.g.*, note the radial distortion in the original image (a). In order to apply the standard pinhole camera model, the original input data needs to be rectified (b). A commonly used method is to estimate the distortion coefficients along with the intrinsic parameters by capturing a target with known geometry, such as shown in (c).

3.2 3D Reconstruction

When projecting 3D real-world points onto 2D image planes, one dimension is discarded. Therefore, recovering 3D information from images is a challenging problem in machine vision. Nevertheless, since many applications depend on extracting scene structure from input images, this problem has been well-studied over the past decades.

Since several tracking algorithms, *e.g.*, [82, 98], have successfully applied knowledge about the 3D structure, we decided to base our proposed multiple object tracking approach on reconstructed geometric information as a primary cue. In the following, we therefore summarize the mathematical background for reconstructing scene structure from images. First, we provide an overview on selected 3D reconstruction techniques in Section 3.2.1. Second, we detail shape from silhouette, as this reconstruction method is able to deal with all limitations imposed by standard surveillance camera networks in Section 3.2.2.

3.2.1 Overview

Although a lot of information is lost when projecting 3D real-world points onto a 2D image plane, images still provide various cues for reconstructing 3D scene structure. These methods are usually summarized as *shape from X*, as they use different cues for reconstructing shape information. In the following, we summarize the concepts of selected shape from X techniques, *i.e.*, stereo matching, shape from shading, photometric stereo, and shape from focus. For a more detailed overview on various reconstruction techniques, we refer the interested reader to the surveys by Dyer [31] and

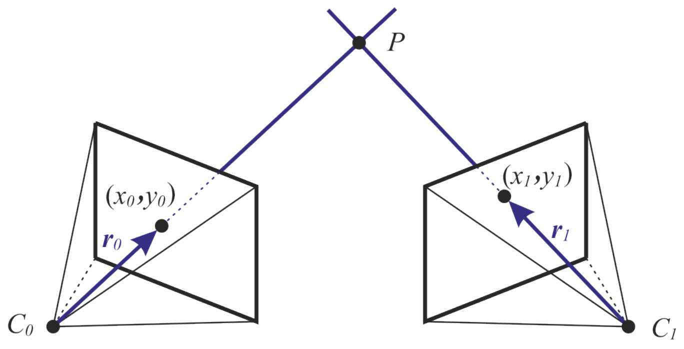


Figure 3.5: 3D point triangulation. Given the matching feature points (x_c, y_c), the corresponding real-world point P can be computed as the intersection of the optical rays $\mathbf{r}_c$. Illustration based on [129].

Slabaugh et al. [121], as well as the more recent surveys by Seitz et al. [117] and Szeliski [129].

Shape from Stereo

Shape recovery using stereo correspondence takes two or more images of the same scene captured by calibrated cameras. The key idea is to find corresponding points in the camera images and obtain the 3D points from triangulation (*e.g.*, see [53, 54]). In general, the correspondences are established by detecting interest points (*e.g.*, Harris corners [50]) and matching the extracted feature descriptors (*e.g.*, using the well-known SIFT [85] descriptor).

Given the corresponding feature matches, triangulation of the 3D points can be done by exploiting the geometric camera calibration, illustrated in Figure 3.5. As can be seen, the optical rays $\mathbf{r}_c$ originate at the camera centers C_c in direction to the feature points (x_c, y_c). Therefore, the corresponding 3D point $P = (X, Y, Z)^\top$ must be located at the intersection of the viewing rays.

In practice, however, uncertainties of the camera calibration have to be considered. Therefore, Sutherland [125] proposed a statistically optimal formulation by minimizing the residual of the measurement equations

$$x_c = \frac{p^{(c)}_{(0,0)}X + p^{(c)}_{0,1}Y + p^{(c)}_{0,2}Z + p^{(c)}_{0,3}W}{p^{(c)}_{2,0}X + p^{(c)}_{2,1}Y + p^{(c)}_{2,2}Z + p^{(c)}_{2,3}W}, \tag{3.17}$$

and

$$y_c = \frac{p^{(c)}_{(1,0)}X + p^{(c)}_{1,1}Y + p^{(c)}_{1,2}Z + p^{(c)}_{1,3}W}{p^{(c)}_{2,0}X + p^{(c)}_{2,1}Y + p^{(c)}_{2,2}Z + p^{(c)}_{2,3}W}, \tag{3.18}$$

where $p_{(i,j)}^{(c)}, i \in \{0,1,2\}, j \in \{0,1,2,3\}$ are the entries of the projection matrix $\mathbf{P}_c$ of camera c.

This minimization problem is usually solved via singular value decomposition (SVD), as the equations become homogeneous if the 3D point is expressed in homogeneous coordinates, *i.e.*, $P = (X, Y, Z, W)^\top$. For more details on multiple view geometry we refer to the text book of Hartley and Zisserman [54].

Shape from Shading and Photometric Stereo

Analyzing illumination and reflectance properties of images yields another cue for reconstructing surface shapes. Based on this idea, two major approaches have evolved. The *shape from shading* [56] approach uses the shading information from a single static image to reconstruct the 3D surface. Obviously, this represents an ill-posed problem and thus, requires several additional assumptions, such as known and constant illumination or a Lambertian reflectance model. However, real-world scenes often do not adhere to these assumptions.

To overcome the limitations imposed from single image analysis, the *photometric stereo* approach [134] uses a sequence of images from a static object with different illuminations. Usually, the direction of illumination is changed, while the viewpoint and camera parameters are kept constant. Thus, the corresponding image points between successive images are known a priori. Since there is no spatial displacement between corresponding points, this method relies on the observed radiance values which vary due to the illumination direction. Therefore, this method is called photometric stereo.

Over the years, both shape from shading and photometric stereo approaches have been widely studied and improved. We refer the interested reader to the survey of Zhang et al. [139] for a detailed overview.

Shape from Focus

In the *shape from focus* technique, surface texture is used to derive depth information. In particular, the goal is to analyze the amount of blur. As can be seen from Figure 3.6, a sharp image can only be obtained if the sensor plane coincides with the image plane, whereas the captured image yields different amounts of blur, depending on the sensor position. Thus, if the focal length f and the distance between the lens and the image plane d_I are known, then the object depth d_P (*i.e.*, the distance between the lens and the object) can be calculated using the *Gaussian lens formula*:

$$\frac{1}{f} = \frac{1}{d_P} + \frac{1}{d_I}. \tag{3.19}$$

Most shape from focus methods, such as [101], require an image sequence of a static object, captured using varying focus settings. Next, a focus

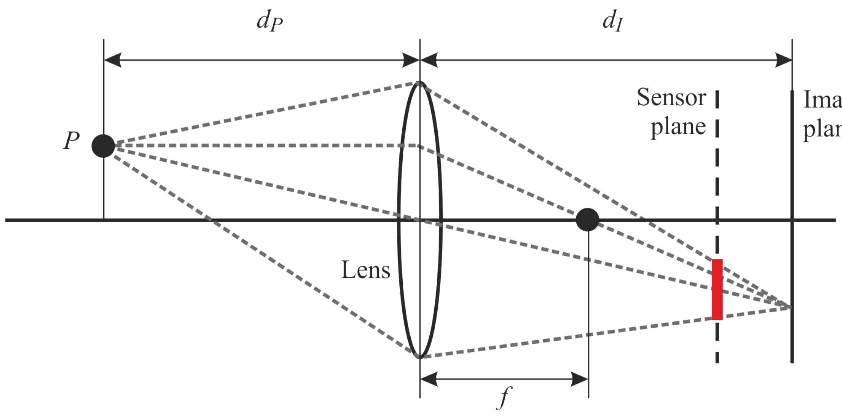


Figure 3.6: Shape from focus. The image of the 3D point P will be blurred if the sensor plane does not coincide with the image plane, as illustrated by the red spot on the sensor plane. By finding the focal length f and the distance d_I at which the projection is sharpest, the object depth d_P can be computed from the Gaussian lens law. Illustration based on [100].

measure (*e.g.*, Tenengrad [76] or sum-modified-Laplacian [100]) is computed to determine the focus quality within local neighborhoods. These measurements are used to determine at which focal length points are in focus. Then, the depth at the corresponding point can be computed directly from Equation (3.19).

Since shape from focus methods provide high-quality reconstruction, they are most commonly used within industrial applications, such as optical microscopy. A major drawback, however, is that most focus-based reconstruction methods require a considerable amount of measurement time and specific lenses are needed to accurately adjust the focus level.

3.2.2 Volumetric Reconstruction

In contrast to above methods, which mostly reconstruct 3D point clouds or depth information, alternative reconstruction techniques are able to produce volumetric reconstructions of a scene, *i.e.*, the scene is represented by a set of volume elements, commonly known as *voxels*. Most of these volumetric approaches are based on the principles of *shape from silhouette* [7, 89] or *voxel coloring* [79, 116].

Shape from silhouette approaches (*e.g.*, [72, 80, 89, 111, 123, 128]) rely on extracted (binary) silhouettes of foreground objects, *e.g.*, obtained via standard background subtraction techniques. The key idea is to reconstruct the scene structure by the intersection of viewing cones spanned by the foreground silhouettes. The reconstructed volume is also referred to as the visual hull [80, 81], *i.e.*, the maximal volume which generates the same silhouettes as observed in the camera views. Note, that the visual hull of an object is usually larger than the object itself, as illustrated in Figure 3.7.

A major drawback of the visual hull is that it cannot represent concavities of the objects, as the intersection of the viewing cones can only generate convex volumes. Therefore, Seitz and Dyer proposed the voxel coloring approach [116], which incorporates color information to overcome this limitation. The key idea is to project each voxel into the camera views and keep only those, which are *photo-consistent*, *i.e.*, the projection of a voxel must have the same color value in each view. Thus, in contrast to the visual hull, these methods reconstruct the *photo hull* [79], *i.e.*, the maximal volume which is photo-consistent with the input images. Although this allows to reconstruct concave objects, it introduces additional practical problems, *e.g.*, cameras should be color-calibrated to robustly compute the photo-consistency criterion.

Nevertheless, considering the tracking problem at hand, we consider shape from silhouette to be superior to voxel coloring. The major reason is that due to the camera placement in standard surveillance camera networks (*i.e.*, few cameras with wide base-lines), as well as dynamic occlusions caused by the objects themselves, voxel coloring may not reconstruct all objects in the scene. For example, if an object is occluded in a single view by another object with significantly different appearance, large parts of the object will be carved away due to the inconsistent colors of the re-projected voxels. On the other hand, shape from silhouette is able to handle such situations, as the visual hull contains all objects which correspond to the provided foreground silhouettes. Furthermore, in tracking applications, usually a convex approximation of the object of interest is sufficient, as we are interested in its position and not an exact reconstruction of its shape. Hence, the limitation that shape from silhouette cannot reconstruct object concavities does not severely affect the tracking accuracy.

Therefore, we base our proposed tracking approach on reconstructions of the objects' visual hull. Thus, in the following, we briefly summarize the shape from silhouette algorithm and discuss its most relevant limitations.

Shape from Silhouette

In order to summarize the visual hull reconstruction via shape from silhouette more formally, we use the following notation. The image observed from camera c is denoted by I_c, where the projection onto the image plane is defined by the camera's projection matrix $\mathbf{P}_c$. Furthermore, $S_c \subseteq I_c$ is the subset of pixels which belong to the foreground silhouette of the objects of interest. Then, the goal of shape from silhouette is to reconstruct the visual hull $\mathcal{V}_{\mathrm{VH}}$, *i.e.*, the set containing all voxels which belong to the visual hull of the segmented objects.

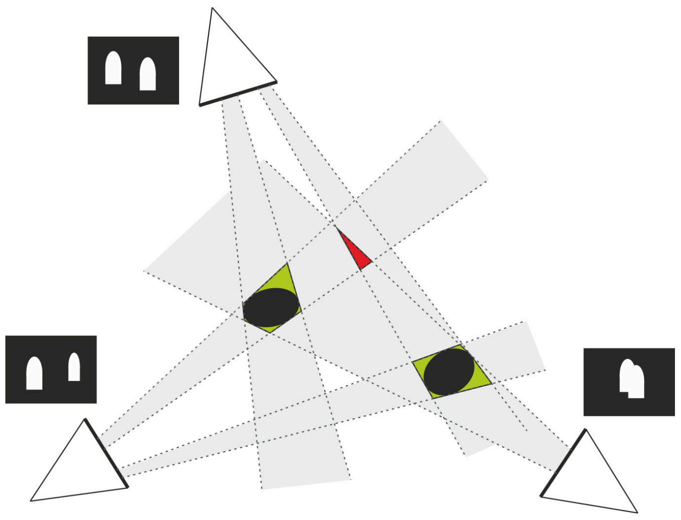


Figure 3.7: Reconstructing the visual hull (green) of objects (black) by intersecting the view cones (gray) which are spanned by the observed silhouettes. Note that the visual hull is usually larger than the real objects and cannot represent object concavities. Furthermore, this reconstruction method may introduce ghost artifacts (red), due to geometric ambiguities.

Therefore, the visual hull is first assumed to fill the whole reconstruction volume $\mathcal{V}$, *i.e.*, $\mathcal{V}_{\text{VH}} = \mathcal{V}$. Next, each voxel $v_i \in \mathcal{V}_{\text{VH}}$ is projected into each camera view, assuming the standard pinhole model (recall Section 3.1.1). The current voxel is now considered part of the visual hull if it projects into each foreground silhouette S_c. Otherwise, the voxel is removed from the set, *i.e.*, $\mathcal{V}_{\text{VH}} = \mathcal{V}_{\text{VH}} \setminus \{v_i\}$. After processing each voxel, $\mathcal{V}_{\text{VH}}$ contains the visual hull of the segmented objects. These steps to reconstruct the visual hull via shape from silhouette are also summarized in Algorithm 3.1.

Note that there are various efficient algorithms for computing the visual hull, *e.g.*, using octree representations [128], or plane-sweeps [79]. Furthermore, each voxel may be processed separately, *i.e.*, no information about neighboring voxels is required to compute the visual hull, and therefore, real-time capable implementations may also be achieved by exploiting this inherent parallelism, *e.g.*, by using a graphical processing unit (GPU).

Limitations

In practice, visual hull reconstructions obtained from volume intersection, such as shape from silhouette, suffer from several limitations. The most relevant, *i.e.*, camera placement and ghosting artifacts, will be summarized in the following.

Algorithm 3.1 Visual Hull Reconstruction.

```
Input: Camera matrices $\mathbf{P}_c$, foreground silhouettes $S_c$, volume $\mathcal{V}$
Output: Visual hull $\mathcal{V}_{\text{VH}}$

1. Set $\mathcal{V}_{\text{VH}} = \mathcal{V}$
2. For each $v_i \in \mathcal{V}$
    • Set $P_i = (X_i, Y_i, Z_i, 1)^\top$, i.e., the homogeneous coordinate vector of voxel $v_i$
    • Set $n = 0$
    • For $c = \{1, \dots, N_c\}$
        – Project $v_i$ into camera view:
          $(\tilde{x}_c, \tilde{v}_c, \tilde{w}_c)^\top = \mathbf{P}_c P_i, \qquad x_c = \frac{\tilde{x}_c}{\tilde{w}_c}, \qquad y_c = \frac{\tilde{y}_c}{\tilde{w}_c}$
        – If $(x_c, y_c)^\top \in S_c$:
              $n = n + 1$
    • If $n < N_c$: $\mathcal{V}_{\text{VH}} = \mathcal{V}_{\text{VH}} \setminus \{v_i\}$, i.e., carve the voxel away if it does not project into each silhouette
```

Camera Placement and Calibration. In order to estimate the object pose from its visual hull, one has to consider the effects of various camera placements. As can be seen from Figure 3.8, the relative position between the cameras has a significant impact on the size of the visual hull. Therefore, one has to take care of the camera placement when setting up camera networks used for applications which are based on visual hull reconstructions, such as the proposed tracking approach.

Another limitation of the original shape from silhouette approach, as pointed out by [93], is that it can only reconstruct objects which are visible in each camera view. However, this issue can easily be handled by minor modifications to the original algorithm, as will be discussed in Chapter 4.

Furthermore, shape from silhouette requires a fully calibrated camera network, *i.e.*, both intrinsic and extrinsic camera parameters must be known. However, as discussed in Section 3.1.3, there are many methods which provide a sufficiently accurate estimate of the camera parameters. Note however, that the camera parameters may change over time, *e.g.*, if the mounting platform moves relative to the scene. In such situations, one has to ensure that the current camera calibration is always available in order to apply shape from silhouette.

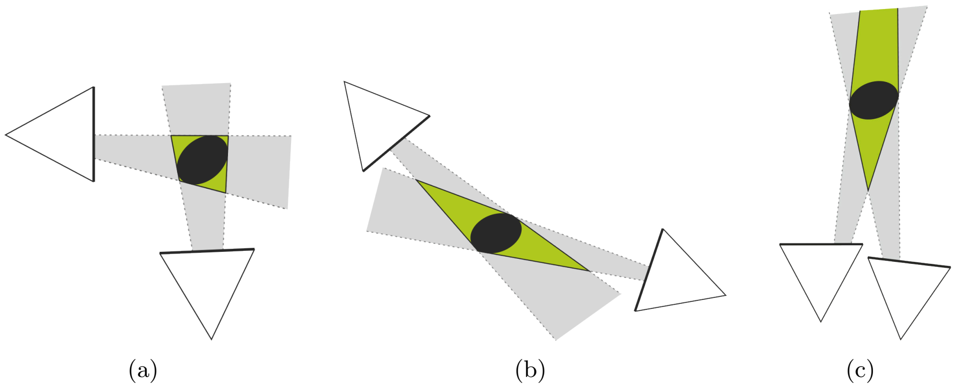


Figure 3.8: Influence of camera placement on the quality of the reconstructed visual hull. If the fields-of-view are nearly orthogonal (a), the visual hull provides a valuable cue for locating the object. However, if the cameras are placed on opposite sides (b) or share the same view point (c), the visual hull is significantly larger than the object, which complicates the exact localization.

Ghosting Artifacts. Visual hull reconstructions may also contain voxels which do not belong to an object, *e.g.*, as illustrated in Figure 3.7. These *ghost* artifacts, also termed *phantom volumes*, are caused by geometric uncertainties of the volume intersection due to noise of the foreground segmentation, or the scene layout itself. An obvious solution to resolve these ambiguities would be to include additional view points. However, due to several restrictions, *e.g.*, hardware costs, this option may not be feasible in every application. Therefore, several post-processing approaches have been proposed to reduce ghost artifacts, *e.g.*, *safe hulls* [96], *real shape hulls* [93], *cleaned visual hulls* [94], or occluder reasoning using Bayesian inference [45, 46].

However, since multiple object configurations may yield the same foreground silhouettes, determining real object volumes from silhouette information (*i.e.*, [93, 94, 96]) is an under-constrained problem. Therefore, these correction methods may remove parts which actually belong to the objects. On the other hand, color-based corrections, such as [45, 46], are highly time-consuming (*e.g.*, more than one minute per frame) and may still fail in situations, where objects have similar appearance.

In contrast to these methods, we introduce an *occupancy volume* which can be derived from the standard visual hull by analyzing local mass densities. As will be discussed in Chapter 4, this provides a valuable cue for tracking and allows to robustly handle both, ghosting artifacts, as well as holes in the visual hull caused by noisy foreground silhouettes.

3.3 Recursive Bayesian Filters

The solution to many scientific problems is based on estimating the state of a system which changes over time. The exact solution is in general unknown, since for most systems only noisy measurements can be obtained, *e.g.*, when using visual input data for object tracking. However, such dynamic systems can be modeled by several methods, such as the *state-space* approach, which allows to estimate the state of the underlying system based on the measurements. A major advantage of this approach is its ability to efficiently handle multivariate data, as well as nonlinear and non-Gaussian processes [133].

In order to analyze dynamic systems using the state-space approach, two models are required, *i.e.*, a *system* or *process model* which describes the evolution of the states over time, as well as a *measurement model* which relates the observed measurements to the system state. Using a Bayesian framework for the dynamic state estimation, the goal is to construct the posterior probability density function (pdf) of the system state based on all available information, including the obtained observations. In principle, an optimal estimate of the state may be inferred from the pdf, as it embodies all statistical information about the system [4].

Considering object tracking, state estimates are usually required frequently, *e.g.*, every time a new observation is available. Therefore, a recursive filtering approach provides a convenient solution, as the observation data can be processed sequentially rather than as a batch. One well-known solution for the state estimation is the *recursive Bayesian filter*, which is widely used for visual tracking, since one of its major advantages is the ability to robustly handle several kinds of uncertainties, *i.e.*, associated with the observations (*e.g.*, noisy measurements from visual input), as well as system dynamics (*e.g.*, motion models do not fit human movement perfectly).

Over the years, many tracking approaches have successfully applied efficient solutions, such as Kalman filtering [62]. However, a major drawback of Kalman filter approaches is that these assume linear system models with Gaussian-distributed noise, which in general is not fulfilled by real-world systems. To overcome these limitations, several approaches, such as [59, 69, 75, 124], rely on sequential Monte Carlo methods, also known as *particle filtering.* Therefore, we also apply a particle filter to form the tracking basis of our proposed approach, as it allows to estimate the state of nonlinear and non-Gaussian systems, which better suits real-world applications.

In the following, we discuss the theory of tracking within a Bayesian framework following the literature [4, 26, 27, 59, 74]. Therefore, in Section 3.3.1, we formulate visual tracking as a dynamic state estimation problem and show how an optimal solution can be derived, *i.e.*, the recursive Bayes filter. Second, we detail the particle filtering approach, which allows for efficiently solving this estimation problem, in Section 3.3.2.

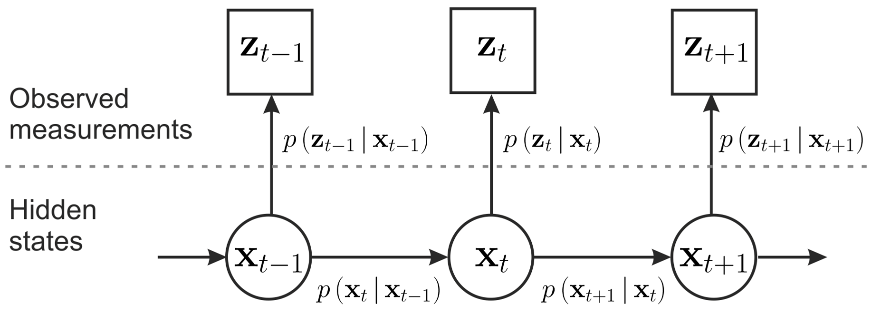

$\mathbf{z}_{t-1}$
$\mathbf{z}_t$
$\mathbf{z}_{t+1}$
Observed measurements
$p(\mathbf{z}_{t-1} \mid \mathbf{x}_{t-1})$
$p(\mathbf{z}_t \mid \mathbf{x}_t)$
$p(\mathbf{z}_{t+1} \mid \mathbf{x}_{t+1})$
Hidden states
$\mathbf{x}_{t-1}$
$\mathbf{x}_t$
$\mathbf{x}_{t+1}$
$p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$
$p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)$

Figure 3.9: Bayesian network of a hidden Markov model. Assuming a first order Markov process, the evolution of the hidden state $\mathbf{x}_t$ of the stochastic dynamic system can be modeled by the conditional probability density $p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$. The corresponding measurements $\mathbf{z}_t$ can be observed according to the probability density function $p(\mathbf{z}_t \mid \mathbf{x}_t)$. Illustration based on [74].

3.3.1 Stochastic Estimation and Recursive Solution

The major goal of object tracking is to obtain information, such as location, about possibly moving targets over time. Thus, tracking can be formulated as an state-space approach, *i.e.*, at every time step, the object state (*e.g.*, its location) must be estimated from the available observations (*e.g.*, by analyzing camera images). More formally, the state of an object at time t is denoted by the random variable $\mathbf{x}_t$. Therefore, starting from the initial object state $\mathbf{x}_0$, the entire sequence of states up to the current is defined as the trajectory $\mathbf{x}_{0:t} = \{\mathbf{x}_0, \dots, \mathbf{x}_t\}$.

In general, the underlying dynamic system is assumed to be a Markov process. Furthermore, the true state $\mathbf{x}_t$ cannot be observed directly. However, we can obtain measurements $\mathbf{z}_t$ from the dynamic system which can be used to reason about the object state, *e.g.*, considering tracking, visual input data is available to estimate the objects' locations. These measurements are also considered random variables due to the uncertainties introduced by the observation process itself. In order to estimate the object state using above hidden Markov model (HMM), a well-known solution is to embed it in a Bayesian network, as illustrated in Figure 3.9. In this Bayesian approach, all information about the state sequence $\mathbf{x}_{0:t}$ can be inferred from the posterior distribution $p(\mathbf{x}_{0:t} \mid \mathbf{z}_{1:t})$ if the prior distribution $p(\mathbf{x}_0)$ is known. The posterior distribution denotes the probability of state sequences $\mathbf{x}_{0:t}$ given all observations $\mathbf{z}_{1:t}$. Note that initially, no observations are obtained, *i.e.*, $\mathbf{z}_0 = \emptyset$ and thus, is usually ignored.

