# Marker-less Motion Capture in General Scenes with Sparse Multi-camera Setups

Ahmed Elhayek

Saarbrücken, Germany

Dissertation
zur Erlangung des Grades des
*Doktors der Ingenieurswissenschaften (Dr.-Ing.)*
der Naturwissenschaftlich-Technischen Fakultäten
der Universität des Saarlandes

March 2015

Dekan - Dean:

Prof. Dr. Markus Bläser
Saarland University
Saarbrücken, Germany

Kolloquiums - Defense

Datum - Date
December 9, 2015, in Saarbrücken

Vorsitzender - Head of Colloquium:
Prof. Dr. Philipp Slusallek

Prüfer - Examiners:

Prof. Dr. Christian Theobalt

Prof. Dr. Hans-Peter Seidel

Protokoll - Reporter:

Dr. Michael Zollhoefer

*To My Mother, Father, Wife and Daughters*

# Abstract

Human motion-capture from videos is one of the fundamental problems in computer vision and computer graphics. Its applications can be found in a wide range of industries. Even with all the developments in the past years, industry and academia alike still rely on complex and expensive marker-based systems. Many state-of-the-art marker-less motion-capture methods come close to the performance of marker-based algorithms, but only when recording in highly controlled studio environments with exactly synchronized, static and sufficiently many cameras. While relative to marker-based systems, this yields an easier apparatus with a reduced setup time, the hurdles towards practical application are still large and the costs are considerable. By being constrained to a controlled studio, marker-less methods fail to fully play out their advantage of being able to capture scenes without actively modifying them.

In the area of marker-less human motion-capture, this thesis proposes several novel algorithms for simplifying the motion-capture to be applicable in new general outdoor scenes. The first is an optical multi-video synchronization method which achieves subframe accuracy in general scenes. In this step, the synchronization parameters of multiple videos are estimated. Then, we propose a spatio-temporal motion-capture method which uses the synchronization parameters for accurate motion-capture with unsynchronized cameras. Afterwards, we propose a motion capture method that works with moving cameras, where multiple people are tracked even in front of cluttered and dynamic backgrounds with potentially moving cameras. Finally, we reduce the number of cameras employed by proposing a novel motion-capture method which uses as few as two cameras to capture high-quality motion in general environments, even outdoors. The methods proposed in this thesis can be adopted in many practical applications to achieve similar performance as complex motion-capture studios with a few consumer-grade cameras, such as mobile phones or GoPros, even for uncontrolled outdoor scenes.

# Kurzfassung

Die videobasierte Bewegungserfassung (Motion Capture) menschlicher Darsteller ist ein fundamentales Problem in Computer Vision und Computergrafik, das in einer Vielzahl von Branchen Anwendung findet. Trotz des Fortschritts der letzten Jahre verlassen sich Wirtschaft und Wissenschaft noch immer auf komplexe und teure markerbasierte Systeme. Viele aktuelle markerlose Motion-Capture-Verfahren kommen der Leistung von markerbasierten Algorithmen nahe, aber nur bei Aufnahmen in stark kontrollierten Studio-Umgebungen mit genügend genau synchronisierten, statischen Kameras. Im Vergleich zu markerbasierten Systemen wird der Aufbau erheblich vereinfacht, was Zeit beim Aufbau spart, aber die Hürden für die praktische Anwendung sind noch immer groß und die Kosten beträchtlich. Durch die Beschränkung auf ein kontrolliertes Studio können markerlose Verfahren nicht vollständig ihren Vorteil ausspielen, Szenen aufzunehmen zu können, ohne sie aktiv zu verändern.

Diese Arbeit schlägt mehrere neuartige markerlose Motion-Capture-Verfahren vor, welche die Erfassung menschlicher Darsteller in allgemeinen Außenaufnahmen vereinfachen. Das erste ist ein optisches Videosynchronisierungsverfahren, welches die Synchronisationsparameter mehrerer Videos genauer als die Bildwiederholrate schätzt. Anschließend wird ein Raum-Zeit-Motion-Capture-Verfahren vorgeschlagen, welches die Synchronisationsparameter für präzises Motion Capture mit nicht synchronisierten Kameras verwendet. Außerdem wird ein Motion-Capture-Verfahren für bewegliche Kameras vorgestellt, das mehrere Menschen auch vor unübersichtlichen und dynamischen Hintergründen erfasst. Schließlich wird die Anzahl der erforderlichen Kameras durch ein neues Motion-Capture-Verfahren, auf lediglich zwei Kameras reduziert, um Bewegungen qualitativ hochwertig auch in allgemeinen Umgebungen wie im Freien zu erfassen. Die in dieser Arbeit vorgeschlagenen Verfahren können in viele praktische Anwendungen übernommen werden, um eine ähnliche Leistung wie komplexe Motion-Capture-Studios mit lediglich einigen Videokameras der Verbraucherklasse, zum Beispiel Mobiltelefonen oder GoPros, auch in unkontrollierten Außenaufnahmen zu erzielen.

# Acknowledgements

First and foremost, I would like to thank my parents Abedelnaseer and Najwa for their continuous support, and my wife Mariam for her patience all the time.

I would like to express my sincere gratitude to my supervisor, Prof. Dr. Christian Theobalt, for introducing me to the topic of marker-less motion capture, for his guidance and support throughout my PhD, and for helping me to build my research skills. I am proud of being part of his group, the graphics, vision and video group.

I would like to thank Prof. Dr. Hans-Peter Seidel and the Max-Planck society for providing such a nice and open working environment, where one has various sources of inspiration and opportunities to interact and cooperate with so many researchers from different countries and fields of research.

I am also thankful to Dr. Carsten Stoll and Prof. Edilson de Aguiar who were my mentors. Their endless patience in answering my technical questions, and their help in writing and revising manuscripts, make my PhD much less difficult.

I also owe special gratitude to other research collaborators: Prof. Bernt Schiele, Prof. Chris Bregler, Dr. Kwang In Kim, Dr. Mykhaylo Andriluka, Dr. Arjun Jain, Dr. Jonathan Tompson, Dr. Nils Hasler and Leonid Pishchulin for their support in my projects. I want to use the opportunity to thank my colleagues Christian Richardt, Michael Zollhöfer, Helge Rhodin, Pablo Garrido, Nadia Robertini, and Srinath Sridhar for proofreading parts of this thesis. I would also like to thank Andreas Baak, Helge Rhodin, and Nils Hasler for allowing us to record their performances and use them for research projects.

I would also like to express my sincere thanks to the administrative staff members, Sabine Budde and Ellen Fries from MPI. They are always kind and generous in supporting me with their professional work. Many thanks to my office-mate Chenglei Wu. It was great fun to share the office with him. Furthermore, I owe thanks to all my colleagues in the computer graphics group at MPI. I am grateful to Mohammed Shaheen and other friends who make my life in Germany very interesting.

# Contents

# Chapter 1

# Introduction

The last decade has seen significant advances in handheld and mobile camera technology. The widespread use of smart phones facilitated casual capturing and sharing any scenes of interest. This abundance of videos resulted in new opportunities and challenges in computer vision and computer graphics. For instance, there are more chances than ever to capture the same scene with multiple cameras: e.g. street performance captured by several spectators. This can significantly broaden the domain of multi-view computer vision and graphics applications such as marker-less human motion capture of any outdoor scene captured with mobile-phone cameras.

Human motion capture is the process of recording the movement of one or several humans from input video. It is one of the fundamental problems in computer vision and computer graphics and has been researched extensively in the past decades. Applications for these methods can be found in a wide range of industries, from entertainment (movies and games) to biomechanics, in sports, and medical sciences. In computer graphics, motion capture is a widely used way to animate virtual human characters. Real-time capture methods made possible through new sensors such as the Microsoft Kinect have opened up new possibilities for human-computer interaction. However, even with all the developments in the past years, for accurate motion capture, industry and academia alike still rely on marker-based optical systems that require complex and expensive setups of cameras and markers.

Recent years have seen a significant improvement of marker-less skeletal human motion capture algorithms [Moeslund *et al.* (2006); Poppe (2007); Sigal *et al.* (2010)]. Many state-of-the-art methods come close to the performance of marker-based algorithms, but only when recording in highly controlled *studio setups*, where 1) there are sufficiently many exactly synchronized high-quality cameras; 2) each camera is

static and scene motion is due to foreground objects only; 3) the background is not cluttered; 4) lighting is controlled; 5) the main foreground actor is seldom occluded.

While relative to marker-based systems, this yields an easier apparatus with a reduced setup time, the hurdles towards practical application are still significant and the costs are still notable. By being constrained to a controlled studio, marker-less methods fail to fully play out their advantage of being able to capture scenes without actively modifying them. Many practical computer graphics and computer vision applications require motions to be captured on site, i.e. the camera system needs to be brought to the set location, because the motion itself cannot be relocated to a studio. Examples are capturing drivers in cars, motion capture on outdoor film sets, recordings of street performances, and the reconstruction of athletes in the field. In such situations, scenes are often cluttered, and foreground and background may be dynamic. Further on, placement and number of cameras may be starkly constrained, cameras are often not synchronized, and they may (have to) move during recording.

In this thesis, I present new methods which address these algorithmic challenges; namely 1) multi-camera synchronization in general scenes; 2) motion capture with unsynchronized cameras; 3) multi-camera tracking in cluttered scenes with dynamic foreground and background; 4) motion capture with very few cameras. I therefore present novel methods for marker-less 3D skeletal human motion capture that succeed in uncontrolled environments and use only a sparse, heterogeneous and weakly constrained camera setup. This implies that our contributions can be adopted in many practical applications to achieve similar performance as the complex and expensive motion capture studios with just a few consumer-grade cameras (e.g. mobile-phone cameras or consumer-grade action cameras, such as GoPro) even in uncontrolled outdoor scenes. This is a significant advance in the field of human motion capture that we feel is required for unlimited number of future applications in a wide range of industries.

## 1.1 Overview

In this thesis, we propose four new methods for solving challenging computer vision and computer graphics problems which are related to generalizing human motion capture setup:

1. an optical multi-video synchronization method which achieves subframe accuracy in general scenes

2. a spatio-temporal motion capture method which works with unsynchronized cameras

3. a method that allows to perform motion capture with moving cameras

4. a ConvNet (Convolutional neural network) based motion capture method that works with very few cameras

It is important to note that each of these methods is strongly related to its preceding method. In particular, the first method estimates multi-video synchronization parameters while the second method uses these parameters to achieve very high motion capture accuracy with unsynchronized cameras. However, the second method fails if at least some of the cameras are moving which is often the case in general outdoor scenes, which is resolved by the third method. Finally, the fourth method works also with very few cameras, whereas the previous method requires 5 cameras to succeed. As a result of these relations, I consider these four approaches as four consecutive steps toward high-quality human motion capture with few unsynchronized handheld cameras. The methods proposed in this thesis have been presented in international research conferences and journals This thesis presents an extended revision of these methods.

### 1.1.1 Optical Multi-Video Synchronization

Our first step toward a simpler human motion capture setup is to estimate the synchronization parameters of several cameras. In fact, there exist several synchronization algorithms. However, these algorithms are limited to specific scenes, where it is possible to track the objects of interest, or to scenes where the objects show specific motions such as ballistic motion [Wedge *et al.* (2006)]. Some approaches are also limited to synchronizing only two sequences. Therefore, we propose a novel algorithm for temporally synchronizing multiple videos capturing the same dynamic scene; details will be discussed in Chapter 4. This algorithm relies on general image features in the scene and it does not require explicit tracking of any specific object. Since such general features usually exist in any video, our algorithm is applicable to general scenes with any number of objects. Moreover, it achieves estimation of the synchronization parameters with sub-frame accuracy. This algorithm can be equally applied to the multi-video case as well as to the two-video case. However, in the multi-video case, additional robustness is achieved by identifying weakly coupled pairs of cameras and removing them from the evaluation of the energy. This leads to

an automatic generation of a graph representing the cameras and their connectivity. The output of this algorithm is the synchronization parameters (i.e. phase shifts and frame rate ratios) of multiple videos. In the experiments, the algorithm succeeds to synchronize datasets that are difficult to synchronize with previous object-tracking-based synchronization techniques.

The novel algorithmic contribution of this synchronization algorithm over previous work are:

1. A set of criteria to filter out noisy and uninformative feature trajectories and pairs of trajectories .

2. An epipolar feature trajectory matching test.

3. A novel strategy for automatic generation of a graph representing the cameras and their connectivity.

### 1.1.2 Motion Capture with Unsynchronized Cameras

The second step uses the synchronization parameters to achieve high motion tracking accuracy despite the unsynchronized cameras. Hasler *et al.* (2009a) have introduced the first method that performs marker-less motion capture with unsynchronized commodity cameras. However, their approach does not make use of sub-frame timing information and instead aligns all frames to the nearest discrete time step. The motion tracking is then performed in the same way as if the cameras were synchronized. This in turn leads to inaccuracies and a reduction of quality in the final results. To address this problem, we propose a new spatio-temporal method for marker-less motion capture; details will be discussed in Chapter 5. This method reconstructs the pose and motion of a character from a multi-view video sequence without requiring the cameras to be synchronized and without aligning captured frames in time. This makes it possible to reconstruct motion in much higher temporal detail than was possible with previous synchronized approaches. If the cameras are running without enforcing synchronization, more samples would be captured in the temporal domain. Therefore, by purposefully running cameras with different offsets in time it is possible to capture very fast motion even at frame rates that off-the-shelf cameras provide. By design, the proposed energy functional used for model-based generative pose estimation is *smooth*. Thus, the derivatives of any order can be computed analytically, allowing effective optimization. In practice, this algorithm simplifies the capture setup in comparison to previous marker-less

approaches, and it enables reconstruction of much higher temporal detail than synchronized capture methods. Thus, slow cameras can be used to capture very fast motion with only little aliasing.

The novel algorithmic contributions of this spatio-temporal motion capture method are:

1. A novel continuous spatio-temporal energy functional that measures model-to-image alignment at any point in time: rather than estimating discrete pose parameters at each time step, it estimates continuous temporal parameter curves that define the motion of the actor.

2. A new method to penalize non-anatomical pose configurations in the continuous pose-curve space.

### 1.1.3 Motion Capture with Moving Cameras

As a third step toward simple human motion capture setup, we aim to work with handheld cameras. To this end, we capture the skeletal motions of humans using a sparse set of potentially moving cameras in an uncontrolled environment; see Chapter 6 for details. This novel algorithm is able to track multiple people even in front of cluttered and dynamic backgrounds with unsynchronized cameras and with varying image quality and frame rate. The algorithm completely relies on optical information and does not make use of additional sensor information (such as depth images or inertial sensors used in some related approaches). The method simultaneously reconstructs the skeletal pose parameters of multiple actors and the motion of each camera. We demonstrate that this algorithm is essential to deal with scenes where cameras, foreground and background can move, and image-based pre-calibration, for example via structure-from-motion (SfM) [Pollefeys *et al.* (2004); Thormählen *et al.* (2008)], fails. The smooth nature and analytic derivatives of the energy functional used to solve for body and camera pose enable continuous and effective optimization. It also enables the automatic detection of the occlusion of body parts either caused by the same person (self-occlusion) or by other people in the same scene. In our experiments, we show qualitatively and quantitatively against ground truth that this algorithm can capture even complex and fast body motion in cluttered outdoor scenes, and that it succeeds with a wide range of heterogeneous, unsynchronized and moving camera systems (such as mobile-phone or outdoor action camera such as *GoPro*) with varying resolution.

The following novel algorithmic contributions over previous work enable this:

1. A new pose-fitting energy function which estimates each camera's motion together with actor pose. In particular, the following extensions over previous section improve the measurement of model-to-image consistency:
   (a) Support for multi-person/multi-camera tracking
   (b) A two-sided similarity term[1]
   (c) Weighting in HSV color space
   (d) Prior on camera motion (smoothness)
2. The pose estimation scheme is using a new and improved occlusion handling approach.
3. A comprehensive evaluation dataset for quantitative comparison. It comprises multi-view video footage recorded with static and moving cameras, ground-truth camera motion data, as well as reference data from a marker-based motion capture system.

### 1.1.4 Motion Capture with a Low Number of Cameras

The previous steps can achieve similar motion capture performance with consumer-grade cameras as the complex and expensive motion capture setups need by professional studios indoors, even in uncontrolled outdoor scenes. In practice, the previous algorithms need at least five cameras to achieve reasonable tracking accuracy, which hinders many practical outdoor motion capture applications. Therefore, in our fourth step toward a simple human motion capture setup, we propose a novel method to capture articulated skeleton motion from input filmed with as few as two cameras; details will be discussed in Chapter 7. This algorithm fuses marker-less skeletal motion tracking with body-part detections from a convolutional network (ConvNet) in order to achieve accurate motion tracking of several subjects in general scenes, indoors and outdoors, even from input captured with much fewer cameras. The algorithm is computationally efficient as poses can be computed very efficiently using iterative local optimization. The result is one of the first algorithms to capture temporally stable, fully articulated joint angles from as little as 2-3 cameras, also with multiple actors in front of moving backgrounds.

The core algorithmic contributions of this method are:

[1]The concept of symmetric similarity was first presented by [Sminchisescu & Telea (2002)]. However, our novel continuous and differentiable two-sided term is essential for moving cameras, and allows for fast tracking.

1. A novel way to combine evidence from a ConvNet-based monocular joint detector [Tompson *et al.* (2014a)] with a model-based articulated pose estimation framework [Stoll *et al.* (2011)].

2. A novel energy term which carefully integrates the body-part detections from all cameras.

## 1.2 Thesis Outline

The rest of this thesis is structured as follows: An overview of related work is provided in Chapter 2. Chapter 3 introduces the fundamental concept of the model-based articulated pose estimation framework by Stoll *et al.* (2011) and the ConvNet-based monocular joint detector by Tompson *et al.* (2014a), which are used as a baseline for the algorithms in this thesis. In Chapter 4, we present a synchronization algorithm which is temporally synchronizing multiple videos capturing the same dynamic scene. This algorithm relies on general image features and it does not require explicitly tracking any specific object, which makes it applicable to general scenes with complex motions.

Our spatio-temporal motion tracking algorithm is presented in Chapter 5. This algorithm takes the synchronization parameters as input, and reconstructs human motion in much higher temporal detail than was possible with previous synchronized approaches. This is achieved by formulating the model-to-image similarity measure as a temporally continuous functional. In Chapter 6, we present an algorithm for capturing the skeletal motions of humans using a set of potentially moving cameras in an uncontrolled environment. This is facilitated by a new energy functional that captures the alignment of the model and the camera positions with the input videos in an analytic way.

We present the ConvNet-based motion capture algorithm in Chapter 7. This algorithm achieves accurate tracking of several subjects in general scenes, indoors and outdoors, even from input captured with as few as two cameras. We conclude this thesis in Chapter 8 and propose future directions for the research on this topic.

## 1.3 List of Publications

The work presented in this thesis has been published in the following papers:

### 1.3 List of Publications

**Elhayek *et al.* (2015a)** **A. Elhayek**, E. Aguiar, A. Jain, J. Tompson, L. Pishchulin, M. Andriluka, C. Bregler, B. Schiele C. Theobalt: Efficient ConvNet-based Marker-less Motion Capture in General Scenes with a Low Number of Cameras, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, USA, 2015.

**Elhayek *et al.* (2014a)** **A. Elhayek**, C. Stoll, K. I. Kim, H.-P. Seidel, C. Theobalt: Outdoor Human Motion Capture by Simultaneous Optimization of Pose and Camera Parameters, Computer Graphics Forum (CGF), ISSN 1467-8659, 2014.

**Elhayek *et al.* (2012a)** **A. Elhayek**, C. Stoll, N. Hasler, K. I. Kim, H.-P. Seidel, C. Theobalt: Spatio-temporal Motion Tracking with Unsynchronized Cameras, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, USA, 2012.

**Elhayek *et al.* (2012c)** **A. Elhayek**, C. Stoll, K. I. Kim, H.-P. Seidel, C. Theobalt: Feature-Based Multi-Video Synchronization with Subframe Accuracy,. In A. Pinz, T. Pock, H. Bischof (Eds.): Pattern Recognition. Lecture Notes in Computer Science, Springer, Berlin, 2012.

# Chapter 2

# Related Work

In this chapter, we introduce previous work related to the content of this thesis. It is generally divided into four parts. Firstly, we introduce video synchronization algorithms related to our multi-camera synchronization algorithm in Chapter 4. In the second part, we discuss marker-less human motion capture algorithms related to Chapter 5, where we introduce a spatio-temporal motion tracking approach. In the third part, outdoors human motion capture algorithms are introduced, which are related to our algorithm in Chapter 6 for capturing the skeletal motions of humans using a sparse set of potentially moving cameras in an uncontrolled environment. In the last part, we introduce hybrid discriminative and generative pose detection algorithms which are related to our ConvNet-based motion capture algorithm (Chapter 7).

## 2.1 Synchronization Algorithms

One of the first video synchronization algorithms is described by Stein (1998) where the algorithm detects static features and tracks moving objects. Based on these detected and tracked features, it estimates the planar alignment as well as the epipolar geometry. This algorithm permits the synchronization of videos which show significantly different view points. However, its usage is limited because it requires explicitly tracking specific objects, and is applicable only to a pair of videos. One or both of these limitations are shared by most existing algorithms. For instance, the algorithms of Dai *et al.* (2006) and Caspi *et al.* (2006) are designed specifically for the two-video case. On the other hand, Sinha & Pollefeys's silhouettes-based algorithm (2004) and Meyer *et al.*'s algorithm for moving cameras (2008) can synchronize multiple cameras, and are based on explicit feature tracking or on the (often

violated) assumption of the existence and detection of reliable (long and clean) trajectories.

In Chapter 4 of this thesis, we present a novel algorithm for temporally synchronizing multiple videos capturing the same dynamic scene. Our algorithm relies on image features in general scenes and it does not require explicit tracking of any specific object, making it applicable to general scenes with complex motion. Most strongly related to the proposed algorithm is the work by Caspi *et al.* (2006), where the concept of feature trajectory matching was introduced for video synchronization. Our algorithm extends this method and explicitly overcomes its two main limitations: 1) our algorithm is applicable when there is arbitrary time shift and frame rate differences, 2) our algorithm enables multi-camera synchronization. Neither of this is directly feasible using Caspi *et al.*'s algorithm since they use grid search of parameters, which is applicable only when one or few parameters need to be estimated. An alternative to video-based synchronization is to exploit additional data, such as audio [Hasler *et al.* (2009b)] or still images obtained with controlled flashes [Shrestha *et al.* (2006)].

## 2.2 Marker-less Motion Capture

Marker-less human motion capture approaches reconstruct human skeletal motion from single or multi-view video and have been studied in the computer vision community over many years. For a detailed discussion and a historical perspective on these techniques, one can consult any of the surveys by Moeslund *et al.* (2006), Poppe (2007) or Sigal *et al.* (2010). The approaches can be roughly divided into methods that rely on multi-view input and methods that try to infer pose from a single view.

The majority of multi-view tracking approaches combine the use of body model, usually represented as a triangle mesh or simple primitives, with silhouette and image features, such as SIFT [Lowe (2004a)], for tracking. These methods estimate pose by optimizing a generative model-to-image similarity. They differ in the type of features used and the way pose optimization is performed. The multi-layer framework proposed by Gall *et al.* (2010) uses a particle-based optimization related to Deutscher & Reid (2005), to estimate the pose from silhouette and color data in the first layer. The second layer refines the pose and extracts silhouettes by local optimization. The approaches by Li *et al.* (2010), Lee & Elgammal (2010) and Bo &

Sminchisescu (2010) require training data to learn either motion models or a mapping from image features to the 3D pose. To evaluate the accuracy of such methods a variety of benchmarks exist such as the HumanEVA [Sigal *et al.* (2010)]. However, almost all multi-view methods to date rely on synchronized multi-view input.