Using above notation, the state-space approach to object tracking can be formally defined by the system model

$$\mathbf{x}_t = \mathbf{f}_t\left(\mathbf{x}_{t-1}, \mathbf{v}_{t-1}\right), \tag{3.20}$$

and the measurement model

$$\mathbf{z}_t = \mathbf{g}_t\left(\mathbf{x}_t, \mathbf{n}_t\right), \tag{3.21}$$

where $\mathbf{f}_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \rightarrow \mathbb{R}^{n_x}$ and $\mathbf{g}_t : \mathbb{R}^{n_x} \times \mathbb{R}^{n_n} \rightarrow \mathbb{R}^{n_z}$ are possibly nonlinear functions, $\{\mathbf{v}_{t-1} \mid t \in \mathbb{N}\}$ is the i.i.d. process noise sequence, $\{\mathbf{n}_t \mid t \in \mathbb{N}\}$ is the i.i.d. measurement noise sequence, and n_x, n_v, n_n, and n_z are the dimensions of the state vector, of the process noise vector, of the measurement noise vector, and of the measurement vector, respectively.

For tracking objects, in general only the current state $\mathbf{x}_t$ of the object at time t is required. Thus, we are only interested in estimating the marginal distribution $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$. Furthermore, it would be beneficial to compute this solution in a recursive manner, as analyzing the whole state sequence for a single estimate would be computationally inefficient. Fortunately, the posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$ may be computed recursively using the previous posterior $p(\mathbf{x}_{t-1} \,|\, \mathbf{z}_{1:t-1})$ in combination with the current observation $\mathbf{z}_t$, which is referred to as *recursive Bayesian filtering.*

In order to obtain a recursive solution for computing the marginal distribution $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$, Bayes' rule is applied to the complete posterior $p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t})$, resulting in:

$$p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}) = \frac{p(\mathbf{x}_{0:t}, \mathbf{z}_{1:t})}{p(\mathbf{z}_{1:t})}. \tag{3.22}$$

Now applying the chain rule to factorize both, the numerator and denominator, *i.e.*,

$$p(\mathbf{x}_{0:t}, \mathbf{z}_{1:t}) = p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}, \mathbf{x}_{0:t}) p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t-1}) p(\mathbf{z}_{1:t-1}), \tag{3.23}$$

and

$$p(\mathbf{z}_{1:t}) = p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}) p(\mathbf{z}_{1:t-1}), \tag{3.24}$$

respectively, Equation (3.22) can be reformulated to:

$$p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}, \mathbf{x}_{0:t}) p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t-1})}{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})}. \tag{3.25}$$

To allow for recursion, the state transition is applied, *i.e.*, $p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t})$ is computed from the previous complete posterior $p(\mathbf{x}_{0:t-1} \,|\, \mathbf{z}_{1:t-1})$ as:

$$p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}, \mathbf{x}_{0:t}) p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathbf{z}_{1:t-1}) p(\mathbf{x}_{0:t-1} \,|\, \mathbf{z}_{1:t-1})}{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})}. \tag{3.26}$$

Marginalizing the complete posterior $p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t})$ in Equation (3.26) over the past states $\mathbf{x}_{0:t-1}$ now gives the posterior of the current state $\mathbf{x}_t$:

$$p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) = \int p(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}) \mathrm{d}\mathbf{x}_{0:t-1} \tag{3.27}$$

$$= \frac{\int p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}, \mathbf{x}_{0:t}) p(\mathbf{x}_t \,|\, \mathbf{x}_{0:t-1}, \mathbf{z}_{1:t-1}) p(\mathbf{x}_{0:t-1} \,|\, \mathbf{z}_{1:t-1}) \mathrm{d}\mathbf{x}_{0:t-1}}{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})} \tag{3.28}$$

$$= \frac{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}, \mathbf{x}_t) \int p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathbf{z}_{1:t-1}) p(\mathbf{x}_{t-1} \,|\, \mathbf{z}_{1:t-1}) \mathrm{d}\mathbf{x}_{t-1}}{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})}. \tag{3.29}$$

Note that in general, the solution of Equation (3.29) cannot be computed recursively, since it depends on the sequence of all observations, *i.e.*, $\mathbf{z}_{1:t-1}$. Therefore, to obtain a proper recursion, two additional assumptions have to be imposed, in particular

$$p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}, \mathbf{x}_t) \triangleq p(\mathbf{z}_t \,|\, \mathbf{x}_t), \tag{3.30}$$

and

$$p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathbf{z}_{1:t-1}) \triangleq p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}). \tag{3.31}$$

The first assumption (Equation (3.30)) states that given the state $\mathbf{x}_t$, the corresponding observation $\mathbf{z}_t$ is conditionally independent from all previous observations $\mathbf{z}_{1:t-1}$, whereas the second assumption (Equation (3.31)) requires that given the previous state $\mathbf{x}_{t-1}$, the current state $\mathbf{x}_t$ is conditionally independent from all previous observations $\mathbf{z}_{1:t-1}$.

These assumptions imply that the state sequence is a first-order Markov process. Thus, Equation (3.29) can finally be rewritten to obtain the *recursive Bayesian filter*:

$$p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t \,|\, \mathbf{x}_t) \int p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1} \,|\, \mathbf{z}_{1:t-1}) \mathrm{d}\mathbf{x}_{t-1}}{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})}. \tag{3.32}$$

In general, the posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$ is obtained recursively in two steps, namely *prediction* and *update*. In the prediction stage, the system model (Equation (3.20)) is used to obtain the pdf of the state $\mathbf{x}_t$ via the Chapman-Kolmogorov relation:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) \mathrm{d}\mathbf{x}_{t-1}. \tag{3.33}$$

Next, the update stage uses the observation $\mathbf{z}_t$, which becomes available at time t, to update the pdf and to obtain the current posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$ via Bayes' rule, as

$$p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t \,|\, \mathbf{x}_t) p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t-1})}{p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})}, \tag{3.34}$$

where the normalization constant $p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})$ is calculated by integrating the numerator over all values of the current state:

$$p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1}) = \int p(\mathbf{z}_t \,|\, \mathbf{x}_t) p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t-1}) \mathrm{d}\mathbf{x}_t. \tag{3.35}$$

Furthermore, $p(\mathbf{z}_t \,|\, \mathbf{z}_{1:t-1})$ is constant relative to the true system state, and therefore, is usually ignored in practice.

Although in principle, an optimal estimate of the current state may be obtained from the pdf using above recursive propagation, it generally cannot be determined analytically. Fortunately, several approaches have been proposed to approximate the optimal Bayesian solution, including the well-known sequential Monte Carlo method. Therefore, in the following section, we describe how this method can be used to solve the recursive propagation of the posterior density (*i.e.*, Equations 3.33 and 3.34).

3.3.2 Sequential Monte Carlo Methods

Common approaches to solve the recursive probability propagation, such as particle filters, are based on sequential Monte Carlo sampling. These methods exploit the simplifying assumption that the continuous state-space is finite and can be discretized into N_s samples. The key idea is to represent the posterior pdf by a finite set of random samples with associated weights, from which the true posterior can be estimated. Following the *law of large numbers*, this Monte Carlo approximation becomes an equivalent representation of the continuous posterior probability when choosing a large sample size [4, 27]. Thus, the sequential Monte Carlo methods approach the optimal Bayesian solution if very large N_s are used.

In the following, we provide a summary of how this discrete approximation can be used to derive the well-known particle filters. Therefore, we first describe sequential importance sampling, which forms the basis of most particle filtering approaches. Second, we discuss the problem of degeneracy, as well as its solutions, since this problem may occur in practice if sequential importance sampling is used. Finally, we conclude this overview on recursive Bayesian filters by deriving the bootstrap particle filter, which is also used in our proposed tracking algorithm.

Importance Sampling

In order to approximate the continuous Bayesian solution by Monte Carlo sampling, a random measure is introduced which characterizes the posterior probability $p\left(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}\right)$. In particular, this random measure, commonly referred to as the *particles*, is defined as

$$\left\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\right\}_{i=1}^{N_s}, \tag{3.36}$$

and consists of N_s support points $\mathbf{x}_{0:t}^{(i)}$ along with the corresponding weights $w_t^{(i)}$, which are normalized such that $\sum_{i=1}^{N_s} w_t^{(i)} = 1$. Using these definitions, the true posterior pdf may be approximated by

$$p\left(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}\right) \approx p_{N_s}\left(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}\right) = \sum_{i=1}^{N_s} w_t^{(i)} \delta\left(\mathbf{x}_{0:t} - \mathbf{x}_{0:t}^{(i)}\right), \tag{3.37}$$

where $\delta(\cdot)$ is the Dirac delta measure.

Following [4, 10], the weights are chosen using the *importance sampling* principle as follows. Suppose that $p(x) \propto \pi(x)$ is a probability distribution function which is difficult to sample from, but may be evaluated up to a proportionality constant, *i.e.*, $\pi(x)$ can be evaluated. Additionally, a distribution function $q(x)$ is introduced, which is easy to draw samples from and has the same support as $p(x)$, *i.e.*, $q(x) \neq 0, \forall x : p(x) \neq 0$. This distribution is commonly referred to as *importance function* or *proposal distribution*, respectively.

Given above definitions, one may now easily draw samples from the importance function, *i.e.*, $x^{(i)} \sim q(x), i = 1, \ldots, N_s$ to obtain a weighted approximation of the pdf $p(x)$ as

$$p(x) \approx \sum_{i=1}^{N_s} w^{(i)} \delta \left(x - x^{(i)} \right), \tag{3.38}$$

where the normalized weight of the i^{th} particle is computed as:

$$w^{(i)} \propto \frac{\pi \left(x^{(i)} \right)}{q \left(x^{(i)} \right)}. \tag{3.39}$$

The importance sampling principle may now be applied to the discrete approximation of the true posterior pdf in Equation (3.37), *i.e.*, the samples $\mathbf{x}_{0:t}^{(i)}$ are drawn from an importance function $q(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t})$. Thus, following Equation (3.39), the weights can be defined as:

$$w_t^{(i)} = \frac{p \left(\mathbf{x}_{0:t}^{(i)} \,|\, \mathbf{z}_{1:t} \right)}{q \left(\mathbf{x}_{0:t}^{(i)} \,|\, \mathbf{z}_{1:t} \right)}. \tag{3.40}$$

Note that usually, the measurements $\mathbf{z}_t$ are received sequentially. Thus, in order to allow for a recursive solution, the importance function must be chosen such that it factorizes into a recursive form, in general:

$$q(\mathbf{x}_{0:t} \,|\, \mathbf{z}_{1:t}) = q(\mathbf{x}_t \,|\, \mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})\, q(\mathbf{x}_{0:t-1} \,|\, \mathbf{z}_{1:t-1}). \tag{3.41}$$

Thus, one obtains the i^{th} sample $\mathbf{x}_{0:t}^{(i)}$ recursively by concatenation of the newly sampled state $\mathbf{x}_t^{(i)}$ and the existing samples $\mathbf{x}_{0:t-1}^{(i)}$, *i.e.*,

$$\mathbf{x}_{0:t}^{(i)} = \left\{ \mathbf{x}_t^{(i)}, \mathbf{x}_{0:t-1}^{(i)} \right\}, \tag{3.42}$$

$$\mathbf{x}_t^{(i)} \sim q\left(\mathbf{x}_t \,|\, \mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t} \right), \tag{3.43}$$

$$\mathbf{x}_{0:t-1}^{(i)} \sim q(\mathbf{x}_{0:t-1} \,|\, \mathbf{z}_{1:t-1}), \tag{3.44}$$

where the corresponding weights are computed from Equation (3.40).

In order to obtain the corresponding weight update equation, the posterior probability is first factored as

$$p\left(\mathbf{x}_{0:t}^{(i)} \,|\, \mathbf{z}_{1:t} \right) \propto p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)} \right) p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)} \right) p\left(\mathbf{x}_{0:t-1}^{(i)} \,|\, \mathbf{z}_{1:t-1} \right), \tag{3.45}$$

which allows to define the weight update equation as:

$$w_t^{(i)} \propto \frac{p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t}\right)} \frac{p\left(\mathbf{x}_{0:t-1}^{(i)} \,|\, \mathbf{z}_{1:t-1}\right)}{q\left(\mathbf{x}_{0:t-1}^{(i)} \,|\, \mathbf{z}_{1:t-1}\right)}. \tag{3.46}$$

Furthermore, this update relation can be rewritten to

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t}\right)}, \tag{3.47}$$

since the rightmost fraction of Equation (3.46) is the weight of the i^{th} particle from the previous time step.

Above formulation is not fully recursive, since it is conditioned on the entire sequence of observations $\mathbf{z}_{1:t}$. In order to obtain a proper recursion, the importance function is usually assumed to be

$$q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t}\right) = q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right), \tag{3.48}$$

which can be applied if only a filtered estimate of $p\left(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}\right)$ is required at each time step. In particular, this is the case for object tracking, as discussed earlier in Section 3.3.1. Furthermore, above modification allows to reduce the memory consumption since both, the trajectory $\mathbf{x}_{0:t-1}^{(i)}$, as well as the previous observations $\mathbf{z}_{1:t-1}$, may be discarded.

Applying this modification leads to the new weight update equation

$$\tilde{w}_t^{(i)} \propto w_{t-1}^{(i)} \frac{p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)}, \tag{3.49}$$

where the weights are not normalized, *i.e.*, $\sum_{i=1}^{N_s} \tilde{w}_t^{(i)} \neq 1$. Therefore, normalization is usually performed in a subsequent step:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{i=1}^{N_s} \tilde{w}_t^{(i)}}. \tag{3.50}$$

Given the modified weight update, the true filtered posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$ can now be discretely approximated as:

$$p\left(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}\right) \approx p_{N_s}(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) = \sum_{i=1}^{N_s} w_t^{(i)} \delta\left(\mathbf{x}_t - \mathbf{x}_t^{(i)}\right). \tag{3.51}$$

This finally leads to the well-known *sequential importance sampling* (SIS) method, summarized in Algorithm 3.2, where the key idea is to recursively

Algorithm 3.2 Sequential Importance Sampling.

```
Input: Previous posterior $p(\mathbf{x}_{t-1} \,|\, \mathbf{z}_{1:t-1}) \approx \left\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{N_s}$
Output: Current posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) \approx \left\{\mathbf{x}_t^{(i)}, w_t^{(i)}\right\}_{i=1}^{N_s}$

1. For $i = \{1, \dots, N_s\}$
    • Sample a new particle: $\mathbf{x}_t^{(i)} \sim q\left(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)$
    • Assign the weight: $\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \frac{p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)}$
2. For $i = \{1, \dots, N_s\}$
    • Normalize weights: $w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N_s} \tilde{w}_t^{(j)}}$
```

propagate the weights $w_t^{(i)}$ and support points $\mathbf{x}_t^{(i)}$ as each observation $\mathbf{z}_t$ is received sequentially. Furthermore, since the posterior probability $p\left(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}\right)$ is approximated by a weighted sample set, the true target state $\mathbf{x}_t$ can be estimated as the weighted average:

$$\mathbf{x}_t \approx \sum_{i=1}^{N_s} \mathbf{x}_t^{(i)} w_t^{(i)}. \tag{3.52}$$

Degeneracy Phenomenon

Implemented as shown in Algorithm 3.2, the sequential importance sampling yields a major drawback, known as *degeneracy* of the particle set. Due to the repeated multiplication of particle weights from previous iterations these weights decrease rapidly and approach zero. In practice, after a few iterations all but one particles have negligible weights. On the one hand, this introduces numerical problems due to limited floating point precision in computer systems. On the other hand, most of the computational effort is spent updating particles, which practically no longer effect the approximation of the posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$.

A commonly used measure to determine the degree of degeneracy is the *effective sample size*, introduced by [10, 84] as

$$N_{\text{eff}} = \frac{N_s}{1 + \operatorname{Var}\left(\overset{*}{w}{}_t^{(i)}\right)}, \tag{3.53}$$

which depends on the variance of the *true weight*:

$$\overset{*}{w}{}_t^{(i)} = \frac{p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{z}_{1:t}\right)}{q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)}. \tag{3.54}$$

A severe degeneracy of the particle set may now be detected if N_{eff} becomes very small. Note that in practice, N_{eff} cannot be computed exactly and therefore, is usually approximated by

$$N_{\text{eff}} \approx \hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_s} \left(w_t^{(i)}\right)^2}, \tag{3.55}$$

where $w_t^{(i)}$ are the normalized weights, obtained by Equation (3.47).

Solutions to the Degeneracy Phenomenon

In order to reduce the effects of the degeneracy phenomenon, the straightforward approach would be to choose a very large number of particles N_s. However, as this is impractical for most applications, two alternative solutions have been established, *i.e.*, choosing a suitable importance function, or resampling the set of particles before the approximation of the posterior deteriorates. In the following, we briefly summarize these two solutions.

Good Choice of the Importance Function. When choosing a suitable importance function $q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t)$, the goal is to minimize the variance of the true weights such that N_{eff} will be maximized. Doucet et al. [26] have shown that the optimal importance function is

$$q_{\text{opt}}\left(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right) = p\left(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right), \tag{3.56}$$

since for any given $\mathbf{x}_{t-1}^{(i)}$, the corresponding weight $w_t^{(i)}$ takes the same value, independent from the sample drawn from $q_{\text{opt}}\left(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)$. This implies that conditioned on $\mathbf{x}_{t-1}^{(i)}$:

$$\text{Var}\left(\overset{*}{w}{}_t^{(i)}\right) = 0. \tag{3.57}$$

Although this choice yields the optimal importance function, there are several major drawbacks to consider, *e.g.*, one needs to be able to sample from $p\left(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)$, which cannot be done in general. Therefore, this optimal choice can only be applied in specific scenarios, *i.e.*, if the state-space of $\mathbf{x}_t$ is finite and discrete, or if the measurement model is linear and both, the system and measurement noise, can be modeled by a Gaussian distribution.

Note that there exists a multitude of suitable importance functions and that the performance of the filtering approach significantly depends on a proper choice. Therefore, we refer to the excellent summaries by Arulampalam et al. [4] and Kristan [74] for a more detailed discussion on suitable importance functions.

Resampling Particles. The basic idea of resampling is to discard particles with low weights and to increase the number of particles with high weights as soon as a significant degeneracy is observed, *i.e.*, when N_{eff} is below a predefined threshold N_τ. In particular, resampling methods generate a new particle set, denoted as

$$\left\{\tilde{\mathbf{x}}_t^{(i)}\right\}_{i=1}^{N_s}, \tag{3.58}$$

by resampling with replacement from the discrete approximation of the posterior, *i.e.*, $p_{N_s}(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$, recall Equation (3.51). The resampling step is performed such that the conditional probability of each state $\mathbf{x}_t^{(i)}$ is $w_t^{(i)}$, denoted by:

$$\Pr\left(\tilde{\mathbf{x}}_t^{(i)} = \mathbf{x}_t^{(i)}\right) = w_t^{(i)}. \tag{3.59}$$

Since the new set is in fact an i.i.d. sample from the discrete probability density $p_{N_s}(\mathbf{x}_t \,|\, \mathbf{z}_{1:t})$, the weights can be reset to $w_t^{(i)} = 1/N_s$. Therefore, resampling can be described by the mapping:

$$\left\{\mathbf{x}_t^{(i)}, w_t^{(i)}\right\}_{i=1}^{N_s} \rightarrow \left\{\tilde{\mathbf{x}}_t^{(i)}, \frac{1}{N_s}\right\}_{i=1}^{N_s}. \tag{3.60}$$

Over the decades, many resampling methods have been proposed, such as *deterministic resampling* [86], *stratified sampling* [84], *regularized sampling* [36], *systematic resampling* [71], or *residual sampling* [84]. Similar to [74], we prefer the deterministic resampling approach, since it is easy to implement and its runtime complexity is $\mathcal{O}(N_s)$. Algorithm 3.3 lists the pseudo-code for this resampling strategy.

In practice, when applying resampling, one has to consider the *sample impoverishment* problem, *i.e.*, particles with high weights will be selected many times and therefore, the diversity among the particles decreases as the drawn sample will contain many repeated points. This problem becomes critical in scenarios, where the process noise is very small and the system states are static. In such situations, all particles collapse to a single point after a few iterations. Therefore, according to [4], particle filters should in general be avoided if the process noise is zero. However, since object tracking usually deals with highly nonlinear systems and dynamic states, particle filtering approaches are well suited for the state estimation in such scenarios.

Algorithm 3.3 Deterministic Resampling.

```
Input: Posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) \approx \left\{\mathbf{x}_t^{(i)}, w_t^{(i)}\right\}_{i=1}^{N_s}$
Output: Resampled posterior $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}) \approx \left\{\tilde{\mathbf{x}}_t^{(i)}, \frac{1}{N}\right\}_{i=1}^{N_s}$

1. Generate cumulative distribution $\left\{c^{(i)}\right\}_{i=1}^{N_s}$, i.e., $c^{(i)} = \sum_{j=1}^{i} w_t^{(j)}$
2. Set $l = 1$
3. For $i = \{1, \dots, N_s\}$
    • While $\left(\frac{i}{N_s} > c^{(l)}\right)$:
          $l = l + 1$
    • Choose $\tilde{\mathbf{x}}_t^{(i)} = \mathbf{x}_t^{(l)}$
    • Set $w_t^{(i)} = \frac{1}{N_s}$
```

Particle Filters

If the sequential importance sampling (Algorithm 3.2) is combined with a resampling strategy, *e.g.*, deterministic resampling (Algorithm 3.3), and furthermore, the effective sample size is used to decide about resampling the particles, one obtains the well-known *generic particle filter*, as summarized in Algorithm 3.4.

Furthermore, if two additional simplifications are imposed, one obtains the widely used *bootstrap particle filter* [44], which is also commonly known as CONDENSATION algorithm [59], or *survival of the fittest* [63]. In particular, the resampling threshold is set to $N_\mathcal{T} = \infty$, *i.e.*, the particles are resampled at every iteration, and the prior probability $p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1})$ is used as the importance function $q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathbf{z}_t)$. Therefore, the weight update rule can be rewritten as:

$$w_t^{(i)} \propto w_{t-1}^{(i)} p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right). \tag{3.61}$$

However, since resampling is done at each time step, the previous weights are $w_{t-1}^{(i)} = 1/N_s$, and thus, the update rule can be simplified to:

$$w_t^{(i)} \propto p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right). \tag{3.62}$$

This particle filter, summarized in Algorithm 3.5, is also used as the basis of the location estimation within our proposed approach.

Algorithm 3.4 Generic Particle Filter.

```
**Input:** Previous posterior $p(\mathbf{x}_{t-1} \,|\, \mathbf{z}_{1:t-1}) \approx \left\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{N_s}$
**Output:** Current posterior $p(\mathbf{x}_{t} \,|\, \mathbf{z}_{1:t}) \approx \left\{\mathbf{x}_{t}^{(i)}, w_{t}^{(i)}\right\}_{i=1}^{N_s}$

1. If $\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_s}\left(w_t^{(i)}\right)^2} < N_\tau$:
    • Resample the input particle set using Algorithm 3.3:
      $\left\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{N_s} \to \left\{\tilde{\mathbf{x}}_{t-1}^{(i)}, \frac{1}{N_s}\right\}_{i=1}^{N_s}$
2. For $i = \{1, \ldots, N_s\}$
    • Sample a new particle: $\mathbf{x}_t^{(i)} \sim q\left(\mathbf{x}_t \,|\, \tilde{\mathbf{x}}_{t-1}^{(i)}, \mathbf{z}_t\right)$
    • Assign the weight: $\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \frac{p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t^{(i)} \,|\, \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_t\right)}$
3. Normalize weights, *i.e.*, compute $w_t^{(i)}$ from Equation (3.50)
```

Algorithm 3.5 Bootstrap Particle Filter.

```
**Input:** Previous posterior $p(\mathbf{x}_{t-1} \,|\, \mathbf{z}_{1:t-1}) \approx \left\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{N_s}$
**Output:** Current posterior $p(\mathbf{x}_{t} \,|\, \mathbf{z}_{1:t}) \approx \left\{\mathbf{x}_{t}^{(i)}, w_{t}^{(i)}\right\}_{i=1}^{N_s}$

1. Resample $\left\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{N_s} \to \left\{\tilde{\mathbf{x}}_{t-1}^{(i)}, \frac{1}{N_s}\right\}_{i=1}^{N_s}$ using Algorithm 3.3
2. For $i = \{1, \ldots, N_s\}$
    • Sample a new particle: $\mathbf{x}_t^{(i)} \sim p\left(\mathbf{x}_t \,|\, \tilde{\mathbf{x}}_{t-1}^{(i)}\right)$
    • Assign the weight: $\tilde{w}_t^{(i)} = p\left(\mathbf{z}_t \,|\, \mathbf{x}_t^{(i)}\right)$
3. For $i = \{1, \ldots, N_s\}$
    • Normalize weights: $w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N_s} \tilde{w}_t^{(j)}}$
```

Chapter 4

Multiple Object Tracking by Volumetric Reconstruction

Contents

In the following chapter, we present our multiple object, multiple camera tracking approach. The key idea of this robust real-time tracker, which we refer to as R2T2[4], is to use geometric information derived from volumetric reconstructions as a primary localization cue. Whenever this information becomes unreliable, *i.e.*, objects move close to each other, we additionally exploit on-line collected appearance information to robustly track individuals without identity switches in complex scenes.

A major advantage of volumetric representations, such as the visual hull obtained via shape from silhouette, is that they impose no assumptions about scene planarity, *e.g.*, no common ground-plane is assumed, and thus are perfectly suited for tracking scenarios where the objects of interest exhibit challenging poses or out-of-plane motion. Nevertheless, the visual hull reconstruction is very sensitive to noisy background subtraction, *i.e.*, missing foreground segmentations may cause holes in the reconstruction, while

[4]"A Jedi must have the most serious mind." - Yoda

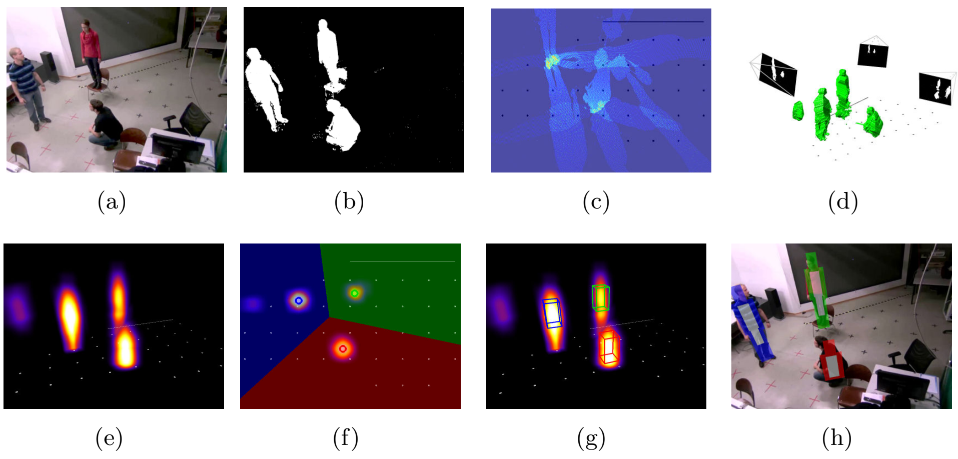


Figure 4.1: Planar projections on background subtracted input images (a,b) may introduce severe artifacts (c) and cannot handle out-of-plane motion, *e.g.*, the person standing on the chair. By exploiting the visual hull (d) to obtain an occupancy volume (e), we overcome these limitations. Furthermore, we split the tracking into two steps: first, we estimate the (x, y) coordinates using particle filters and Voronoi partitioning (f), followed by locating the vertical mass center (g). The re-projected tracking result is shown in (h).

false positive segmentations may introduce additional phantom volumes. Therefore, related tracking approaches based on visual hull reconstructions additionally rely on appearance information – either to locate the objects despite noisy reconstructions (*i.e.*, [82]), or to improve the visual hull reconstruction (*i.e.*, [46]). However, these methods require appearance information available at any time. Additionally, improving the visual hull reconstruction is a highly time-consuming task.

In contrast to these approaches, we derive an *occupancy volume* from the visual hull reconstruction, which is less sensitive to noisy foreground segmentation. Therefore, we analyze local mass densities of the reconstruction volume. As will be shown in Section 4.1, this provides a valuable cue for locating the objects. The actual tracking step is then performed using a particle filtering approach in combination with Voronoi partitioning, as presented in Section 4.2. This allows for efficiently tracking multiple objects solely based on the geometric information obtained via the occupancy volume, as illustrated in Figure 4.1.

Furthermore, we exploit the 3D scene structure to collect samples for each individual on-line, *i.e.*, samples are extracted whenever the corresponding object is fully visible in a camera view. As discussed in Section 4.3, we resort to this appearance information solely on-demand, *i.e.*, whenever the primary geometric cue becomes unreliable. This provides an additional benefit, as we do not require appearance information to be available for each individual at every frame. Finally, in Section 4.4, we demonstrate how the occupancy volume can be exploited to automatically initialize trackers for objects entering the scene.

4.1 Volumetric Mass Density

In the following, we demonstrate how the visual hull reconstruction can be exploited to obtain a valuable cue for multi-object tracking. Therefore, we first summarize the required modifications for the standard shape from silhouette technique in Section 4.1.1. Then, we introduce the occupancy volume based on the local mass densities of the visual hull in Section 4.1.2.

4.1.1 Visual Hull Reconstruction

As mentioned in Section 3.2.2, shape from silhouette can only reconstruct the visual hull of objects within the overlap of all camera views. To overcome this limitation, we modify the original formulation as follows.

First, we introduce a *visibility function* to compute the subset of cameras which are able to view the corresponding voxel v_i. In particular, we evaluate whether the corresponding voxel v_i lies within a camera's field-of-view. Thus, we define the visibility function as

$$\text{visible}(v_i) = \{c \mid \text{visible}_c(v_i) = 1\}_{c=1}^{N_C}, \tag{4.1}$$

$$\text{visible}_c(v_i) = \begin{cases} 1 & \text{if } \text{project}_c(v_i) \in I_c \\ 0 & \text{otherwise} \end{cases}, \tag{4.2}$$

where N_C is the number of cameras, I_c denotes the image of camera c, and $\mathbf{P}_c$ is the corresponding projection matrix. Furthermore, $\text{project}_c(v_i)$ computes the imaged coordinates of the voxel v_i via the standard pinhole camera model[5]

$$\text{project}_c(v_i) = \left(\frac{\tilde{x}_i^{(c)}}{\tilde{w}_i^{(c)}}, \frac{\tilde{y}_i^{(c)}}{\tilde{w}_i^{(c)}} \right)^\top, \tag{4.3}$$

$$\begin{pmatrix} \tilde{x}_i^{(c)} \\ \tilde{y}_i^{(c)} \\ \tilde{w}_i^{(c)} \end{pmatrix} = \mathbf{P}_c \begin{pmatrix} v_{i,x} \\ v_{i,y} \\ v_{i,z} \\ 1 \end{pmatrix}, \tag{4.4}$$

where $v_{i,x}$, $v_{i,y}$, and $v_{i,z}$ denote the x, y, and z-coordinate of the i^{th} voxel, respectively. Note that this visibility information is constant for static camera networks and therefore, may be pre-computed to increase the runtime performance.

[5]See Section 3.1 for a detailed discussion of the image formation process.

Algorithm 4.1 Adapted Visual Hull Reconstruction.

```
Input: Camera matrices $\mathbf{P}_c$, foreground silhouettes $S_c$
Output: Visual hull $\mathcal{V}_{\text{VH}}$

1. For each $v_i \in \mathcal{V}_{\text{VH}}$
     • Set $n_i = 0$
     • For each $c \in \text{visible}\,(v_i)$
         – Project $v_i$ into camera view:
           $(\tilde{x}_c, \tilde{v}_c, \tilde{w}_c)^\top = \mathbf{P}_c(v_{i,x}, v_{i,y}, v_{i,z}, 1)^\top$
           $x_c = \frac{\tilde{x}_c}{\tilde{w}_c}, \qquad y_c = \frac{\tilde{y}_c}{\tilde{w}_c}$
         – If $(x_c, y_c)^\top \in S_c$:
               $n_i = n_i + 1$
     • If $n_i = |\,\text{visible}\,(v_i)\,|$:
           $v_i = 1$
       else:
           $v_i = 0$
```

Using this additional visibility information, we can easily adapt the standard shape from silhouette reconstruction as summarized in Algorithm 4.1. To obtain the required foreground silhouettes, we apply the approximate median background model as proposed by McFarlane and Schofield [92]. In particular, we compute the background models on two different color spaces for each view, *i.e.*, one using only intensity values (gray-scale) and one on the saturation channel of the HSV color space. By combining these two background models, we obtain a more robust segmentation, as the effects of weak reflections and penumbras, *i.e.*, soft shadows, are significantly reduced.

Additionally, we slightly modify the visual hull representation. Instead of removing voxels from the resulting volume, we set $v_i = 0$ or $v_i = 1$ for voxels which belong to the background or the visual hull, respectively. The reason for this alternative representation will become obvious in the following section, where we introduce the occupancy volume.

Furthermore, volumetric reconstruction approaches obviously require that the objects of interest are placed in front of the cameras. However, this assumption may be violated for tracking scenarios due to the unconstrained camera placement. For example, consider situations where objects move behind a single camera but are still visible in other views. Using the standard pinhole projection, the object points may be projected onto valid image coordinates, although the object itself is out of the camera's field-of-view. Hence, it is beneficial to additionally incorporate the relative position of the voxel *w.r.t.* the camera's center into the visibility function.

This can be done by evaluating the voxel's *chirality* [52], *i.e.*, the sign of the voxel's depth $d_{v_i}^{(c)}$ *w.r.t.* the camera c. Following Hartley and Zisserman [54], the depth can be computed as

$$d_{v_i}^{(c)} = \frac{\operatorname{sign}\left(\det\left(\mathbf{M}_c\right)\right)\tilde{w}_i}{\left\|\mathbf{m}_3^{(c)}\right\|}, \tag{4.5}$$

where $\mathbf{M_c} = \mathbf{K}_c\mathbf{R}_c$ is the left 3×3 sub-matrix of the camera's projection matrix $\mathbf{P}_c$, and $\mathbf{m}_3^{(c)}$ is the 3rd row of $\mathbf{M}_c$. The voxel v_i is now considered to be in front of the camera[6], if $d_{v_i}^{(c)} > 0$. Note that we are actually only interested in the sign of $d_{v_i}^{(c)}$, and not the depth itself. Therefore, according to [54], we only need to evaluate

$$d_{v_i}^{(c)} \doteq \operatorname{sign}\left(\det\left(\mathbf{M}_c\right)\right)\tilde{w}_i, \tag{4.6}$$

where $\doteq$ denotes equality of the sign. Thus, we obtain the final visibility function as

$$\operatorname{visible}\left(v_i\right) = \left\{c \,\middle|\, \operatorname{visible}_c\left(v_i\right) = 1 \wedge \operatorname{sign}\left(d_{v_i}^{(c)}\right) > 0\right\}_{c=1}^{N_C}, \tag{4.7}$$

by additionally incorporating the position of a voxel *w.r.t.* to the camera.

4.1.2 Occupancy Volume

In general, visual hull reconstructions are very sensitive to noisy foreground segmentations. To overcome this problem and furthermore obtain a valuable cue for object tracking, we introduce the 3D *occupancy volume* as follows.

Given the modified visual hull representation $\mathcal{V}_{\mathrm{VH}}$ obtained from Algorithm 4.1, *i.e.*, $v_i = 1$ denotes a part of the visual hull, and $v_i = 0$ defines background, we derive the occupancy volume $\mathcal{V}_{\mathrm{O}}$ by computing the local mass density values m_D for each voxel v_i. More formally,

$$\mathcal{V}_{\mathrm{O}} = \left\{m_D(v_i) \,|\, \forall v_i \in \mathcal{V}_{\mathrm{VH}}\right\}, \tag{4.8}$$

$$m_D(v_i) = \frac{\sum_{v_j \in N_{v_i}} v_j}{|N_{v_i}|}, \tag{4.9}$$

where the local neighborhood N_{v_i} depends on the objects of interest.

For example when considering persons, one can observe that people tend to align their torso upright, *e.g.*, while standing, walking, and even while

[6]Note that we use the same convention as [52, 54], *i.e.*, cameras are calibrated s.t. $\det\left(\mathbf{M}_c\right) > 0$. If the camera calibration does not follow this convention, *i.e.*, $\det\left(\mathbf{M}_c\right) < 0$, one has to switch the sign accordingly, *e.g.*, by multiplying the projection matrix by -1.

crouching. Thus, for the task of tracking humans, a natural choice would be to model the torso by an upright cylinder. Hence, the neighborhood may be defined as

$$N_{v_i} = \left\{ v_j \;\middle|\; \sqrt{(v_{j,x} - v_{i,x})^2 + (v_{j,y} - v_{i,y})^2} \leq r \wedge |v_{j,z} - v_{i,z}| \leq \frac{h}{2} \right\}, \quad (4.10)$$

where r, h denote the radius and height of the cylinder, respectively. By choosing $h \geq 3r$, we increase the emphasis on the vertical neighborhood and thus incorporate the upright alignment of the human torso. Note that the mass density defines a likelihood relationship on the position of an object's center, *i.e.*, the objects' mass centers correspond to high local density values within the occupancy volume, as can be seen in Figure 4.1(e).

However, although a cylindrical neighborhood may provide a tight bound on the human torso, its implementation is highly time-consuming. Therefore, we propose to approximate the human torso by an axis-aligned cuboid as

$$N_{v_i} = \left\{ v_j \;\middle|\; |v_{j,x} - v_{i,x}| \leq r \wedge |v_{j,y} - v_{i,y}| \leq r \wedge |v_{j,z} - v_{i,z}| \leq \frac{h}{2} \right\}, \quad (4.11)$$

which allows to use efficient integral image [25] representations for computing the mass densities. In practice, both neighborhood formulations provide similar valuable cues for tracking. However, the cuboid significantly outperforms the cylinder in terms of runtime efficiency and thus, should be preferred.

Although ghost artifacts may occur during reconstruction of the visual hull, their effects are significantly reduced by computing the local mass densities. Furthermore, the mass densities of ghosts vary over time, *i.e.*, these masses are unstable and usually lower compared to real objects. Additionally, we can safely assume that objects enter and leave the scene at known locations, *i.e.*, they cannot suddenly appear in the middle of the scene. Combining this closed-world assumption with the lower mass densities of ghosts allows for robustly handling these reconstruction artifacts.

4.2 Tracking from Geometric Cues

In the following, we show how we use the occupancy volume to robustly track multiple objects. For clarity, we first discuss single object tracking using a particle filtering approach in Section 4.2.1. Then, we demonstrate how Voronoi partitioning can be exploited for efficient multiple object tracking in Section 4.2.2.

4.2.1 Single Object Tracking

In order to overcome the computationally expensive search within the 3D occupancy volume, we split the actual tracking part into two separate steps. This $(2 + 1)$D tracking can be accomplished as follows.

First, we locate the objects within the Cartesian plane, *i.e.*, we estimate the corresponding (x, y) coordinates. We derive a top-view occupancy map $\mathcal{M}$ by assigning the maximum local mass density value along the z axis for a given position. More formally,

$$\mathcal{M}_{x,y} = \max m_D(v_i), \quad v_{i,x} = x, \quad v_{i,y} = y, \quad v_{i,z} \in [z_{\min}, z_{\max}], \tag{4.12}$$

where $z_{\min}, z_{\max}$ define the vertical extent of the tracking volume. As already observed in Section 4.1.2, high local mass densities indicate the location of the object's mass center. Thus, by using the vertical maxima, we obtain valuable peaks which correspond to the objects' true positions.

To estimate the target's (x, y) coordinate, we then use a particle filtering approach [59] operating on $\mathcal{M}$. In particular, we use a bootstrap particle filter[7] to estimate the state $\mathbf{x}_{i,t} = (x_i, y_i, \hat{v}_{i,x}, \hat{v}_{i,y})^\top$ of the i^{th} target at time t, where $(x_i, y_i)^\top$ is the object's location within the Cartesian plane, and $(\hat{v}_{i,x}, \hat{v}_{i,y})^\top$ describes the object's velocity. These velocities are modeled as a Gaussian distribution with zero mean and a motion dependent standard deviation which allows to adjust for a wide range of tracking scenarios, including standard surveillance tasks (*i.e.*, usually walking or standing people) as well as sports games (*i.e.*, fast moving players).

Given the mass density observations $\mathbf{z}_{1:t}$ of the occupancy map $\mathcal{M}$, we then approximate the posterior probability $p\left(\mathbf{x}_{i,t} \,|\, \mathbf{z}_{1:t}\right)$ using a set of weighted particles obtained via Monte Carlo sampling,

$$p\left(\mathbf{x}_{i,t} \,|\, \mathbf{z}_{1:t}\right) \approx \left\{\mathbf{x}_{i,t}^{(l)}, w_{i,t}^{(l)}\right\}_{l=1}^{N_s}, \tag{4.13}$$

where N_s is the number of sampled particles, *e.g.*, in our experiments, we achieve a good trade-off between runtime efficiency and approximation accuracy by choosing $N_s = 30$.

Similar to Pérez et al. [108], we choose a second-order auto-regressive model to represent the transition probability $p\left(\mathbf{x}_{i,t} \,|\, \mathbf{x}_{i,t-1}\right)$:

$$\mathbf{x}_{i,t+1} = \mathbf{A}\mathbf{x}_{i,t} + \mathbf{B}\mathbf{x}_{i,t-1} + \mathbf{v}_t. \tag{4.14}$$

This means that the state transition depends on the previous positions and velocities, *i.e.*, defined by the drift coefficients in matrices $\mathbf{A}$ and $\mathbf{B}$, as well as a random component, modeled as $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where the covariance matrix $\mathbf{\Sigma}$ contains the corresponding variances for the x and y-dimensions. Obviously, these transition parameters depend on the tracking scenario, *i.e.*, expected motion of the targets (*e.g.*, walking versus running people), frame rate of the video stream, and the spatial resolution of the reconstruction volume. For example, when tracking mostly walking or slowly running people

[7] Recall Section 3.3 for a detailed discussion of particle filters.

we assume a maximum speed of approximately 14 km/h. Thus, at a frame-rate of 20 frames per second, the particle transition at the occupancy map can be limited by a radius of 0.2 m between consecutive frames.

Furthermore, to compute the corresponding weights for the bootstrap particle filter according to Algorithm 3.5

$$w_{i,t}^{(l)} \propto p\left(\mathbf{z}_t \mid \mathbf{x}_{i,t}^{(l)}\right), \tag{4.15}$$

we assume that the likelihood model is exponentially distributed

$$p\left(\mathbf{z}_t \mid \mathbf{x}_{i,t}^{(l)}\right) \approx \exp\left(-\lambda\left(1-\sum_{\hat{x},\hat{y}}\frac{\mathcal{M}_{\hat{x},\hat{y}}}{\rho}\right)\right), \tag{4.16}$$

where λ is a constant rate parameter of the distribution, *e.g.*, we set $\lambda = 5$ in our experiments. Furthermore, the data likelihood is computed over a local neighborhood, *i.e.*, $\hat{x}, \hat{y} \in N\left(\mathbf{x}_{i,t}^{(l)}\right)$, defined by the particle position and size. In particular, we assume particles of size $0.2\,\text{m} \times 0.2\,\text{m}$. Finally, ρ is a normalization constant depending on the spatial resolution of the occupancy map.

Given above definitions, we can now estimate the object location within the xy-plane via particle filtering. In particular, the true state of the i^{th} object at time t can be approximated using its weighted particles as:

$$\mathbf{x}_{i,t} \approx \sum_{l=1}^{N_s} \mathbf{x}_{i,t}^{(l)} w_{i,t}^{(l)}. \tag{4.17}$$

The final step in approximating the 3D object position is to compute the corresponding z coordinate. Therefore, we search for the mass center along the z axis within a local neighborhood of the corresponding estimate. This additionally allows for robustly tracking objects which exhibit out-of-plane motion, *e.g.*, people stepping on ladders, as well as challenging poses, *e.g.*, crouching or sitting people.

Note that a dense computation of the occupancy volume, *i.e.*, evaluating the local mass densities for each voxel, is computationally expensive. To reduce the complexity and to further increase the runtime performance, we propose to evaluate the local mass densities only at specific heights in a first step. By choosing suitable height levels, we ensure that no object is missed, as illustrated in Figure 4.2. This allows for an efficient approximation of the occupancy map to estimate the object's (x, y) position. Afterwards, we perform a refined search for the vertical mass center at the estimated position to obtain the corresponding z-coordinate.

For example, when tracking people, we set the vertical extent of the tracking volume to $z_{\min} = 0\,\text{m}, z_{\max} = 2.2\,\text{m}$, and approximate the human torso by a cuboid of size $r = 0.2\,\text{m}, h = 0.6\,\text{m}$. To efficiently approximate

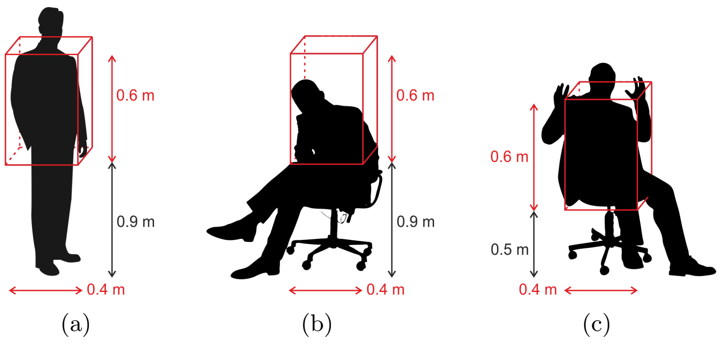


Figure 4.2: Approximation of the human torso by an axis-aligned cuboid. Evaluating the local mass densities only at specific height levels (a,b) allows for efficiently deriving the occupancy map to locate the object within the xy-plane. Afterwards, the corresponding z-coordinate is obtained by a refined search for the vertical mass center, restricted to a local neighborhood of the estimated (x, y) coordinate (c). Human silhouettes were provided by *All Silhouettes*[8].

the occupancy map, we compute the local mass densities at three height levels, *i.e.*, $z = 0.45\,\mathrm{m}$ (to cover crawling or crouching people), $z = 1.2\,\mathrm{m}$ (to cover walking people), and $z = 1.9\,\mathrm{m}$ (to handle out-of-plane motions, *e.g.*, jumping people). As demonstrated by our experimental results, this simplification allows for tracking at high frame rates while achieving excellent tracking performance.

4.2.2 Multiple Object Tracking

The particle filter tracking described so far works well for single instances. However, collisions of multiple objects cannot be handled. In fact, if objects move close to each other, the respective modes at the occupancy map may coalesce into a single blob, once their visual hulls cannot be separated. Thus, collision handling techniques, such as the iterative repulsion scheme [112], are required. However, by exploiting the assumption that multiple objects cannot occupy the same location in space at the same time, we can use an efficient approach based on Voronoi partitioning of the hypotheses space, as inspired by Kristan et al. [75].

In general, when tracking N_O objects, we are interested in the state of all objects i at time t, which is called the *joint-state*:

$$\mathbf{X}_t = \{\mathbf{x}_{i,t}\}_{i=1}^{N_O} . \tag{4.18}$$

Then, the goal of multiple object tracking is to estimate the posterior probability of the joint-state $p\left(\mathbf{X}_t \,|\, \mathbf{z}_{1:t}\right)$, based on the available observations.

[8]`http://all-silhouettes.com/` (accessed November 25, 2012)

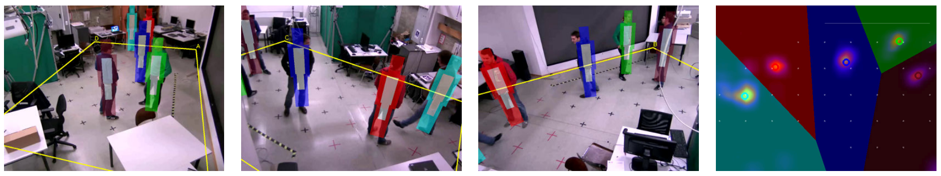


Figure 4.3: Input images with superimposed tracking results (left) and the corresponding occupancy map with Voronoi partitions (right) for $t = 1810$ of the Chap dataset. Note that the colors of the Voronoi partitions correspond to the tracker labels.

From applying the closed-world assumption that multiple objects cannot occupy the same location at the same time, it follows that the occupancy map $\mathcal{M}$ can be partitioned into non-overlapping regions, s.t. each partition is occupied by exactly one object.

Therefore, we apply a Voronoi partitioning on $\mathcal{M}$ (see Figure 4.3), using the currently estimated object positions $(x_i, y_i)^\top$. More formally, we obtain the set $\mathcal{T}$ of pairwise-disjoint convex regions as

$$\mathcal{T} = \left\{\hat{\mathcal{T}}_i\right\}_{i=1}^{N_O}, \tag{4.19}$$

$$\hat{\mathcal{T}}_i = \left\{m \in \mathcal{M} \mid d\left(m, (x_i, y_i)^\top\right) \leq d\left(m, (x_j, y_j)^\top\right), \forall j \neq i\right\}, \tag{4.20}$$

where $d(\cdot)$ denotes the Euclidean distance. In particular, we use the efficient *parallel banding* algorithm by Cao et al. [21] to compute the Voronoi partitions.