In a second category of approaches, methods try to infer poses [Andriluka *et al.* (2010); Ionescu *et al.* (2011)] from single-view images, or motions from monocular video [Wei & Chai (2010)]. Most of the methods for human pose estimation are based on some form of probabilistic body model such as the pictorial structures (PS) model [Felzenszwalb & Huttenlocher (2005); Fischler & Elschlager (1973)] that represents the body configuration as a collection of rigid parts and a set of pairwise part connections. A large number of algorithms have been proposed [Andriluka *et al.* (2009); Dantone *et al.* (2013); Eichner & Ferrari (2009); Sapp & Taskar (2013); Yang & Ramanan (2011)]. Yang & Ramanan (2011) proposed a flexible mixture of templates based on linear Support vector machine (SVM). Approaches that model yet higher-order body-part dependencies have been proposed more recently. Pishchulin *et al.* (2013a,b) model spatial relationships of body-parts using *Poselet* Bourdev & Malik (2009) priors and a deformable part model (DPM) based part-detector. Sapp & Taskar (2013) propose a multi-modal model which includes both holistic and local cues for mode selection and pose estimation. Similar to the *Poselets* method, using a semi-global classifier for part configuration, the *Armlets* approach by Gkioxari *et al.* (2013) shows good performance on real-world data, however, it is demonstrated only on arms. This category of approaches has gained more attention in the past few years, even though the results do not yet reach the accuracy of multi-view methods and usually do not use character models with many degrees of freedom. Furthermore, all these approaches suffer from the fact that the features used (HoG features, edges, contours, and color histograms) are hand-crafted and not learnt.

There are also recent works on human motion capture from depth cameras, such as the Kinect [Baak *et al.* (2011); Ganapathi *et al.* (2010); Shotton *et al.* (2011); Wei *et al.* (2012)]. These methods are designed for real-time use. However, they only reconstruct coarse skeletal motion and coarse surface geometry [Taylor *et al.* (2012)]. High-quality pose and shape reconstruction is not their goal. Moreover, most depth cameras work only indoors, and have a very limited range and accuracy. Earlier vision methods such as Plänkers & Fua (2001) attempted to capture human skeletal motion from stereo footage, but did not achieve as high-quality poses and reconstructions as recent methods. Recent approaches such as Wu *et al.* (2013) use a sparse camera system, for example a stereo setup, to achieve high-quality poses and reconstructions. This method exploit bidirectional reflectance distribution function

(BRDF) information and scene illumination for accurate pose tracking and surface refinement. It relies on a foreground segmentation approach that combines appearance, stereo, and pose tracking results to segment out actors from the background.

Tracking without silhouette information is typically approached by combining segmentation with a shape prior and pose estimation. While Bray *et al.* (2006) use graph-cut segmentation, Brox *et al.* (2010) and Gall *et al.* (2008) rely on level-set segmentation together with motion features or an analysis-by-synthesis approach. While these approaches iterate over segmentation and pose estimation, the energy functional commonly used for level-set segmentation can be directly integrated in the pose estimation scheme to speed-up the computation [Schmaltz *et al.* (2011)]. The approach by Stoll *et al.* (2011) introduced an analytic formulation for calculating the model-to-image similarity based on a Sums-of-Gaussians model. Both body model and images are represented as collection of Gaussians with associated colors. The energy functional is continuous in parameter space and allows for near real-time tracking of complex scenes.

The only work addressing the necessity for complex and expensive synchronized multi-view camera setups for tracking is by Hasler *et al.* (2009a). In their work, sub-frame accurate synchronization is achieved by optimizing correlation of the audio channels of each video. However, during the human pose estimation stage, the sub-frame information is discarded and the videos are treated as synchronized with one-frame accuracy (i.e. all images taken at the same time instant) for further processing. The estimation step creates silhouettes using a level-set segmentation and uses these for pose optimization. As we show in Chapter 5, this approximation is not valid for fast motion, and we propose an algorithm that overcomes the limitation of frame-level synchronization in [Hasler *et al.* (2009a)]. By representing the pose parameters as an analytic function of time, tracking becomes possible with heterogeneous and unsynchronized but stationary cameras at sub-frame accuracy.

## 2.3 Outdoor Motion Capture with Moving Cameras

In the previous section, we discussed many marker-less motion capture algorithms. Nevertheless, all of these algorithms, except [Hasler *et al.* (2009a)], do not work with moving cameras in an uncontrolled outdoor environment. Pons-Moll *et al.* (2011) introduce an outdoor human motion capture system that combines video input with sparse inertial sensor input. As it employs an annealing particle-based optimization

scheme, its idea is to use orientation cues derived from the inertial input to sample particles from the manifold of valid poses. Then, visual cues derived from the video input are used to weight these particles and to iteratively derive the final pose. However, this method does not work with moving cameras.

Only few approaches deal with tracking human motion from moving cameras. As mentioned before, Hasler *et al.* (2009a) proposed an algorithm for motion tracking with unsynchronized cameras. In this algorithm, the input sequences are recorded with handheld video cameras. However, camera synchronization and calibration problems were decoupled from pose estimation by explicitly solving these problems before pose estimation. The camera parameters for each set of (synchronized) video frames are estimated using a structure-from-motion approach (SfM). A different approach was taken by Shiratori *et al.* (2011) who mount outwards facing cameras to the limbs of an actor and estimate the skeletal pose based on structure-from-motion of the actor's environment. These approaches have several limitations: structure-from-motion fails in case of cluttered scenes with dense moving background (e.g., crowds of people), motion blur due to hand-held camera shaking, and small camera translation or pure rotational motion. Furthermore, frame-level synchronization might be insufficient for heterogeneous cameras as demonstrated in Chapter 5 (i.e., sub-frame accurate synchronization leads to a significant improvement), and body-mounted cameras mean unwanted active modification of the scene.

Ye *et al.* (2012) presented an algorithm which tracks human motion with multiple consumer depth sensors (i.e. Kinects). They simultaneously optimize skeletal pose and sensor position based on image correspondences from feature tracking and geometric correspondences between the point clouds and the performer's surface. However, due to the use of depth sensors, the method cannot be applied in outdoor scenarios, and fails if no stable image features can be found in the background. To enable rendered fly-arounds in virtual replays, Germann *et al.* (2010) tracked articulated billboard models of soccer players from TV cameras in a soccer stadium. However, their algorithm is not fully automatic and taylored to soccer pitches where foreground separation is easier. Compared with those approaches, the method proposed in Chapter 6 does not depend on structure-from-motion and is instead based on a new generative skeletal pose tracker that minimizes a single model-to-image consistency measure simultaneously in the skeletal actor poses and the poses of moving cameras. We demonstrate that this strategy is essential to deal with scenes where cameras foreground and background can move, and image-based pre-calibration (such as structure-from-motion) fails.

## 2.4 Hybrid Discriminative and Generative Pose Detection

We discussed so far many multi-view tracking approaches which combine a body model with silhouette or image features for tracking. Most of these approaches, however, still rely on a sufficiently high number of cameras and they would fail if only a small number of cameras is available, even when recording simple scenes. On the other side, we discussed many methods that try to infer pose from a single view. However, all these approaches suffer from the fact that the features used are hand-crafted and not learnt.

Convolutional networks are by far the best-performing algorithms for many vision tasks such as object detection, image segmentation, video classification, pose estimation, and face recognition. The state-of-the-art methods for human-pose estimation are also based on Convolutional networks [Chen & Yuille (2014); Jain *et al.* (2014a,b); Tompson *et al.* (2014a); Toshev & Szegedy (2014)]. Toshev & Szegedy (2014) formulate the problem as a direct regression to joint location. Chen & Yuille (2014) improve over Toshev & Szegedy (2014) by adding an image-dependent spatial prior. Jain *et al.* (2014a) train an image patch classifier which is run in a sliding-window fashion at run time. Tompson *et al.* (2014a) use a multi-resolution ConvNet architecture to perform heat-map likelihood regression which they train jointly with a graphical model. However, apart from the new advances of these approaches, they still do not reach the same accuracy of multi-view methods, mainly due to the uncertainty in the part detections. In addition, they usually work only on very simplified models with few degrees of freedom, and the results often exhibit jitter over time.

Only a few methods in the literature are able to combine the individual strengths of both strategies. Using a depth camera, Baak *et al.* (2011) introduce a data-driven hybrid approach combining local optimization with global pose retrieval from a database for real-time full body pose reconstruction. Sridhar *et al.* (2013) also uses a hybrid solution, combining a discriminative part-based pose retrieval technique with a generative pose estimation method, for articulated hand-motion tracking using color and depth information. However, to the best of our knowledge, the method proposed in Chapter 7 presents one of the first algorithm to fuse marker-less skeletal motion tracking with body-part detections from a convolutional network for efficient and accurate marker-less motion capture with a few consumer cameras. This enables us to accurately capture full articulated motion of multiple people with as little as 2-3 cameras in front of moving backgrounds.

# Chapter 3

# Preliminaries

In this chapter, we will introduce some fundamental concepts and notations that the following work is based on. We will first give a brief introduction to motion-capture. Here we focus on the difference between marker-based and marker-less human motion capture algorithms. Then, we will introduce the generative model-based marker-less motion-capture approach by [Stoll *et al.* (2011)]. This approach is the baseline of our motion-capture algorithms. The next section will give an overview over the Convolutional neural network (ConvNet) approach which we use later for 2D body part detection.

## 3.1 Motion Capture

*Motion capture* is the process of recording the movement of one or several performers from input video. It has many applications, for instance in sports, biomedical research, or computer animation. The goal of motion-capture is to record the movement of a performer in a compact and usable manner [Gleicher & Ferrier (2002)]. This can be achieved by approximating the human body by a kinematic skeleton which consists of a small number of rigid segments that are connected by joints. Based on this approach, the task of motion-capture is reduced to finding the correct 3D skeletal configuration given a stream of video observations of a performer [Menache (1999)]. The reduction of the motion of a person to a set of skeletal joint parameters makes the problem of capturing the movement tractable, as it reduces drastically the dimensionality of the representation (typical skeletal representations used for motion-capture have somewhere between 30 and 50 degrees of freedom). Although, this reduction does not reflect the full complexity of human anatomy, it simplifies capturing and editing the motion and form a good compromise between

accuracy and model complexity [Stoll (2009)]. In the rest of this section, we will present a brief overview of the difference between marker-based and marker-less motion-capture.

### 3.1.1 Marker-based Motion Capture

Despite the significant amount of research which has been devoted to increasing the accuracy of marker-less motion-capture methods, the industry standard for human motion-capture (HMC) is by using marker-based systems. These systems require a set of markers to be placed on the performers body. HMC systems are classified into two categories based on the type of employed markers [Canton-Ferrer *et al.* (2010)]: the first type is nonoptical (inertial, magnetic, and mechanic) which usually requires special suits embedding rigid skeletal-like structures [Kirk *et al.* (2005)], magnetic or accelerometric devices or multisensor fusion algorithms [Roetenberg (2006)]; the second type is image-based or optical systems which are based on photogrammetric methods. These systems allow much larger freedom of movement and are less intrusive. Therefore, they are more common compared to the nonoptical ones. There are two types of optical markers: passive markers, that usually consist of retro-reflective tape reflecting under infrared lights [Vicon (2014)]; and active markers, that consist of infra-red LEDs [Phasespace (2014)].

In general, tracking requires the actor to wear a special suit to which a set of markers have been attached; see Fig. 3.1. The markers are designed carefully in order to make it easy to locate them in the video streams of the cameras recording the scene. Each marker has a predefined position on the body and is associated with a specific bone of the kinematic skeleton. This allows to triangulate the 3D position of the markers in each frame and to estimate the pose of the skeleton. Although, there are many problems (e.g. disambiguation, occlusions and missing markers) which need to be solved with this type of setup, marker-based systems allow to record the pose and the motion of a performer very accurately. However, these systems are also limited in their application range because the user is required to wear the special marker suit, which is an intrusive process [Stoll (2009)]. Additionally, these systems are usually expensive and require a dedicated hardware. Therefore, they can not be used in many outdoor motion-capture applications.

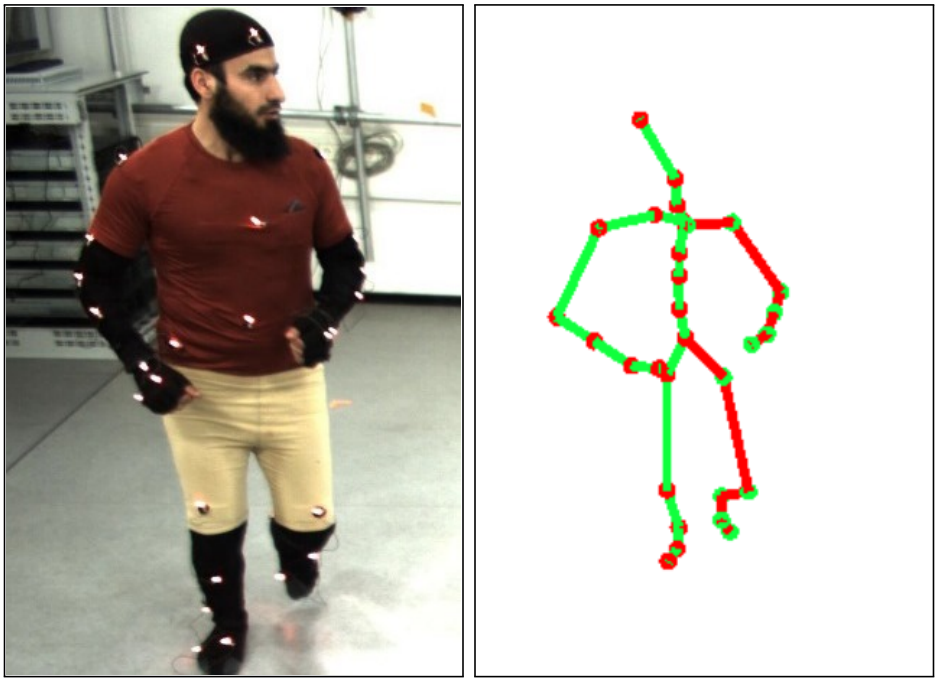


Figure 3.1: Maker based motion-capture. Left: A photograph of the subject during the capture session where markers are attached to a special suit. Right: The resulting kinematic skeleton corresponding to when the picture was taken.

### 3.1.2 Marker-less Motion Capture

As a first step to address some of the limitations of marker-based systems, marker-less motion-capture systems were introduced. Instead of using the markers in the images to estimate the skeletal pose, the marker-less systems use computer vision techniques to extract features directly from the video without interfering with the scene appearance. Classically, these methods use a 3D model of the human body. The model comprises a kinematic skeleton that defines the degrees-of-freedom $\Theta$ of the human model, and a representation of the shape and appearance of the human (e.g. geometric primitives or a detailed triangle mesh). In general, $\Theta$ is estimated by maximizing the similarity between the input images and projections of the human model to the corresponding views; see Fig. 3.2. Therefore, marker-less systems are more flexible than marker-based systems, which increases the number of possible applications of human motion-capture. However, it remains difficult for marker-less systems to achieve the same level of accuracy as marker-based systems. Moreover, image features may be very difficult to extract from the input videos and contain a

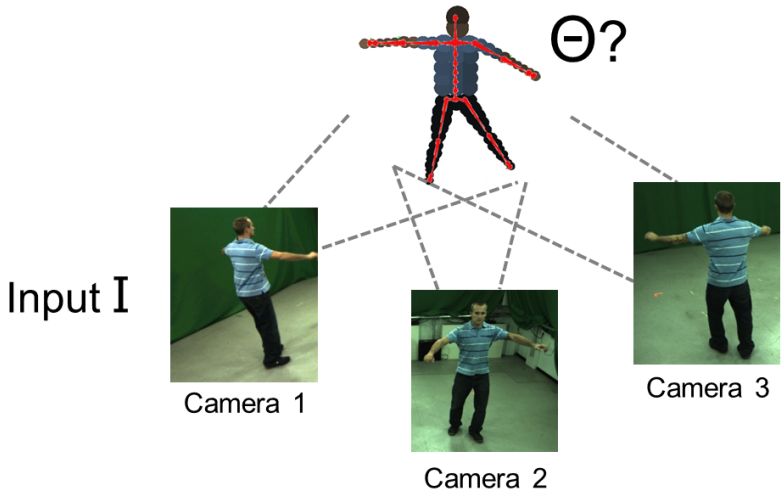



Figure 3.2: General concept of marker-less motion-capture. The pose parameters $\Theta$ are estimated by maximizing the similarity between each input image and a corresponding projection of the human 3D model.

high level of noise and inaccuracies, limiting the quality of the resulting motion unless recorded in a controlled studio environment [Stoll (2009)]. Thus, a lot of research has been devoted to developing accurate and fast marker-less methods which can track the motion accurately despite these algorithmic challenges. We present one of these methods in the following section.

## 3.2 Sums of Gaussians Tracker

In this thesis, we present three marker-less motion-capture methods. The baseline of these methods is the *Sums of Gaussians (SoG) Tracker* [Stoll *et al.* (2011)]. I introduce the basic concept of this marker-less motion-capture algorithm in this section. In the past, a lot of effort has been devoted to developing marker-less motion-capture algorithms. These efforts have addressed several aspects of marker-less motion capture algorithms like the human model [Plankers & Fua (2003)], the optimization approach [Bregler *et al.* (2004)], the image features [Ballan & Cortelazzo (2008)] or motion priors [Sidenbladh & Black (2003)].

In [Stoll *et al.* (2011)], the authors revisit the human model that is used for tracking. Many methods focus on realistic 3D models of humans. Although, such models can be easily derived from full body 3D scans [Anguelov *et al.* (2005)], they

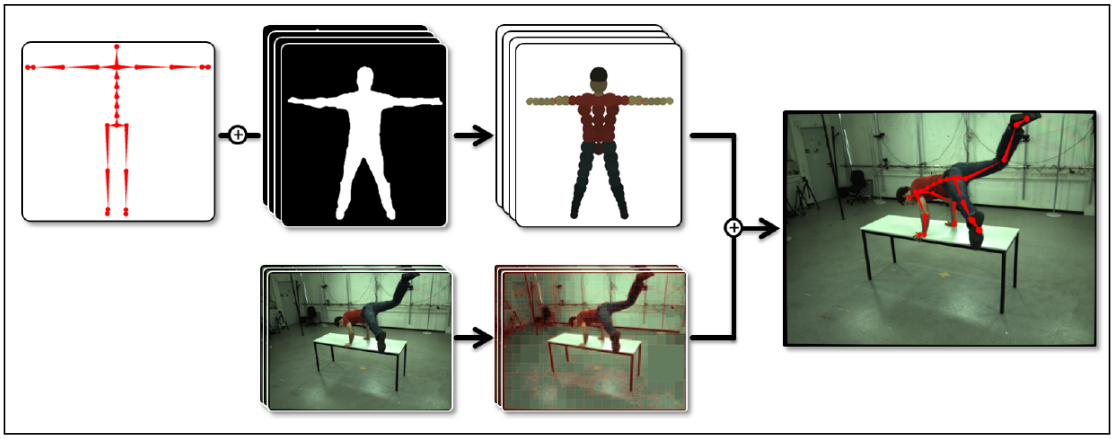


Figure 3.3: SoG tracker method overview: An actor-specific human 3D body model based on SoG is constructed from a sparse set of multi-view input images in a pre-processing step (top, Section 3.2.1). The input video streams are converted into a 2D SoG using a quad-tree (bottom, Section 3.2.2), and are used with the 3D human body model to estimate the skeletal pose of the actor in the frames (right, Section 3.2.3). [Stoll *et al.* (2011)]

decrease the computational efficiency. On the other hand, the simple models, such as [Wren *et al.* (1997)], which relied on simple spatial 2D blob models, allow to achieve real-time performance. [Wren *et al.* (1997)] does not rely on silhouettes obtained by background subtraction as many current methods e.g. [Sigal *et al.* (2010)]. In contrast to [Wren *et al.* (1997)] which estimates the articulated pose only in 2D, [Stoll *et al.* (2011)] extend the simple and fast 2D blob model to 3D.

In [Stoll *et al.* (2011)], the human model is represented by a set of spatial Gaussians (SoG). The model is equipped with a color model to represent the shape and appearance of the human and a kinematic skeleton that defines the degrees-of-freedom (DoF) of the human model. The person-specific model can be reconstructed from a sparse set of images. Similar to the human model, the input images are also represented as SoG that model color consistent image blobs. Based on the SoG models of the image and the human body, a continuous and differentiable model-to-image similarity measure is introduced. This allows to perform fast marker-less motion-capture even for many camera views by optimizing the parameters of the model such that the model-to-image similarity is maximized.

The outline of the processing pipeline of the SoG tracker is illustrated in Fig. 3.3 This pipeline can be divided into three steps: the first is SoG-based model

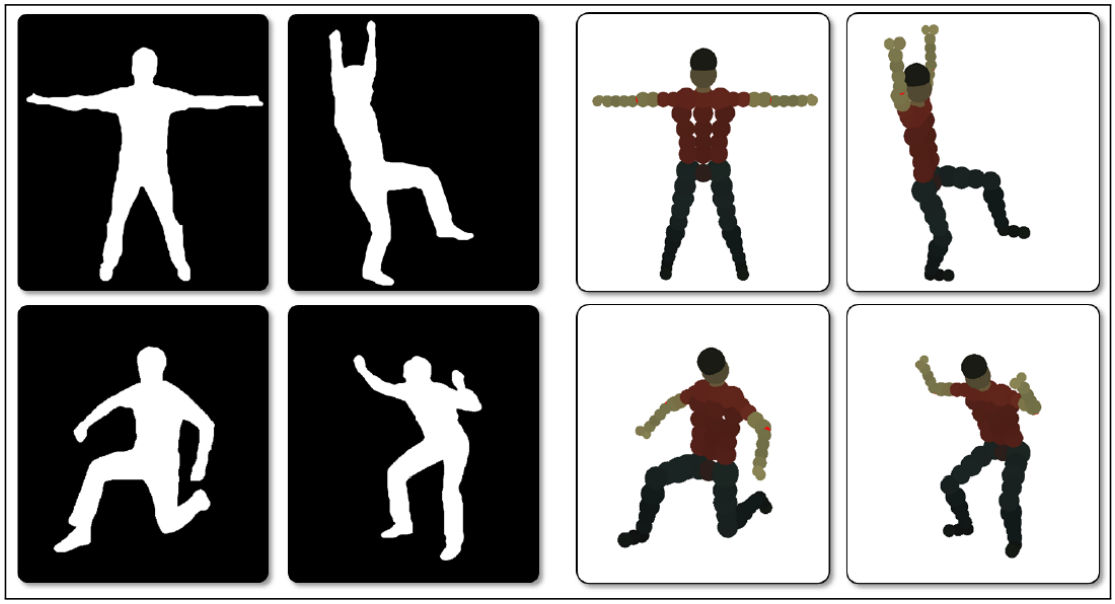


Figure 3.4: Estimating an actor specific model from example pose images. Left: Single segmented input image of the multi-view sets for each pose. Right: Resulting actor-specific body model after optimization and color estimation. [Stoll *et al.* (2011)]

estimation (Section 3.2.1) where a low number of manually segmented multi-view images are used to estimate an actor specific body model; the second step is SoG-based image approxiamtion (Section 3.2.2) where each image of multi-view input videos is converted into a SoG representation; the last step is SoG-based motion tracking (Section 3.2.3) where the similarity between the SoG model and the SoG images is used for tracking the articulated motion of the actor. The tracking step starts with the estimated pose of the model in the previous frame, and optimizes the parameters such that the overlap similarity at the current frame is maximized.

### 3.2.1 SoG-based Body Model

In [Stoll *et al.* (2011)], a default SoG-based human model is manually designed. This model consists of a kinematic skeleton to which a 3D SoG approximation of the performer's body is attached. The skeleton consists of 58 joints. Each joint is defined by an offset to its parent joint and a rotation represented in axis-angle form. In total, the model has 61 parameters $\Lambda$ (58 rotational and 3 translational). The skeleton further features a separate degree of freedom (DoF) hierarchy, consisting of

$n_{DoF}$ pose parameters $\Theta$. The degrees of freedom are mapped to the joint parameters using a $61 \times n_{DoF}$ matrix $\mathscr{M}$:

$$\Lambda = \mathscr{M}\Theta. \tag{3.1}$$

where each entry of $\mathscr{M}$ defines the influence weight that the parameters of $\Theta$ have on the joint angles $\Lambda$. All results in [Stoll *et al.* (2011)] were reported with a DoF hierarchy consisting of $n_{DoF} = 43$ pose parameters. Anatomically implausible pose configurations are prevented by modeling an allowable parameter range $l_l$ to $l_h$ for each DoF. This construction allows the model to reproduce natural deformation of the spine, as a single DoF can model smooth bending. It also allows straight-forward creation of several different levels of detail without having to edit the kinematic joint hierarchy itself.

The shape of the human model is represented using 63 3D Gaussians, where each Gaussian is attached to exactly one bone in the articulation hierarchy, resulting in a SoG model $\mathscr{K}_m$ that is parametrized by the pose parameters $\Theta$ of the kinematic skeleton. In a pre-processing step, the default model is adapted to generate an actor specific body model that roughly represents the shape and color statistics for each person we want to track. To this end, a low number of temporally not subsequent, multi-view images of example poses are manually segment; see Fig. 3.4 (Left). Thereafter, the pose parameters $\Theta$ are roughly initialized to correspond to the initial poses manually. A common set of shape parameters $\Theta_{shape}$ defines bone lengths as well as the positions and variances of the Gaussian model for a total of 216 degrees of freedom. Since the model acquisition is just a special case of the tracking approach, both the pose parameters $\Theta$ and shape parameters $\Theta_{shape}$ are optimized by maximizing the similarity measure (Eq. 3.7) based on the binary color values $c_i$ of the silhouette; see Section 3.2.3. Fig. 3.4 (Right) shows an actor-specific model that has been acquired from a set of manually segmented images of specific body poses.

### 3.2.2 SOG-based Image Approximation

To reduce the computational cost, the input images are also approximated based on 2D SoG using a fast quad-tree based clustering method. The simplest approach of approximating an input image $I$ by a SoG $K_I$ is to define a single Gaussian $\beta_i$ for each image pixel $p_i$ and assign to each Gaussian the color value $\mathbf{c}_i \in R^3$ of the pixel. However, to reduce the computational cost, a quad-tree structure is used to efficiently cluster image pixels with similar color into larger regions and each of these regions is then approximated using a single Gaussian $\beta_i$; see Fig. 3.5. In

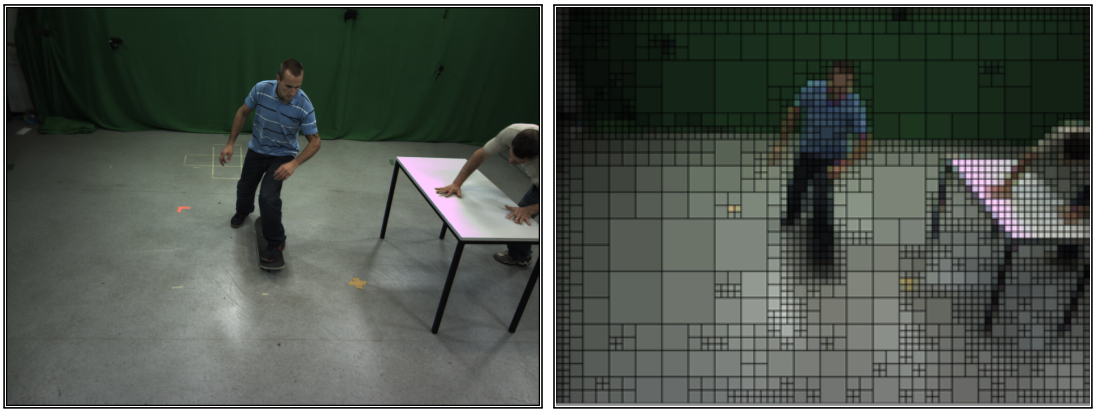


Figure 3.5: SoG-based image approximation. Left: Input image. Right: Quad-tree structure with average colors used to generate the SoG. Each square is represented by a single Gaussian.

[Stoll *et al.* (2011)] a threshold of the standard deviation of colors $\epsilon_{col} = 0.15$ is used to determine which pixels to cluster together. Thus, each node is subdivided into four sub-nodes when the standard deviation of colors on a quad-tree node is larger than $\epsilon_{col}$. The quad-tree depth is limited by a maximum depth of 8. Then each square-shaped cluster is represented by a Gaussian $\beta_i$ where $\mu$ is the center of the cluster and $\sigma^2$ is set to be the square of half the side-length of the node. Moreover, each Gaussian is assigned the average color $c_i$ of the cluster.

### 3.2.3 SOG-based Motion Capture

The proposed tracking algorithm adopts an energy maximization approach. It uses an energy functional which measures the similarity between the projections of the SoG 3D model and the SoG approximation of the input sequence. Each single Gaussian in the SoG sets is associated with a color $\mathbf{c_i}$ that can be used to measure the color similarity between two blobs. For each time step, measuring the similarity between a 3D SoG and a 2D SoG is facilitated by projecting the 3D SoG of the body model into the corresponding image plane and performing the comparison in 2D.

**Model to Image Similarity Measure:** For two given $2D$ SoGs $\mathscr{K}_a$ and $\mathscr{K}_b$ provided with colors $\mathbf{c}$ for each Gaussian blob, respectively, their similarity is defined

as

$$
\begin{aligned}
& E(\mathscr{K}_a, \mathscr{K}_b) \\
& = \int_\Omega \sum_{i \in \mathscr{K}_a} \sum_{j \in \mathscr{K}_b} d(\mathbf{c}_i, \mathbf{c}_j) \mathscr{B}_i(\mathbf{x}) \mathscr{B}_j(\mathbf{x}) \, \mathrm{d}\mathbf{x} \\
& = \sum_{i \in \mathscr{K}_a} \sum_{j \in \mathscr{K}_b} E_{ij}, \qquad (3.2)
\end{aligned}
$$

where $\mathscr{B}(\mathbf{x})$ is a Gaussian basis function

$$
\mathscr{B}(\mathbf{x}) = \exp\left( -\frac{\|\mathbf{x} - \mu\|^2}{2\sigma^2} \right). \qquad (3.3)
$$

$E_{ij}$ is the similarity between a pair of Gaussians $\mathscr{B}_i$ and $\mathscr{B}_j$ given their colors $\mathbf{c}_i$ and $\mathbf{c}_j$:

$$
\begin{aligned}
E_{ij} &= d(\mathbf{c}_i, \mathbf{c}_j) \int_\Omega \mathscr{B}_i(\mathbf{x}) \mathscr{B}_j(\mathbf{x}) \, \mathrm{d}\mathbf{x} \\
&= d(\mathbf{c}_i, \mathbf{c}_j) 2\pi \frac{{\sigma_i}^2 {\sigma_j}^2}{{\sigma_i}^2 + {\sigma_j}^2} \exp\left( -\frac{\|\mu_i - \mu_j\|^2}{{\sigma_i}^2 + {\sigma_j}^2} \right). \qquad (3.4)
\end{aligned}
$$

The color similarity function $d(\mathbf{c}_i, \mathbf{c}_j)$ measures the Euclidean distance between $\mathbf{c}_i$ and $\mathbf{c}_j$ in the HSV color space and feeds the result into a Wendland function [Wendland (1995)]. This renders $d$ to a smooth function bounded in $[0, 1]$ (0 for dissimilar input and 1 for similar input).

To measure the similarity between a given pose $\Theta$ of our body model $\mathscr{K}_m(\Theta)$ and a given input image SoG $\mathscr{K}_I$, we first need to project the body model into the respective camera image plane using the projection operator $\Psi$. Given a camera $\varsigma_l$ with respective $3 \times 4$ camera projection matrix $P_l$ and focal length $f_l$, we define the *projected* 2D Gaussian $\mathscr{B} = \Psi_l(\tilde{\mathscr{B}})$ corresponding to the 3D Gaussian $\tilde{\mathscr{B}}$ based on the following operations:

$$
\mu = \begin{pmatrix} [\tilde{\mu}^p]_x / [\tilde{\mu}^p]_z \\ [\tilde{\mu}^p]_y / [\tilde{\mu}^p]_z \end{pmatrix} \qquad \sigma = \tilde{\sigma} f_l / [\tilde{\mu}^p]_z \qquad (3.5)
$$

with $\tilde{\mu}^p = P_l\tilde{\mu}$ being the perspective-transformed 3D Gaussian mean. However, this projection function ignores possible selfocclusions that may happen when projecting the 3D model onto the 2D image plane. Several Gaussians may be projected onto overlapping 2D positions and thereby contribute several times to the energy function. In [Stoll *et al.* (2011)], this issue is implicitly resolved by defining the following model to image similarity:

$$E_{sim}(\mathscr{K}_I, \mathscr{K}_m(\Theta)) = \sum_{i\in\mathscr{K}_I} \min\left(\left(\sum_{j\in\Psi(\mathscr{K}_m)} E_{ij}\right), E_{ii}\right). \tag{3.6}$$

To prevent overlapping projected 3D SoGs from contributing multiple times in the above sum and thereby distorting the similarity function accordingly, [Stoll *et al.* (2011)] clamp the similarity to be at most $E_{ii} = \pi\sigma_i^2$, which is the similarity of the image Gaussian with itself. This can be seen as a simple approximation of an occlusion term. This approximation is intuitively motivated in Fig. 3.6. Using this SoG-based formulation as a basis has the advantage that the original formulation is by definition smooth in space. It does not rely on calculating and updating any image features or silhouette correspondences.

**Objective Function:** The ultimate goal of this algorithm is to estimate the pose-parameters $\Theta$ of the kinematic skeleton given $n_{cam}$ cameras $\varsigma_l$ with respective SoG approximation of the input images $(\mathscr{K}_l, C_l)$ and the 3D SoG body model $(\mathscr{K}_m; C_m)$. To this end, it is important to define an energy function $E(\Theta)$ that evaluates how accurately the model described by the parameters $\Theta$ represents what is in the images. Thus, the most important part of $E(\Theta)$ is measuring the similarity of the model $(\mathscr{K}_m; C_m)$ in the pose defined by $\Theta$ with all input images $(\mathscr{K}_l, C_l)$. The authors of [Stoll *et al.* (2011)] define this similarity function $E(\Theta)$ as

$$E_{sim}(\Theta) = \frac{1}{n_{cam}} \sum_{l=1}^{n_{cam}} \frac{1}{E_{sim}(\mathscr{K}_l, \mathscr{K}_l)} E_{sim}(\mathscr{K}_l, \Psi_l(\mathscr{K}_m(\Theta)), C_l, C_m). \tag{3.7}$$

In addition to $E_{sim}(\Theta)$ the final energy function $E(\Theta)$ includes a skeleton and motion-specific term:

$$E(\Theta) = E_{sim}(\Theta) + w_l E_{lim}(\mathscr{M}\Theta) + w_a E_{acc}(\Theta). \tag{3.8}$$

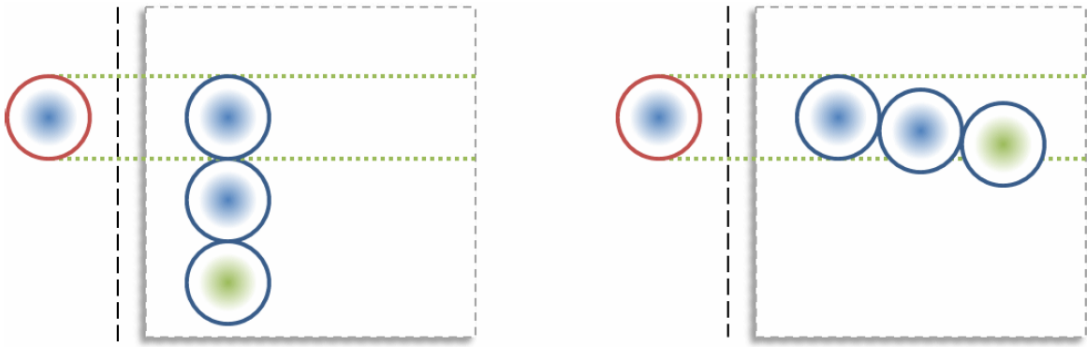


Figure 3.6: Self-occlusion approximation. Inside boxes: Top view of 3D model SoG. Left of dotted line: Image plane with 2D Gaussian. **Left column (no occlusion):** As long as no occlusions happen, (Eq. 3.2) calculates a correct overlap of a single element. In this example, the color (blue) and the shape are identical, yielding the similarity $E_{ii}$. **Right column (occlution approximation):** If several 3D model Gaussians project to the same screen space coordinate, their contribution is cumulative, yielding a similarity larger than $E_{ii}$, even though two of the model Gaussians should be occluded. Using (Eq. 3.6) correctly limits the contribution of a single 2D image Gaussian, yielding the same similarity $E_{ii}$ for both cases. [Stoll *et al.* (2011)]

where $E_{lim}(\Lambda)$, with $\Lambda = \mathscr{M}\Theta$ (Eq. 3.1), is a soft constraint on the joint limits and $E_{acc}$ is a smoothness term that penalizes high acceleration in the parameter space. The weights $w_l$ and $w_a$ influence the strength of these constraints and were set to $w_l = 1$ and $w_a = 0.05$.

This similarity measure is smooth in space and accordingly the analytical derivatives of any order can be computed easily with respect to the pose parameters $\Theta$. Therefore, it is possible to calculate the analytic gradient of $E(\Theta)$ efficiently and use it in a gradient ascent optimization procedure. However, simple gradient ascent tends to be very slow when optimizing energy functions that consist of long narrow valleys in the energy landscape, as it tends to "zig-zag" between opposing walls. In order to enhance the performance of the algorithm, an efficient conditioned gradient ascent is applied to optimize $E(\Theta)$. To this end, a conditioning vector is introduced into the optimization to increase step-size in directions where the gradient sign is constant, and decrease it if the ascent is "zig-zagging". (c.f. [Stoll *et al.* (2011)] for more details).

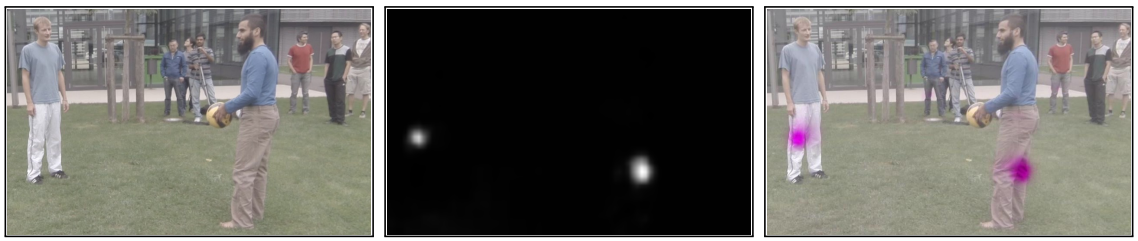


Figure 3.7: Input and output of the ConvNet body part detector. **Left:** Input image. **Middle:** Output heat-map of the right knee. **Right:** Input image overlaid with the heat-map.

## 3.3 ConvNet Body Part Detector

Recently, deep-learning discriminative architectures have achieved state-of-the-art performance on many difficult vision tasks [Razavian *et al.* (2014); Taigman *et al.* (2014); Zeiler & Fergus (2014)]. In particular the works [Chen & Yuille (2014); Jain *et al.* (2013); Tompson *et al.* (2014a, 2015); Toshev & Szegedy (2014)] have recently shown that convolutional network (ConvNet) architectures are well suited for the task of human-body pose detection and in most cases out-perform traditional graphical model based techniques. Furthermore, due to the availability of modern Graphics Processing Units (GPUs), it is possible to perform Forward Propagation (FPROP) of deep ConvNet architectures at interactive frame-rates (for instance the work of [Tompson *et al.* (2014a)] can perform single frame joint inference at 12 frames per second on an NVIDIA Titan GPU).

In practice, the SoG tracker fails with less than five cameras, which hinders many practical motion-capture applications. Therefore, in Chapter 7, we propose a novel algorithm to capture articulated skeleton motion from input filmed with as few as two cameras. This algorithm fuses marker-less skeletal motion tracking with 2D body part detections. Therefore, we briefly summarize the approach of [Tompson *et al.* (2014a)], which we use for part detection. This approach achieves state-of-the-art results on several public benchmarks, and is formulated as a convolutional network [LeCun *et al.* (1998a)] to infer the location of 13 joints in monocular RGB images.

ConvNets are biologically inspired variants of multilayered perceptrons. They exploit spatial correlation in natural images by extracting features generated by localized convolution kernels [Tompson *et al.* (2014b)]. Since the human body tends to have many repeated local image features (for instance left and right hands and

legs), ConvNets are well suited to perform feature extraction since multi-layered feature banks can share common features, thereby reducing the number of required free parameters.

Following the work of [Tompson *et al.* (2014a)], instead of training the ConvNet to detect the 13 body parts at once, the full human body-pose recognition problem is recast as an intermediate collection of easier individual body-part recognition problems, which can be more easily learned by ConvNets. Instead of directly inferring the UV pixel location of all 13 joints at once, the ConvNet infers a distribution over the pixel locations for each joint (or a set of heat-maps), where the detection energy at each pixel location is an independent term in the objective function used to train the ConvNet.

Empirically, we have found that inferring a heat-map output is less prone to over-fitting. A likely explanation is that in the presence of strong outlier detections (i.e. for ambiguous poses where left and right joint detections are visually similar and thus ambiguous), for a ConvNet to infer a single UV location it must arbitrarily choose a single detection or - more likely - choose the spatial mean of the two UV locations. Such an output results in a large Mean Squared Error (MSE) value. To minimize this error during training, the network is then prone to over-fitting, which hinders generalization performance. On the other hand, inferring a heat-map output allows for "softer" errors during training, since the MSE over independent detections for each pixel location is less strict on outlier detections. Additionally, the ConvNet is better at handling occlusions; by learning robust compound, high-level image features, the ConvNet is able to infer the approximate position of an occluded and otherwise unseen feature (for instance, when tracking multiple subjects, occluded joint locations can be inferred by the locations of its parent joints in the kinematic chain).

The model is a fully convolutional network and is therefore a translation invariant part detector (see [Tompson *et al.* (2014a)] for details). It takes as input a single RGB image, creates a 3 level Gaussian pyramid and outputs 13 heat-maps $H_{j,c}$ describing the per-pixel likelihood for each of the 13 joints; see Fig. 3.7. Since the network consists of two $2 \times 2$ MaxPooling layers, the output heat-maps are at a decimated resolution.

For [Jain *et al.* (2014b); Tompson *et al.* (2014a)], the part-detection network is trained using supervised learning via batched Stochastic Gradient Descent (SGD) with Nesterov Momentum. A MSE criterion is used to minimize the distance between the inferred response-map activation and a ground truth response-map. The target is a 2D Gaussian with a small variance and mean centered at the ground-truth

### 3.3 ConvNet Body Part Detector

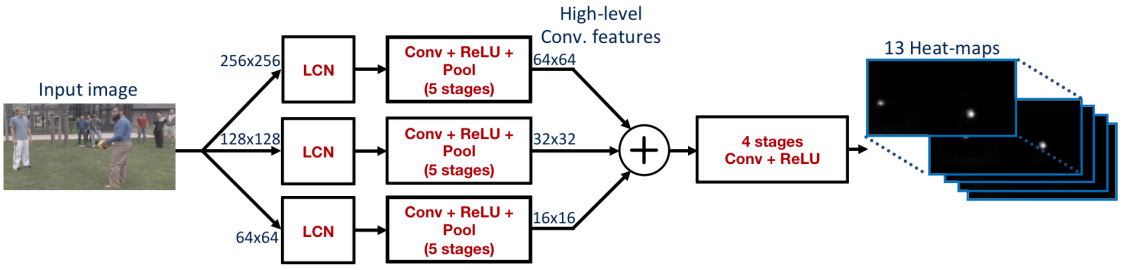



Figure 3.8: Convolution Network Archiecture

joint locations. In order to prevent network over-training and improve generalization performance, random perturbations of the input images (randomly flipping and scaling the images) is performed. The network was trained on the MPII Human Pose Dataset [Andriluka *et al.* (2014)], which consists of 28,821 training annotations of people in a wide variety of poses and static scenes. Note that training on our own sequences (or sequences similar to ours) may increase accuracy even further.

After downsampling (with anti-alisiang) to produce a Gaussian pyramid input, the first layer of each resolution bank is a local contrast normalization (LNC) layer. In conjunction with the Gaussian pyramid input, this layer creates 3 resolution images with non-overlapping spectral content (since the 5x5 LCN kernel is the same for each resolution bank). The advantage of this input representation is that it promotes specialization amongst the 3 banks, reducing network redundancy and thus improving generalization performance. Furthermore, the use of multiple resolutions increases the amount of spatial context seen by the network without a significant increase in the number of trainable parameters. Each of the 3 images is processed through a 5 stage Convolution-Non-Linearity-MaxPooling network which creates a dense and high-level feature representation for each of the multi-resolution images.

Each resolution bank is comprised of 5 convolution modules, 5 piecewise non-linearity modules, and 2 max-pooling modules. Please note that, not all convolution stages are followed by pooling with decimation. Each convolution module uses a stack of learned convolution kernels with an additional learned output bias to create a set of output feature maps (please refer to [LeCun *et al.* (1998b)] for an in-depth discussion). For all non-linearity layers, a rectified linear activation [Nair & Hinton (2010)] is used, which has been shown to improve training speed and discrimination performance in comparison to the standard sigmoid units. Each max-pooling module sub-samples its input image by taking the maximum in a set of

non-overlapping rectangular windows. The max-pooling [Nagi *et al.* (2011)] is used since it effectively reduces computational complexity at the cost of spatial precision, however in practice the tradeoff between pooling size and generalization performance is incredibly complex. Interested readers should refer to [Tompson *et al.* (2015)] for an in-depth discussion on max-pooling for detection networks.

Each resolution bank other than the highest, is fed through a nearest-neighbor up sampling layer to bring the feature maps into canonical resolution. Then, these resolution banks are input to a pixel-wise addition. The combination of these three operations (convolution-up-sampling-addition) is an approximation of the first fully-connected stage in a patch-based detector architecture (see [Tompson *et al.* (2014a)] for details). Lastly, the resultant feature maps are then feed through a 4 layer Convolution-Non-Linearity network (each with 1x1 convolution kernels) to create the final 13 heat-map images. The effective input-patch size (or alternatively "receptive-field size") that this network approximately simulates is $136 \times 136$ pixels in the input resolution. To handle persons of different size, heat-maps $H^s_{j,c}$ are precomputed at 4 different scales $s$.

# Chapter 4

# Optical Multi-Camera Synchronization

The last ten years have observed significant advances in mobile camera technology. The widespread use of smart phones facilitated casually capturing and sharing scenes of interest. The abundance of these data resulted in new opportunities and challenges in computer vision and computer graphics. For instance, there are more chances than ever to capture the same scene with multiple cameras: e.g., capturing a street show with several spectators. This can significantly broaden the domain of multiple-camera computer vision and graphics applications (e.g., marker-less motion capture [Stoll *et al.* (2011)] and video-based rendering [Ballan *et al.* (2010)]). However, it should be noted that computer vision and graphics algorithms typically assume that the cameras are synchronized, i.e., the ratio between the frame rates and the relative temporal offsets are known. In general uncontrolled settings, this may not be true: the cameras hardware may be heterogeneous and accordingly the recorded sequences (videos) have different frame rates. Sometimes, we only have the sequences with unknown source cameras. Furthermore, it is unlikely that the recorded sequences have the same offset. This makes automatic synchronization a necessity.

In the literature, there exist several synchronization algorithms. However, these algorithms are limited to specific scenes where it is possible to track the objects of interest, or to scenes where the objects show specific motions such as ballistic motion [Wedge *et al.* (2006)], or to synchronizing two sequences only. Therefore, in this chapter, we will present a feature-based multi-video synchronization algorithm which is our first step towards human motion capture in general scenes with unsynchronized cameras. This novel algorithm temporally synchronizes multiple

videos capturing the same dynamic scene. It relies on general image features and it does not require explicitly tracking any specific object, making it applicable to general scenes with complex motion. This is facilitated by our new trajectory filtering and matching schemes that correctly identify matching pairs of trajectories (inliers) from a large set of potential candidate matches, of which many are outliers. We find globally optimal synchronization parameters by using a stable RANSAC-based optimization approach. For multi-video synchronization, the algorithm identifies an informative subset of video pairs which prevents the RANSAC algorithm from being biased by outliers. The work presented here was published in [Elhayek *et al.* (2012c)].

## 4.1 Method Overview

The first step of our multi-video synchronization algorithm is feature-based matching: we extract a set of features and track them in each video, which constitute a set of feature trajectories. Then, the problem of synchronization is cast into spatio-temporally matching the trajectories across different sequences. Since such general features usually exist in any video, our algorithm is applicable to general scenes with any number of objects. Moreover, the dynamic properties of these trajectories enable the algorithm to achieve sub-frame accuracy of the synchronization parameters.