According to [75], given the Voronoi partitioning $\mathcal{T}_t$ at time t, the objects' states become conditionally independent and thus, the posterior probability of the joint-state factorizes as:

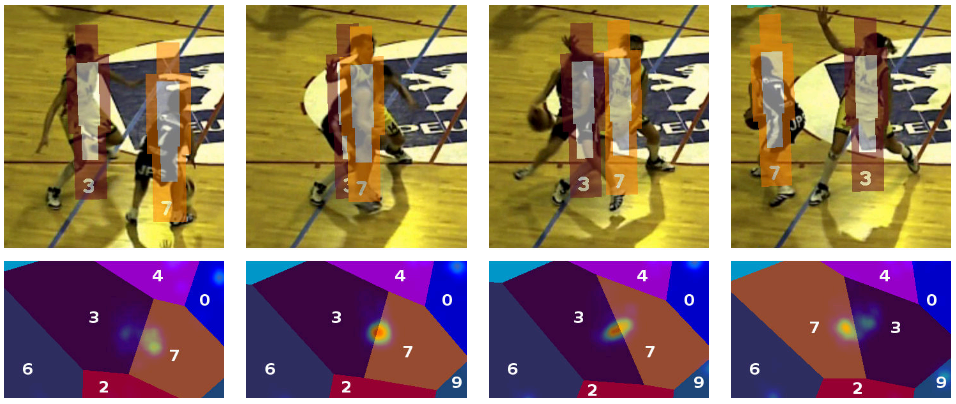
$$p\left(\mathbf{X}_t \mid \mathbf{z}_{1:t}, \mathcal{T}_t\right) = \prod_{i=1}^{N_O} p\left(\mathbf{x}_{i,t} \mid \mathbf{z}_{1:t}, \mathcal{T}_t\right). \tag{4.21}$$

This implies that given the Voronoi partitioning, each object can be tracked by a single-object tracker restricted to its corresponding partition.

In order to restrict a particle filter's state transition to its respective Voronoi partition, we use a mask function derived from the partitioning. More formally, we obtain the mask for the i^{th} particle filter as:

$$\text{mask}\left(i, \mathcal{T}\right) = \begin{cases} 1 & \forall m \in \hat{\mathcal{T}}_i \\ 0 & \forall m \in \hat{\mathcal{T}}_j, j \neq i \end{cases}. \tag{4.22}$$

Note that the partitioning is of special importance if a target is fully occluded by other objects, *i.e.*, it is not visible in any camera view. Hence, the particle filter keeps the correct position and cannot drift to nearby modes on the occupancy map.



(a) $t = 1046$. (b) $t = 1051$. (c) $t = 1054$. (d) $t = 1060$.

Figure 4.4: Sample merge-split situation at the APIDIS dataset. The colors of the visual results (top) and Voronoi partitions on the derived occupancy map (bottom) correlate. During an attack situation (a,b), the tracker starts drifting (c). By using the discriminative classifier, the trajectories are correctly re-established (d), preventing the identity switch.

4.3 Resolving Geometric Ambiguities

So far, the proposed algorithm operates solely on the geometric information derived from the binary foreground segmentations. However, in real-world scenarios it is often not possible to correctly assign identities by just using geometric information, *e.g.*, consider the situation shown in Figure 4.4. To cope with this problem, we developed a *merge-split* approach where geometrically ambiguous situations are identified and resolved as soon as possible by incorporating additional appearance information.

Therefore, we exploit the scene structure to collect samples on-line, as detailed in Section 4.3.1. Then, we discuss the required steps to robustly reassign the tracked identities in ambiguous situations using logistic regression in Section 4.3.2.

4.3.1 On-line Feature Extraction

To obtain visual representations of the objects, we collect samples by exploiting the 3D scene structure, as well as the actual tracking positions of the individual objects. By using the depth information we can identify occlusions in the individual views, and thus extract information only from valuable samples, *i.e.*, samples where the corresponding object is fully visible. The samples are extracted at the torso region, illustrated by the white overlay in Figure 4.4.

As features we use histograms over the hue and saturation (HS) channels of the HSV color space to describe an object's appearance. Note that we also evaluated different color spaces, such as RGB, RG-Chroma, and CIE-Lab. However, throughout our experiments we achieved the best performance using HS histograms with 64 bins per channel.

To allow robust identification of the objects, we introduce a *feature bag* $\mathcal{F}_i$ for each object, where we store N_F feature representations for each camera. More formally, the feature bag of the i^{th} object is defined as

$$\mathcal{F}_i = \left\{\mathcal{F}_i^{(c)}\right\}_{c=1}^{N_C}, \tag{4.23}$$

$$\mathcal{F}_i^{(c)} = \left\{\mathbf{f}_l^{(c)}\right\}_{l=1}^{N_F}, \tag{4.24}$$

where $\mathbf{f}_l^{(c)}$ is the l^{th} feature, *i.e.*, the HS histogram, extracted from camera c. As proposed by [82], we also evaluated combined features, *i.e.*, the histograms are normalized over all cameras. However, these are significantly less discriminative, especially in scenarios where the cameras are exposed to different environmental illumination conditions. Therefore, we use separate feature bags for each camera view.

To ensure that a feature bag will always contain the most up-to-date appearance information of the corresponding object, we update the feature bags frequently, following a first-in-first-out (FIFO) strategy. Thus, once the maximum number of samples for a feature bag have been collected, *i.e.*, $\left|\mathcal{F}_i^{(c)}\right| = N_F$, newly on-line extracted samples will replace the oldest ones. This allows to robustly handle changing appearance of the objects (*e.g.*, see Chap dataset). In particular, we set $N_F = 20$ and update the feature bags once every second throughout our experiments.

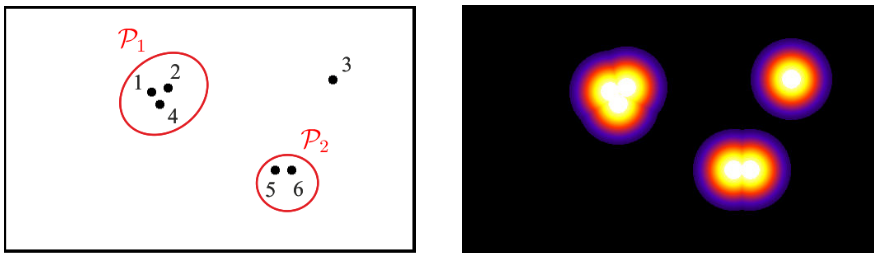
4.3.2 Logistic Regression

In order to resolve the geometric ambiguities, we propose a *merge-split* approach as follows[9]. First, in the *merge* step, we identify potential *conflicts* where a robust identity assignment cannot be guaranteed solely based on the geometric information. In practice, these potential conflicts may be detected by observing the 2D occupancy map. Therefore, we denote an object to be *in conflict* with other objects, if the distance between their currently estimated positions $(x_i, y_i)^\top$ is below a pre-defined threshold $\tau_\mathcal{C}$, *e.g.*, $\tau_\mathcal{C} = 0.75\,\text{m}$. Hence, we can easily compute the set of conflicted objects *w.r.t.* the i^{th} object as

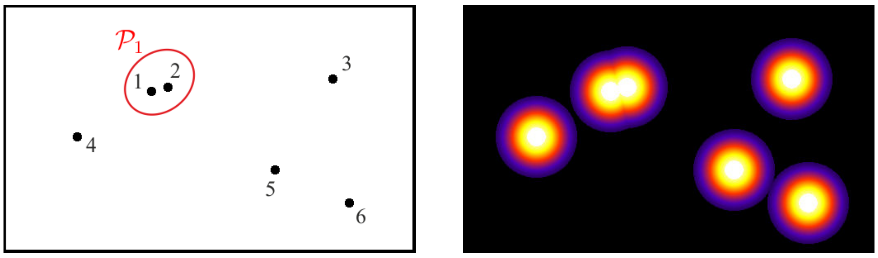
$$\mathcal{C}_i = \left\{j \mid d\left((x_i, y_i)^\top, (x_j, y_j)^\top\right) < \tau_\mathcal{C}, \forall j \neq i\right\}, \tag{4.25}$$

where $d(\cdot)$ is the Euclidean distance between the estimated positions.

[9]For clarity, we assume that there is only one camera and therefore, drop the corresponding superscripts. In practice, however, there are N_C classifiers for each object.



(a) The objects form two separate crowds which results in two conflict pools, *i.e.*, $\mathcal{P}_1 = \{1, 2, 4\}$ and $\mathcal{P}_2 = \{5, 6\}$. Note that object 3 can be reliably tracked using geometric information alone.



(b) After objects 4, 6 move away, separate local maxima can be detected. Hence, assuming the trackers could be re-assigned based on the extracted features at the local maxima, the pools are now $\mathcal{P}_1 = \{1, 2\}$ and $\mathcal{P}_2 = \emptyset$.

Figure 4.5: Synthetic example illustrating conflict pools before (a) and after (b) objects part from crowds. Left images show the true object positions (top-view), whereas the corresponding occupancy maps are shown right.

Next, we group the conflicted objects into *conflict pools* $\mathcal{P}_m$, *i.e.*, subsets of the tracked objects which are conflicted with nearby objects. This means that there will be one pool for each crowd within the observed tracking region, as illustrated in Figure 4.5. The proposed pooling procedure is summarized in Algorithm 4.2.

Now we train a discriminative classifier for each involved object using the on-line collected features. Therefore, we use L2-regularized logistic regression classifiers of the publicly available LIBLINEAR toolbox[10] by Fan et al. [34], which allow to efficiently solve binary classification problems. In particular, the logistic regression classifier for the i^{th} object is based on the probability model

$$p_i\left(y_{i,l} \,|\, \mathbf{f}_l, \mathbf{w}_i\right) = \frac{1}{1 + e^{-y_{i,l}\mathbf{w}_i^\top \mathbf{f}_l}}, \tag{4.26}$$

which means that given a feature representation $\mathbf{f}_l$ and the model weights $\mathbf{w}_i$, we obtain the posterior probability of the instance label $y_{i,l} \in \{+1, -1\}$. In fact, we use the convention that $y_{i,l} = +1$, if $\mathbf{f}_l$ has been extracted from the i^{th} object. Furthermore, we can exploit the conflict pools to reduce the

[10]`http://www.csie.ntu.edu.tw/~cjlin/liblinear/`(accessed October 10, 2012)

Algorithm 4.2 Conflict Detection and Pooling.

```
Input: Currently tracked objects with labels $i \in \mathbb{N}$
Output: Conflict pools $\mathcal{P}_m$, s.t. for each crowd there is exactly one pool

1. For each object label $i$
    • Set $\mathcal{C}_i = \emptyset$
    • For each object label $j \neq i$
        – Compute distance $d(P_i, P_j)$
        – If $d(P_i, P_j) < \tau_{\mathcal{C}}$:
              $\mathcal{C}_i = \mathcal{C}_i \cup \{j\}$
2. Set $m = 1$
3. For each $\mathcal{C}_i, |\mathcal{C}_i| > 0$
    • Set $\mathcal{P}_m = \{i\}$
    • For each $j \in \mathcal{P}_m$
        – Set $\mathcal{P}_m = \mathcal{P}_m \cup \mathcal{C}_j$
        – Set $\mathcal{C}_j = \emptyset$
      Note: $\mathcal{P}_m$ may change during iterations, hence ensure that each
            added label is processed
    • Set $m = m + 1$
```

classification complexity, as we only need to distinguish between the objects of the corresponding pool. Thus, we train a one-vs-all classifier for each object within the conflict pool $\mathcal{P}_m$, where the training samples are provided via the corresponding feature bags, *i.e.*, $\mathbf{f}_l \in \mathcal{F}_i, i \in \mathcal{P}_m$. Therefore, we obtain the corresponding class labels to train the i^{th} classifier as:

$$y_{i,l} = \begin{cases} +1 & \forall \mathbf{f}_l \in \mathcal{F}_i \\ -1 & \forall \mathbf{f}_l \in \mathcal{F}_j : \forall j \in \mathcal{P}_m, j \neq i \end{cases} . \tag{4.27}$$

In the training stage, we now estimate the weights $\mathbf{w}_i$ of the model p_i by minimizing the negative log-likelihood:

$$\min_{\mathbf{w}_i} \sum_l \log\left(1 + e^{-y_{i,l}\mathbf{w}_i^\top \mathbf{f}_l}\right). \tag{4.28}$$

Usually, according to Lin et al. [83], an additional L2-regularization term is added to obtain good generalization abilities. Therefore, we actually solve the unconstrained optimization problem

$$\min_{\mathbf{w}_i} \frac{1}{2}\mathbf{w}_i^\top \mathbf{w}_i + C_i \sum_l \log\left(1 + e^{-y_{i,l}\mathbf{w}_i^\top \mathbf{f}_l}\right), \tag{4.29}$$

where $C_i > 0$ is a penalty parameter.

While several trackers are in a conflicted state, *i.e.*, their visual hull volumes are merged, the corresponding particle filters share one coalesced maximum, as can be seen in Figures 4.4 and 4.5. Thus, in the *split* step we resolve these geometric ambiguities by correctly re-assigning the conflicted trackers based on the visual appearance. Therefore, we search for separate local maxima, restricted by the Voronoi partitions of the involved objects, *i.e.*, all objects within the corresponding conflict pool $\mathcal{P}_m$. In particular, we use an efficient block-based non-maximum suppression (NMS) as proposed by Neubeck and Van Gool [102] to detect local maxima within the 2D occupancy map.

Once the objects move away, their visual hull volumes split again and separate local maxima may be detected. If there are more than one separate maxima for a conflict pool $\mathcal{P}_m$, we extract features $\mathbf{f}_{\text{NMS}}$ at the corresponding positions. Given these candidate features and the trained logistic regression classifiers, we can now robustly re-assign the conflicted trackers. Therefore, for each local maxima, we seek the object identifier $\hat{\imath} \in \mathcal{P}_m$ which yields the maximum posterior probability:

$$\hat{\imath} = \arg\max_i \, p_i\left(y_{i,\text{NMS}} = +1 \,|\, \mathbf{f}_{\text{NMS}}, \mathbf{w}_i\right). \tag{4.30}$$

Furthermore, we re-assign the corresponding tracker only if $p_{\hat{\imath}} > \tau_R$, where τ_R is a pre-defined threshold. For our experimental evaluations, we achieved excellent results by choosing τ_R to be in the range $[0.7, 0.8]$. Finally, whenever a tracker has successfully been re-assigned, we remove it from the conflict pool, *i.e.*, $\mathcal{P}_m = \mathcal{P}_m \setminus \{\hat{\imath}\}$.

4.4 Autonomous Tracking

In order to initialize and cancel trajectories, we define *entry regions* near the entrances of the scenes, similar to related approaches, *e.g.*, [9]. This also conforms to the closed-world assumption that objects cannot suddenly appear at the middle of the scene, which we exploited to reduce the effect of ghosts in the visual hull reconstruction. Exemplary entry regions are shown in Figure 4.6.

Whenever objects leave the observed region of interest, the corresponding tracker is canceled. Furthermore, we store the corresponding features $\mathcal{F}_i$

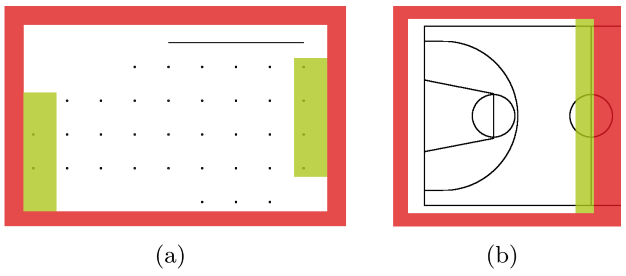


Figure 4.6: Layout of the entry regions, *i.e.*, green areas, used for the ICG laboratory datasets (a) and the APIDIS basketball court (b). Note that trackers will be canceled if objects leave the tracking area, *i.e.*, denoted by red boundaries.

to allow for re-identification if an object re-enters the scene. Therefore, in contrast to related approaches, such as [82], objects may leave and re-enter via completely different entry points of the scene.

For the automatic initialization, we observe the occupancy map at the defined entry areas by extracting maximally stable extremal regions [90]. Therefore, we compare for each candidate region, whether its mass density corresponds to that of the average object of interest. Note that there are many alternative initialization methods which may be applied to identify entering objects, *e.g.*, considering multi-person tracking, one may incorporate state-of-the-art pedestrian detectors (*e.g.*, Felzenszwalb et al. [37]) to observe the entry regions and obtain locations for initialization.

Before assigning a new tracker, we match the appearance model of the entering person against those of objects, which left the scene previously. Since we use HS histograms to represent the objects' appearance, we evaluated several commonly used distance metrics for histogram matching. In particular, we computed the *Earth Mover's Distance* [114], different histogram intersection scores (*i.e.*, [61, 127]), as well as the χ^2 distance. A tracker is only re-assigned to the entering person if the distance is below a pre-defined threshold which depends on the used metric. Note that in our experiments, we achieved similar re-identification performance with all metrics. In practice, however, the normalized histogram intersection [61] may be preferred, as it allows for a more intuitive adjustment of the threshold, *i.e.*, the measure lies in the interval $[0, 1]$.

Chapter 5

Experiments

Contents

In this chapter, we demonstrate our proposed multi-object tracker on several challenging people tracking scenarios. For evaluation of the tracking performance, we use the standard CLEAR[11] performance metrics, which have also been used by several other multi-object tracking approaches, *e.g.*, Berclaz et al. [9]. These metrics allow for principled evaluation, as well as intuitive comparison between different approaches. We summarize the evaluation process in Section 5.1. Details on the conducted comparison to the state-of-the-art will be presented in Section 5.2.

For evaluation, we use both, synthetic and real-world datasets. In particular, we captured several sequences within our laboratory to demonstrate common tracking challenges, such as out-of-plane motion or crowded situations. Additionally, we use two publicly available datasets, namely the APIDIS basketball dataset and the MVL Lab5 laboratory sequence. Specific challenges of these datasets will be summarized in Section 5.3, where we also present illustrative tracking results. Finally, we discuss the tracking and runtime performance more detailed in Sections 5.4 and 5.5.

[11] Classification of Events, Activities, and Relationships (CLEAR) Evaluation and Workshop, `http://www.clear-evaluation.org` (accessed August 6, 2012).

5.1 Evaluation Metrics

For evaluation of our tracking algorithm and comparison to the state-of-the-art, we compute the standard CLEAR multiple object tracking performance metrics, *i.e.*, *Multiple Object Tracking Accuracy* and *Precision* (MOTA and MOTP). These measures have been proposed by Bernardin and Stiefelhagen [11] to objectively compare tracker characteristics, with a special focus on their estimated object location precision, as well as their accuracy in consistently labeling objects throughout the sequences.

The precision metric MOTP evaluates the alignment of true positive trajectories *w.r.t.* the ground truth, *i.e.*, it states the position error averaged by the number of valid object-hypothesis matches. More formally, this metric is defined as

$$\mathrm{MOTP} = \frac{\sum_{i,t} d_t^i}{\sum_t c_t}, \tag{5.1}$$

where d_t^i is the position error for the i^{th} match at time t and c_t is the number of valid matches. The position error can be computed for any tracking result format, *e.g.*, using an overlap criterion for returned bounding boxes, or a distance measure between predicted locations and ground-truth positions on the ground-plane. As all evaluated approaches locate the objects on the 2D real-world ground-plane, we use the Euclidean distance between hypothesized and labeled object positions to compute the MOTP scores for comparison. Thus, the reported MOTP values are measured in meters, where lower values indicate a better alignment with the labeled ground truth positions. A tracker hypothesis is considered valid if it lies within a radius defined by a distance threshold τ_d of an annotated ground truth position. Note that this distance threshold also defines the upper bound on the reported precision metric MOTP.

The accuracy metric MOTA on the other hand accounts for all object configuration errors, *i.e.*, false positives, false negatives (misses), and identity switches (mismatches). It is defined as

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FP}_t + \mathrm{FN}_t + \mathrm{IDS}_t)}{\sum_t g_t}, \tag{5.2}$$

using the number of false positives FP_t, the number of false negatives FN_t, the number of identity switches IDS_t, and the number of labeled objects g_t at time t, respectively. Thus, higher MOTA values indicate a better performance, with 1 representing a perfect tracking result. This accuracy metric is derived from three individual error ratios, averaged by the total number of objects present in the whole sequence. These individual error ratios are the ratio of false positives

$$\overline{\mathrm{FP}} = \frac{\sum_t \mathrm{FP}_t}{\sum_t g_t}, \tag{5.3}$$

the ratio of missed objects

$$\overline{\mathrm{FN}} = \frac{\sum_t \mathrm{FN}_t}{\sum_t g_t}, \tag{5.4}$$

and the ratio of identity switches

$$\overline{\mathrm{IDS}} = \frac{\sum_t \mathrm{IDS}_t}{\sum_t g_t}. \tag{5.5}$$

Note that both MOTA and MOTP are independent of each other, *i.e.*, MOTA gives an intuitive performance measure *w.r.t.* detecting objects of interest and keeping their identity, independent of the localization precision, whereas MOTP evaluates the localization error without considering false object configurations. For a more detailed discussion of the design choices behind these metrics, we refer the interested reader to [11, 64].

In order to obtain the ground truth object positions for the real-world datasets, we used the provided ground truth data for both, the APIDIS dataset and the MVL Lab5 dataset, and manually annotated every 10th frame of the ICG laboratory scenarios. Therefore, we annotated the foot point locations (x_i^c, y_i^c) of every object i in each camera view c and projected them onto the real-world ground-plane, as

$$(\hat{x}_i^c, \hat{y}_i^c, \hat{w}_i^c)^\top = H_c^{-1}(x_i^c, y_i^c, 1)^\top, \tag{5.6}$$

where H_c defines the homography between the world plane at $z = 0$ and the image plane of camera c. According to [54], this transformation is defined as

$$H_c = [p_c^1, p_c^2, p_c^4], \tag{5.7}$$

where p_c^j is the jth column of the corresponding camera's projection matrix P_c. The real-world ground truth positions $(\hat{x}_i, \hat{y}_i, 1)^\top$ are estimated as the mean over the individual projections:

$$\hat{x}_i = \frac{1}{C}\sum_{c=1}^{C} \frac{\hat{x}_i^c}{\hat{w}_i^c}, \tag{5.8}$$

$$\hat{y}_i = \frac{1}{C}\sum_{c=1}^{C} \frac{\hat{y}_i^c}{\hat{w}_i^c}. \tag{5.9}$$

For the synthetic dataset, we used the foot point location of the human models in every frame.

5.2 Comparison to the State-of-the-Art

The K-Shortest Paths (KSP) tracker of Berclaz et al. [9] is considered to be state-of-the-art in multiple object tracking and has been shown to perform well in complex scenarios, such as poorly illuminated passageways, or crowded indoor scenarios. Therefore, we compare our proposed tracking algorithm with their publicly available implementation[12].

The KSP tracker operates on a discretized top view representation (grid) and uses peaked probabilistic occupancy maps, which define the probability that an object is present at a specific grid position. Similar to the original formulation, we obtain the input probability maps using the publicly available implementation of the probabilistic occupancy map (POM) detector[13] of Berclaz et al. [38].

In order to ensure a fair comparison, we use the same foreground segmentation as input to both, our tracking algorithm and the POM detector. For KSP/POM, we divide the top view representation into a grid of 40cm $\times$ 40cm cells. The spatial distance between the cell centers was varied from 10cm to 20cm. We set the maximum number of iterations for the POM detector to 1000 and varied the input parameters σ (which accounts for the quality of the foreground segmented input images) and the prior probability parameter. We then used the KSP tracker to link the POM detections. Therefore, we also evaluated the KSP tracker with varying input parameters, *i.e.*, different limits on the maximum movement between consecutive frames, as well as different entry point setups. The best performing results are reported and used as a baseline for comparison.

Note that we also tried to obtain additional implementations and datasets of related approaches, such as COST [47], M_2-Tracker [98], or the space carving based approaches of [46] and [82]. Unfortunately, none of these implementations or datasets has been provided for comparison. However, we were allowed to use the MVL Lab5 dataset of Mandeljc et al., including their tracking results as presented in [88]. Thus, we are able to compare our proposed approach to another recently published multiple object tracker.

5.3 Datasets

We used several datasets to demonstrate the capabilities of the proposed tracking approach, namely a synthetic dataset, six challenging real-world scenarios captured at our institute's laboratory, the public APIDIS basketball dataset, as well as the very recently published MVL Lab5 dataset (see [88]). The characteristics of these datasets are listed in Table 5.1, while the following Sections 5.3.1-5.3.4 summarize the main challenges and present illustrative tracking results. A detailed performance evaluation and runtime analysis will be presented in the following Sections 5.4 and 5.5, respectively.

[12]`http://cvlab.epfl.ch/software/ksp/` (accessed September 20, 2012)
[13]`http://cvlab.epfl.ch/software/pom/` (accessed September 20, 2012)

Dataset	N_C	**Frames**	N_O	**FPS**	**Resolution**	**Section**
Wiper	4	75	8	20	1280×960	5.3.1
Chap	4	3760	5	20	1024×768	5.3.2
Leaf 1	4	1800	4	20	1024×768	
Leaf 2	4	2400	5	20	1024×768	
Much	4	2400	5	20	1024×768	
Pose	4	1820	6	20	1024×768	
Table	4	1760	5	20	1024×768	
APIDIS	7	1500	12	25	1600×1200	5.3.3
MVL Lab5	4	7840	5	20	2048×1536	5.3.4

Table 5.1: Dataset characteristics indicating the number of cameras N_C, the total number of frames, the maximum number of simultaneously visible objects N_O, and the resolution of the video streams. The last column refers to the sections explaining the corresponding datasets.