The technical challenges lie in the fact that the tracked trajectories are in general very noisy, *e.g.*, the tracked location of detected feature points are not precisely aligned in a video and tracking could fail. Furthermore, since there can be many trajectories in a given set of videos, identifying correctly matching pairs of trajectories across different videos is challenging. One of our main contributions is a method for resolving these problems. We propose a set of criteria to filter out noisy and uninformative trajectories and pairs of trajectories (details will be discussed in Section 4.3). As a result, a set of tentative trajectory pairs is generated. Among them, the correct subset (inliers) is identified by minimizing a global energy based on RANSAC-type optimization. The two-video version of our synchronization algorithm is summarized in Fig. 7.1. Since the energy is defined for any number of sequences, our algorithm can be equally applied to the multi-video case as well as to the two-video case. However, in the former case, additional robustness is achieved by identifying weakly coupled pairs of cameras and removing them from the evaluation of energy. This leads to an automatic generation of a graph representing the cameras and their connectivity. In the experiments, we demonstrate the effectiveness of

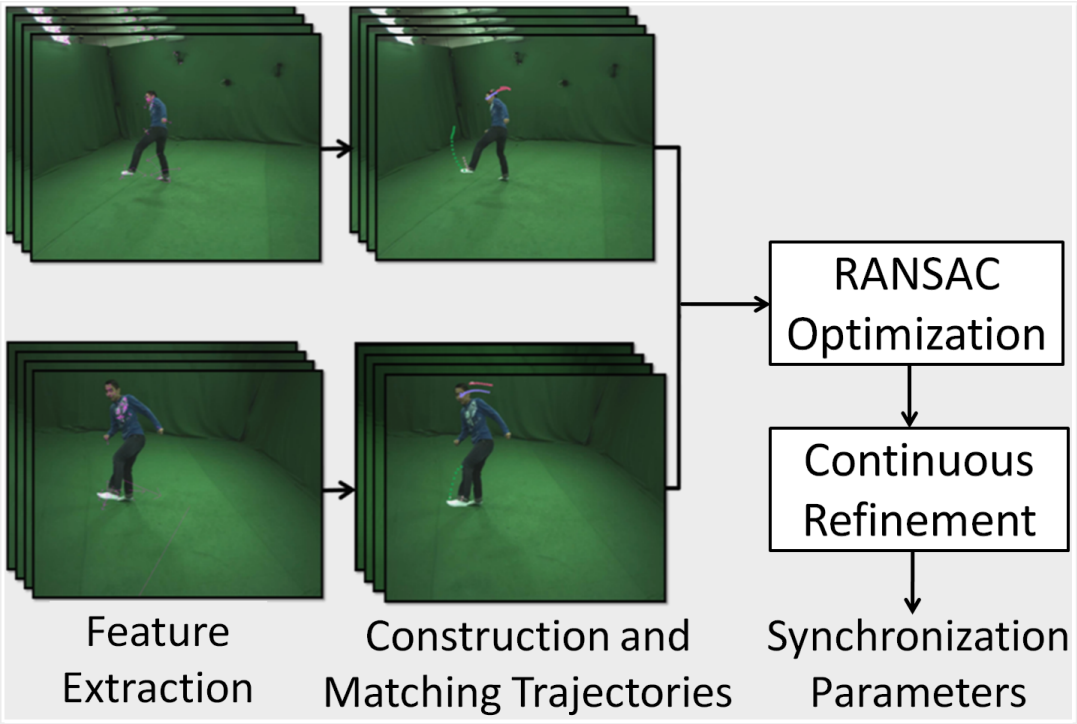


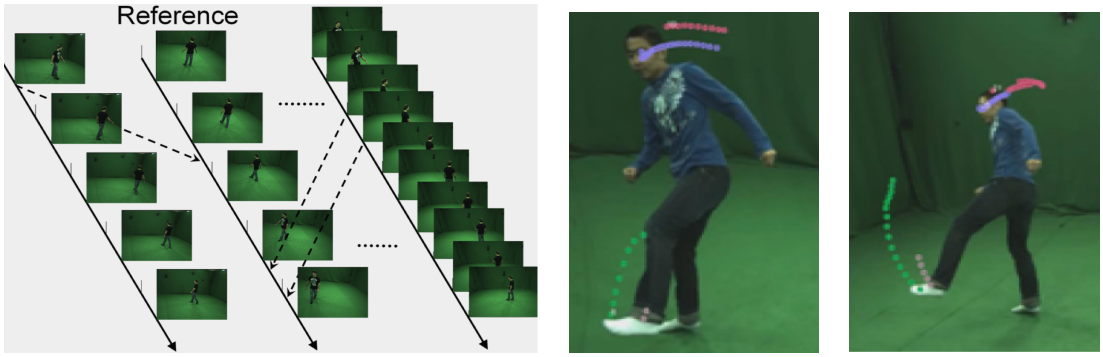

Figure 4.1: Two-video synchronization overview. We extract feature-points from each frame (left), construct in-sequence trajectories by matching these features across consecutive frames, and use epipolar matching to find the corresponding trajectories between two videos (middle). Then, the synchronization parameters are estimated based on RANSAC optimization, which are refined by continuous optimization (right).

our algorithm with datasets that are difficult to synchronize with the existing object tracking based synchronization techniques.

## 4.2 Problem Formulation

Similar to other synchronization methods [Caspi *et al.* (2006); Pádua *et al.* (2010)], we assume that each video is recorded by a camera which has a constant frame rate. In this case, the temporal misalignment between a set of videos occurs if they have time-shifts (offsets) between their start times, and/or when they have different frame rates (Fig. 4.2(a)). Accordingly, there is an affine relationship between the time lines (time coordinate values) of each pair of sequences.

For the two-video case, synchronization can be performed by setting one sequence




(a) (b) (c)

Figure 4.2: **(a)** Schematic diagram of multi-video synchronization. Time lines of different sequences with different frame rates are mapped to non-integer points along a single reference time line, as indicated by the arrows. **(b)** and **(c)** show two temporally corresponding frames from two different video sequences with some corresponding space-time trajectories resulting from the actor's motion in the previous frames.

as a reference (denoted as $S_r$) and estimating the relative offset $\theta_i$ and the frame rate ratio $R_i$ of the other sequence (denoted as $S_i$) with respect to the reference time line $t_r$ of $S_r$:

$$t_r = R_i \cdot t_i + \theta_i, \tag{4.1}$$

where $t_i$ is the time line of $S_i$. For general multi-video synchronization, consistent comparison of multiple sequences can be facilitated by establishing a global reference time line. While any global parametrization should work, for a given set of unsynchronized input sequences $\mathscr{S} = \{S_0, \ldots, S_N\}$, we simply set the time line of the first sequence $S_0$ as the reference. This sequence and the corresponding time line will henceforth be denoted as $S_r$ and $t_r$, respectively. With this representation, our algorithm produces an estimate of synchronization parameters $\{\theta_i, R_i\}$ (with respect to $t_r$) for each sequence $S_i \in \mathscr{S} \setminus S_r$.

Since the sequences in $\mathscr{S}$ capture the same scene, there is a geometrical relationship between the appearances of the scene components in each pair of sequences: Let $\mathbf{x}_r = (x_r, y_r, t_r)$ be a space-time point in the reference sequence $S_r$ and $\mathbf{x}_i = (x_i, y_i, t_i)$ be the corresponding points in $S_i \in \mathscr{S} \setminus S_r$ (i.e., $t_r$ and $t_i$ are related based on Eq. 4.1). Then, they should satisfy the fundamental geometrical relationship given below:

$$p_r(t_r)^\top F_i(t_i) p_i(t_i) = 0 \tag{4.2}$$

where, $p_r(t_r)$ is a vector consisting of the spatial coordinate values (i.e., $\{x_r, y_r, 1\}$) of $\mathbf{x}_r$, and $F_i$ is the fundamental matrix relating the reference camera and the $i$-th camera. Throughout this chapter, we assume static cameras. For the general moving camera case, $F$ has to be defined for each pair of corresponding frames. This can be done by updating $F$ based on the motion of the corresponding cameras. In Chapter 6, we address the problem of camera motion estimation.

## 4.3 General Synchronization Algorithm

This section presents our synchronization algorithm. We first discuss the two-video synchronization setting and illustrate the essential idea. Then, the extension of this framework to multi-video is discussed.

### 4.3.1 Two-video Synchronization

Our algorithm is based on matching trajectories of features appearing in a pair of videos given their fundamental matrix $F$. First, a set of features (SIFT features) are extracted from each frame of a sequence. Then, we use Best-Bin-First (BBF)-based feature matching to establish correspondences between features appearing in each pair of consecutive frames; see [Lowe (2004b)] for details. If the features corresponding to a single 3D-point are matched across more than two consecutive frames, the corresponding trajectory is constructed. Each trajectory is represented based on spatial coordinates of the corresponding feature points, each of which is assigned with the corresponding frame index. For instance, a trajectory in $S_r$ can be represented as

$$T_r = \{p_r(t), p_r(t+1), p_r(t+2), ..., p_r(t+k)\},$$

where $k+1$ is the length of the trajectory (i.e., tracking is successful for $k+1$ consecutive frames).

Matching a pair of trajectories implies establishing the correspondence between two sets of points contained in the two trajectories, respectively. Precisely matching a pair of *non-trivial* trajectories (details will be discussed shortly), uniquely defines the spatial parameters (i.e. fundamental matrix; *c.f.* Eq. 4.2), and since each point is assigned with the time index, the corresponding temporal parameters (offset and frame rate ratio).

In general, the construction of trajectories is noisy. For example, usually the locations of detected features do not precisely correspond to each other across the

consecutive frames and the tracking can be erroneous. Accordingly, the constraint (4.2) might not be exactly satisfied. Alternatively, one could minimize the following residual error with respect to those parameters [Caspi *et al.* (2006)]:

$$E(F_i, \theta_i, R_i) = \sum_{t_i \in support(T_i)} d_{F_i} \left( p_r(R_i \cdot t_i + \theta_i), p_i(t_i) \right), \tag{4.3}$$

where $d_F(A, B)$ is the Euclidean distance between a feature $A$ and the epipolar line corresponding to a feature $B$ mapped based on $F$ (see Fig. 4.4(c)).

The strategy described above is applicable only when a correct pair of trajectories (each from a single sequence) is identified. In general, there are multiple trajectories constructed in each sequence and the correspondences between them are not known *a priori*. Suppose that $m$ and $n$ trajectories are constructed from $S_r$ and $S_i$, respectively. Then there are $m \times n$ potential matching pairs of trajectories, only a few of which are correct. We therefore use RANSAC [Fischler & Bolles (1981)] which can effectively filter out the outliers matches. However, naively feeding all potential matches into a RANSAC step does not yield a proper parameter estimate: there exist several *trivial* trajectories which geometrically match many other trajectories. Moreover, the large number of trivial trajectories decreases the computational efficiency of the method. Therefore, we introduce three trajectory filtering steps. Firstly, we remove very short trajectories which are shorter than a specific number of frames (5 frames in our experiments). The second filter removes trajectories corresponding to static feature points. We call this type of trivial trajectories space-static trajectories. A trajectory is removed if the variance of its spatial coordinate values is small (i.e. less than 15 pixels in our experiments).

Finally, we remove all trajectories which may generate ambiguous matches. We call this type of trivial trajectories epipolar-static trajectories. This happens when the tangents of trajectory point's are nearly parallel to the points epipolar line defined by the fundamental matrix of the camera pair. It is very difficult to distinguish any motion along that line in the other camera. This may lead to the feature match being classified as an inlier with low energy even for wrong matches. To detect such ambiguous (trivial) trajectories, we check each trajectory by computing the angles between its tangents at each of its points and the epipolar line of these points in its own camera. If the sum of these angles is too small, the corresponding trajectory may erroneously match many trajectories in the other sequence. Thus, we reject any trajectory if the score $\sum_{t_i \in sup(T_i)} 1 - cos(angle)$ is less than 0.32. Fig. 4.3(a) shows an example of such a trivial trajectory where we check the angles between the tangent to the trajectory at point $p_2$ (i.e. the line defined by $p_2$ and $p_3$) and

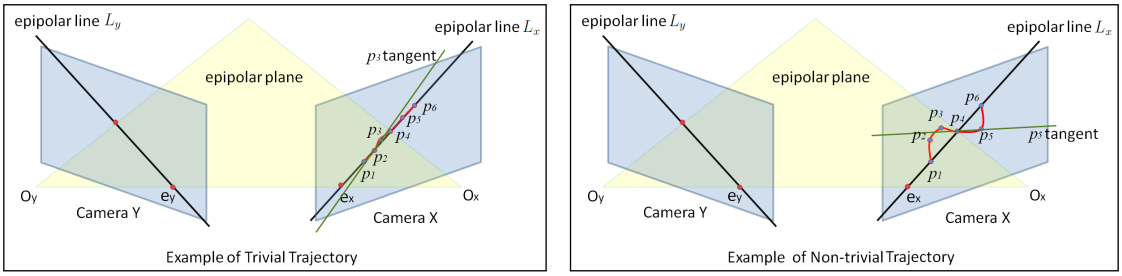



Figure 4.3: Epipolar trivial trajectory filter. **(a) Left:** Example of trivial trajectory in camera X which is nearly parallel to its epipolar line $L_x$ defined by the fundamental matrix of this camera pair. This trajectory may match any trajectory along $L_y$. However, it can be filtered by checking the angles between the tangents of each point along the trajectory (e.g. the green line for point $p_2$) and the epipolar line of these points (e.g. $L_x$ for $p_2$). **(a) Right:** Example of non-trivial trajectory where the angles between the tangents of the trajectory points and their epipolar lines are large.

the epipolar line for $p_2$ (the line defined by $p_2$ and the epipole $e_x$). Note that, for each point in this trajectory, the sum of the angles is too small which may lead to many incorrect matches with trajectories along $L_y$ in the other sequence. Therefore, we filter this trajectory before the epipolar matching step of our algorithm. On the other hand, Fig. 4.3(b) shows an example of a non-trivial trajectory where the sum of the angles is large (e.g. the angle corresponding to $p_4$).

Even after the trajectory filtering stage, erroneous candidate trajectory pairs may remain. These may negatively influence the run-time of a RANSAC optimization, and for a prescribed finite run-time, can bias RANSAC towards an unreliable solution. It should be noted that in order for a pair of trajectories to match, they have to overlap with each other in space and in time. Checking this can quickly filter out most wrong matches: Given a candidate match, we intersect the epipolar line corresponding to each feature point in the shorter trajectory with the longer trajectory (Fig. 4.4(a)). Since the frame rates of corresponding source videos are fixed, the consecutive epipolar lines should intersect with the longer trajectory such that the points of intersection are roughly equally spaced.[1] To check this, we first calculate the hypothetical frame rate ratios of two videos (denoted as $R_T$) based on the entire interval of intersection. For instance, in Fig. 4.4(a), $R_T$ is calculated by dividing the number of feature points lying between $F_i \cdot p_1$ and $F_i \cdot p_7$ on the longer

[1]Note that in case of corresponding trajectories from identical cameras (i.e. equal frame rates) the distances between consecutive points of intersection along the time dimension must be 1.

Table 4.1: Two-video synchronization algorithm

1. Extract features from each frame
2. Construct in-sequence trajectories
3. Filter out trivial trajectories:
   - short trajectories
   - space-static trajectories
   - epipolar-static trajectories
4. Build a table of tentative matches based on epipolar geometry
5. RANSAC-based optimization:
   (a) Randomly sample two pairs of matching trajectories and estimate parameters accordingly
   (b) Compute the number of inliers
   (c) Repeat steps (5.a) and (5.b) and choose the parameters which show the highest number of inliers
6. (optionally) Refine the RANSAC estimate using continuous optimization

trajectory by 7 which is the number of intersecting epipolar lines. In the same way, we calculate hypothetical frame rate ratios from each consecutive interval on the trajectory (e.g., $[F_i p_1, F_i p_2]$). All of these estimated frame rate ratios should agree roughly with $R_T$: we decide that a new hypothetical frame rate ratio $R_N$ agrees with $R_T$ if $|R_T - R_N| < 0.5R_T$.

Then, the degree of overlap between two trajectories is measured based on the number of consecutive epipolar lines ($P_{min}$) which satisfies the above described condition. When, $P_{min}$ is smaller than 5 (threshold found by experimental validation), the corresponding trajectory pair is rejected. It should be noted that in general, an epipolar line can intersect with a trajectory more than once (Fig. 4.4(a)). This case can be dealt with by retaining multiple hypothetical frame rate ratios ($R_T$) accordingly. The result of this step is a table of tentative matching trajectories.

The extension of the energy functional (4.3) for the multiple trajectory case,

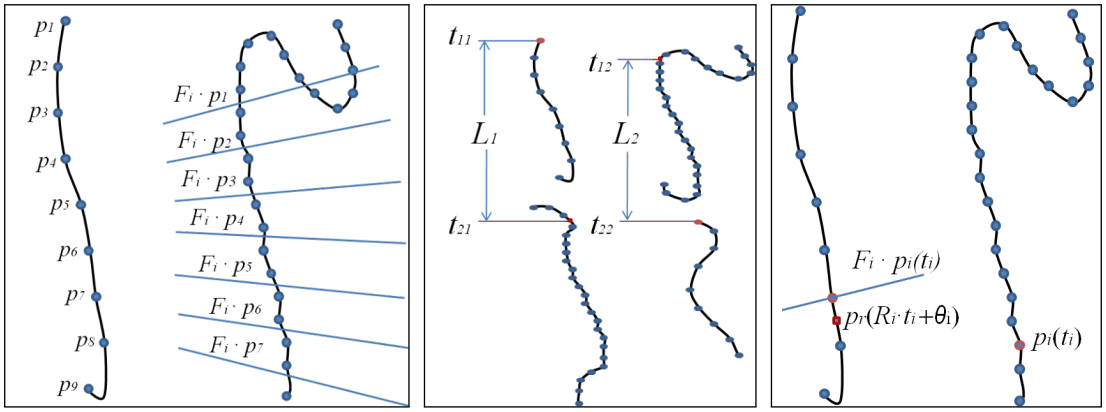



Figure 4.4: **(a) Left:** Epipolar trajectory matching test. **(b) Center:** Estimation of synchronization parameters based on two distant pairs of matching trajectories. **(c) Right:** Trajectory point residual error measured as the distance between the point $p_r(R_i \cdot t_i + \theta_i)$ and the epipolar line $F_i \cdot p_i(t_i)$.

given a precomputed fundamental matrix $F_i$ for static cameras, is as follows:

$$E(\theta_i, R_i) = \sum_{T_i \in \Gamma_i} \sum_{t_i \in support(T_i)} d_{F_i} \left( p_r(R_i \cdot t_i + \theta_i), p_i(t_i) \right), \tag{4.4}$$

where $\Gamma_i$ is the set of trajectories for the $i$-th video, minimizing it with the tentative matches does not correctly estimate the synchronization parameters since the tentative matches still contain a lot of outliers. Accordingly, we apply the RANSAC algorithm instead. It should be noted that each iteration of RANSAC requires generating hypothetical synchronization parameters. This can be determined from two pairs of corresponding feature points. These can be sampled from a single pair of matching trajectories, but we select them from two distinct candidate matches. This turned out to be more robust; see Fig. 4.4(b). Then, the hypothetical parameters are computed by solving the following equations for the two unknowns:

$$\begin{aligned} t_{11} &= R_i * t_{12} + \theta_i, \\ t_{21} &= R_i * t_{22} + \theta_i \end{aligned}$$

and the corresponding residual error is used to classify the tentative matches into inliers and outliers; see Fig. 4.4(c). The number of iterations of RANSAC adaptively changes based on the number of inliers [Hartley & Zisserman (2004a)]. At the end of the RANSAC loop, the parameters with the highest number of inliers are selected.

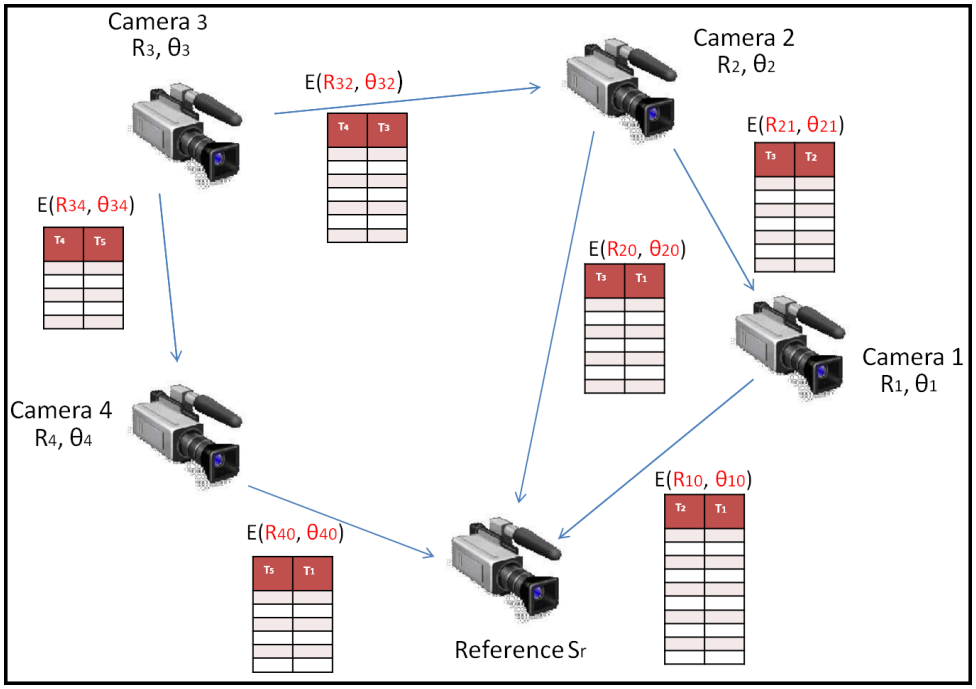



Figure 4.5: An example of the multi-video connectivity graph constructed by our algorithm.

The estimated parameters are further refined by continuously optimizing (4.4) with only inliers:

$$E(\theta_i, R_i) = \sum_{T_i \in \Gamma_i} \int_{t_i \in support(T_i)} d_{F_i} \left( p_r(R_i \cdot t_i + \theta_i), p_i(t_i) \right) d_{\theta_i} d_{R_i}, \tag{4.5}$$

We first render the problem into continuous optimization by interpolating each trajectory with cubic-splines. Then a standard gradient descent is performed. However, our preliminary experiments revealed that the continuous optimization step does not significantly improve the result over the initial RANSAC estimate and it is therefore not performed in general. Our two-video synchronization algorithm is summarized in Table 4.1.

### 4.3.2 Multi-video Synchronization

Once the global time coordinate is established, the extension of two-video synchronization framework to the multi-video case is straightforward. In this case, the global energy functional can be defined as the sum of pair-wise energies of the form

(Eq. 4.4) for any possible pair $i$:

$$E_g(\theta_i, R_i) = \sum_i E(\theta_i, R_i). \tag{4.6}$$

However, naively optimizing this energy functional is sub-optimal: some pairs of videos have more matching candidates and, accordingly, they are more informative than the other pairs. For instance, for two videos showing the same scene from significantly different viewpoints, the number of candidate trajectory matches might be very small. In this case, the parameters estimated by emphasizing the error corresponding to this camera pair might not be reliable. The remainder of this section discusses a strategy for solving this problem.

The relationships between a set of videos (or cameras) can be represented as a graph (see Fig. 4.5) in which a node corresponds to a sequence and an edge represents a set of tentative matching pairs of trajectories plus the corresponding synchronization parameters (of one node, with the other node treated as a reference). In this case, there are as many sets of parameters as the number of edges (i.e. local edge parameters), while the actual number of sets of parameters should correspond to the number of nodes (i.e. global parameters related to the reference time line).

To ensure that a consistent global parameters can be recovered from a set of local edge parameters, in each RANSAC step, we remove any cycle in the graph. This can be done, in principle, by randomly building a spanning tree. However, we have empirically observed that the accuracy of the estimated synchronization parameters between a pair of videos decreases with increasing distance between the cameras. Specifically, the lower the number of tentative matches between a pair of sequences, the less accurate the resulting estimation of synchronization parameters becomes. We exploit this observation by pre-filtering edges between distant pairs of cameras based on the number of tentative matches (35 in our experiments). An example of the resulting *connectivity graph* is shown in Fig. 4.5.

Figure 4.5 exemplifies a single step of RANSAC iteration. The global parameters $R_2$ and $\theta_2$ (with respect to the reference sequence $S_0$) can be estimated based on the paths $e_{21}$, $e_{10}$ and $e_{20}$. In general, the pairwise estimates of local parameters for each of these edges conflict with each other. To rule this out, in the RANSAC step, we construct a random spanning-tree, e.g., by removing edges $E_{20}$ and $E_{10}$.