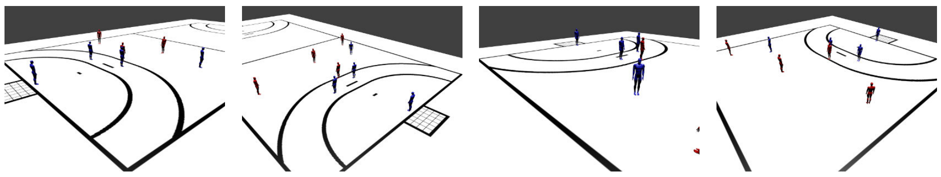
MOTP [m]	**MOTA**	**TP**	**FP**	**FN**	**IDS**
0.126	1.000	592	0	0	0

Table 5.2: Qualitative and quantitative tracking results for the synthetic feasibility test case Wiper. This table lists the precision metric MOTP (lower is better) and accuracy metric MOTA (higher is better), as well as the total number of true positives (TP), false positives (FP), false negatives (FN), and identity switches (IDS). The reported scores were achieved at a distance threshold of $\tau_d = 0.50\,\text{m}$.

5.3.1 Synthetic Dataset

As an early step in evaluating the proposed algorithm, we decided to demonstrate the feasibility of the space carving based approach. Therefore, we created a synthetic dataset, as this allows to neglect common sources of errors, such as imprecise camera calibrations or lens distortion effects. The dataset should exhibit dynamic occlusion, *i.e.*, caused by the objects, a large tracking area, as well as proximity between tracked objects. Thus, our synthetic dataset shows an attack scenario of a European handball game.

This dataset shows the left-half of a European handball field, resulting in a tracking region of approximately $20\,\text{m} \times 20\,\text{m}$, captured by four cameras at a resolution of 1280×960. On the field, we placed two differently colored teams (red and blue) with four players each. The scene depicts the well-known *wiper* attack strategy, conducted by the players of the red team. Sample input images and illustrative tracking results are shown in Figures 5.1 and 5.2, respectively. In the following, we refer to this dataset as Wiper.


(a) Input images.

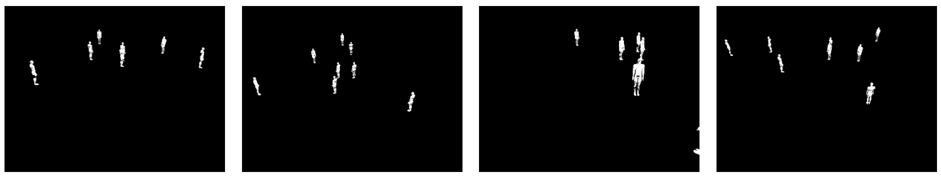
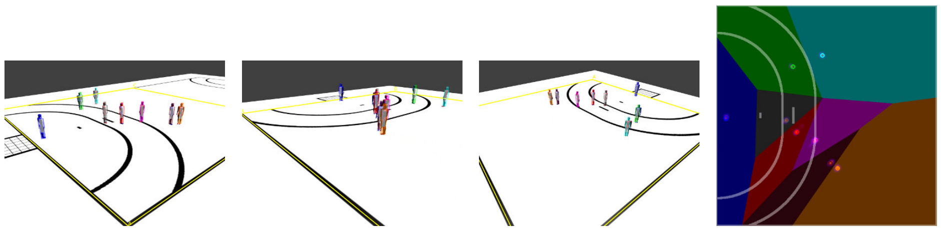

(b) Foreground segmentation.

Figure 5.1: Sample input images (a) and corresponding foreground segmentation (b) for frame 5 of the synthetic WIPER scenario, which depicts an attack strategy in a European handball game.


(a) $t = 45$.

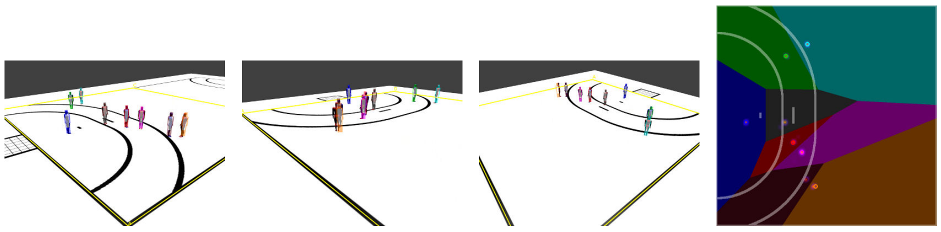

(b) $t = 60$.

Figure 5.2: Illustrative tracking results for the synthetic WIPER scenario. The first three columns show re-projected tracking results into selected camera views (*i.e.*, cameras 1, 3, and 4), whereas the last column illustrates the ground-plane occupancy map with corresponding Voronoi partitions, used to estimate the 2D object coordinates. The yellow rectangle limits the tracking region.

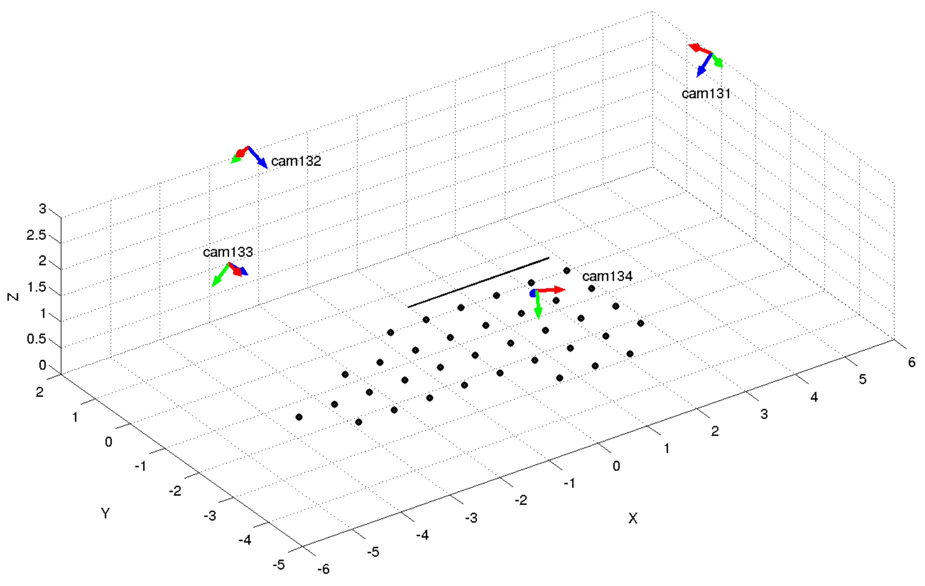


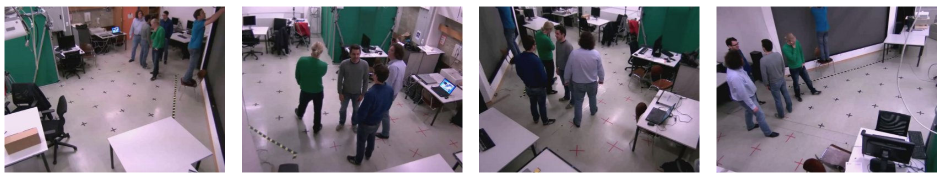
Figure 5.3: Camera setup of all ICG laboratory datasets. The blue arrow depicts the orientation of each camera's optical axis, whereas the red and green axes illustrate the alignment of each camera's image plane. The black dots and line correspond to the actual markers on the laboratory's ground-plane.

The 3D models of the players were created using MakeHuman™, a freely available[14] open source modeling tool. Next, these models were used to render the attack scenario using the free open source 3D content creation suite Blender[15]. As can be seen from the performance metrics listed in Table 5.2, we are able to accurately locate and track all 8 players, despite the dynamic occlusions, as well as the large tracking area. Note that we did not consider a realistic animation of the players' articulations but instead focused on creating a demonstration showing multiple objects and frequent dynamic occlusions. Therefore, we do not consider this sequence for comparison to the state-of-the-art.

5.3.2 ICG Datasets

We recorded several real-world datasets at the *Deskotheque* laboratory using 4 static Axis P1347 cameras at 20 fps and a resolution of 1024×768. The size of the tracking region is approximately $7\,\text{m} \times 4\,\text{m}$. Figure 5.3 illustrates the camera setup used for capturing all laboratory scenarios. Although the

[14]`http://www.makehuman.org/` (accessed July 11, 2012)
[15]`http://www.blender.org/` (accessed July 11, 2012)


(a) Input images.

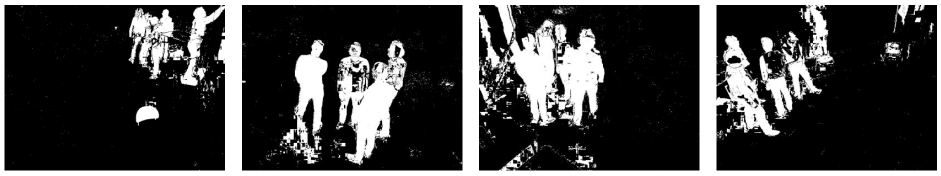

(b) Foreground segmentation.

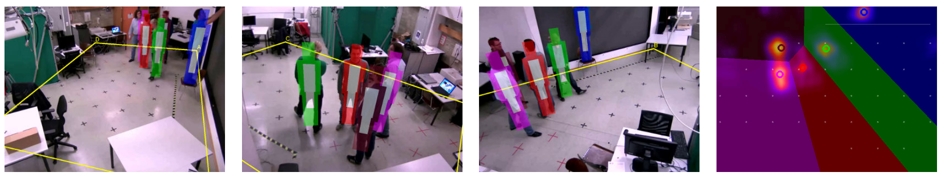
Figure 5.4: Sample input images (a) and corresponding foreground segmentation (b) for frame 565 of the CHAP scenario which demonstrates out-of-plane motion, *i.e.*, the person standing on the chair.

cameras were placed slightly above head-level, at approximately 2.9 m, all sequences exhibit significant occlusions. We calibrated the camera network using the publicly available toolbox by Bouguet [14]. The motivation behind these datasets is to demonstrate violations of common tracking assumptions by out-of-plane motion, as well as different poses and articulations, *e.g.*, sitting or crouching. In particular, we captured six challenging datasets, which will be discussed in the following.

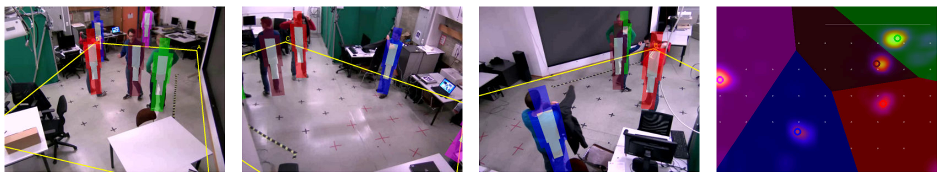
Changing Appearance (CHAP)

This sequence depicts a standard surveillance scenario, where five people move unconstrained within a laboratory. Throughout the scene, the people change their visual appearance by putting on jackets with significantly different colors than their sweaters. Since people move close to each other after changing their appearance, these situations impose additional challenges to color based object tracking approaches, as fixed color models cannot deal with changing appearances.

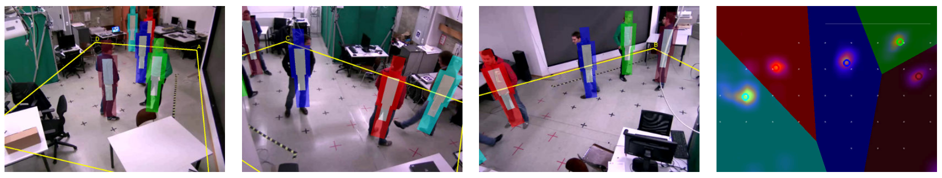
Furthermore, this sequence demonstrates the effect of out-of-plane motion, *i.e.*, a person steps on a chair to fix a poster above his head, as shown in Figure 5.4. As can be seen from the illustrative tracking results in Figure 5.5, the proposed algorithm is able to robustly keep track of the people, despite the violation of the ground-plane assumption, or the significantly different visual appearance. Note that the main focus of this sequence clearly lies on the changing appearance of the people. Therefore, we set up additional scenarios with an emphasis on out-of-plane motion, as well as complex poses and articulations.



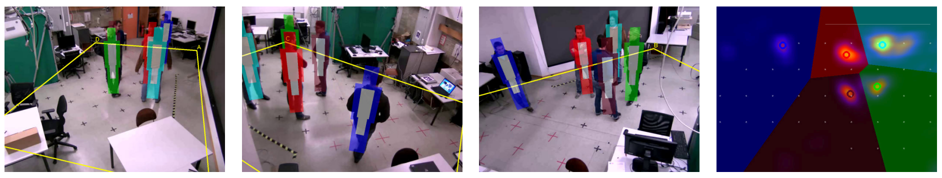
(a) $t = 565$.



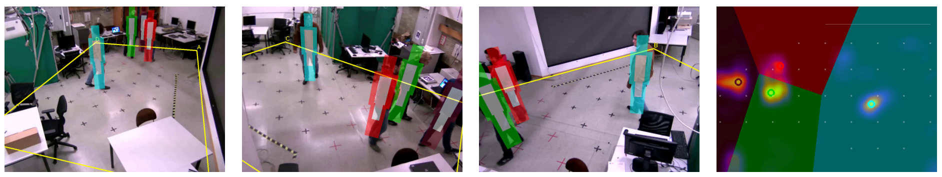
(b) $t = 1490$.



(c) $t = 1810$.

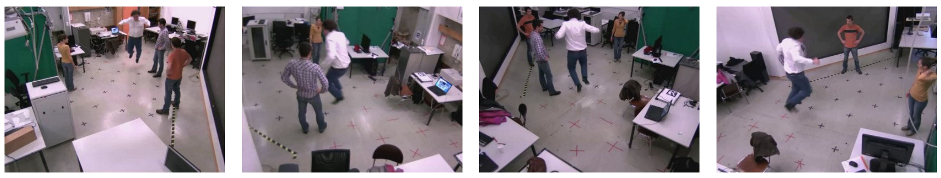


(d) $t = 2960$.



(e) $t = 3565$.

Figure 5.5: Illustrative tracking results for the Chap scenario. These situations demonstrate out-of-plane motion (a), people putting on jackets (b), as well as an identity switch (c) caused by the person re-entering the scene after changing his appearance outside the tracking region. After changing clothes, the individuals again form crowds (d), before leaving the scene (e). The yellow rectangle limits the tracking region. Note that the top view occupancy map has been cropped for better visualization.


(a) Leaf 1, $t = 968$.

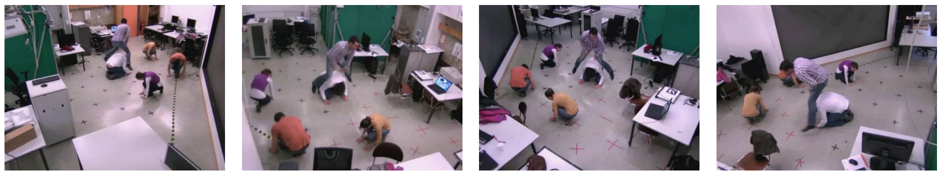

(b) Leaf 2, $t = 868$.

Figure 5.6: Sample input images for the Leaf 1 (a) and Leaf 2 (b) scenarios. Note the out-of-plane motion, *i.e.*, the jumping person, and the challenges caused by the kneeling people, as well as the proximity of the individuals.

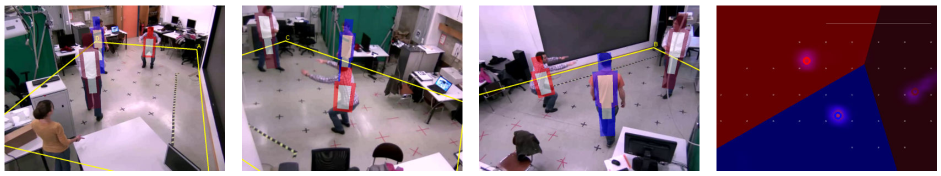
Leapfrogs (Leaf 1 & 2)

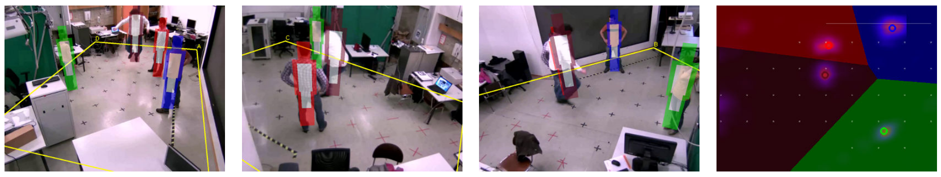
In the well-known leapfrog game, players vault over each other's stooped backs, as shown in Figure 5.6. Therefore, a player bends over – either with hands rested on the knees, or crouching – and the next one leaps over by straddling legs wide apart on each side of the other player's back. For several reasons – such as the proximity of the people while performing a leapfrog, the crouching pose, or the out-of-plane motion while jumping – tracking the players of this game is very challenging. Therefore, we decided to capture two scenarios, referred to as Leaf 1 & 2, which depict leapfrogging people.

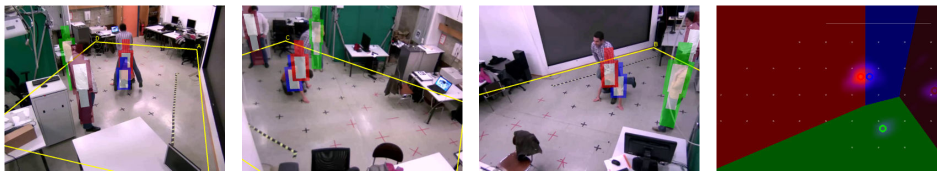
As these datasets are designed to exhibit an increasing difficulty level, the Leaf 1 scenario shows a total of four people preparing for the following game. Therefore, they perform several squats, as well as warm-up jumps before they continue with a few leapfrog exercises. Although there are only four people involved, this sequence nevertheless features several challenging situations due to the complex poses, out-of-plane motion, or fast motion of the players while jumping.

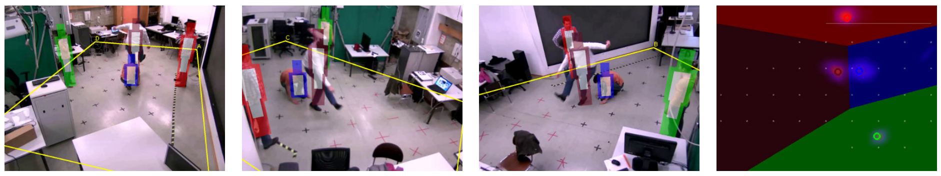
For the more complex Leaf 2 scenario, five people arrange in a circle and each one performs leapfrogs over all other participants. Extracting appearance information for robust re-identification of the targets is hampered by the fact that non-leaping players are crouching most of the time and do not stand upright until it is their turn to perform the leapfrogs.

Note that the leapfrog scenarios violate the closed-world assumption that multiple objects cannot occupy the same location at the same time. Considering the occupancy map used to estimate the object positions on the ground-plane, as described in Section 4.2, this assumption is violated for


(a) $t = 670$.


(b) $t = 968$.


(c) $t = 1353$.


(d) $t = 1540$.

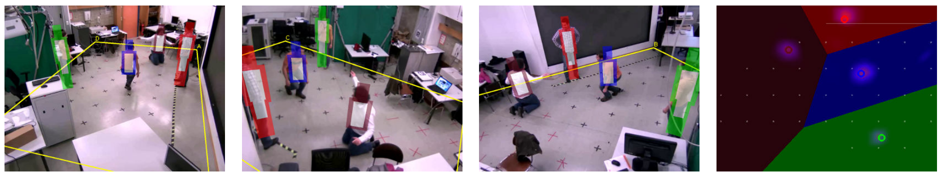
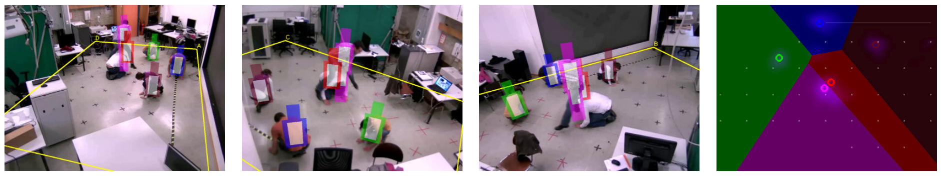
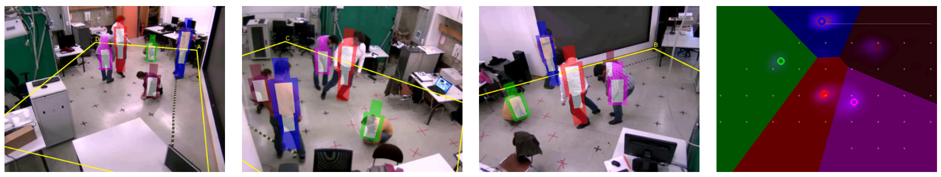
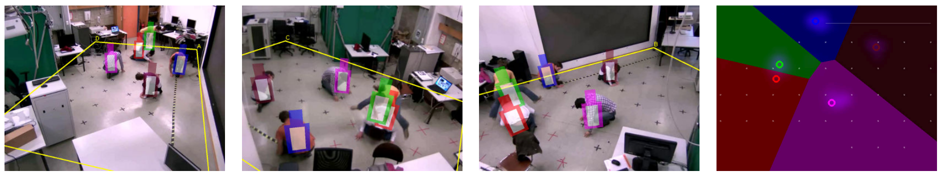
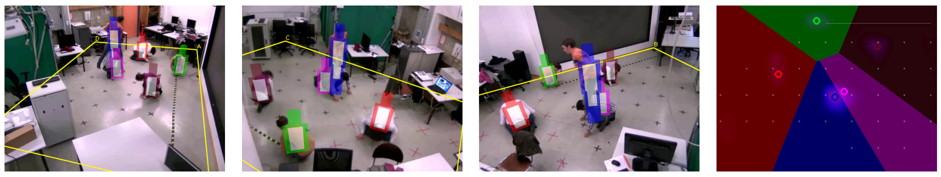

(e) $t = 1590$.

Figure 5.7: Illustrative tracking results for the Leaf 1 scenario. These situations illustrate the squats (a) and jumps (b) of the warm-up phase and demonstrate that leapfrogs (c,d) are physically demanding exercises (e). Note that the top view occupancy map has been cropped for better visualization.


(a) $t = 868$.


(b) $t = 1070$.


(c) $t = 1525$.


(d) $t = 1648$.

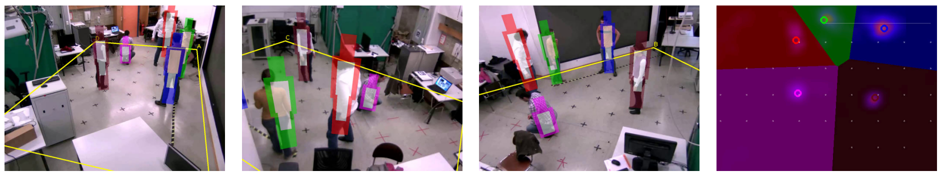
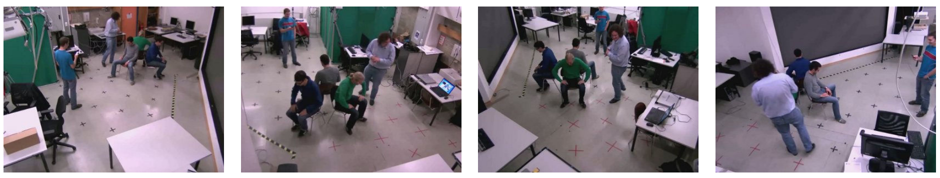
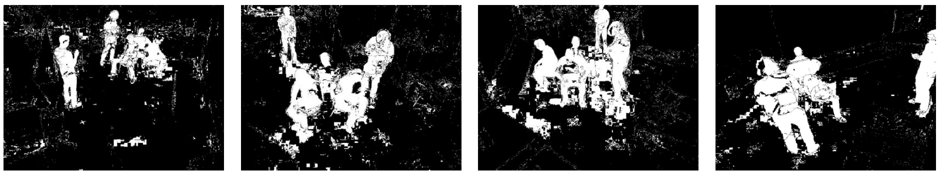

(e) $t = 2110$.

Figure 5.8: Illustrative tracking results for the Leaf 2 scenario, depicting the challenging poses inherent to the leapfrog game (a-d), as well as an individual leaving the scene, depicting an unusual pose (e). Note that the top view occupancy map has been cropped for better visualization.



(a) Input images.



(b) Foreground segmentation.

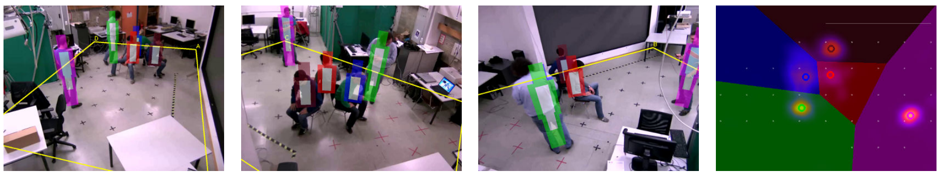
Figure 5.9: Sample input images (a) and corresponding foreground segmentation (b) for frame 940 of the Much scenario. The situation illustrates the result after the first race for the chairs.

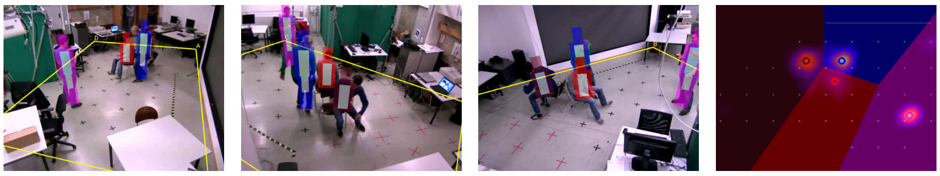
players which currently perform leapfrogs, since their mass centers are located on top of each other. Nevertheless, the proposed algorithm is able to robustly track these scenarios, as illustrated in Figures 5.7 and 5.8. In such situations, the corresponding modes on the occupancy map coalesce and the corresponding trackers move towards the single peak. However, multiple trackers may share the same mode, as all particles of an individual tracker are limited by their corresponding Voronoi cell. Since the geometric information cannot be used to reliably distinguish the involved objects, we use the on-line collected appearance information to resolve these situations and prevent identity switches, as discussed in Section 4.3.

Musical Chairs (Much)

Another challenging situation arises for multiple object trackers whenever objects move fast or at close quarters. To demonstrate these issues, we recorded a *musical chairs* game, also known as *Going to Jerusalem* or *Trip to Jerusalem*. In this game, N players move in unison around a circle of $N - 1$ chairs, while a non-playing moderator is playing music. As soon as the moderator stops the music, all players race towards the chairs and try to sit down on one of them. The player left without a chair is then eliminated, as shown in Figure 5.9, and the game continues with one chair less. Once there is only one player left, the game is over and the remaining player is the winner.

We captured this game sequence, referred to as Much, showing four people playing *musical chairs* and a non-playing moderator who starts and stops the recorded music. Due to the nature of this game, this sequence exhibits


(a) $t = 940$.


(b) $t = 1405$.

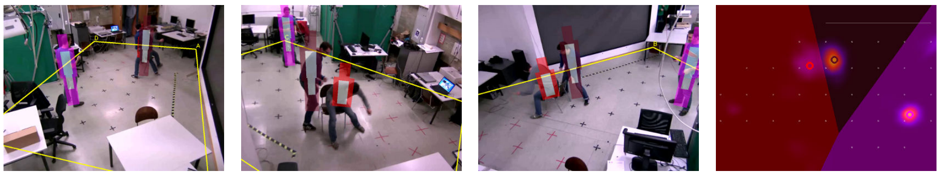
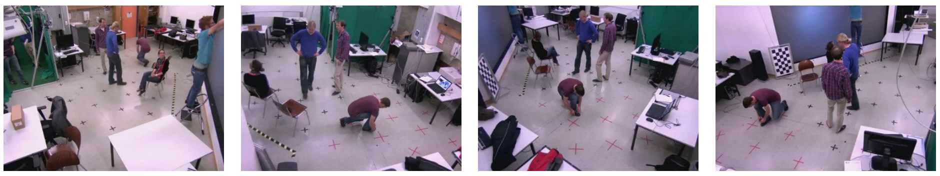

(c) $t = 1915$.

Figure 5.10: Illustrative tracking results for the MUCH scenario. The frames show the game results after the first (a), second (b), and final (c) race for the chairs. Note that the top view occupancy map has been cropped.

fast motions, as well as crowded situations, *e.g.*, when all players race to the available chairs. Furthermore, sitting on chairs violates the commonly used constraint of standing persons. Additionally, regarding background modeling, there are dynamic background items, *i.e.*, the chairs which are removed after each round, as well as a static foreground object, *i.e.*, the moderator stands almost still throughout the whole sequence. Despite these challenges, we are able to robustly track all players and the moderator throughout the game, as illustrated in Figure 5.10.

Challenging Poses (POSE)

This sequence shows up to 6 people in various poses, such as standing, walking, kneeling, crouching, crawling, sitting, and stepping on ladders. Additionally to these poses, which again violate common tracking assumptions such as upright standing pedestrians or a common ground-plane, a changing background illumination causes further challenges *w.r.t.* robust foreground segmentation, as can be seen from the input images in Figure 5.11. Nevertheless, the tracking approach based on the local mass density proves to be robust in such situations, as illustrated in Figure 5.12.


(a) Input images.

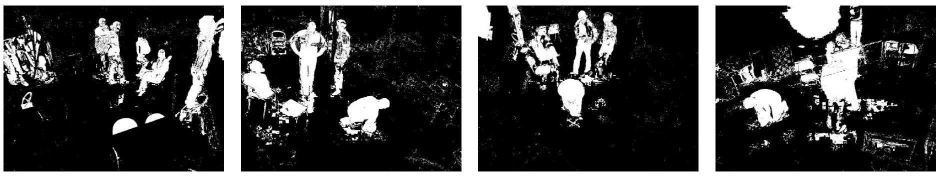
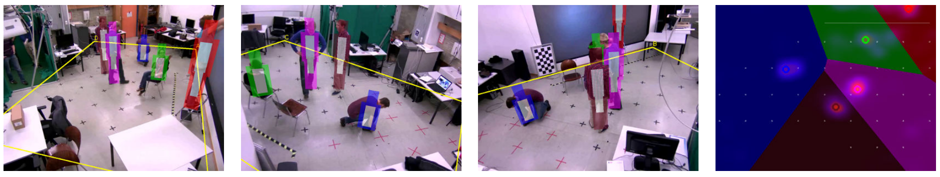
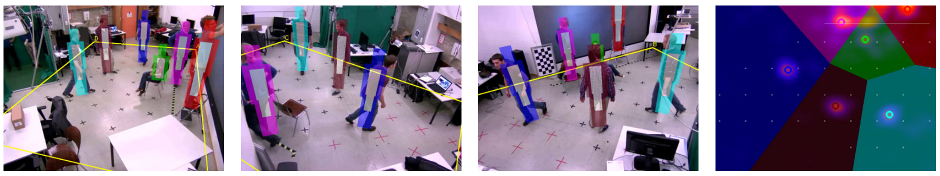

(b) Foreground segmentation.

Figure 5.11: Sample input images (a) and corresponding foreground segmentation (b) for frame 875 of the Pose scenario, showing sitting and kneeling people, as well as out-of-plane motion, *i.e.*, the person standing on the ladder.


(a) $t = 875$.


(b) $t = 950$.

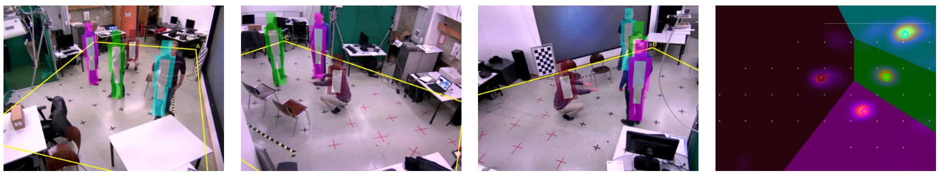
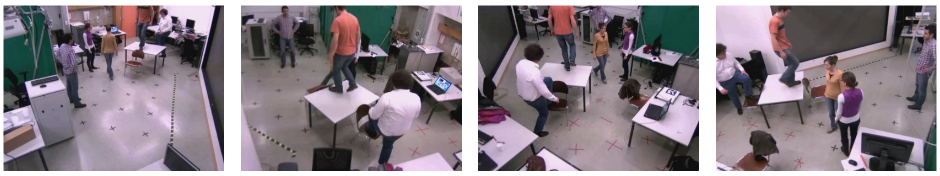
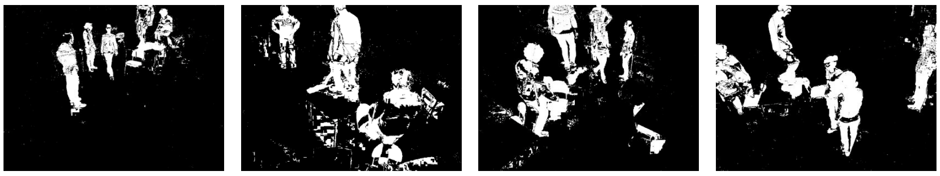

(c) $t = 1280$.

Figure 5.12: Illustrative tracking results for the Pose scenario. The frames illustrate different poses, such as sitting and kneeling (a), out-of-plane motion (b), as well as a person performing squats (c). Note that the top view occupancy map has been cropped for better visualization.



(a) Input images.



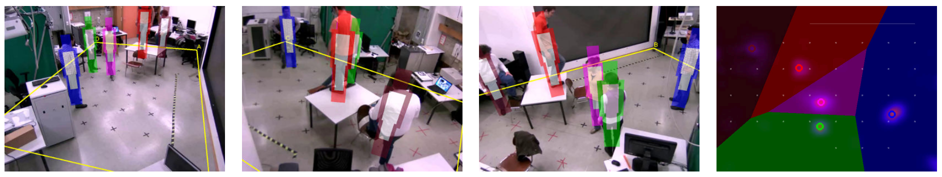
(b) Foreground segmentation.

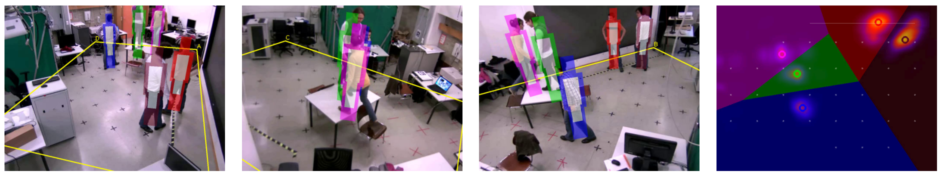
Figure 5.13: Sample input images (a) and corresponding foreground segmentation (b) for frame 460 of the TABLE scenario. Note that the heads of people standing on the table are visible in at most one camera, due to the limited fields-of-view.

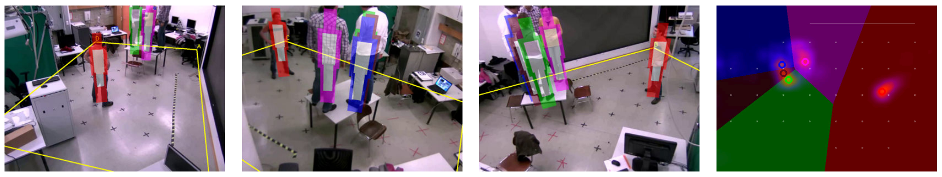
Out-of-Plane Motion (TABLE)

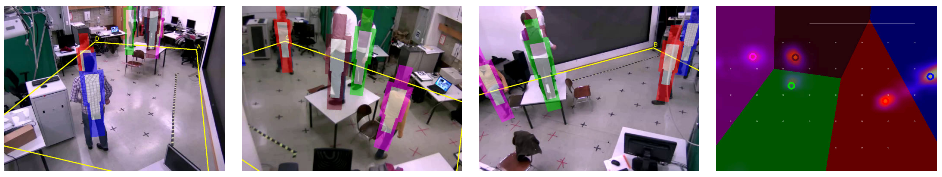
We designed this scenario with a special focus on out-of-plane motion, demonstrated by people stepping on a table in the middle of the laboratory. This violation of the common ground-plane assumption cannot be handled robustly by tracking approaches which use simple geometric transformations, since these transformations are only valid for points lying on the ground-plane.

Throughout the sequence, five people frequently walk over a table in the laboratory, as shown in Figure 5.13. The major challenges of this scenario are caused by the out-of-plane motion and the occlusions caused by both, the table and the people in the scene. Furthermore, this scenario exhibits a densely crowded situation, *i.e.*, four people gather at the table, before two of them jump away. These fast jumps complicate the tasks of keeping track of the objects, as well as resolving the crowded situation robustly. However, as can be seen from the illustrative results in Figure 5.14, the combination of geometric information for localization and appearance information for re-identification proves sufficient to robustly resolve this ambiguous situation correctly.


(a) $t = 460$.


(b) $t = 610$.


(c) $t = 1090$.


(d) $t = 1290$.

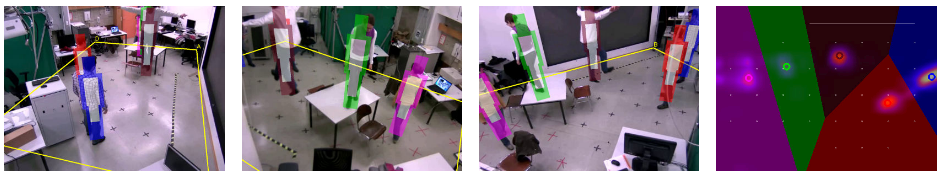
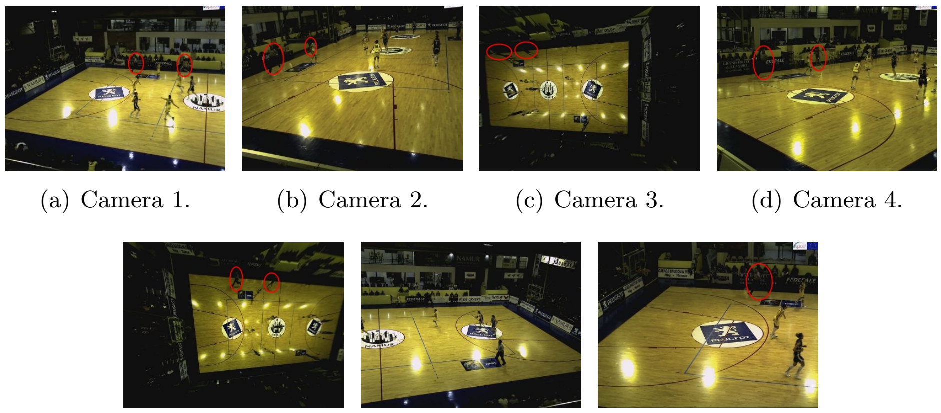

(e) $t = 1315$.

Figure 5.14: Illustrative tracking results for the Table scenario. These frames show the out-of-plane motion caused by people walking over the table (a,b), as well as the crowded situation (c), which can be robustly resolved using the on-line collected appearance information, as soon as the people disperse (d,e). Note that the top view occupancy map has been cropped for better visualization.


(a) Camera 1. (b) Camera 2. (c) Camera 3. (d) Camera 4.

(e) Camera 5. (f) Camera 6. (g) Camera 7.

Figure 5.15: Rectified input images for frame 515 of the public APIDIS dataset. Note that a player of the blue team, as well as a referee – both highlighted by the red ellipses for visualization – are hardly visible, which complicates the automatic initialization. As we evaluate our approach on the left-half of the basketball court, cameras 3 and 6 are not used.

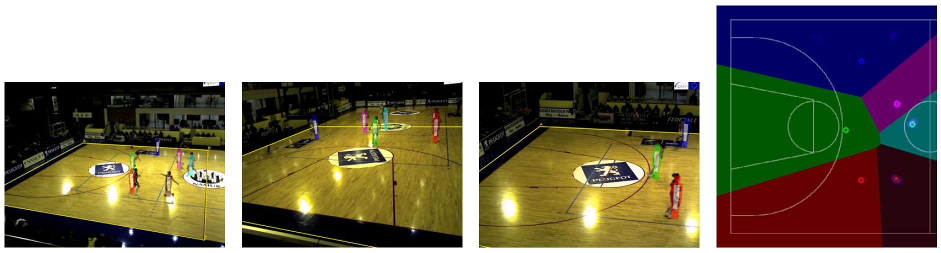
5.3.3 APIDIS Dataset

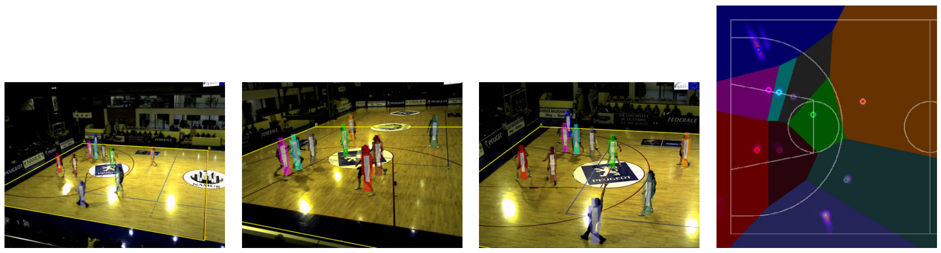
The public APIDIS dataset[16] shows an attack scenario of a basketball game, monitored monitored by 7 cameras (see Figure 5.15). This dataset contains various challenges like heavy occlusions, densely crowded situations as well as complex poses and articulations, or abrupt motion changes. Further challenges are caused by the similar appearance of all players of a team, as well as strong shadows and significant reflections on the floor. Furthermore, some cameras share almost the same viewpoint, *e.g.*, cameras 1 and 7, which provides nearly no additional visual information. These similar camera poses complicate the exact localization for the proposed approach, as already discussed in Section 3.2.2.

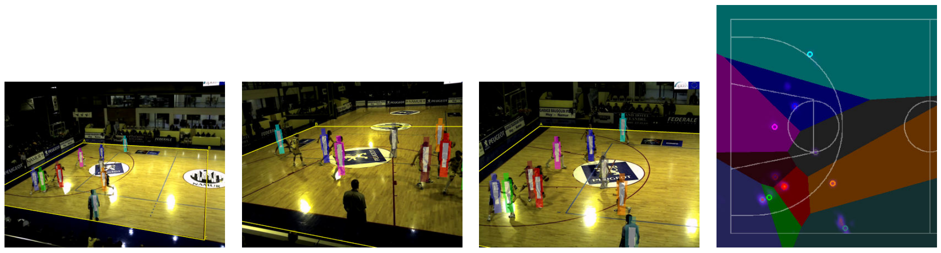
Similar to object detection approaches, such as [2, 107], which demonstrate their algorithms on the APIDIS dataset, we evaluate the performance on the left-half of the basketball court, as this side is covered by the larger number of cameras, *i.e.*, cameras 1, 2, 4, 5, and 7. This results in a region of interest of approximately $15\,\text{m} \times 15\,\text{m}$.

Additional challenges are caused by the location of the cameras, as these are placed at the back line and only one side line, thus this dataset does not provide high quality imagery of the area near the opposing side line. In combination with the difficult segmentation task, *e.g.*, players of the blue

[16]Provided by the European project on Autonomous Production of Images based on Distributed and Intelligent Sensing (APIDIS), `http://www.apidis.org/Dataset/` (accessed September 11, 2012).


(a) $t = 515$.


(b) $t = 645$.


(c) $t = 785$.

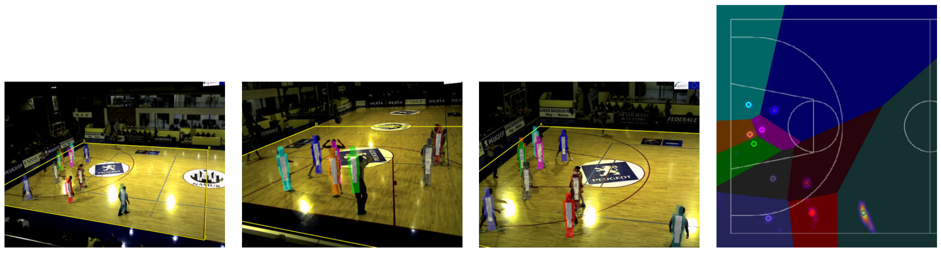
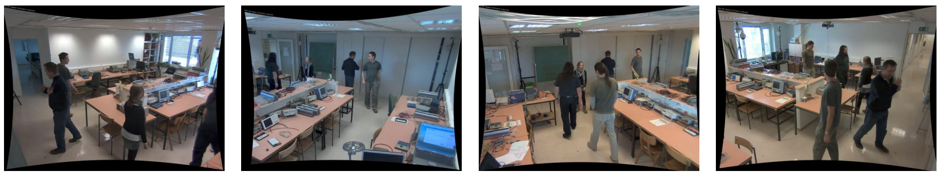

(d) $t = 1115$.

Figure 5.16: Illustrative tracking results for the APIDIS scenario. These frames show a failure case of the automatic initialization (a), *i.e.*, the referee cannot be segmented robustly, as well as the tracking results after all players entered the left-half of the court (b,c) and the situation after the shot attempt (d).


(a) Input images.

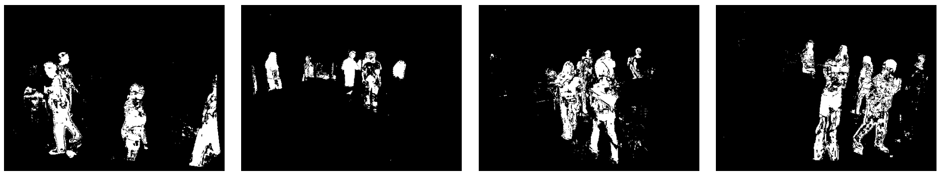

(b) Foreground segmentation.

Figure 5.17: Rectified input images for frame $t = 5450$ of the MVL Lab5 dataset. Note that the cluttered desks cause significant occlusions, which complicates the tracking task.

team can hardly be segmented when standing in front of the grandstand, this significantly complicates the automatic initialization. Despite all imposed challenges, we still achieve competitive performances and outperform the state-of-the-art, as will be shown in the discussion (see Section 5.4).

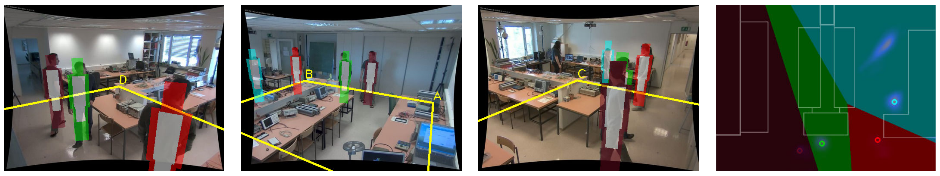
5.3.4 MVL Dataset

MVL Lab5 is a multi-modal indoor person localization dataset, recently published[17] by the authors of [87, 88]. This dataset contains visual data obtained from four static Axis P1346 network cameras, as well as location events from a radio-based localization system. In particular, they used a Ubisense localization system[18], which is based on ultra-wide band radio technology. As the scenario depicts several challenges for visual tracking approaches, such as significant occlusions, the additional radio information is intended to improve the localization of the individuals.

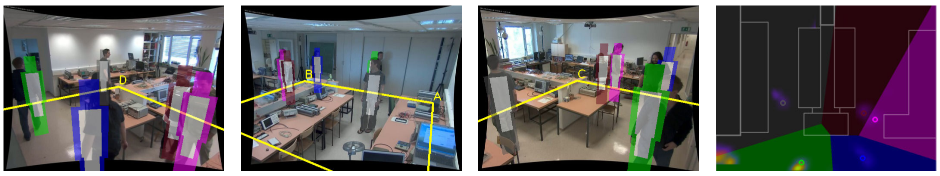
The scenario shows five people walking around a realistically cluttered workspace, as illustrated in Figure 5.17. The major challenges for visual tracking algorithms are caused by the similar appearance of the individuals, changing illumination conditions, as well as severe occlusions, *i.e.*, static occlusions are caused by the office furniture, while the individuals additionally frequently occlude each other.

[17]Provided by the Machine Vision Laboratory (MVL) at University of Ljubljana, `http://vision.fe.uni-lj.si/research/mvl_lab5/` (accessed November 28, 2012).

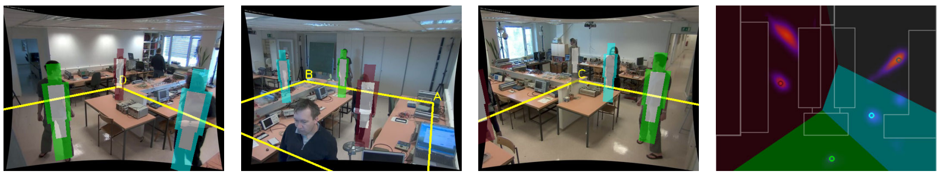
[18]For a description of the ultra-wideband measurement devices, see `http://www.ubisense.net/en/resources/factsheets/series-7000-ip-sensors.html` (accessed December 12, 2012).



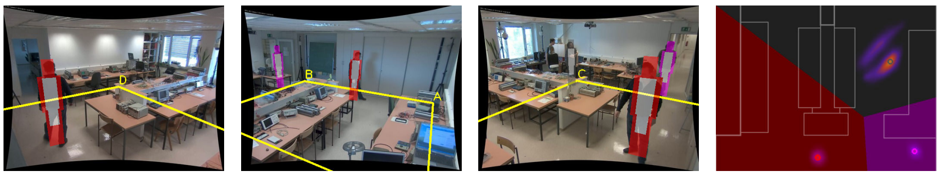
(a) $t = 4095$.