The estimated local edge parameters are converted to the global parameters using the relations

$$R_{xy} = \frac{R_x}{R_y}, \quad \textit{and} \quad \theta_{xy} = \frac{\theta_x - \theta_y}{R_y}, \tag{4.7}$$

Table 4.2: Multi-video synchronization algorithm

1. Extract features from each frame
2. Construct in-sequence trajectories
3. Filter out trivial trajectories
4. Build connectivity graph:
   (a) Build a table of tentative matches for each pair of cameras
   (b) Remove edges between distant cameras (i.e. the camera pairs with low number of tentative matching trajectories)
5. RANSAC-based optimization:
   (a) Estimate the synchronization parameters based on a random spanning tree of the graph
   (b) Compute the number of inliers from the table of tentative matches
   (c) Repeat steps (5.a) and (5.b) and choose the parameters which show the highest number of inliers
6. (optionally) Refine the RANSAC estimate using continuous optimization

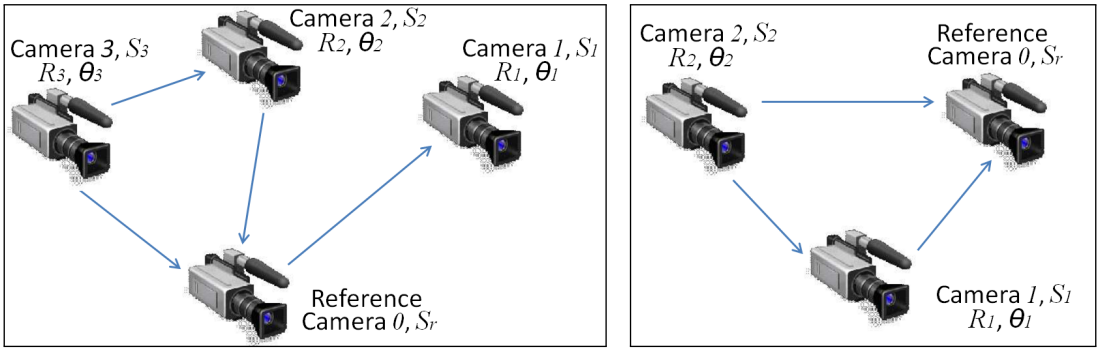



Figure 4.6: Connectivity graphs resulting from our algorithm for $\mathscr{S}^1$ (left) and $\mathscr{S}^2$ (right).

where $R_{xy}$ and $\theta_{xy}$ are the parameters of the edge between any two nodes $x$ and $y$. Once the global synchronization parameters are constructed, they are evaluated based on the number of inliers using every edge in the graph, i.e, the trajectory pairs which are not contained in the spanning tree are used as well. After the RANSAC iteration, the set of global parameters corresponding to the highest number of inliers is selected. The multi-video version of the algorithms is summarized in Tables 4.2. It should be noted that the first three steps of the multi-video synchronization algorithm are identical to those of the two-video algorithm.

## 4.4 Experimental Evaluation

In this section, we evaluate our algorithm based on two sets of unsynchronized videos capturing different scenes with different number of moving persons. The total number of videos in the sets is 7, and the resolution of each frames is 1296 x 968. To facilitate quantitative evaluation, we set the cameras up such that accurate timestamps for each frame can be obtained, which provide the corresponding ground-truth synchronization parameters for each set of videos. Once features are extracted, our algorithm took on average 20 seconds and 3.5 minutes for two-video and four-video synchronization, respectively.

In our evaluation, we show the *residual error* of the parameters (i.e. offset $\theta_i$ and frame rate ratio $R_i$) as well as the average and maximum *frame errors*, that are computed by aligning each frame of the synchronized video to the reference time line and by computing the deviations of the corresponding frame numbers from the ground-truth.

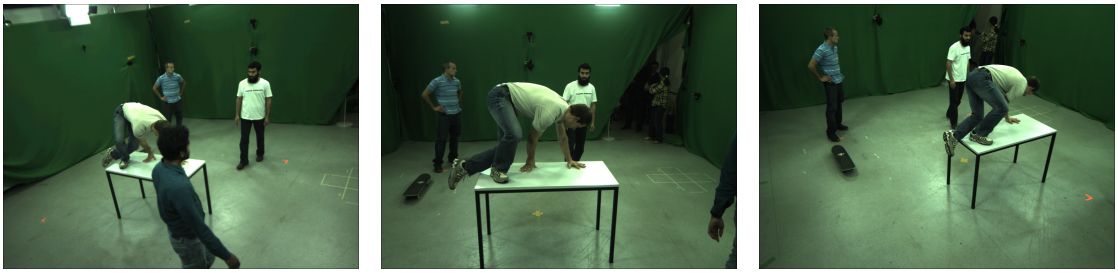


Figure 4.7: An example of synchronization for three videos ($\mathscr{S}^2$). Each frame (taken from each synchronized video) has the same global time-coordinate. The average video length is 280 frames, while the frame resolution is 1296 x 968.

In the first set of experiments, we evaluated the performance of our algorithm for the two-video case. To gain an insight into the role of individual filtering steps (Section 4.3), we constructed two different versions of our algorithm - one of them is constructed by removing the static filtering, the other one by removing the epipolar filtering stage from the original algorithm. We have also performed experiments with known frame rates, which are assumed to be known for most existing synchronization algorithms. Table 4.3 summarizes the result of our two-video synchronization experiments. We selected five pairs from two sets of videos, which show two different scenes containing four ($\mathscr{S}^1 = \{S_r^1, S_1^1, S_2^1, S_3^1\}$) and three ($\mathscr{S}^2 = \{S_r^2, S_1^2, S_2^2\}$) video-sequences, respectively. It was not possible to perform any experiments without static filtering for the case of $\mathscr{S}^2$ since the videos in this set contain many trajectories and accordingly the number of potential matches were prohibitively high[1]. The results suggest that both filtering stages, most notably the epipolar filtering, are critical to the performance of our algorithm and that once the frame rates are known, significant improvement can be gained.

Table 4.4 summarizes the result of the multi-video synchronization experiments for the video sets $\mathscr{S}^1$ and $\mathscr{S}^2$. This result demonstrates the effectiveness of our multi-video synchronization algorithm. The average frame error is less than two frames except for one pair of cameras: the average error for the sequence $S_3^1$ is rather high, which is most likely caused by the significantly different viewpoint from the rest of the videos in $\mathscr{S}^1$. Fig. 4.6 shows examples of connectivity graphs constructed based on our algorithm while Fig. 4.7 shows an example of three-video synchronization with synchronized frames.

[1]The continuous optimization step improved the average error by only 0.01 from the RANSAC results with significant additional computation. Accordingly, for the rest of the experiments, we do not adopt this stage.

Table 4.3: The results of additional two-video synchronization experiments.

| Video pairs | Experimental setup description | Ground truth | Residual error ($\theta_i/R_i$) | Average frame error | Maximum frame error |
|---|---|---|---|---|---|
| $S_0^1$,$S_1^1$ | Without static filtering | -50.00/1 | 2.74 /0.014 | 1.32 | 2.73 |
| | Without epipolar filtering | | 9.70 / 0.052 | 4.55 | 9.70 |
| | Complete algorithm | | 1.57 / 0.008 | 0.75 | 1.57 |
| | With given $R_i$ | | 0.19 / 0.000 | 0.19 | 0.19 |
| $S_0^1$,$S_3^1$ | Without static filtering | 80.00/2 | 0.93/0.020 | 1.27 | 3.03 |
| | Without epipolar filtering | | 5.42 / 0.179 | 13.22 | 30.17 |
| | Complete algorithm | | 0.71 / 0.005 | 0.29 | 0.711 |
| | With given $R_i$ | | 1.50 / 0.000 | 1.50 | 1.50 |
| $S_2^2$,$S_3^2$ | Without epipolar filtering | 29.11/1 | 0.95/ 0.001 | 1.09 | 1.24 |
| | Complete algorithm | | 0.52 / 0.004 | 0.24 | 0.52 |
| | With given $R_i$ | | 0.89 / 0.000 | 0.89 | 0.89 |
| $S_0^2$,$S_4^2$ | Without epipolar filtering | 27.38/1 | 3.70 / 0.066 | 3.93 | 8.56 |
| | Complete algorithm | | 0.46 / 0.026 | 3.65 | 6.83 |
| | With given $R_i$ | | 1.08 / 0.000 | 1.08 | 1.08 |
| $S_1^1$,$S_2^1$ | Without static filtering | 60.00/1 | 1.61 / 0.026 | 1.46 | 3.52 |
| | Without epipolar filtering | | 1.29 / 0.022 | 1.31 | 3.16 |
| | Complete algorithm | | 1.32 / 0.023 | 1.35 | 3.26 |
| | With given $R_i$ | | 2.13 / 0.000 | 2.13 | 2.13 |

### 4.4 Experimental Evaluation

Our algorithm is capable of exploiting the relationship among more than two video streams, and accordingly, it is naturally suited for multi-video applications. However, it should be noted that it is always possible to decompose a given multi-video synchronization problem into a set of two-video problems: one could first build a spanning tree and estimate the local pairwise synchronization parameters for each edge. Then, a globally consistent set of synchronization parameters can be estimated based on (Eq. 4.5) which corresponds to a single step of our multi-video RANSAC iteration. In general, the performance of multi-video synchronization should be better than this two-video synchronization-based approach, since the former can exploit all the available pairwise relationships, most of which are discarded when building a spanning tree. To exemplify this, we have selected three pairs of videos (namely $\{S_r^1, S_1^1\}$, $\{S_r^1, S_2^1\}$ and $\{S_r^1, S_3^1\}$), estimated pair-wise synchronization parameters, and obtained the global synchronization parameters based on (Eq. 4.5). The performance of this algorithm is significantly worse than of our new multi-video synchronization algorithm: the average frame errors for $S_1^1$ , $S_2^1$ and $S_3^1$ were 0.753, 0.298 and 37.5, respectively. Especially, the two-video algorithm completely failed for $S_3^1$ since, as mentioned above, the camera's viewpoint is very different from the rest of the cameras; only one edge in the graph is not sufficient to compute reasonable estimate of the parameters.

In a final experiment, we evaluated the performance of a variant of our algorithm which determines the parameters based on grid search: each parameter is sampled at regular grid and the parameter set corresponding to the largest number of inlier is selected for multi-video synchronization. This can be regarded as an instantiation of [Caspi *et al.* (2006)] in our feature-based setting. We found out that the grid search algorithm needs much longer computation time to yield results of comparable accuracy than our method because of the high dimensionality of the parameter space. For instance, for four videos in $\mathscr{S}^1$, to achieve a comparable runtime efficiency to

Table 4.4: Multi-video synchronization results.

| Video | Ground truth $(\theta_i/R_i)$ | Estimated parameters | Average frame error |
|---|---|---|---|
| $S_1^1$ | -50.00 / 1.000 | -50.84 / 1.005 | 0.35 |
| $S_2^1$ | 80.00 / 2.000 | 80.35 / 1.999 | 0.24 |
| $S_3^1$ | -30.00 / 1.000 | -23.85 / 0.969 | 2.61 |
| $S_1^2$ | 79.20 / 1.000 | 78.41 / 1.027 | 1.67 |
| $S_2^2$ | 50.12 / 1.000 | 51.06 / 1.001 | 1.01 |

our original algorithm, we had to choose a very coarse grid spacings of more than 50 and 0.5 for $\theta_i$ and $R_i$, respectively (with reasonable search ranges of parameters $[-150, 150]$ and $[0.1, 2]$ for $\theta_i$ and $R_i$, respectively). The parameters $S_1^1$ , $S_2^1$ and $S_3^1$ optimized in this way are $-150/1.6$, $-150/0.1$ and $-50/1.1$, respectively, which are considerably worse than the results of our original algorithm.

## 4.5 Discussion

We have presented a multi-video synchronization algorithm that succeeds on multi-video sets comprising two or more views of general scenes. It does not require tracking of a specific object but utilizes feature trajectories tracked in individual cameras that are matched across views. To enable this, we contribute a robust trajectory filtering and energy minimization framework based on RANSAC for the multi-camera case. Moreover, we propose a novel strategy for identifying an informative subset of video pairs which further improves the multi-camera synchronization performance and prevents the RANSAC algorithm from being biased by outliers. In the following chapters, we propose human motion-capture algorithms which need the output of our synchronization algorithm to achieve high accuracy results in general scenes, indoors and outdoors. The algorithm proposed in this chapter is subject to a few limitations. Currently, it can not synchronize videos captured with moving or uncalibrated cameras. However, using the static background feature trajectories to estimate the calibration between each pair of corresponding frames, would enable our algorithm to work with these uncalibrated videos.

# Chapter 5

# Motion Capture with Unsynchronized Cameras

Human pose estimation from videos is one of the fundamental problems in computer vision and computer graphics which has been researched extensively in the past decades. Applications for these methods can be found in a wide range of industries, from entertainment (movies and games) to biomechanics, in sports, and medical sciences. Real-time capture methods made possible through new sensors such as the Microsoft Kinect have opened up new possibilities for human-computer interaction. However, even with all the developments in the past years, for accurate motion capture both industry and academia still rely on marker-based optical systems that require complex and expensive setups of cameras and markers.

A significant amount of research has thus been devoted to simplifying the setup and accuracy of marker-less methods [Moeslund *et al.* (2006); Poppe (2007); Sigal *et al.* (2010)]. However, these methods often rely on recording videos with synchronized cameras. Further, these setups require special hardware, and cannot make use of commodity camera hardware with limited frame rates. They are also often expensive and difficult to set up. Hasler *et al.* (2009a) have introduced a method that performs marker-less capture with unsynchronized commodity cameras. Their approach does not make use of sub-frame timing information and instead aligns all frames to the nearest discrete time step. The motion tracking is then performed in the same way as if the cameras were synchronized. This in turn leads to inaccuracies and a reduction of quality in the final results.

Another limitation of marker-less methods is that modern video cameras still have a limited frame rate. Marker-based systems often capture motion with over 120 frames per second or even higher, allowing them to accurately capture fast and

subtle motions alike. In contrast, most commodity video camera systems usually capture images with 30 Hz, with specialized vision systems capturing up to 60 frames per second at reasonable resolutions. This means that fast motion is harder to capture accurately with a marker-less setup. If the cameras are run without enforcing synchronization, more samples would be captured in the temporal domain, but spatial coherence will be lost, as in general no two cameras capture at the same time instance.

In Chapter 4, we have presented a multi-video synchronization algorithm that succeeds on multi-video sets comprising two or more views of general scenes. This algorithm estimate the synchronization parameters $\{\theta_c, R_c\}$ (with respect to reference time line $t_r$), for each camera $c$. As a second step toward human motion capture in general scenes with sparse multi-camera setups, we introduce a new spatio-temporal method for marker-less motion capture. Given an estimate of synchronization parameters for each camera, we reconstruct the pose and motion of a character from a multi-view video sequence without requiring the cameras to be synchronized. Therefore, our method allows cameras to capture videos with different sub-frame time offsets and even varying frame rates. In contrast to [Hasler *et al.* (2009a)], we use the sub-frame timing information instead of aligning all frames to the nearest discrete time step. At the same time, we are able to reconstruct motion in much higher temporal detail than was possible with previous synchronized approaches [Stoll *et al.* (2011)]. By purposefully running cameras unsynchronized we can capture even very fast motion at the frame rate that off-the-shelf cameras provide.

Our main contribution is the introduction of a continuous spatio-temporal energy functional that measures model-to-image alignment at any point in time: Rather than estimating discrete pose parameters at each time step, we estimate continuous temporal parameter curves that define the motion of the actor. By design, the energy functional is *smooth* and accordingly the derivatives of any order can be computed analytically, allowing effective optimization. Similar to [Stoll *et al.* (2011)], we represent both the actors body as well as the input images as Sums-of-Gaussians (SoG). We also present a method to enforce joint limits in the continuous pose-curve space. In the experiments we show that our approach can simplify the capture setup in comparison to previous marker-less approaches and that it enables reconstruction of much higher temporal detail than synchronized capture methods. Because of this, slow cameras can be used to capture very fast motion with only little aliasing. These contributions and results have been published in [Elhayek *et al.* (2012a)].

## 5.1 Method Overview

Multi-view tracking methods usually capture the performance of an actor with $n_{cam}$ synchronized video cameras (Fig. 5.1a). The human body is modeled using a kinematic skeleton and an approximation of the body geometry, using, for example, a triangle mesh from a laser scan [Gall *et al.* (2009)], a statistical model [Balan *et al.* (2007)], simple primitives like cylinders [Sidenbladh *et al.* (2002)], or a continuous function [Ilic & Fua (2006)]. For each frame $i$ at time $t_i$ of the synchronized input video streams the parameters of the kinematic skeleton $\Theta_{t_i}$ are optimized to maximize similarity of the pose with the input images. This can be measured with an energy functional $E_{t_i}(\Theta_{t_i})$ that is minimized.

Our approach instead considers unsynchronized video streams where each image is taken at a different time $t$ (Fig. 5.1b). Note that all cameras may run at different frame rates as well. We assume that timestamps $t_i$ for each image are given. These could be obtained using our optical multi-camera synchronization algorithm; see Chapter 4. Under specific assumptions, methods such as the audio-synchronization method from [Hasler *et al.* (2009a)] or the image based methods [Carceroni *et al.* (2004); Meyer *et al.* (2009)] can be used.

When recording unsynchronized video, it is possible to sample more densely in time compared to synchronized video. This comes at the cost of losing spatial information at each time instant (Fig. 5.1b). This poses a new challenge, as in the extreme case for a given time step, only a single view will be available. Exclusively fitting pose parameters to a single image at each time step would lead to unstable tracking since the problem is underdetermined due to ambiguities and occlusions. Instead of estimating the pose parameters $\Theta$ for each discrete time step, we estimate a smooth function $\Theta = X(t)$, which for each given time instance $t$, represents the corresponding vector of pose parameters. This representation enables us to aggregate information collected from nearby images in time, such that for each time step, the determination of pose parameter becomes well-posed. Effectively, we are trading spatial resolution for higher temporal resolution but we will show that we only lose a little spatial resolution and gain a lot in temporal accuracy.

As fitting a single continuous function to the whole sequence at once would require a very complex function and be difficult to optimize, we instead divide the sequence into overlapping segments $\mathscr{S}_j$ of length $l_{seg}$ and fit a set of simple polynomial functions to each segment (Fig. 5.1c). A globally continuous function is then computed by blending the segments with a partition of unity method (Fig. 5.1d).

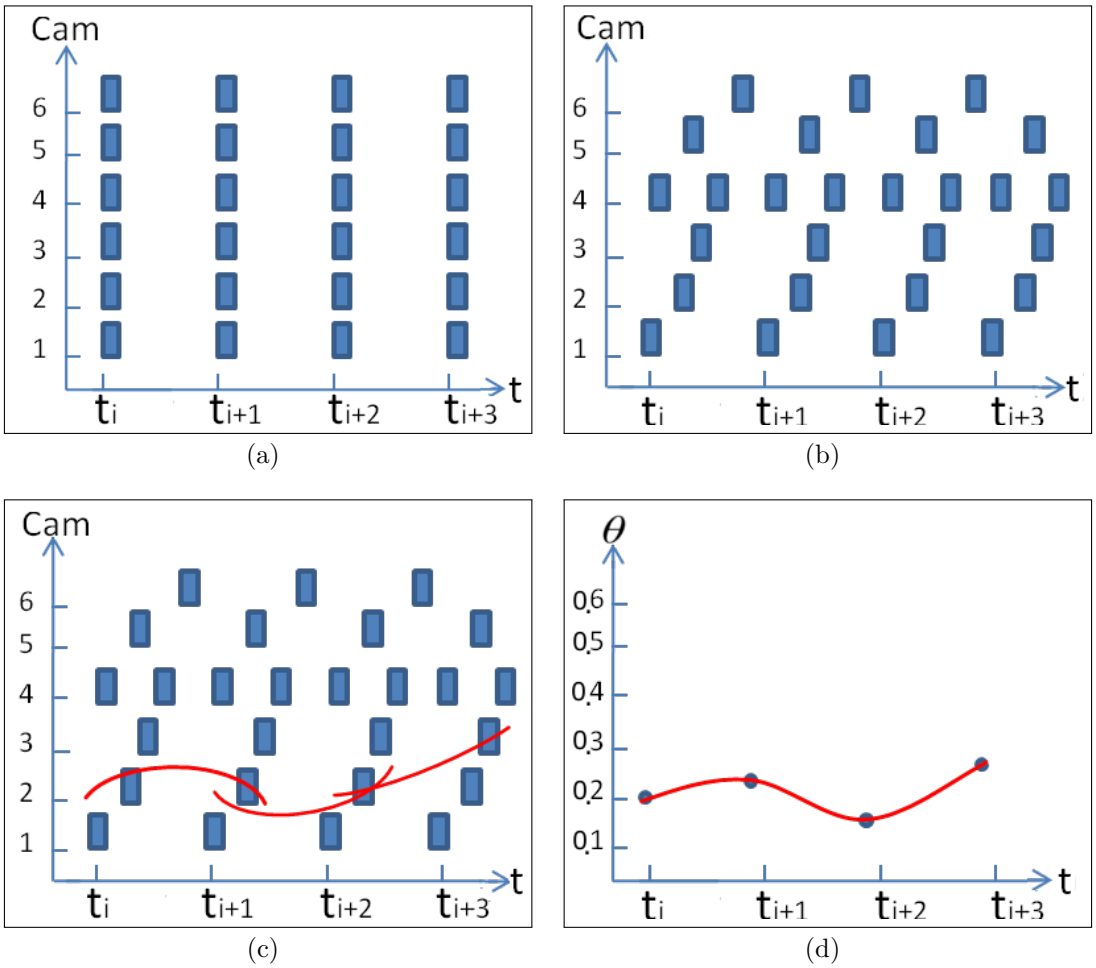



Figure 5.1: Basic concept: *(a)* Image distribution of synchronized cameras. Each blue rectangle corresponds to a frames in a video. *(b)* Image distribution of unsynchronized cameras after mapping to a single time line. *(c)* The interval functions for a single pose parameter. *(d)* After blending, we have reconstructed a continuous pose function for the entire domain.

## 5.2 Spatio-Temporal Tracking

The proposed tracking algorithm adopts an energy-minimization approach. We use an energy functional which measures the dissimilarity between a human body model and the input sequence. As described shortly, the energy functional is continuous both in space and in time such that the evaluation of the model (i.e. measuring the disagreement from the input) is possible at any given time (*c.f.* Section 5.2.1). To

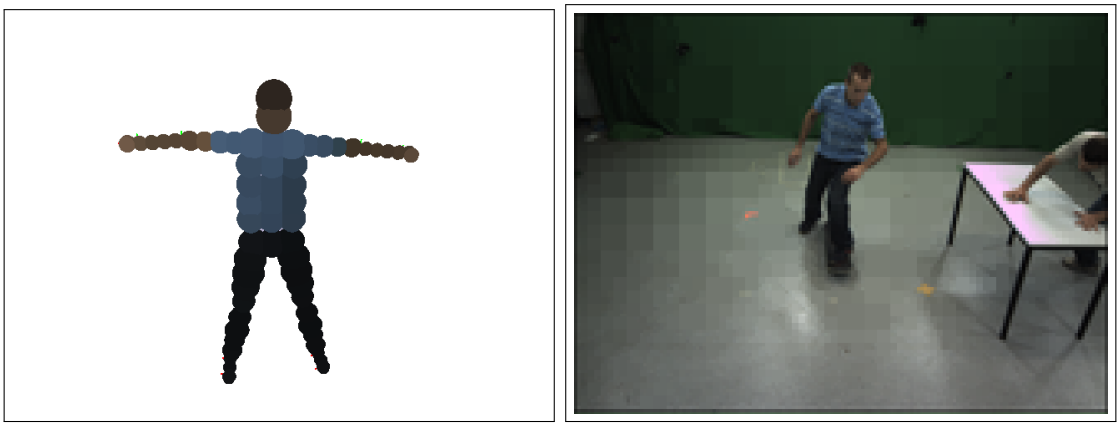


Figure 5.2: SoG model overview. *Left:* Body model generated from example input images. *Right:* Image SoG approximation generated from a quad-tree (each cell represents one Gaussian).

facilitate this, we represent the model based on continuous functions. Specifically, we adopt the Sums-of-Gaussians (SoG) representation as proposed by Stoll *et al.* (2011) and described in Section 3.2. Human articulation is modeled by a kinematic skeleton and its shape is represented using a 3D SoG, where each 3D Gaussian is attached to exactly one bone in the articulation hierarchy. The model is generated by fitting it to a set of example images (Fig. 5.2 left). To reduce the computational cost, the input images are also approximated based on 2D SoG using a fast quad-tree based clustering method (Fig. 5.2 right). Each single Gaussian in the SoG sets is associated with a color $\mathbf{c}$ that can be used to measure color similarity between two blobs. For each time step, measuring the similarity between a 3D SoG and a 2D SoG is facilitated by projecting the 3D SoG of the body model into the corresponding image plane and performing the comparison in 2D (Section 3.2.3). Using this SoG-based formulation as a basis has the advantage that the original formulation is already smooth in space. It does not rely on calculating and updating any image features or silhouette correspondences. As a result, extending the approach to the temporal domain comes naturally. It can also handle tracking highly complex articulated models; see Section 3.2 for details.

### 5.2.1 Spatio-Temporal Similarity Measure

As estimating a single continuous function for a whole sequence quickly becomes intractable, we first divide the sequence into overlapping time segments $\mathscr{S}_j$ of length $l_{seg}$. We represent each of the $n_{DoF}$ parameters of the kinematic skeleton for each

segment using a polynomial $X(t, \psi_j)$ of degree $n_{deg}$, where $\psi_j = \left[\chi_l^k\right]$ with $k \in 1 \dots n_{DoF}$ and $l \in 1 \dots n_{deg}$ are the coefficients of the polynomial. We call the function $X(t, \psi_j)$ the *motion function* for time segment $j$ (see Fig. 5.1c). Choosing a low degree polynomial as local motion function presents a good compromise between function smoothness and function complexity.

Given an input image SoG $\mathscr{K}_I^i$ with its respective timestamp $t_i$ and coefficients $\psi_j$ of the current motion function we can estimate the similarity between the two using (Eq. 3.6) as

$$E_{sim}(\mathscr{K}_m(X(t_i, \psi_j)), \mathscr{K}_I^i). \tag{5.1}$$

We can now sum up the similarity of all $n_{img}$ image SoGs $\mathscr{K}_I$ which belong to the segment $\mathscr{S}_j$ to get a spatio-temporal similarity measure over the entire segment:

$$E_{sim}(\psi_j) = \frac{1}{n_{img}} \sum_{t_l \in \mathscr{S}_j} \frac{1}{E_{sim}(\mathscr{K}_I^{t_l}, \mathscr{K}_I^{t_l})} E_{sim}(\mathscr{K}_m(X(t_i, \psi_j)), \mathscr{K}_I^{t_l}). \tag{5.2}$$

It should be noted that this similarity measure is smooth in space and time and accordingly the analytical derivatives of any order can be computed easily with respect to the coefficients $\psi_j$ of the model's motion functions.

### 5.2.2 Spatio-Temporal Joint Limits

An important component of articulated motion tracking systems is enforcing anatomically correct joint motion. All joints in the human body only have a limited amount of articulation. To prevent anatomically implausible poses, tracking systems usually penalize poses that exceed certain joint limits. This happens either by adding a penalty to the energy that is being optimized or by limiting the admissible range of DoF parameters through box constraints. Modeling these limits in the discrete case is straightforward, but becomes more involved in the spatio-temporal formulation from Section 5.2.1.

We want to penalize motion functions $X(t, \psi_j)$ where parts of the functions lie outside an admissible limit range $[l_l, l_h]$ for $t \in \mathscr{S}_j$ (see Fig. 5.3 for examples). We can define a penalty function $E_{lim}(j)$ that measures the area of the functions that

exceeds the limits within the segment as

$$E_{lim}(\psi_j) = \left( \int_{t \in \mathscr{S}_j \wedge X(t,\psi_j) < l_l} l_l - X(t,\psi_j)\,\mathrm{d}t + \int_{t \in \mathscr{S}_j \wedge X(t,\psi_j) > l_h} X(t,\psi_j) - l_h\,\mathrm{d}t \right)^2. \tag{5.3}$$

As can be seen in Fig. 5.3, this penalty function has to handle 10 different cases depending on the position of the curve with respect to the limits and the segment boundaries. Moreover, solving for a quadratic function can result in a linear or even constant function. Since the quadratic intersection with the DoF limits is undefined in such cases, it is necessary to handle them as additional limit cases. However, each case has a compact analytical solution and derivatives with respect to the curve coefficients $\psi_j$.

### 5.2.3 Segment Tracking

We combine the spatio-temporal similarity measure $E_{sim}$ and the limit penalty term $E_{lim}$ into a single energy functional

$$E(\psi_j) = -E_{sim}(\psi_j) + \alpha E_{lim}(\psi_j), \tag{5.4}$$

where $\alpha$ is a weight factor which determines how strongly we want to penalize non-anatomical pose configurations during tracking. As we can calculate analytical derivatives of both energy terms, we can calculate the gradient $\nabla E(\psi_j)$ efficiently. We find the minimum of $E(\psi_j)$ using a simple conditioned gradient descent method similar to [Stoll *et al.* (2011)]:

$$\psi_j^{i+1} = \psi_j^i + diag(\sigma_i)\nabla E(\psi_j^i). \tag{5.5}$$

The conditioner $\sigma_i$ is updated after every iteration according to the rules:

$$\sigma_{i+1}^{(l)} = \begin{cases} \sigma_i^{(l)}\mu^+ & \text{if } \nabla E(\psi_j^i)\nabla E(\psi_j^{i-1}) > 0 \\ \sigma_i^{(l)}\mu^- & \text{if } \nabla E(\psi_j^i)\nabla E(\psi_j^{i-1}) \leq 0. \end{cases} \tag{5.6}$$

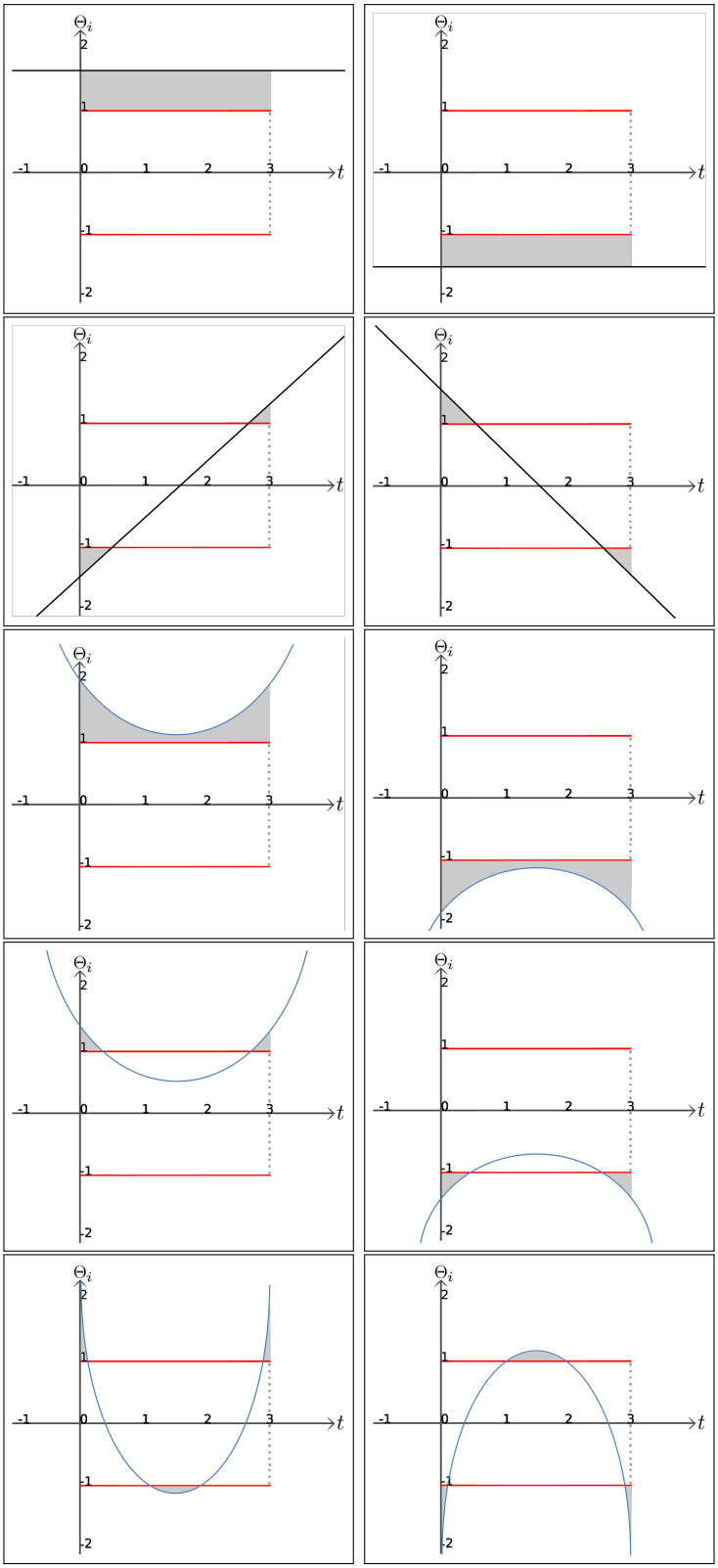



Figure 5.3: Limit violation cases. The red lines are the DoF limits boundaries. We compute the DoF function in the interval $\mathscr{S}$=[0,3]. The proposed error measure is the integral of the gray areas.

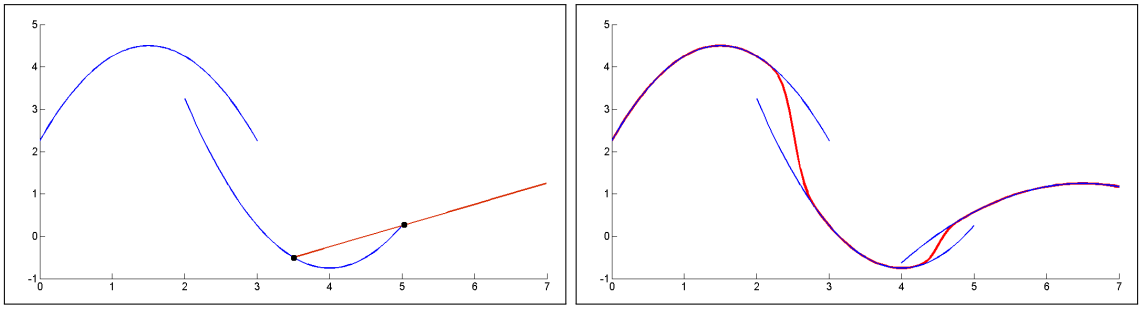


Figure 5.4: Motion functions. *Left:* Initialization of new segment (red) from previous segment function (blue). *Right:* Blended global motion function (red) generated from three local motion functions (blue).

Using the conditioner increases the convergence rate of the gradient descent method in long and narrow valleys of the objective function, as it effectively dampens oscillations and increases step-size in the direction of the valley. We found that this simple approach needs more iterations to converge than higher order optimization schemes, but is still faster in many cases as each iteration is simpler to calculate.

We assume that the actor in each sequence starts in a known pose (for example T-Pose) and is not moving for a brief moment. We find the parameters for the first segment $\mathscr{S}_0$ by initializing the body model pose to the known pose and only optimizing the constant coefficients $\chi_0^k$ of the motion function (Fig. 5.4). We ignore all linear and higher order coefficients and set them to 0. This essentially optimizes for a constant pose without any motion in the current segment.

Each following segment $\mathscr{S}_j$ is placed so that it overlaps with the previous segment by $l_{overlap}$, which is given as percentage of the segment length (Fig. 5.4). We initialize the coefficients of our current segment to be a linear extrapolation of the motion in the previous segment (Fig. 5.4). We then run the optimization for all parameters $\chi_l^k$ until convergence.

#### 5.2.3.1 Motion Function Blending

The estimated continuous functions for each segment $\mathscr{S}_j$ may not agree with each other in the overlapping regions (Fig. 5.4b in blue). To generate a globally smooth motion function we therefore blend all local motion functions together using a partition-of-unity approach (Fig. 5.4b in red). We define a weight function $w_j(t)$ for each segment that is 1 at the center and falls off smoothly to 0 at the segments boundaries, and is 0 everywhere else. Using the $\mathbf{C}^2$ smooth Wendland radial basis

function $\varphi_{3,1}(x)$ Wendland (1995) the final global motion function is defined as

$$X_{global}(t) = \frac{\sum_{\forall \mathscr{S}_j} w_j(t) X(t, \psi_j)}{\sum_{\forall \mathscr{S}_j} w_j(t)}. \tag{5.7}$$

Blending the motion function is a post-processing step and is performed after all segments have been optimized. The resulting motion function $X_{global}(t)$ is $\mathbf{C}^2$ smooth in $t$ and represents the tracking result of our algorithm.

## 5.3 Experiments

We evaluated our method on 9 sequences recorded with 11 unsynchronized cameras at a resolution of $162 \times 121$ pixels with varying frame rates between 45 and 70 frames per second with a total of about $\sim 6000$ frames of video. The camera setup used for our experiments provides us with accurate timestamps for each image. When using setups without this possibility, we could estimate timestamps using our optical multi-camera synchronization algorithm; see Chapter 4. Other methods such as the audio-synchronization method from [Hasler *et al.* (2009a)] or the image based synchronization methods [Carceroni *et al.* (2004); Meyer *et al.* (2009)] can be used in specific scenes which satisfy each method's assumptions. We estimated kinematic skeletons and Gaussian body models for 3 actors and used the quad-tree based image conversion from [Stoll *et al.* (2011)] to convert the input images to SoG models; see Chapter 3.

The recorded scenes cover a wide range of different motions, from simple walking/running, over fast acrobatic motions, to scenes with as many as 6 people featuring strong occlusions. The tracking approach does not rely on an explicit background subtraction and implicitly separates actors from background using the colors of the SoG body models. The green screen visible in part of the background is not used for explicit segmentation.

Our non-optimized, single-threaded implementation of the spatio-temporal tracker requires on average between 1 and 5 seconds to find the optimal parameters for each segment per actor. The specific run time per segment depends mainly on the motion complexity, i.e., fast motions with large frame-to-frame pose changes take longer to track.

Figure 5.5 shows pose estimation results of our algorithm for some of the sequences from different camera views. Our method tracked all sequences successfully with the same settings used for segment size $l_{seg} = 2.0$ frames of the slowest frame

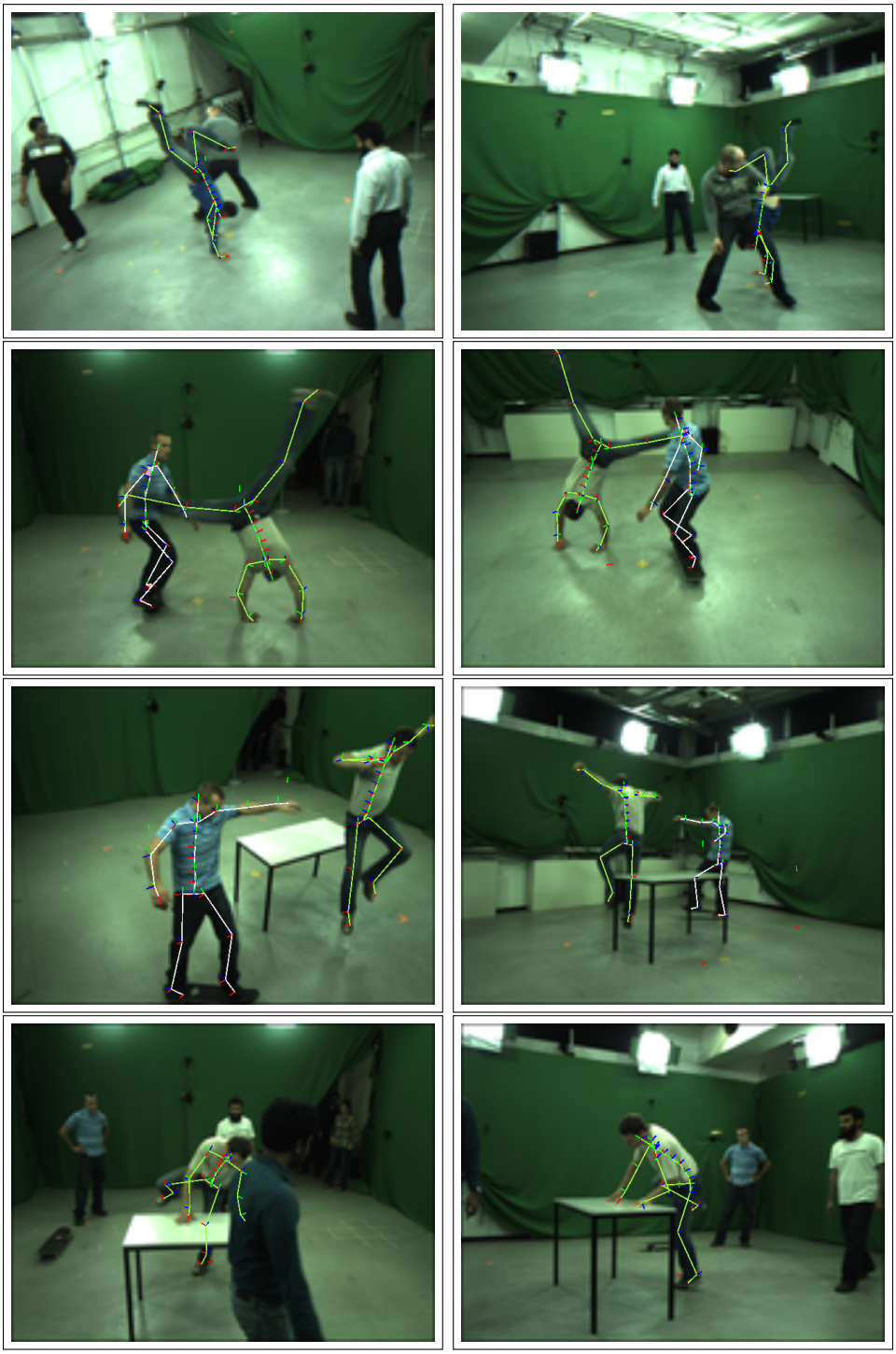


Figure 5.5: Complex motion tracking with 11 cameras. Tracking results of the proposed method on unsynchronized sequences shown as skeleton overlay over the input images. We successfully tracked actors in several challenging scenarios, including sequences with multiple people interacting closely, heavy occlusions, and fast motion from acrobatics and skateboarding.

rate, overlap of $l_{overlap} = 0.6$, and joint limit weight of $\alpha = 0.1$. The figure also shows results for tracking multiple people in the same sequence. Here, we tracked each actor separately without specifically modeling character interactions (such as contact and occlusion) or segmenting the input images.

Compared to results created by aligning multiple images to a single time-step and using a discrete tracking approach, our spatio-temporal formulation creates more accurate results. The discrete tracker also fails to correctly track some sequences with complex occlusions and fast motions.

**Quantitative Evaluation:** To evaluate our method quantitatively we recorded a sequence $\mathscr{S}_{ref}$ with the actor walking with increasing speed with a synchronized camera setup running at 70 frames per second (Fig. 5.6a). We then created an unsynchronized sequence $\mathscr{S}_{unsync}$ from this scene by temporally subsampling the input video such that only a single camera image is kept at each time instant (Fig. 5.6b). The downsampled sequence effectively has each camera recording at $\sim 7$ frames per second, slightly offset to each other. This represents an extreme case, as for all but the slowest motions, the cameras will see vastly different poses for the actor. Finally, we also created a synchronized low-speed sequence $\mathscr{S}_{low}$ which contains only every 11th frame for each camera (Fig. 5.6d). All three downsampled sequences contain the same number of images.

We used the full sequence $\mathscr{S}_{ref}$ to create a baseline synchronized tracking results $\mathscr{T}_{ref}$ using the method from [Stoll *et al.* (2011)]. We then tracked the actor from the unsynchronized sequence $\mathscr{S}_{unsync}$ with our spatio-temporal approach to generate a result $\mathscr{T}_{cont}$. We also generated tracking results by aligning all 11 cameras of $\mathscr{S}_{unsync}$ to the same time-step (Fig. 5.6c) and using the synchronized tracker to generate $\mathscr{T}_{aligned}$, and tracked sequence $\mathscr{S}_{low}$ to generate $\mathscr{T}_{low}$.

As can be seen in the results video in [Elhayek *et al.* (2012b)], both $\mathscr{T}_{aligned}$ and $\mathscr{T}_{low}$ fail to track the sequence correctly until the end. On the other hand, our spatio-temporal tracking result $\mathscr{T}_{cont}$ successfully tracks the motion of the actor even when the actor is moving extremely fast towards the end of the sequence. Figure 5.8 shows the per frame joint position error compared to the baseline result $\mathscr{T}_{ref}$ for the spatio-temporal result (red), the aligned discrete tracker (blue), and the low fps synchronized tracker (green). We used linear interpolation to create parameters for all frames of the sequence for the two discrete tracking approaches. Our approach has a slightly higher joint position error in the beginning of the sequence, where the motion of the actor is slow and aligning all frames to a single time instant is still a good approximation. However, as soon as the motion of the actor becomes faster,

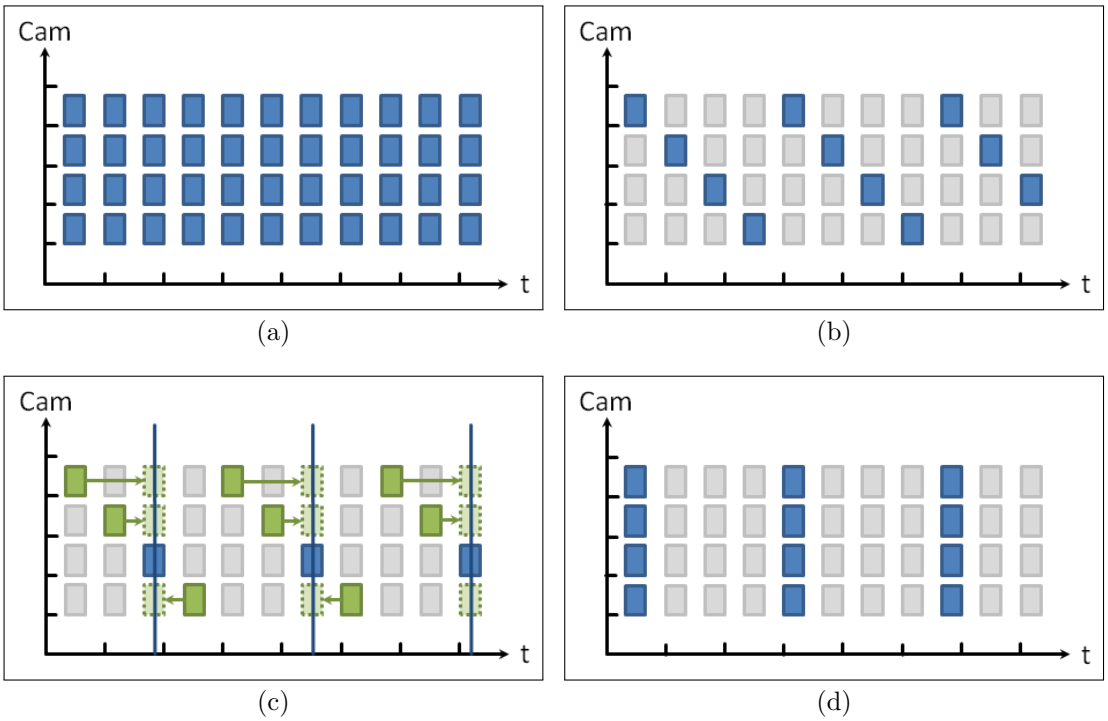



Figure 5.6: Evaluation sequence: (a) Synchronized baseline sequence. (b) Subsampled unsynchronized sequence. (c) Aligned sequence created from the unsynchronized sequence. (d) Low frame rate sequence.

the discrete tracker's error increases until it fails to produce correct poses at around frame 1800 (*c.f.* results video in [Elhayek *et al.* (2012b)]).

**Inaccurate Timestamps:** As accurate timestamps for each image may not be available when we estimate the synchronization from the video contents, we evaluated the influence of noise on the timestamps on tracking accuracy. We added Gaussian noise with a variance of 1/3rd to the phase of each camera (as the frame rate of the video is usually accurately determined). As can be seen in Fig. 5.9, the tracking accuracy of the noisy timestamps (red) does not significantly vary from the accurate timestamps (green). Our method is stable with respect to the expected inaccuracy of estimated timestamps.