(b) $t = 5500$.



(c) $t = 6430$.



(d) $t = 7338$.

Figure 5.18: Illustrative tracking results for the MVL Lab5 scenario. Note that the proposed approach is able to robustly track the people in regions where the camera fields-of-view support the visual hull reconstruction, *i.e.*, half of the room, as indicated by the clear modes on the occupancy map (a,b). On the other hand, the robust localization is hampered in regions where only opposing cameras view the individuals. Despite these issues, we are still able to track the people by following the corresponding peaks on the occupancy maps (c). Since people frequently leave and re-enter the tracking region, we have to rely on the appearance information to re-identify them correctly, as we do not use the provided radio localization events. However, as shown in (d), the similar appearance of the individuals significantly complicates the re-identification and thus leads to identity switches.

Further challenges arise due to the placement of the cameras, as these are mounted at the corners of the room, such that parts of the tracking region are only visible in opposing camera views. As discussed in Section 3.2.2, these view constellations cause the visual hull reconstructions to be generally much larger than the corresponding objects. Nevertheless, since the proposed occupancy volume provides a valuable clue for tracking, we are able to locate the individuals at a high precision, as can be seen from Figure 5.18.

The additional radio-based localization data may be used to prevent identity switches, as shown by [88]. Although every individual carries a radio identification tag, precise localization solely based on radio information is not feasible, due to the presence of radio-reflective metallic surfaces, as well as obstacles which hamper the signal transmission. Usually, however, only visual information is available for object tracking tasks. Therefore, we ignore the additional radio information during evaluation of our approach.

5.4 Results and Discussion

We evaluated the performance metrics on all datasets for two different distance thresholds, *i.e.*, $\tau_d = 0.5\,\mathrm{m}$ and $\tau_d = 0.75\,\mathrm{m}$, to demonstrate the tracking precision of the compared approaches. The evaluation results are listed in detail in Tables 5.3 and 5.4 for the ICG laboratory datasets (*i.e.*, Chap, Leaf 1 & 2, Much, Pose, and Table), while results for the APIDIS and MVL Lab5 datasets are listed in Tables 5.5 and 5.6, respectively. Figure 5.19 illustrates the results of the performance evaluation on all real-world datasets. Note that we presented the results of the synthetic feasibility test Wiper already in Section 5.3.1, as this dataset is not used for comparison with the state-of-the-art.

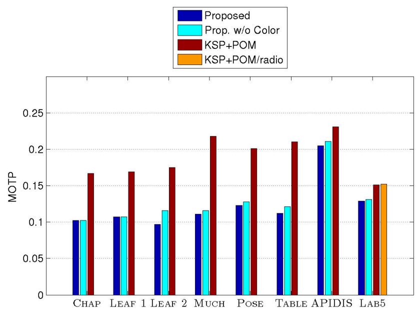
As can be seen from the overall scores, the proposed algorithm achieves state-of-the-art performance at standard visual surveillance scenarios (*i.e.*, Chap and Leaf 1), whereas we significantly outperform the KSP tracker at more complex scenarios, *i.e.*, APIDIS, Leaf 2, Much, Pose, and Table. On the other hand, our approach suffers from improper camera placement which hampers the visual hull reconstruction, as indicated by the lower performance at the MVL Lab5 dataset. Additionally, the people within this dataset are similarly vested which makes the identification difficult, especially in the frequently arising situations where people leave and re-enter the tracking region. To overcome this problem, we may incorporate the additional radio identification data to improve the tracking accuracy *w.r.t.* identity switches, as suggested by the evaluations of [88]. Nevertheless, despite the specific challenges of this dataset, we still achieve competitive tracking accuracy. Furthermore, regarding the tracking precision, our proposed approach outperforms the KSP tracker on all evaluated datasets.

Dataset	τ_d **[m]**	**Algorithm**	**MOTP [m]**	**MOTA**
Chap	0.50	Proposed	**0.102**	**0.994**
		Prop. w/o Color	**0.102**	0.719
		KSP+POM	0.167	0.952
	0.75	Proposed	**0.102**	**0.996**
		Prop. w/o Color	0.106	0.728
		KSP+POM	0.173	0.966
Leaf 1	0.50	Proposed	**0.107**	**0.991**
		Prop. w/o Color	**0.107**	0.721
		KSP+POM	0.169	0.976
	0.75	Proposed	**0.108**	**0.996**
		Prop. w/o Color	0.110	0.725
		KSP+POM	0.188	0.994
Leaf 2	0.50	Proposed	**0.097**	**0.916**
		Prop. w/o Color	0.116	0.727
		KSP+POM	0.175	0.819
	0.75	Proposed	**0.099**	**0.924**
		Prop. w/o Color	0.128	0.731
		KSP+POM	0.187	0.839
Much	0.50	Proposed	**0.111**	**0.977**
		Prop. w/o Color	0.116	0.736
		KSP+POM	0.218	0.754
	0.75	Proposed	**0.113**	**0.987**
		Prop. w/o Color	0.140	0.764
		KSP+POM	0.234	0.790
Pose	0.50	Proposed	**0.123**	**0.944**
		Prop. w/o Color	0.128	0.822
		KSP+POM	0.201	0.555
	0.75	Proposed	**0.128**	**0.956**
		Prop. w/o Color	0.148	0.886
		KSP+POM	0.216	0.615
Table	0.50	Proposed	**0.112**	**0.898**
		Prop. w/o Color	0.121	0.816
		KSP+POM	0.210	0.719
	0.75	Proposed	**0.142**	**0.962**
		Prop. w/o Color	0.148	0.805
		KSP+POM	0.229	0.767

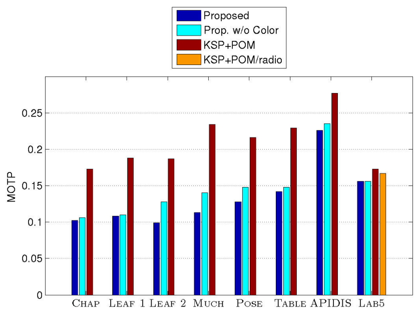
Table 5.3: Tracking performance on ICG datasets. For each evaluated dataset, we state the precision metric MOTP (lower is better) and accuracy metric MOTA (higher is better). The best scores are highlighted.

Dataset	τ_d [m]	Algorithm	TP	FP	FN	IDS
Chap	0.50	Proposed	1555	**2**	**6**	**1**
		Prop. w/o Color	1316	193	241	4
		KSP+POM	**1607**	50	21	7
	0.75	Proposed	1558	**1**	**4**	**1**
		Prop. w/o Color	1326	188	234	2
		KSP+POM	**1626**	41	9	5
Leaf 1	0.50	Proposed	464	**2**	2	**0**
		Prop. w/o Color	436	83	44	7
		KSP+POM	**495**	6	**1**	5
	0.75	Proposed	465	1	**1**	**0**
		Prop. w/o Color	437	82	43	7
		KSP+POM	**500**	**0**	3	**0**
Leaf 2	0.50	Proposed	**930**	**41**	**41**	**0**
		Prop. w/o Color	856	115	117	34
		KSP+POM	913	87	66	24
	0.75	Proposed	**934**	**37**	**37**	**0**
		Prop. w/o Color	859	112	115	35
		KSP+POM	926	81	56	21
Much	0.50	Proposed	**783**	**9**	**9**	**0**
		Prop. w/o Color	694	99	99	11
		KSP+POM	770	139	32	26
	0.75	Proposed	**787**	**5**	**5**	**0**
		Prop. w/o Color	705	90	88	9
		KSP+POM	**787**	123	19	27
Pose	0.50	Proposed	**485**	**14**	**14**	**0**
		Prop. w/o Color	456	42	44	3
		KSP+POM	427	156	31	17
	0.75	Proposed	**489**	**12**	**10**	**0**
		Prop. w/o Color	474	27	26	4
		KSP+POM	446	149	16	15
Table	0.50	Proposed	**621**	**32**	**28**	**6**
		Prop. w/o Color	596	57	55	8
		KSP+POM	573	105	58	14
	0.75	Proposed	**641**	**13**	**10**	**2**
		Prop. w/o Color	607	76	45	6
		KSP+POM	598	97	36	15

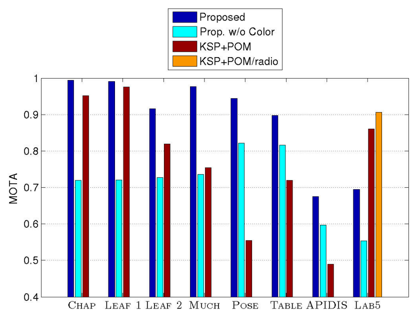
Table 5.4: Tracking results on ICG datasets. For each evaluated dataset, we state the total number of true positives (TP), false positives (FP), false negatives (FN), and identity switches (IDS). The best scores are highlighted.



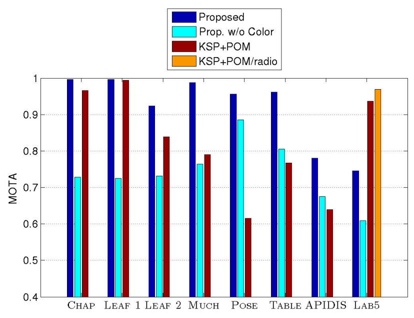
(a) MOTP, $\tau_d = 0.50\,\text{m}$.



(b) MOTP, $\tau_d = 0.75\,\text{m}$.



(c) MOTA, $\tau_d = 0.50\,\text{m}$.



(d) MOTA, $\tau_d = 0.75\,\text{m}$.

Figure 5.19: Performance metrics MOTP (a,b) and MOTA (c,d) of the compared approaches on the real-world datasets for the different distance thresholds τ_d. Note that the results for KSP on top of standard POM in combination with radio localization events is only available for the multi-modal MVL Lab5 dataset.

One of the major reasons for the significantly lower performance of the KSP tracker in the more complex scenarios is that the POM detector suffers from the various challenges of the evaluated datasets. In particular, as already reported by Alahi et al. [2], we observed a large number of false positives of the POM detector if noisy foreground segmentations are used as input, *e.g.*, caused by changing illumination. Furthermore, in situations where people exhibit challenging poses, missed detections occur frequently. In such situations, the KSP tracker is often not able to link the true positive detections correctly or starts drifting after several frames of missed detections. These issues especially become evident by the significantly lower tracking accuracy at the APIDIS and Pose scenarios. In contrast to KSP, our proposed approach is able to handle such complex poses and articulations more robustly by exploiting the volumetric information. Furthermore,

(a) CLEAR metrics.

τ_d [m]	**Algorithm**	**MOTP [m]**	**MOTA**
0.50	Proposed	**0.205**	**0.675**
	Prop. w/o Color	0.211	0.597
	KSP+POM	0.231	0.490
0.75	Proposed	**0.226**	**0.780**
	Prop. w/o Color	0.235	0.675
	KSP+POM	0.277	0.639

(b) Absolute performance values.

τ_d [m]	**Algorithm**	**TP**	**FP**	**FN**	**IDS**
0.50	Proposed	**656**	**88**	**172**	**9**
	Prop. w/o Color	625	121	202	10
	KSP+POM	607	156	220	46
0.75	Proposed	**699**	**47**	**130**	**5**
	Prop. w/o Color	658	88	170	11
	KSP+POM	630	64	198	37

Table 5.5: Tracking results (a,b) of the evaluated approaches on the APIDIS dataset. The best scores are highlighted.

our tracking algorithm is not sensitive to noisy foreground segmentation, as we consider the local neighborhoods of the visual hull reconstruction for computing the occupancy volume.

Another drawback of the KSP tracker in combination with POM detections is the lack of incorporating color information into the trajectory linking process. This becomes apparent by the high number of identity switches, especially in densely crowded scenarios but also in situations where people move very close to each other, as demonstrated in Figure 5.20. Therefore, to allow a fair comparison, we evaluated the proposed approach without discriminative appearance models for resolving geometrically ambiguous situations, *i.e.*, trajectory assignment is solely based on the geometric information derived from the occupancy volume. The corresponding tracking results are reported as *Prop. w/o Color*. Although only relying on the binary foreground segmentation, we achieve similar performances compared to the KSP approach on moderately complex scenarios, and even outperform it significantly on datasets, which exhibit challenging poses and out-of-plane motion (*i.e.*, APIDIS, MUCH, and POSE), as the robust occupancy volume provides a valuable cue for tracking.

By additionally using a discriminative classifier to resolve these ambiguous situations, we achieve excellent tracking results, especially *w.r.t.* the number of identity switches. This advantage of the proposed approach can be seen in situations, where people move very close to each other, *e.g.*,

(a) CLEAR metrics.

τ_d [m]	**Algorithm**	**MOTP [m]**	**MOTA**
0.50	Proposed	**0.129**	0.695
	Prop. w/o Color	0.131	0.554
	KSP+POM	0.151	0.861
	KSP+POM/radio	0.152	**0.906**
0.75	Proposed	**0.156**	0.746
	Prop. w/o Color	0.156	0.609
	KSP+POM	0.173	0.937
	KSP+POM/radio	0.167	**0.969**

(b) Absolute performance values.

τ_d [m]	**Algorithm**	**TP**	**FP**	**FN**	**IDS**
0.50	Proposed	14231	3191	1528	92
	Prop. w/o Color	14244	5422	1515	86
	KSP+POM	18809	1396	1396	**18**
	KSP+POM/radio	**19272**	**938**	**933**	22
0.75	Proposed	14774	2828	1126	82
	Prop. w/o Color	14886	5085	1065	88
	KSP+POM	19583	646	622	12
	KSP+POM/radio	**19889**	**316**	**316**	**4**

Table 5.6: Tracking results (a,b) for performance comparison on the MVL Lab5 dataset. Note that the results for comparison (KSP+POM and KSP+POM/radio) have been provided by the authors of [88]. The best scores are highlighted.

	Wiper	**Chap**	**Leaf 1**	**Leaf 2**	**Much**
N_A	3	9	14	48	12
N_R	2	7	12	47	12

	Pose	**Table**	**APIDIS**	**MVL Lab5**
N_A	12	34	27	222
N_R	10	32	23	171

Table 5.7: Resolving geometric ambiguities. For each dataset, we state the number of ambiguous situations N_A (*i.e.*, how often individuals cannot be identified solely based on the geometric information), as well as the number of resolved conflicts N_R. Note that a conflict is considered resolved, whenever all corresponding individuals are re-identified with a high confidence of the corresponding classifiers.

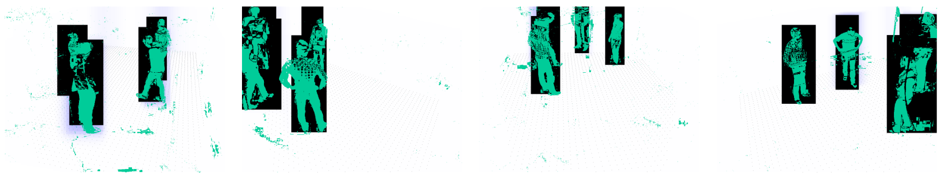
APIDIS, Much, Leaf 1 & 2, or Table. Note that the identity switches of the Table scenario occur when the individuals form a crowd at the table. As the corresponding visual hull reconstructions do not allow separating each individual, the tracker shortly drifts until the people disperse and the previous identities can be assigned correctly (recall the corresponding illustrative tracking results in Figure 5.14). Thus, the identity switches are caused by the drifting trackers, as indicated by the better performance results when considering a higher distance threshold of $\tau_d = 0.75\,\mathrm{m}$.

Furthermore, as we collect and update the appearance information of the individuals on-line by exploiting the 3D scene structure, we are able to robustly handle situations where people change their visual appearance significantly, *e.g.*, demonstrated by the Chap scenario. Note that the single identity switch at this sequence occurs after a person leaves the tracking region, changes his clothes outside, and then re-enters the scene.

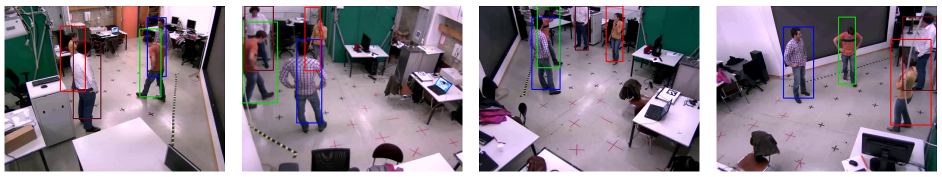
The importance of incorporating appearance information for consistent labeling of the trajectories is also indicated by Table 5.7, which lists the number of geometrically ambiguous situations in relation to confidently resolved conflicts. As can be seen, the appearance information is required to robustly re-identify the individuals once geometric cues can no longer be used to distinguish between them. This is also confirmed by the significantly higher tracking performance of the proposed approach compared to the evaluation without using appearance information (*i.e.*, *Prop. w/o Color*).

Considering the tracking precision, the proposed approach achieves very accurate results, *i.e.*, the average distance between real object positions and estimated positions is approximately 0.1 m. Since the KSP tracker is based on a discretized top view representation, it is constrained by the spatial resolution of the grid. However, as it operates offline on a graph built over all frames, it cannot handle arbitrarily dense grids, due to memory limitations. In contrast, our voxel based approach operates on-line and can be used with high resolution volumes, *i.e.*, we set the voxel size to 1cm $\times$ 1cm $\times$ 5cm for the evaluated scenarios.

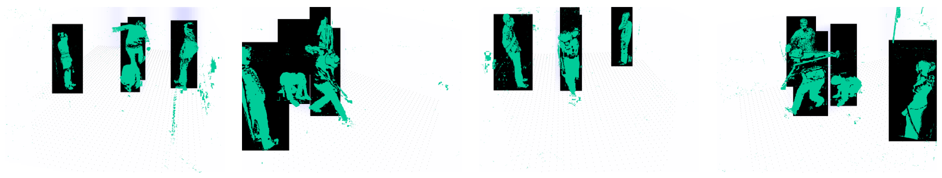
As discussed in Section 5.3.3, the APIDIS basketball dataset depicts a complex scenario for object tracking approaches. However, as can be seen from the tracking results, we still achieve very accurate and precise tracking results, despite the challenges caused by shadowing effects and heavy reflections, as well as the complex and fast movement of the players. Since the POM detector is more sensitive to the noisy foreground segmentation of this dataset, the KSP tracker is significantly outperformed by the proposed approach. Nevertheless, although the on-line sample collection facilitates correctly tracking players of different teams, identity switches occur due to the similar appearance of players within a team. Therefore, more discriminative visual features specifically designed for sports analysis, *e.g.*, obtained via jersey number recognition (*e.g.*, as shown by [119]), may further improve the overall performance on this scenario.



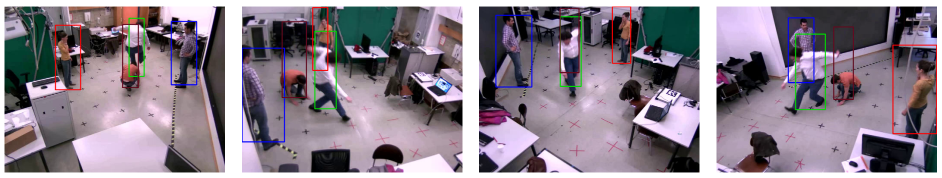
(a) POM detections, $t = 1200$.



(b) KSP tracking results, $t = 1200$.



(c) POM detections, $t = 1540$.



(d) KSP tracking results, $t = 1540$.

Figure 5.20: Illustrative tracking results of the state-of-the-art KSP tracker in combination with POM detections. As demonstrated by this leapfrog exercise of the LEAF 1 dataset, identities of the individuals cannot be kept consistently. Note that black rectangles at the POM detection results (a,c) denote a high probability that a person is present at the corresponding location. The colors of the re-projected KSP tracking results (b,d) correspond to the identities of the tracker hypotheses.

	Proposed	Prop. w/o Color	KSP	POM	KSP +POM
Wiper	5.10	-	-	-	-
Chap	9.89	12.67	43.49	0.02	0.02
Leaf 1	9.88	10.34	63.84	0.04	0.04
Leaf 2	7.65	9.04	229.77	0.05	0.05
Much	12.08	13.21	185.28	0.06	0.06
Pose	10.27	12.99	132.49	0.05	0.05
Table	8.03	9.60	208.51	0.07	0.07
APIDIS	4.42	6.16	80.70	0.03	0.03
MVL Lab5	8.23	12.91	-	-	-

Table 5.8: Runtime performance of the evaluated approaches, measured in frames per second (fps). Note that the spatial grid density is a major runtime factor for both, KSP and POM. Therefore, these runtimes may vary.

5.5 Runtime Analysis

Table 5.8 lists the runtime performance in frames per second (fps) evaluated on a standard PC with a 3.2 GHz Intel CPU, 16 GB RAM, and a GeForce GTX580. We achieve frame rates of up to 12 fps for standard tracking scenarios, although only the visual hull reconstruction and the occupancy volume are computed on the GPU, exploiting the inherent parallelism. Therefore, the current performance suggests additional speed-up, if the specific steps of the proposed algorithm are further optimized.

The major bottleneck of the proposed approach is obviously caused by the size of the underlying volume. Therefore, depending on the extent of the tracking region in combination with the spatial resolution of the voxels, the runtime performance of the proposed approach varies between the differently scaled scenes, *e.g.*, the reconstruction volume used for the APIDIS dataset is approximately 8 times larger than for the ICG laboratory scenarios. Note that we limited the vertical extent of the tracking region to $z_{\max} = 3.5\,\mathrm{m}$ in order to robustly handle out-of-plane motions, such as jumping people. Furthermore, the resolution of the input images affects the runtime performance, as we need to re-project the tracking results into the individual camera views to extract the appearance information on-line. Nevertheless, we achieve near real-time performance on all evaluated datasets, despite the high resolution input images and the large tracking regions.

The KSP tracker achieves very high frame rates due to the efficient shortest path computation. We report the runtimes for those KSP/POM configurations, which achieve the best tracking performance. Thus, the reported frame rates vary for scenarios with similar input data, as the KSP runtime depends on the spatial grid density.

In contrast, the POM detector exhibits a significantly lower frame rate caused by the high resolution of the input images, as well as the required parameter configurations to handle the noisy foreground segmentation. Thus, the combination of KSP and POM does not achieve real-time capability on the evaluated scenarios. Furthermore, the KSP tracker requires the detection probabilities for all time steps in advance for constructing the underlying graph structure. Thus, it can only be computed off-line, whereas the proposed approach works completely on-line at high frame rates.

Chapter 6

Conclusion

Contents

In the following, we conclude this thesis and provide an outlook on future work encouraged by the experimental results so far.

6.1 Summary

In this thesis, we proposed a real-time capable approach for multi-object tracking from a calibrated camera network with partially overlapping views. The primary cue to locate the objects of interest is 3D information obtained from visual hull reconstructions. Therefore, we demonstrated how the local mass densities of the visual hull can be exploited to robustly obtain location estimates for multi-object tracking. Additionally, we incorporate discriminative appearance information on-demand to provide a consistent labeling in geometrically ambiguous situations.

In particular, we use binary foreground masks obtained via standard background subtraction as input for a coarse 3D reconstruction via shape from silhouette, *i.e.*, we reconstruct the objects' visual hull. In order to reduce the effect of ghost artifacts and to robustly handle noisy foreground masks, we introduce the *occupancy volume*, based on local mass densities of the visual hull reconstructions. This allows to estimate the location of the object's mass center and thus, provides a valuable cue for tracking.

Therefore, we track objects by exploiting the local mass density scores within a particle filtering approach. In particular, each individual object is tracked by a separate particle filter. However, in contrast to most multi-object tracking approaches, we constrain nearby trackers by a Voronoi partitioning, inspired by [75]. In contrast to common iterative particle repulsion schemes (*e.g.*, [112]), this allows to efficiently handle situations where the tracked objects move very close to each other. Furthermore, we collect appearance information on-line, which is used on-demand to train discriminative classifiers if pure geometric information cannot be used to reliably distinguish the objects.

We demonstrated our tracking approach on several challenging real-world datasets which exhibit challenging poses, out-of-plane motion, densely crowded scenes, as well as similar appearance of the objects. In particular, we used publicly available multi-person tracking datasets (*i.e.*, APIDIS and MVL Lab5) and additionally captured challenging scenarios at our laboratory (*i.e.*, Chap, Leaf 1 & 2, Much, Pose, and Table). These datasets are also used to compare our approach to the state-of-the-art, *i.e.*, the publicly available KSP tracker [9] in combination with the well-known POM detector [38]. As can be seen from the experimental evaluations, we outperform state-of-the-art methods in terms of precision, accuracy, as well as runtime, on most complex datasets.

Since the KSP tracker does not incorporate color information, we also evaluated our approach solely based on the geometric information derived from the visual hull reconstruction, *i.e.*, appearance information is not used to resolve geometric ambiguities. As shown by our experimental evaluations, the geometric information alone provides a valuable cue for multi-object tracking. In fact, we achieve state-of-the-art performance by only exploiting the geometric information for tracking. However, there are situations where a consistent assignment of the tracker hypotheses to the objects cannot be guaranteed solely on the geometric information, *e.g.*, if objects move very close to each other, such as in an attack scenario of a sports game.