**Polynomial Degree for Local Motion Functions:** We investigated the influence of the degree of the polynomial of the local motion functions on the accuracy of the tracking result in our quantitative evaluation sequence. As can be seen in

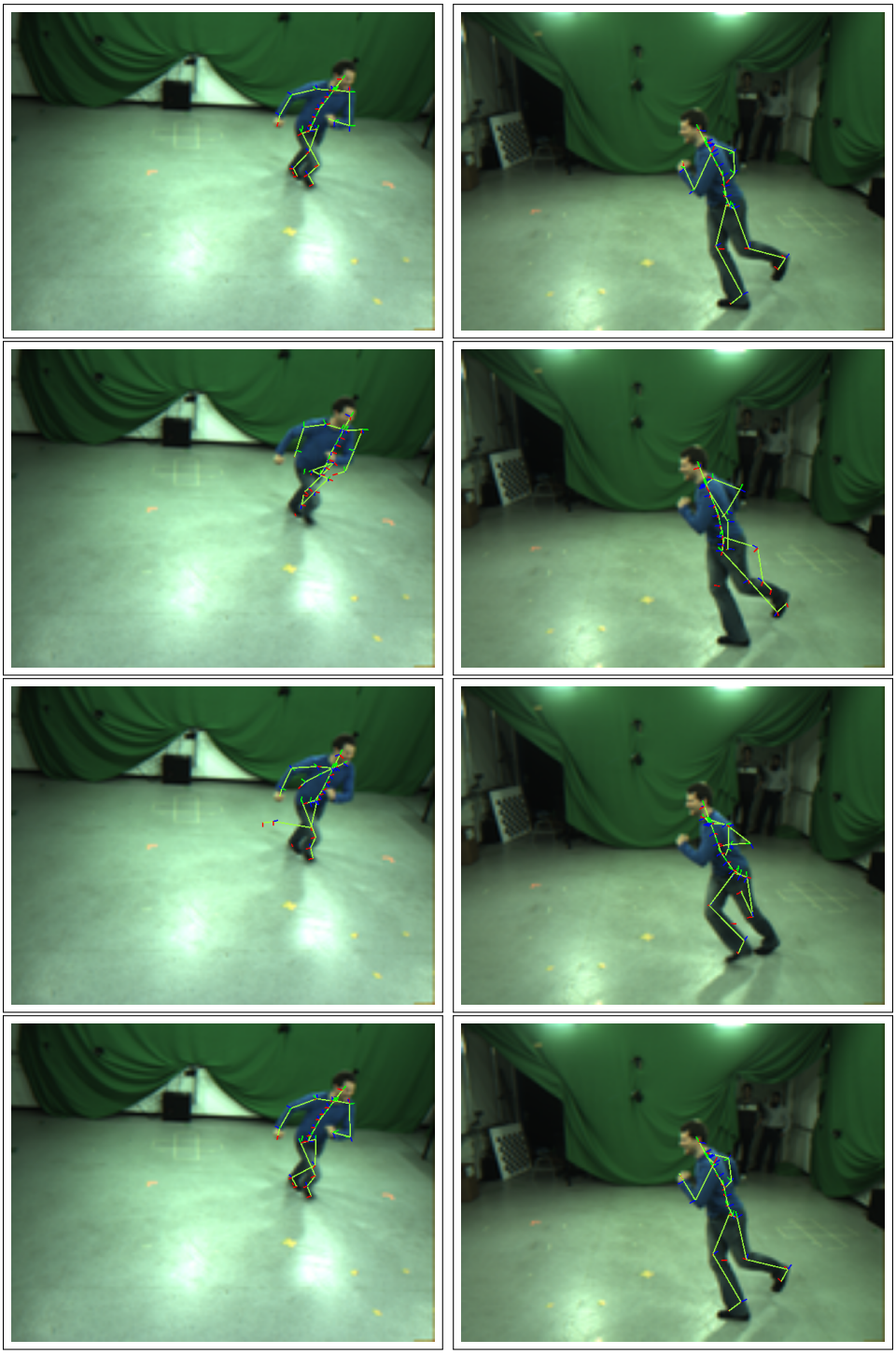


Figure 5.7: Quantitative evaluation. From top to bottom: Baseline tracking result $\mathscr{T}_{ref}$, aligned tracking $\mathscr{T}_{aligned}$, subsampled tracking result $\mathscr{T}_{low}$ and our tracking result $\mathscr{T}_{cont}$. Only our spatio-temporal tracking method is able to successfully track the whole sequence.

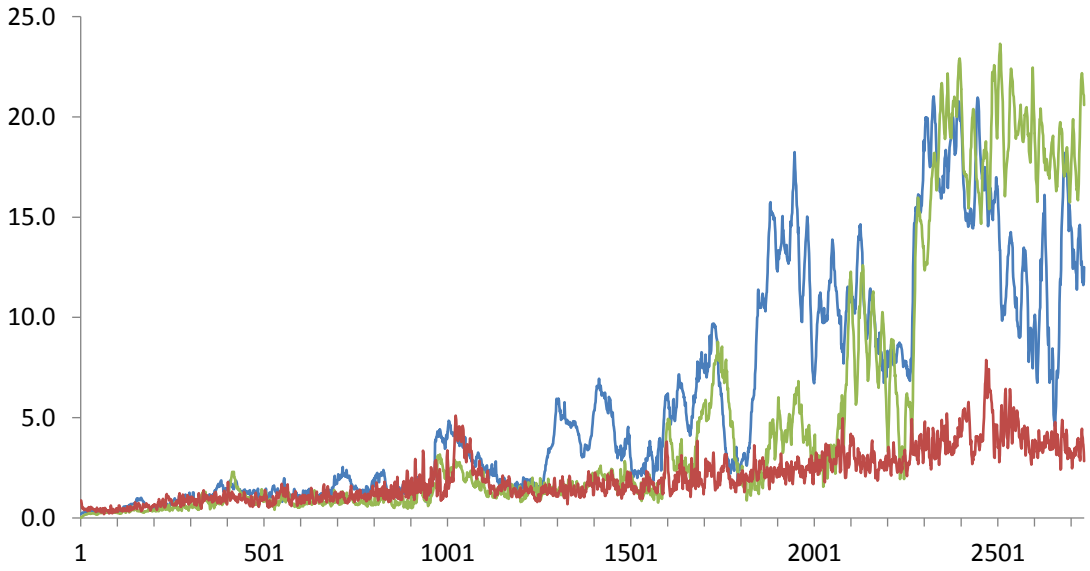



Figure 5.8: Quantitative comparison between our spatio-temporal tracking approach (red), synchronized tracking with unsynchronized input images (blue), and synchronized tracking with 7fps input (green). The vertical axis shows average joint position error in cm compared to the baseline result in the respective video frame. All tracking approaches use the same number of input images. As the actors motion becomes faster towards the end of the sequence, only our spatio-temporal approach is able to track the sequence correctly.

Fig. 5.10, both linear (blue curve) and quadratic (green curve) polynomials produce similar result. However, in some situations the linear motion estimation may produce strong inaccuracies (for example around frame 1700). We found that quadratic polynomials present a good compromise between fitting accuracy and computational complexity.

## 5.4 Discussion

We have introduced a spatio-temporal approach to articulated motion tracking from unsynchronized multi-view video. Unlike previous approaches that rely on synchronized input video, our method makes use of the additional temporal resolution to successfully track fast moving actors with low frame rate cameras. Our approach shows that using unsynchronized cameras not only enables us to use lower frame rate cameras for tracking, but also increases the tracking quality for fast motion as our quantitative evaluation shows. Despite this simpler setup, by running the cameras

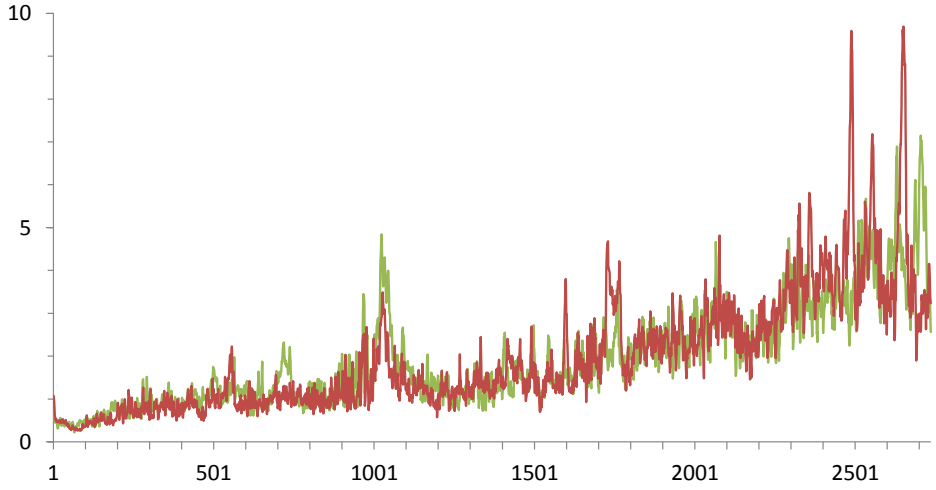



Figure 5.9: Comparing accurate timestamps (green) against inaccurate timestamps with added noise (red) on our quantitative evaluation sequence. The vertical axis is the average joint position error in cm with respect to the baseline, the horizontal axis is the frame of the sequence.

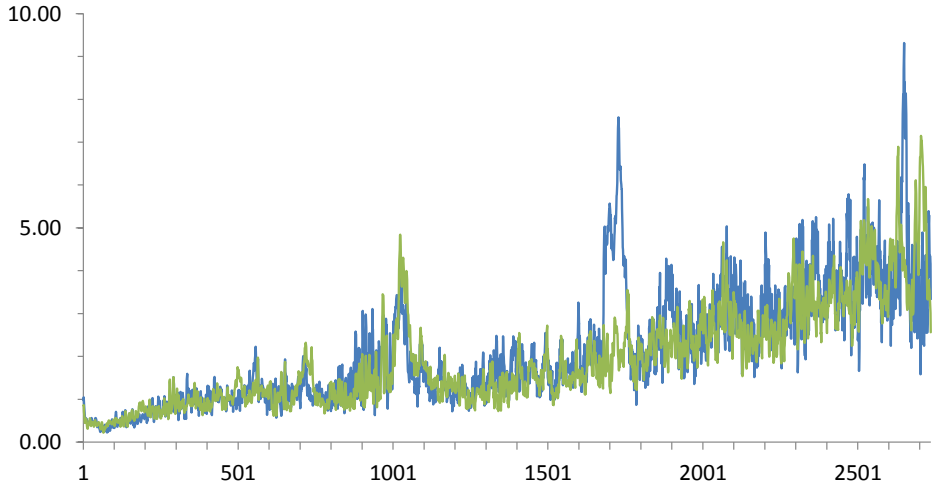



Figure 5.10: Comparison of linear motion functions (blue) and quadratic motion functions (green) on our quantitative evaluation sequence. The vertical axis is the average joint position error in cm with respect to the baseline, the horizontal axis is the frame of the sequence.

purposefully out-of-sync, the continuous tracker reconstructs fast motion at much higher quality as Fig. 5.8 shows. In practical situations, for example when capturing with camcorders, it will not be possible to control the sub-frame alignment of the camera shutters. Depending on the alignment the result will have more spatial accuracy (when nearly synchronized) or more temporal resolution (with unaligned input images). Our method also enables setting up simpler and cheaper capture setups, as there is no need anymore for hardware based synchronization and high frame rate cameras.

In practice, the tracking accuracy of the proposed algorithm decreases with input filmed using less than five cameras. Therefore, in Chapter 7, we propose a novel marker-less motion capture algorithm which achieves high tracking accuracy from input filmed with as few as two cameras. Another limitation of the algorithm proposed in this chapter is that it works only with static cameras which hinders many practical outdoor motion capture applications where cameras may need to be moved during capture for practical reasons. Thus, in the Chapter 6, we propose an extension of this algorithm which allows to capture the skeletal motions of multiple people even in front of cluttered and non-static backgrounds using a sparse set of potentially moving cameras in an uncontrolled environment.

As our method is using a simple local optimization approach, it may fail in complicated cases with many occlusions and few cameras. Although our approach is often more reliable than the synchronized implementation in [Stoll *et al.* (2011)], in our experience, we sometimes get stuck in a local minimum and are not able to recover. Using more advanced global optimization schemes such as presented in [Gall *et al.* (2009)], would enable us to detect and correct these errors. We also rely on the color of the actor being sufficiently different from the background in our error function, which could be improved upon by using more advanced color models for each Gaussian, such as color histograms. Despite these limitations, in most cases our algorithm successfully tracked even complex motions under severe occlusions with unsynchronized cameras.

To estimate a globally continuous function representing the motion parameters, we first construct local polynomials and then blend them using a partition of unity approach. This leads to a computationally efficient algorithm since the optimization of each local polynomials can be done independently. However, from a theoretical perspective, this approach is sub-optimal in the sense that the optimization does not take advantage of all available observed data (i.e. images). It is future work, to explore different possibilities of trading the computational complexity and the optimality of the parameter function in this context.

# Chapter 6

# Outdoor Motion Capture with Moving Cameras

Many computer vision applications require motions to be captured on site (i.e. in general outdoor environment) with moving cameras. Moreover, in computer graphics, motion capture is a widely used way to animate virtual human characters. Unfortunately, traditional marker-based motion capture systems are expensive and cumbersome to use.

Recent years have seen a significant improvement of *marker-less skeletal human motion capture* algorithms [Moeslund *et al.* (2006); Poppe (2007); Sigal *et al.* (2010)]. Many state-of-the-art methods come close to the performance of marker-based algorithms, but only when recording in highly controlled *studio setups*, where 1) there are sufficiently many exactly synchronized high-quality cameras; 2) each camera is static and scene motion is due to foreground objects only; 3) the background is not cluttered; 4) lighting is controlled; 5) the main foreground actor is seldomly occluded.

While relative to marker-based systems, this yields an easier apparatus with a reduced setup time, the hurdles towards practical application are still large and the costs are still notable. By being constrained to a controlled studio, marker-less methods fail to fully play out their advantage of being able to capture scenes without actively modifying them. A lot of practical computer vision and computer graphics applications require motions to be captured on site, i.e. the camera system needs to be brought to the location, because the motion itself cannot be relocated to a studio. Examples are capturing drivers in cars, motion capture on outdoor film sets, recordings of street performances, or the reconstruction of athletes in the field. In such situations, scenes are often cluttered and fore- and backgrounds

may be dynamic. Further on, placement and number of cameras may be starkly constrained, cameras can often not be synchronized, and they may (have to) move during recording for instance in order to follow a moving object. Some methods succeed in uncontrolled recording scenarios and allow certain camera motion (also outdoors [Hasler *et al.* (2009a)]), but have limited accuracy and would fail in case of 1) cluttered scenes and with unconstrained sparse camera sets; 2) small camera translation or pure rotational motion; 3) motion blur due to hand-held camera shaking.

We have introduced, in the previous chapter, a spatio-temporal approach to articulated motion tracking from unsynchronized multi-view video. As a third step toward a new widely applicable human motion capture setup, we aim to work with handheld cameras in general scenes. We therefore present a method for markerless 3D skeletal human motion capture that succeeds in uncontrolled environments and uses only a sparse, heterogeneous and weakly constrained camera setup. The algorithm reliably captures even complex 3D skeletal body motion 1) with potentially as little as five cameras (e.g. mobile-phone cameras); 2) with camera setups that are unsynchronized and of differing makes, resolutions and frame rates; 3) in cluttered indoor and outdoor scenes where backgrounds are dynamic and the actor can be occluded; 4) without using specialized auxiliary sensor information, such as depth images or inertial sensors; 5) with any type of camera motion even including notable shaking.

The core algorithmic contribution is a new generative skeletal pose tracker that minimizes a single model-to-image consistency measure simultaneously in the skeletal actor poses *and* the poses of moving cameras (see Section 6.2). We demonstrate that this strategy is essential to deal with scenes where cameras, foreground, and background can move and image-based pre-calibration (e.g. via structure-from-motion (SfM),e.g. [Pollefeys *et al.* (2004); Thormählen *et al.* (2008)]), fails. 3D model and 2D image data compared during consistency measurement are based on the Sums-of-Gaussian model used previously for indoor tracking with static cameras; see Chapter 3 and Chapter 5. However, the energy function and the minimization strategy have been profoundly extended to match this more challenging scenario. The smooth nature of our energy functional with analytic derivatives enables continuous optimization. It also enables the automatic detection of the occlusion of body parts either caused by the same person (self-occlusion) or by the other people in the same scene (Section 6.3). This is properly taken into account in the optimization.

While this is not the first method for outdoor motion capture, to the best of our knowledge, our algorithm is the first that aims for motion reconstruction with

moving cameras, unsynchronized video streams in an uncontrolled environment with uncontrolled cameras motion, and of multiple characters, all at once. In summary, the algorithm proposed in this chapter augments our algorithm in Chapter 5 which does not succeed under the aforementioned challenging conditions. The novel algorithmic contributions over previous work, that enable this, are:

1. A new pose fitting energy function extended to estimate each camera's motion together with actor pose; see (Section 6.2.2). In particular, the following extensions were done to the measurement of model-to-image consistency
   (a) Support for multi-person/multi-camera tracking
   (b) A two-sided similarity term [1]
   (c) Weighting in HSV color space
   (d) Prior on camera motion (smoothness)
2. The pose estimation scheme is using a new and improved occlusion handling.

In our experiments, we show qualitatively and quantitatively against ground truth that our algorithm can capture even complex and fast body motion in cluttered outdoor scenes, and that it succeeds with a wide range of heterogeneous, unsynchronized and moving camera systems with varying resolution, also just a few mobile phone or outdoor action cameras, such as *GoPro*. We also contribute with a comprehensive evaluation dataset for quantitative comparison. It comprises multi-view video footage recorded with static and moving cameras, ground truth camera motion data, as well as reference data from a marker-based motion capture system. This chapter closely follows [Elhayek *et al.* (2014a)], where the concepts presented here have been published.

## 6.1 Method Overview

Input to our algorithm is a set of video streams of the same scene, yielding a set of frames $\mathscr{I} = \{I_1, \ldots, I_n\}$ obtained from several cameras (camera indices omitted for readability). These cameras can be of varying make, resolution and frame rate, and they can move during recording. Video streams are not expected to be temporally

[1] The concept of symmetric similarity was presented in [Sminchisescu & Telea (2002)]. However, our novel continuous and differentiable two-sided term is essential in case of moving cameras and allows for fast tracking.

synchronized (see Section 7.4) and the global time stamps are explicitly estimated, as discussed shortly. We assume that intrinsic cameras parameters are known (e.g. through calibration before recording).

As opposed to studio-based methods, we assume that lighting can vary mildly during recording, large part of the background can be dynamic, and the tracked person can be occluded for the duration of a few frames. The output of our algorithm is a continuous *motion function* $X(t) : \mathbb{R} \rightarrow \mathbb{R}^n$ that returns an $n$-dimensional pose vector for a given time stamp $t$. It is important to note that $X$ is a short notation for $X_{global}(t)$, which is defined in (Eq. 5.7). In contrast to our algorithm in Chapter 5, here, $n$ is given as $n = 6c + m$, where $m = 43$ is the number of degrees of freedom of the skeletal model (pose and joint angles, thus describing the *pose*; Section 3.2 for more details) and $c$ is the number of moving cameras in the scene. The 6 parameters for each moving camera describe translation and rotation. Due to the ambiguity between camera and performer motion in a single camera view, we can represent camera motion as an additional rigid transformation to the pose of the actor in a specific single view. This simplifies the optimization, as camera parameter optimization can be handled in the same way as actor motion. Note that in our setting we represent joint parameters as continuous temporal curves, thus they can be calculated for every sub-frame time instant of the motion (see Section 6.2).

For each tracked actor, the template body model must be shape-adjusted, which we do in a semi-automatic way from a set of calibration poses prior to motion recording; see Section 3.2 for details. It could also be done manually in case one has no control over the footage and actor motion.

For tracking multiple people in a scene, we initialize the pose of each actor independently. Then, our algorithm estimates a single combined motion function $X$ that concatenates the motion functions of individual actors. This is different from our algorithm in Chapter 5 where we run a single-person tracker for each actor. With this setting, the occlusions caused by different actors would not be taken into account. However, by estimating a single large motion function, we handle multiple people tracking exactly in the same way as the self-occlusion (see Section 6.3). Accordingly, the remainder of this and the next sections focus only on the single actor case without loss of generality for multiple actors.

Before tracking commences, we first synchronize video streams up to a frame-level accuracy by using the audio stream [Hasler *et al.* (2009a)]. We refine this initial result by our global multi-view image-based synchronization method which yields frame rates and offsets at sub-frame precision; see Chapter 4 for details.

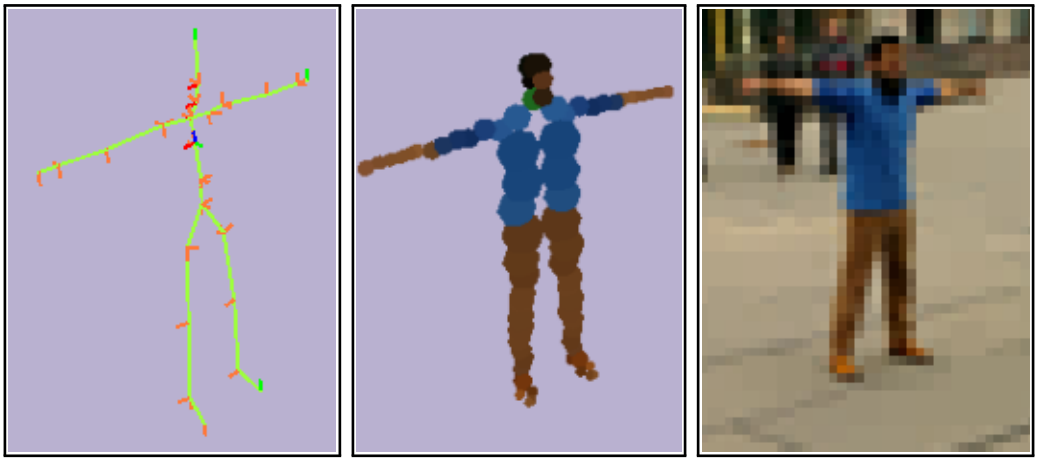


Figure 6.1: *Left:* skeletal representation of a performer. *Center:* a 3D SoG representation approximating the body shape of the performer. *Right:* image SoG representation (each box represents one 2D Gaussian $G^2$).

In the beginning, we also expect a small amount of user interaction to obtain an extrinsic camera calibration $C_{ext}$ for one multi-view frame of each camera at the nearest time stamps (after temporal alignment). We employ a bundle-adjustment with manually marked features in the scene background [Hartley & Zisserman (2004b)]. Note that this is only needed for one set of frames.

The core of our algorithm is a new energy minimization approach where a model-to-image consistency energy functional is jointly optimized with respect to camera pose and skeletal pose parameters (Section 6.3). The energy functional is based on the Sums-of-Gaussians scene representation of Chapter 5 which we profoundly extended to deal with our more general scene conditions, such as moderate appearance variations, occlusion, and dynamic background, as well as the sparse visual evidence from only few cameras (Section 6.2.2). In this Chapter, we briefly restate important concepts from prior work that we build on, but focus on the newly developed extensions. Employing a space-time optimization strategy is essential (Section 6.2.1) to deal with the lack of exact frame synchronization, and to be able to benefit from larger temporal baselines to regularize tracking with few cameras. With few cameras, occlusions of the actor, even for a short period, can lead to catastrophic failure of joint angle and camera optimization (see Section 7.4 for examples). We explicitly detect occlusions by monitoring the energy variation in time. Once an occlusion is detected, the corresponding camera is disabled for optimization and

does not contribute to the energy anymore. In case it is a moving camera, its pose parameters are re-initialized based on corresponding linearly interpolated parameter values (Section 6.3).

## 6.2 Tracking with Moving and Unsynchronized Cameras

One of the most important aspects of motion tracking with casually captured videos is that the cameras may move, and accordingly, the camera parameters have to be estimated for each frame in the video. In existing approaches for this scenario, estimation of these two sets of parameters is decoupled by pre-estimating camera parameters (e.g. performing SfM) and subsequently optimizing the pose (of the actors) given these known cameras. Unfortunately, this strategy cannot be exercised when the background is cluttered, the camera translation motion is not sufficient or the videos are blurry due to shaking cameras unless these conditions SfM fail (see Fig. 6.7 and results video in [Elhayek *et al.* (2014b)]). Our video streams (e.g. with as little as five cameras) are sparse and frame capture is not synchronized. Furthermore, there are many inherent ambiguities in model-image-matching which aggravate finding an optimal solution: the free motion of the body cannot be decoupled from the ego-motion of the cameras. For instance, it is often impossible to distinguish between an actor moving towards a static camera and a moving camera approaching a static actor.

A core innovation of our tracking algorithm is the simultaneous optimization of skeletal pose parameters and the pose parameters of every moving camera. Both camera and skeletal pose parameters are separately retained, but the effect of changing one set of parameters could still be compensated by the change of the other. Capturing the scene with one or more static cameras resolves this ambiguity. However, when there are no static cameras, the final results can only represent relative motion to the cameras and will not have a fixed global coordinate space. Our system uses the body model as common reference point to optimize skeletal pose and the pose of moving cameras. We are thus not forced to rely on unstable background features.