Therefore, we additionally incorporate discriminative appearance information, *i.e.*, we exploit the 3D scene structure to robustly extract hue-saturation histograms to describe the objects' appearance. We then resort to this additional cue whenever geometric information cannot be used to reliably identify the objects. Furthermore, note that the proposed approach is able to handle scenarios where the objects change their visual appearance, since we resort to the color information solely on-demand and update the collected appearance information on a frequent time basis.

Nevertheless, since we rely on 3D information derived from the input images, we require a proper placement of the cameras, which may be considered the major drawback of the proposed approach. For example, if the cameras share similar view-points or large parts of the scene are only visible in opposing camera views, the reconstructed visual hull is usually much larger than the real object, which complicates precise localization. Despite the unsuitable camera placement, *e.g.*, demonstrated by the MVL Lab5 dataset, we are still able to robustly track single objects. However, resolving crowded scenes becomes significantly more difficult in such situations, which manifests itself mainly by a lower tracking accuracy. Therefore, camera placements which lead to considerably inaccurate visual hull reconstructions should be avoided at first, *i.e.*, when setting up the camera network, in order to track at the best performance.

Overall, the experimental evaluations show that our proposed approach is able to robustly track multiple objects in complex scenarios. In particular, using the volumetric reconstructions allows to efficiently handle out-of-plane motion (*i.e.*, object motion is not constrained to the ground-plane), as well as different articulations of the objects (*e.g.*, sitting, jumping, crawling, or kneeling people). By computing the local mass densities from the visual hulls, we obtain a valuable cue for localizing the objects at a high precision and are less sensitive to noisy foreground segmentations compared to related approaches. Furthermore, we can also reliably resolve crowded situations by using on-line collected appearance information. Therefore, the presented results encourage further research which will be summarized in the following.

6.2 Outlook

Although our proposed approach achieves excellent tracking performance in complex environments, future research may improve the tracking performance for specific challenges. For example, the visual hull reconstruction may be improved by incorporating information about the static scene structure, *e.g.*, obtained from a 3D model of the static scene or by using the probabilistic occluder inference approach by Guan et al. [45, 46]. This would allow to identify static occluders, *e.g.*, office furniture, which otherwise cause large parts of the objects' visual hull to be carved away.

Furthermore, the experimental results so far encourage improvements to robustly deal with similar appearance of objects, *e.g.*, as demonstrated by the APIDIS or MVL Lab5 datasets. A possible solution would be to extract different features, specifically designed for the individual tracking problem. For example, jersey number recognition in combination with jersey color extraction allows unique identification of players in a sports game and thus, may improve the tracking accuracy at the APIDIS dataset.

Another possible improvement would be to additionally incorporate pan-tilt-zoom (PTZ) cameras. Since these cameras are very flexible, *i.e.*, they can observe an almost 360° horizontal field-of-view by repositioning the camera head, they may be used to provide additional fields-of-view on-demand. This additional input may reduce the geometric ambiguities on the one hand, and provide high quality imagery for resolving labeling conflicts on the other hand. Therefore, in order to capture these additional information at the right time, a detailed analysis of automatic view-planning methods would be beneficial.

Bibliography

[1] J. K. Aggarwal and Q. Cai. Human Motion Analysis: A Review. *Computer Vision and Image Understanding (CVIU)*, 73(3):428–440, 1999.

[2] Alexandre Alahi, Laurent Jacques, Yannick Boursier, and Pierre Vandergheynst. Sparsity Driven People Localization with a Heterogeneous Network of Cameras. *Journal of Mathematical Imaging and Vision (JMIV)*, 41(1-2):39–58, 2011.

[3] Saad Ali and Mubarak Shah. Floor Fields for Tracking in High Density Crowd Scenes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008.

[4] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing (TSP)*, 50(2):174–188, 2002.

[5] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Robust Object Tracking with Online Multiple Instance Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(7):1324–1338, 2011.

[6] Anup Basu. Active Calibration: Alternative Strategy and Analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1993.

[7] Bruce Guenther Baumgart. *Geometric Modeling for Computer Vision.* PhD thesis, Stanford University, Computer Science Department, 1974.

[8] Jérôme Berclaz, François Fleuret, and Pascal Fua. Robust People Tracking with Global Trajectory Optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[9] Jérôme Berclaz, François Fleuret, Engin Türetken, and Pascal Fua. Multiple Object Tracking using K-Shortest Paths Optimization. *IEEE*

Transactions on Pattern Analysis and Machine Intelligence (PAMI), 33(9):1806–1819, 2011.

[10] Niclas Bergman. *Recursive Bayesian Estimation: Navigation and Tracking Applications*. PhD thesis, Linköping University, Department of Electrical Engineering, 1999.

[11] Keni Bernardin and Rainer Stiefelhagen. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. *European Association for Signal Processing (EURASIP) Journal on Image and Video Processing*, 2008.

[12] Jules Bloomenthal and Jon Rokne. Homogeneous Coordinates. *The Visual Computer*, 11(1):15–26, 1994.

[13] Andrea Bottino, Aldo Laurentini, and P. Zuccone. Toward Non-intrusive Motion Capture. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 1998.

[14] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`, 2010. Online; accessed July 26, 2012.

[15] Michael D. Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Online Multi-Person Tracking-by-Detection from a Single, Uncalibrated Camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(9):1820–1833, 2010.

[16] Duane C. Brown. Decentering Distortion of Lenses. *Photogrammetric Engineering*, 32(3):444–462, 1966.

[17] Duane C. Brown. Close-Range Camera Calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.

[18] Q. Cai and J. K. Aggarwal. Tracking Human Motion in Structured Environments Using a Distributed-Camera System. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 21(11): 1241–1247, 1999.

[19] Yizheng Cai, Nando de Freitas, and James J. Little. Robust Visual Tracking for Multiple Targets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006.

[20] Fabrice Caillette and Toby Howard. Real-Time Markerless Human Body Tracking with Multi-View 3-D Voxel Reconstruction. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2004.

[21] Thanh-Tung Cao, Ke Tang, Anis Mohamed, and Tiow-Seng Tan. Parallel Banding Algorithm to compute exact distance transform with the GPU. In *ACM Symposium on Interactive 3D Graphics (SI3D)*, 2010.

[22] T. A. Clarke and J. G. Fryer. The Development of Camera Calibration Methods and Models. *The Photogrammetric Record*, 16(91):51–66, 1998.

[23] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-Time Tracking of Non-Rigid Objects using Mean Shift. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[24] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-Based Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(5):564–577, 2003.

[25] Franklin C. Crow. Summed-Area Tables for Texture Mapping. In *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1984.

[26] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[27] Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[28] Lisa Dron. Dynamic Camera Self-Calibration from Controlled Motion Sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1993.

[29] Fenglei Du and Michael Brady. Self-Calibration of the Intrinsic Parameters of Cameras for Active Vision Systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1993.

[30] Wei Du and Justus Piater. Multi-camera People Tracking by Collaborative Particle Filters and Principal Axis-Based Integration. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2007.

[31] Charles R. Dyer. Volumetric scene reconstruction from multiple views. *Foundations of Image Understanding*, pages 469–489, 2001.

[32] Ran Eshel and Yeal Moses. Tracking in a Dense Crowd Using Multiple Cameras. *International Journal of Computer Vision (IJCV)*, 88(1): 129–143, 2010.

[33] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. A Mobile Vision System for Robust Multi-Person Tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[34] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research (JMLR)*, 9:1871–1874, 2008.

[35] Olivier Faugeras. *Three-Dimensional Computer Vision.* MIT Press, 1993.

[36] Paul Fearnhead. *Sequential Monte Carlo methods in filter theory.* PhD thesis, University of Oxford, Department of Statistics, 1998.

[37] Pedro F. Felzenszwalb, David McAllester, and Deva Ramanan. A Discriminatively Trained, Multiscale, Deformable Part Model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[38] François Fleuret, Jérôme Berclaz, Richard Lengagne, and Pascal Fua. Multi-Camera People Tracking with a Probabilistic Occupancy Map. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(2):267–282, 2008.

[39] Pierre F. Gabriel, Jacques G. Verly, Justus H. Piater, and André Genon. The State of the Art in Multiple Object Tracking Under Occlusion in Video Sequences. In *Proceedings of the International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 2003.

[40] Jürgen Gall and Victor Lempitsky. Class-Specific Hough Forests for Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[41] Jürgen Gall, Nima Razavi, and Luc Van Gool. On-line Adaption of Class-specific Codebooks for Instance Tracking. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2010.

[42] Dariu M. Gavrila. The Visual Analysis of Human Movement: A Survey. *Computer Vision and Image Understanding (CVIU)*, 73(1):82–98, 1999.

[43] Martin Godec, Peter M. Roth, and Horst Bischof. Hough-based Tracking of Non-Rigid Objects. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[44] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *Radar & Signal Processing*, 140(2):107–113, 1993.

[45] Li Guan, Jean-Sébastien Franco, and Marc Pollefeys. 3D Occlusion Inference from Silhouette Cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[46] Li Guan, Jean-Sébastien Franco, and Marc Pollefeys. Multi-Object Shape Estimation and Tracking from Silhouette Cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[47] Abhinav Gupta, Anurag Mittal, and Larry S. Davis. COST: An Approach for Camera Selection and Multi-Object Inference Ordering in Dynamic Scenes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007.

[48] Mei Han, Wei Xu, Hai Tao, and Yihong Gong. An Algorithm for Multiple Object Trajectory Tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[49] Ismail Haritaoglu, David Harwood, and Larry S. Davis. W^4: Who? When? Where? What? A Real Time System for Detecting and Tracking People. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition (AFGR)*, 1998.

[50] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference*, 1988.

[51] Richard Hartley. Self-Calibration of Stationary Cameras. *International Journal of Computer Vision (IJCV)*, 22(1):5–23, 1997.

[52] Richard Hartley. Chirality. *International Journal of Computer Vision (IJCV)*, 26(1):41–61, 1998.

[53] Richard Hartley and Peter Sturm. Triangulation. *Computer Vision and Image Understanding (CVIU)*, 68(2):146–157, 1997.

[54] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, second edition, 2004.

[55] Janne Heikkilä and Olli Silvén. A Four-step Camera Calibration Procedure with Implicit Image Correction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1997.

[56] Berthold K. P. Horn. *Shape from Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from One View.* PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering, 1970.

[57] Min Hu, Jianguang Lou, Weiming Hu, and Tieniu Tan. Multicamera Correspondence Based on Principal Axis of Human Body. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2004.

[58] Stephen S. Intille and Aaron F. Bobick. Visual Tracking Using Closed-Worlds. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1995.

[59] Michael Isard and Andrew Blake. CONDENSATION - Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision (IJCV)*, 29(1):5–28, 1998.

[60] Michael Isard and John Philip MacCormick. BraMBLe: A Bayesian Multiple-Blob Tracker. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2001.

[61] Anil K. Jain and Aditya Vailaya. Image Retrieval using Color and Shape. *Pattern Recognition (PR)*, 29(8):1233–1244, 1996.

[62] Rudolf Emil Kálmán. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[63] Keiji Kanazawa, Daphne Koller, and Stuart Russel. Stochastic simulation algorithms for dynamic probabilistic networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 1995.

[64] Rangachar Kasturi, Dmitry Goldgof, Padmanabhan Soundararajan, Vasant Manohar, John Garofolo, Rachel Bowers, Matthew Boonstra, Valentina Korzhova, and Jing Zhang. Framework for Performance Evaluation of Face, Text, and Vehicle Detection and Tracking in Video: Data, Metrics, and Protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(2):319–336, 2009.

[65] Vera Kettnaker and Ramin Zabih. Bayesian Multi-camera Surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[66] Saad M. Khan and Mubarak Shah. A Multiview Approach to Tracking People in Crowded Scenes using a Planar Homography Constraint. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006.

[67] Saad M. Khan and Mubarak Shah. Tracking Multiple Occluding People by Localizing on Multiple Scene Planes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(3):505–519, 2009.

[68] Sohaib Khan, Omar Javed, Zeeshan Rasheed, and Mubarak Shah. Human Tracking in Multiple Cameras. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2001.

[69] Zia Khan, Tucker Balch, and Frank Dellaert. MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(11):1805–1819, 2005.

[70] Kyungnam Kim and Larry S. Davis. Multi-camera Tracking and Segmentation of Occluded People on Ground Plane Using Search-Guided Particle Filtering. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006.

[71] Genshiro Kitagawa. Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models. *Journal of Computational and Graphical Statistics*, 5(1):346–351, 1996.

[72] Kalin Kolev, Thomas Brox, and Daniel Cremers. Fast Joint Estimation of Silhouettes and Dense 3D Geometry from Multiple Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(3):493–505, 2012.

[73] Nils Krahnstoever and Paulo Mendonça. Autocalibration from Tracks of Walking People. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2006.

[74] Matej Kristan. *Tracking people in video data using probabilistic models.* PhD thesis, University of Ljubljana, Faculty of Electrical Engineering, 2008.

[75] Matej Kristan, Janez Perš, Matej Perše, and Stanislav Kovačič. Closed-world tracking of multiple interacting targets for indoor-sports applications. *Computer Vision and Image Understanding (CVIU)*, 113 (5):598–611, 2009.

[76] Eric Krotkov. Focusing. *International Journal of Computer Vision (IJCV)*, 1(3):223–237, 1988.

[77] John Krumm, Steve Harris, Brian Meyers, Barry Brumitt, Michael Hale, and Steve Shafer. Multi-Camera Multi-Person Tracking for EasyLiving. In *Proceedings of the IEEE International Workshop on Visual Surveillance (VS)*, 2000.

[78] Cheng-Hao Kuo and Ram Nevatia. How does person identity recognition help multi-person tracking? *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[79] Kiriakos N. Kutulakos and Steven M. Seitz. A Theory of Shape by Space Carving. *International Journal of Computer Vision (IJCV)*, 38 (3):199–218, 2000.

[80] Aldo Laurentini. The Visual Hull Concept for Silhouette-Based Image Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 16(2):150–162, 1994.

[81] Aldo Laurentini. How Far 3D Shapes Can Be Understood from 2D Silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 17(2):188–195, 1995.

[82] Martijn Liem and Dariu M. Gavrila. Multi-person tracking with overlapping cameras in complex, dynamic environments. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.

[83] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust Region Newton Methods for Large-Scale Logistic Regression. *Journal of Machine Learning Research (JMLR)*, 9:627–650, 2008.

[84] Jun S. Liu and Rong Chen. Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

[85] David Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

[86] John Philip MacCormick. *Probabilistic modelling and stochastic algorithms for visual localisation and tracking.* PhD thesis, University of Oxford, Department of Engineering Science, 2000.

[87] Rok Mandeljc, Stanislav Kovačič, Matej Kristan, and Janez Perš. Non-Sequential Multi-View Detection, Localization and Identification of People using Multi-Modal Feature Maps. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2012.

[88] Rok Mandeljc, Stanislav Kovačič, Matej Kristan, and Janez Perš. Tracking by Identification Using Computer Vision and Radio. *SENSORS*, 13(1):241–273, 2013.

[89] Worthy N. Martin and J. K. Aggarwal. Volumetric Descriptions of Objects from Multiple Views. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 5(2):150–158, 1983.

[90] Jiří Matas, Ondrej Chum, Martin Urban, and Tomáš Pajdla. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2002.

[91] Stehpen J. Maybank and Olivier D. Faugeras. A Theory of Self-Calibration of a Moving Camera. *International Journal of Computer Vision (IJCV)*, 8(2):123–151, 1992.

[92] Nigel J. B. McFarlane and Charles Patrick Schofield. Segmentation and tracking of piglets in images. *Machine Vision and Applications (MVA)*, 8(3):187–193, 1995.

[93] Brice Michoud, Saïda Bouakaz, Erwan Guillou, and Hector Briceño Pulido. Largest Silhouette-Equivalent Volume for 3D Shapes Modeling without Ghost Object. In *Proceedings of the Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications (M2SFA2)*, 2008. (in conj. ECCV).

[94] Brice Michoud, Erwan Guillou, Saïda Bouakaz, Mathieu Barnachon, and Alexandre Meyer. Towards Removing Ghost-Components from Visual-Hull Estimations. In *Proceedings of the IEEE International Conference on Image and Graphics (ICIG)*, 2009.

[95] Ivana Mikić, Mahan Trivedi, Edward Hunter, and Pamela Cosman. Human Body Model Acquisition and Tracking Using Voxel Data. *International Journal of Computer Vision (IJCV)*, 53(3):199–223, 2003.

[96] Gregor Miller and Adrian Hilton. Safe Hulls. In *Proceedings of the IEEE European Conference on Visual Media Production (CVMP)*, 2007.

[97] Anurag Mittal and Larry S. Davis. Unified Multi-camera Detection and Tracking Using Region-Matching. In *Proceedings of the IEEE Workshop on Multi-Object Tracking*, 2001.

[98] Anurag Mittal and Larry S. Davis. M_2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene. *International Journal of Computer Vision (IJCV)*, 51(3):189–203, 2003.

[99] Thomas B. Moeslund and Erik Granum. A Survey of Computer Vision-Based Human Motion Capture. *Computer Vision and Image Understanding (CVIU)*, 81(3):231–268, 2001.

[100] Shree K. Nayar and Yasuo Nakagawa. Shape from focus. Technical Report CMU-RI-TR-89-27, Carnegie Mellon University, The Robotics Institute, 1989.

[101] Shree K. Nayar and Yasuo Nakagawa. Shape from Focus. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 16(8): 824–831, 1994.

[102] Alexander Neubeck and Luc Van Gool. Efficient Non-Maximum Suppression. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2006.

[103] Katja Nummiaro, Esther Koller-Meier, and Luc Van Gool. Color Features for Tracking Non-Rigid Objects. *Chinese Journal of Automation*, 29:345–355, 2003.

[104] Kenji Okuma, Ali Taleghani, Nando de Freitas, James J. Little, and David G. Lowe. A Boosted Particle Filter: Multitarget Detection and Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2004.

[105] Kazuhiro Otsuka and Naoki Mukawa. Multiview Occlusion Analysis for Tracking Densely Populated Objects Based on 2-D Visual Angles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[106] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll Never Walk Alone: Modeling Social Behavior for Multi-target Tracking. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

[107] Peixi Peng, Yonghong Tian, Yaowei Wang, and Tiejun Huang. Multi-camera Pedestrian Detection with a Multi-view Bayesian Network Model. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2012.

[108] Patrick Pérez, Carine Hue, Jaco Vermaak, and Michel Gangnet. Color-Based Probabilistic Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2002.

[109] Fatih Porikli. Achieving Real-Time Object Detection and Tracking Under Extreme Conditions. *Journal of Real-Time Image Processing*, 1(1):33–40, 2006.

[110] Horst Possegger, Matthias Rüther, Sabine Sternig, Thomas Mauthner, Manfred Klopschitz, Peter M. Roth, and Horst Bischof. Unsupervised Calibration of Camera Networks and Virtual PTZ Cameras. In *Proceedings of the Computer Vision Winter Workshop (CVWW)*, 2012.

[111] Surya Prakash and Antonio Robles-Kelly. A Semi-supervised Approach to Space Carving. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2010.

[112] Wei Qu, Dan Schonfeld, and Magdi Mohamed. Real-Time Interactively Distributed Multi-Object Tracking Using a Magnetic-Inertia Potential Model. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005.

[113] Markus Quaritsch, Markus Kreuzthaler, Bernhard Rinner, Horst Bischof, and Bernhard Strobl. Autonomous Multicamera Tracking on Embedded Smart Cameras. *European Association for Signal Processing (EURASIP) Journal on Embedded Systems*, 2007.

[114] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision (IJCV)*, 40(2):99–121, 2000.

[115] Paul Scovanner and Marshall F. Tappen. Learning Pedestrian Dynamics from the Real World. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

[116] Steven M. Seitz and Charles R. Dyer. Photorealistic Scene Reconstruction by Voxel Coloring. *International Journal of Computer Vision (IJCV)*, 35(2):151–173, 1999.

[117] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[118] Yongduek Seo, Sunghoon Choi, Hyunwoo Kim, and Ki-Sang Hong. Where are the ball and players? Soccer Game Analysis with Color-based Tracking and Image Mosaick. In *Proceedings of the International Conference on Image Analysis and Processing (ICIAP)*, 1997.

[119] Horesh Ben Shitrit, Jérôme Berclaz, François Fleuret, and Pascal Fua. Tracking Multiple People under Global Appearance Constraints. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[120] Sudipta N. Sinha and Marc Pollefeys. Camera Network Calibration and Synchronization from Silhouettes in Archived Video. *International Journal of Computer Vision (IJCV)*, 87(3):266–283, 2010.

[121] Greg Slabaugh, Bruce Culbertson, Tom Malzbender, and Ron Schafer. A Survey of Methods for Volumetric Scene Reconstruction from Photographs. In *Proceedings of the International Workshop on Volume Graphics*, 2001.

[122] Kevin Smith, Daniel Gatica-Perez, and Jean-Marc Odobez. Using Particles to Track Varying Numbers of Interacting People. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[123] Dan Snow, Paul Viola, and Ramin Zabih. Exact Voxel Occupancy with Graph Cuts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[124] Sabine Sternig, Thomas Mauthner, Arnold Irschara, Peter M. Roth, and Horst Bischof. Multi-camera Multi-object Tracking by Robust Hough-based Homography Projections. In *Proceedings of the IEEE International Workshop on Visual Surveillance (VS)*, 2011. (in conj. CVPR).

[125] Ivan E. Sutherland. Three-Dimensional Data Input by Tablet. *Proceedings of the IEEE*, 62(4):453–461, 1974.

[126] Tomáš Svoboda, Daniel Martinec, and Tomáš Pajdla. A Convenient Multi-Camera Self-Calibration for Virtual Environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422, 2005.

[127] Michael J. Swain and Dana H. Ballard. Color Indexing. *International Journal of Computer Vision (IJCV)*, 7(1):11–32, 1991.

[128] Richard Szeliski. Rapid Octree Construction from Image Sequences. *Computer Vision, Graphics, and Image Processing (CVGIP): Image Understanding*, 58(1):23–32, 1993.

[129] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, first edition, 2010.

[130] Christian Theobalt, Marcus Magnor, Pascal Schüler, and Hans-Peter Seidel. Combining 2D Feature Tracking and Volume Reconstruction for Online Video-Based Human Motion Capture. In *Proceedings of the Pacific Conference on Computer Graphics and Applications*, 2002.

[131] Roger Y. Tsai. A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses. *IEEE Journal of Robotics and Automation*, 3(4): 323–344, 1987.

[132] Jaco Vermaak, Arnaud Doucet, and Patrick Perez. Maintaining Multi-Modality through Mixture Tracking. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003.

[133] Mike West and Jeff Harrison. *Bayesian Forecasting and Dynamic Models*. Springer, second edition, 1997.

[134] Robert J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19(1):139–144, 1980.

[135] Bo Wu and Ram Nevatia. Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet based Part Detectors. *International Journal of Computer Vision (IJCV)*, 75 (2):247–266, 2007.

[136] Danny B. Yang, Héctor H. González-Baños, and Leonidas J. Guibas. Counting People in Crowds with a Real-Time Network of Simple Image Sensors. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003.

[137] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object Tracking: A Survey. *ACM Journal of Computing Surveys (CSUR)*, 38(4):Article No. 13, 2006.

[138] Qian Yu, Gérard Medioni, and Isaac Cohen. Multiple Target Tracking Using Spatio-Temporal Markov Chain Monte Carlo Data Association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[139] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from Shading: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 21(8):690–706, 1999.

[140] Zhengyou Zhang. Flexible Camera Calibration by Viewing a Plane from Unknown Orientations. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1999.

[141] Tao Zhao and Ram Nevatia. Tracking Multiple Humans in Crowded Environment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

Acronyms

APIDIS	Autonomous Production of Images based on Distributed and Intelligent Sensing
CHAP	Changing Appearance
CLEAR	Classification of Events, Activities, and Relationships
CUDA	Compute Unified Device Architecture
EM	Expectation-Maximization
EMD	Earth Mover's Distance
FIFO	First-In-First-Out
FOV	Field-of-View
FPS	Frames per Second
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
ICG	Institute for Computer Graphics and Vision
KSP	K-Shortest Path
LEAF	Leapfrog
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
MOT	Multiple Object Tracking
MOTA	Multiple Object Tracking Accuracy
MOTP	Multiple Object Tracking Precision
MSER	Maximally Stable Extremal Regions

MUCH	Musical Chairs
MVL	Machine Vision Laboratory
NMS	Non-Maxima Suppresion
PDF	Probability Density Function
POM	Probabilistic Occupancy Map
PTZ	Pan-Tilt-Zoom
R2T2	Robust Real-Time Tracking
SFS	Shape from Silhouette
SIS	Sequential Importance Sampling
SVD	Singular Value Decomposition