Our approach is instantiated as an energy minimization algorithm:

$$E(X_i) = -E_{sim}(X_i) + \lambda_1 E_{Lim}(X_i) + \lambda_2 E_{Smooth}(X_i), \tag{6.1}$$

where $E_{sim}(X_i)$ measures the similarity of model parameter $X_i$ to data $\mathscr{S}$ in each segment $\mathscr{S}_i$, as discussed shortly. $E_{Lim}$ and $E_{Smooth}$ are *prior terms* that enforce limits and smoothness on joint angles and camera poses, respectively. The hyper-parameters $\lambda_1$ and $\lambda_2$ are set to 0.1 and 0.01, respectively (see Section 7.4 for discussion on tuning hyper-parameters).

The individual components of the energy are detailed in the following. We will also detail the continuous pose parametrization and specific representation of image and shape data we employ. Optimizing continuous curves for skeletal and camera poses is essential since our data are not frame-synchronized and are spatially sparse. We can stabilize optimization by considering image data from larger temporal baselines.

### 6.2.1 Continuous Parameterization and Scene Representation

We extend concepts from Chapter 5 and instead of identifying parameter vectors for each discrete time stamp (corresponding to synchronized frame index), we construct a continuous, parameter vector-valued *motion function* $X(t)$ of time $t$ representing both skeletal and camera pose parameters. In this setting, the similarity between a specific motion function $X$ and a set of images is evaluated by sampling $X$ at the time stamp $t_i$ of each image $I_i \in \mathscr{I}$.

For representing the 3D spatial extent of the body model, as well as the 2D input images, we employ the Sums-of-Gaussians (SoG) model; see Section 3.2 for details. Fig. 6.1 shows an example of a smooth (i.e., continuously differentiable as many times as desired) 3D SoGs representation of human body model and 2D SoGs representation of input image, respectively. We captured the corresponding sequence in outdoors scene.

Since estimating a continuous motion function $X$ for a long sequence is computationally intractable, we divide the sequence into overlapping time segments $\{\mathscr{S}_1, \ldots, \mathscr{S}_{n_s}\}$ (of length 2/30 Section for each, with an overlap interval of 0.6/30 sec) and estimate the local motion functions $X_i$ as quadratic polynomials for each $\mathscr{S}_i$ independently. This results in $3 \times n$ parameters for each $X_i$ (3 being the number of parameters for a quadratic polynomial; there is no coupling between different parameters). A globally smooth motion function $X$ is implicitly reconstructed by blending the $X_i$ at overlap using partition-of-unity; see Section 5.2.3 for more details. Accordingly, the variables to be optimized are the coefficients of the polynomial $\psi_i$ for each segment $i$.

### 6.2.2 Model-to-image Similarity Term

We now explain the similarity term $E_{sim}(X_i)$ which measures the similarity of the 3D model defined by the motion function $X_i$ to each input image $I$ which belong to segment $\mathcal{S}_i$. This term is an extension of the spatio-temporal similarity term (Eq. 5.2) needed in Chapter 5. We represent a video frame $I_j$ with time stamp $t_j$ as the 2D SoG $\mathcal{K}_I^{t_j}$ and the respective model which is defined by the motion function $X_i(t_j)$ as 3D SoG $\mathcal{K}_m(X_i(t_j))$. Then, the similarity is calculated by projecting each 3D Gaussian function of $\mathcal{K}_m(X_i(t_j))$ into the image plane of $I_j$ and measuring the overlap to all 2D Gaussians of $\mathcal{K}_I(t_j)$.

In contrast to (Eq. 5.2), the positions of each 3D Gaussian with respect to any camera is a function of both the skeletal pose parameters and the parameters of that camera which are encoded in $X(t_j)$. However, it should be noted that the skeletal parameters are optimized based on the data of every camera, while the parameters of each moving camera are optimized based on the data of that camera only (i.e. by maximizing the similarity between this camera's 2D SoG and the projected 3D SoG). Then we can define the similarity term $E_{sim}(X_i)$ of the motion function $X_i$ as the sum of similarities of $\mathcal{K}_m(X_i(t_j))$ and $\mathcal{K}_I(t_j)$ for each $t_j \in \mathcal{S}_i$:

$$E_{sim}(X) = \frac{1}{n_{img}} \sum_{t_j \in \mathcal{S}} \frac{E_{sim}(\mathcal{K}_m(X_i(t_j)), \mathcal{K}_I^{t_j})}{E_{sim}(\mathcal{K}_I^{t_j}, \mathcal{K}_I^{t_j})}, \tag{6.2}$$

where $n_{img}$ is the total number of images in the segment $\mathcal{S}_i$, $E_{sim}(\mathcal{K}_m(X_i(t_j)), \mathcal{K}_I^{t_j})$ is the similarity of a 3D SoG and a 2D SoG, which is defined in (Eq. 3.6). Since every $\mathcal{K}_I^{t_j}$ for $t_j \in \mathcal{S}_i$ consists of a different number of 2D Gaussians, we normalize $E_{sim}(\mathcal{K}_m(X_i(t_j)), \mathcal{K}_I^{t_j})$ by the similarity of $\mathcal{K}_I^{t_j}$ with itself. The general similarity of two 2D SoGs is defined in (Eq. 3.2).

**Weighting in HSV color space.** As illumination in outdoor scenes can vary more strongly than in studio setups, we use a new color similarity that is more resistant to intensity changes. The similarity $d$ for two HSV values $\mathbf{a}$ and $\mathbf{b}$ is defined as

$$d(\mathbf{a}, \mathbf{b}) = 2\varphi_{3,1}(\|\mathbf{a} - \mathbf{b}\|_W) - 1, \tag{6.3}$$

where $\varphi(\cdot)_{3,1} : \mathbb{R} \to [0, 1]$ is the smooth, compactly supported Wendland function [Wendland (1995)] and $\|\mathbf{a}-\mathbf{b}\|_W^2 := (\mathbf{a}-\mathbf{b})^\top W(\mathbf{a}-\mathbf{b})$ with $W = \text{diag}([1, 1, 0.2]^\top)$,

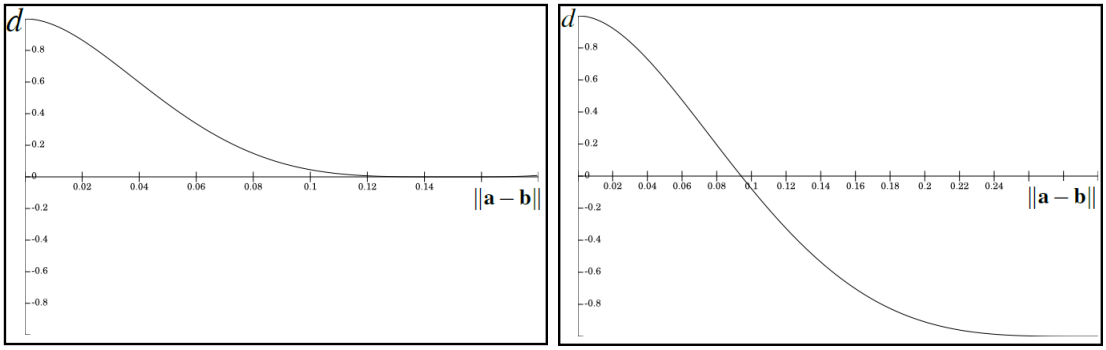



Figure 6.2: Two-sided vs. one-sided color similarity term. *Left:* The one-sided similarity term $d(\mathbf{a}, \mathbf{b}) = \varphi_{3,1}(\|\mathbf{a} - \mathbf{b}\|)$ only adds positive values to the energy if the color differences $\|\mathbf{a} - \mathbf{b}\|$ are less than a Wendland support of 0.15. *Right:* The two-sided similarity term (Eq. 6.3) additionally penalize dissimilar colors.

and diag($\mathbf{v}$) builds a diagonal matrix consisting of the entries of $\mathbf{v}$. The down-weighting of the *value* component (using $W$) in the HSV model is decided based on preliminary experiments: In outdoor scenes, the value component was most severely affected by changes of illumination (e.g., shading, highlight, and specularity) and we experimentally determined a factor of 0.2, i.e. 20% to be a good choice.

**Two-sided color similarity term.** In contrast to the one-sided color similarity term used in Chapter 5 where $d \geq 0$, the *two-sided* color similarity term (Eq. 6.3) can be negative when the two input colors are distinct (see Fig. 6.2). This is important in case of moving cameras, where each camera has its own pose (i.e. translation and rotation) parameters which are optimized based on its own data only (in contrast, the human pose parameters are constrained with data from several views). Figure 6.3 shows that with the one-sided similarity term, for a given skeletal pose and camera parameters, one can erroneously increase the similarity between the model and the image by moving the camera towards the object. In this case, the corresponding projected 3D Gaussians become larger and accordingly they lead to higher similarities in (Eq. 6.2) in general. In contrast, the two-sided term solves this ambiguity by penalizing the dissimilarity between the projected object and the background. In addition, the two-sided term is also generally important to enable reliable tracking with only very few static cameras, as we show later.

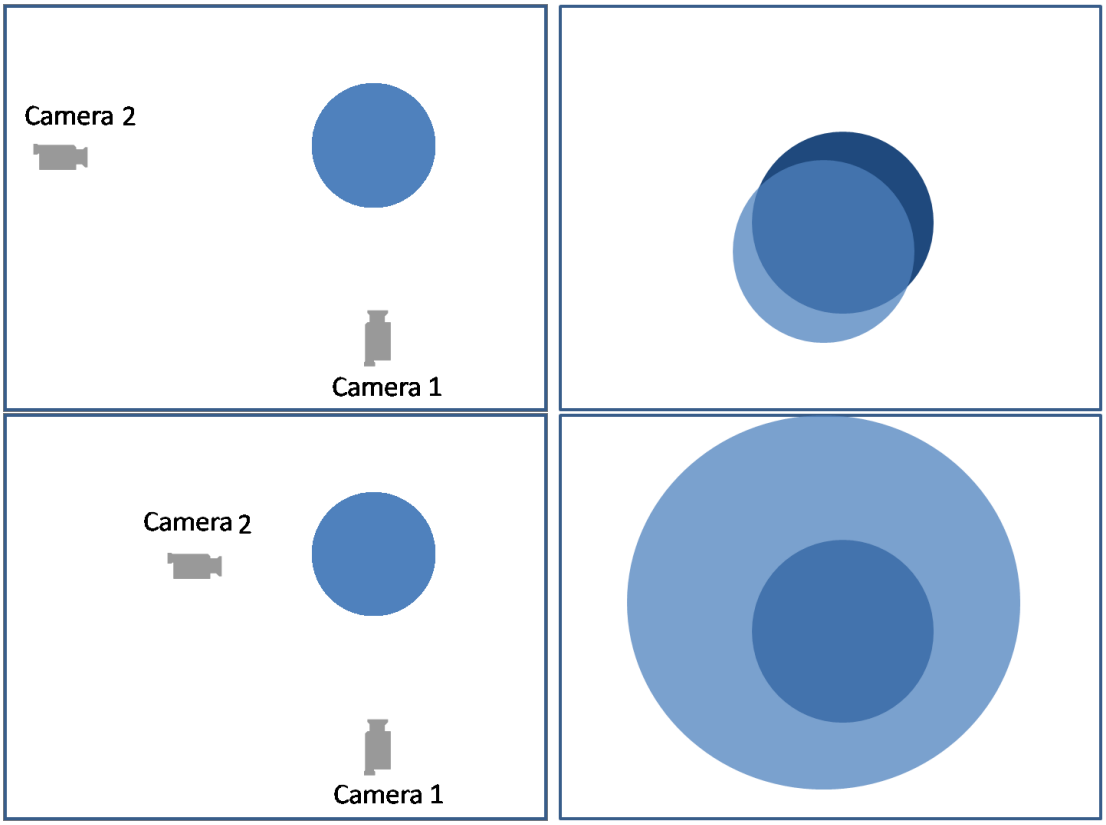



Figure 6.3: Importance of two-sided energy. Tracking an object with two cameras. *Left:* locations of cameras with respect to the object and *Right:* the constant input image from camera 2 (dark blue circle) overlaid with the projection of the 3D object (light blue) which is a function of the camera and pose parameters. If we move camera 2 closer to the object (bottom row), the virtual object is larger than the input which increase the overlapping between them. This can erroneously increase the similarity if we do not penalize the dissimilarity with the white background, which is achieved with the two-sided similarity term (Section 6.2.2).

### 6.2.3 Prior on Camera Motion

As motions of camera and actor are inherently ambiguous (see earlier discussion), estimation of the camera and pose parameters is inherently ill-posed. During the optimization, the effect of a change in camera position may be canceled out in the energy by opposite global pose change of the body. Temporal drifting over subsequent segments and irrevocable convergence to an erroneous local minimum could thus easily happen (see Fig. 6.9d for an example).

We approach this problem by enforcing first-order temporal smoothness over the parameters. This prevents any rapid change in parameters and the above-mentioned problem can be prevented as our experiments confirm (see Section 7.4):

$$E_{Smooth}(X) = \sum_{i=1}^{n_s} \sum_{j:S_j \cap S_i \neq \emptyset} \|X(l_i) - X(l_j)\|^2, \tag{6.4}$$

where $l_i$ is the middle point of a segment $\mathscr{S}_i$ in time and $n_s$ is the number of segments. $E_{smooth}$ requests similar values for model parameters at midpoints of overlapping segments from all camera sources. The other prior term $E_{Lim}$ used in (Eq. 6.1) constrains joint angles to an anatomically plausible range as in Chapter 5.

## 6.3 Combined Camera and Pose Optimization

We optimize the energy functional (Eq. 6.1) using the conditioned gradient descent approach presented in [Stoll *et al.* (2011)]; see Section 3.2 for more details. At the beginning of a motion sequence, we expect that the body model is shape initialized, and a rough manual initialization of the pose $\mathscr{S}_0$ in which the actor stands is given. This shape initialization is performed as described in Chapter 5.

**Occlusion handling.** As explained earlier, detecting and handling the case that a person is occluded from a camera view is crucial for our method. By design, the contribution of each camera to the similarity term is clearly separated from the other cameras (Eq. 6.2) and is smooth over time. This enables occlusion detection by monitoring the variation of each model Gaussian's energy component in time.

During the optimization, we inspect the similarities between the projected 3D Gaussians of the model $\mathscr{K}_M$ and the 2D image Gaussians of each camera $\mathscr{K}_I$: For a given image SoG $\mathscr{K}_{I(k)}$ (corresponding to a camera $\mathscr{C}_k$), the model Gaussian $G_i \in \mathscr{K}_M$ is marked as *false-projected* when $\max_{G_j \in \mathscr{K}_{I(k)}} E_{ij} < T_o$ for a given threshold $T_o$,

where the similarity $E_{ij}$ is calculated for the pair $\Psi(G_i)(G_i \in \mathscr{K}_M)$ and $G_j \in \mathscr{K}_{I(k)}$. when the number of false-projected Gaussians is larger than a threshold $T_n$, we decide that an occlusion has occurred in the image $I_k$. In this case, we exclude this camera from the optimization, as the occlusion may otherwise negatively influence the pose estimation of the other cameras. If the occluded camera is non-static, we also do not optimize the corresponding camera parameters. The pose optimization is then continued with the remaining cameras, and the parameters of the camera in which the skeleton is occluded are linearly extrapolated. The parameters $T_o$ and $T_n$ were held fixed at 0.6 and 30 during the experiments.

During the occlusion, the extrapolated parameters of the cameras are compared with the corresponding projected SoGs of the human model (as estimated based on unoccluded cameras). In this way, the end of the occlusion can be detected (i.e. when the number of false-projected Gaussians is less than a threshold $T_n$). In case the occluded camera is moving, once the occlusion is finished, the camera tracking starts again with the extrapolated parameters as initialization. This strategy succeeds in most of our test cases where occlusions are short and camera motion smooth. In all other cases (e.g. Fig. 6.13), more time-consuming global pose optimization would be needed after the occlusion ends.

## 6.4 Experiments

We evaluated our algorithm on seven real world sequences, which we recorded in an uncontrolled outdoor scenario with varying complexity: The sequences vary in the numbers and identities of actors to track, the existence and number of moving objects in the background, and the number of moving and static cameras. Sequences also differ in the makes, the frame resolutions, and the frame rates of cameras. By quad-tree decomposition, all images are effectively downsampled to a resolution of $160 \times 90$ before tracking. Moreover, we recorded two additional sequences in the studio for marker-based quantitative evaluation of both skeletal motion, as well as camera motion reconstruction accuracy. Table 7.1 summarizes the specification of the sequences used in the experiments. Since the cameras are not sub-frame-level synchronized, it is unlikely that frames from all cameras are available for a given time stamp. Accordingly, the time complexity can only be measured based on an average over a time span. Due to the more elaborate energy (in particular the two-sided term), and a larger parameter space, the runtime of our algorithm is slightly lower than the approach in Chapter 5. Further, the run-time of our

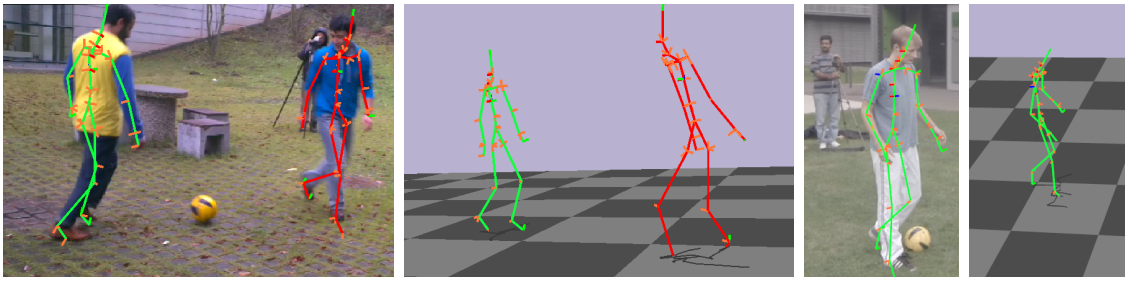


Figure 6.4: Examples of multi-person tracking with moving cameras. (Left two images) two actors, and two moving and three static cameras (*Soccer*1). (Right two images) One actor, and three moving and two static cameras (*Walk*2).

Table 6.1: Specification of the sequences used in the experiments where $W$ is short for *Walk* (i.e. $W1$ is $Walk1$).

| Sequence | *Soccer*1 | *Soccer*2 | *W*1 | *W*2 | *W*3 | *W*4 | *Run* | *W*5 | *W*6 |
|---|---|---|---|---|---|---|---|---|---|
| # moving cams. | 2 | 1 | 1 | 3 | 8 | 2 | 1 | 0 | 3 |
| # static cams. | 3 | 4 | 4 | 2 | 0 | 6 | 6 | 8 | 5 |
| frame rates | 23.8 | | | 120 | | | | 25 | |
| camera types | mobile-phone (HTC One X) | | | *GoPro* | | | | *studio cam.* | |
| frame resol. | 1280 × 720 (original); 160 × 90 (operating resol.) | | | | | | | 256 × 256 | |
| # tracked objs. | 2 | | 1 | | | | | | |
| # moving background objs. | 0 | | 9 | | | | | 0 | |

algorithm depends on the number of cameras and actors, and the complexity of the scene, e.g. the number of Guassians needed in 2D. For a single actor and five cameras, our algorithm takes around a minute for processing a single segment $\mathscr{S}$ (Section 6.2.1) that contains around two frames captured from each camera. Using the discrete (non-space-time) optimization algorithm of [Stoll *et al.* (2011)] (see also Section 6.4.3) on a similar sequence recorded in studio (i.e. lower scene complexity) with 8 cameras, our method performs at 13 seconds per frame. Apart from body model initialization which requires the user to apply a few strokes to background segment the images of four actor initialization poses (see Section 3.2), tracking is fully-automatic.

Figures 6.4, 6.5, 6.6, and 6.7 show example poses tracked from sequences *Walk1*,

*Walk2*, *Walk4*, *Run*, *Soccer1*, and *Soccer2* (see also the results in the video in [Elhayek *et al.* (2014b)]). Our algorithm successfully estimated the pose parameters of actors as well as the positions of the moving cameras in these sequences. In particular, our algorithm successfully tracked the two actors in *Soccer2* who often occlude each other (Fig. 6.9) and the actors in highly cluttered scenes ( *Walk2*, *Walk4*, and *Run* each of which contains 9 moving background people). When tracking the moving camera from the *Soccer1* sequence, SfM failed to successfully estimate the camera motion due to motion blur as shown in Fig. 6.7: Since the hand-held camera is shaking, motion blur occurred across several frames which causes feature tracking to fail. In contrast, our method was able to successfully track both camera and actor pose, even with the challenging background. Only on some isolated frames with stark occlusion, the arm or head are incorrectly tracked, as expected.

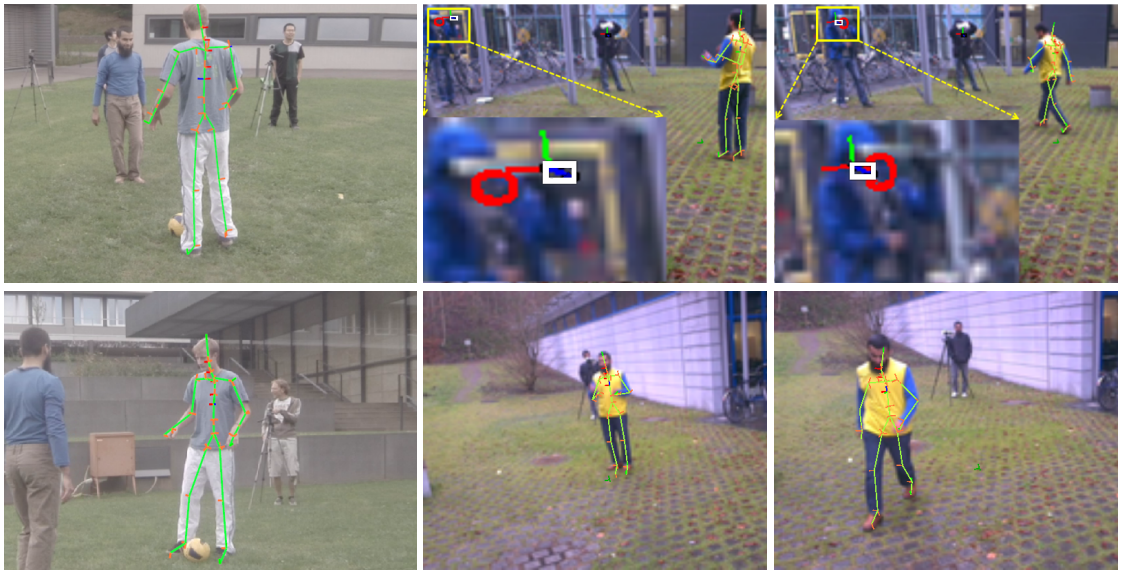
To evaluate whether using moving cameras in addition to static cameras, actually improves the quality of the pose reconstruction, we tracked sequence *Soccer1* once with only 3 static cameras, and compared the results to the full tracking with 3 static and 2 moving cameras (Fig. 6.8). While moving cameras add unknowns to the optimization problem, the additional images provide enough information to increase the tracking accuracy and estimate the camera dynamics.

### 6.4.1 Evaluation of Algorithmic Design Choices

We qualitatively evaluated the importance of the various components of our algorithm (see Secs. 6.1 and 6.2) on the *Soccer2* sequence. Figure 6.9a compares tracking with our new (non-positive definite) color similarity measure (Eq. 6.3) (top row) against tracking with the old color measure of [Elhayek *et al.* (2012a); Stoll *et al.* (2011)] (bottom row). The new color measure is crucial for successfully estimating the motion of moving cameras (see Section 6.2.2).

In outdoor recordings, the observed brightness of the objects and the background can change. By making our color similarity measure more resistant to changes in brightness (Fig. 6.9b, top row), tracking becomes more stable compared to the original brightness-sensitive color measurement (Fig. 6.9b, bottom row). When the Euclidean distance is used instead (bottom), the color of the body model is not distinctive enough from the background color as the change of incident illumination (due to shadows; see results video in [Elhayek *et al.* (2014b)]) leads to a large variation in the value component, that causes a tracking error.

Figure 6.9c demonstrates the importance of our occlusion handling. When the number of cameras is limited, the erroneous contribution of a camera under occlusion



(a) (b) (c)

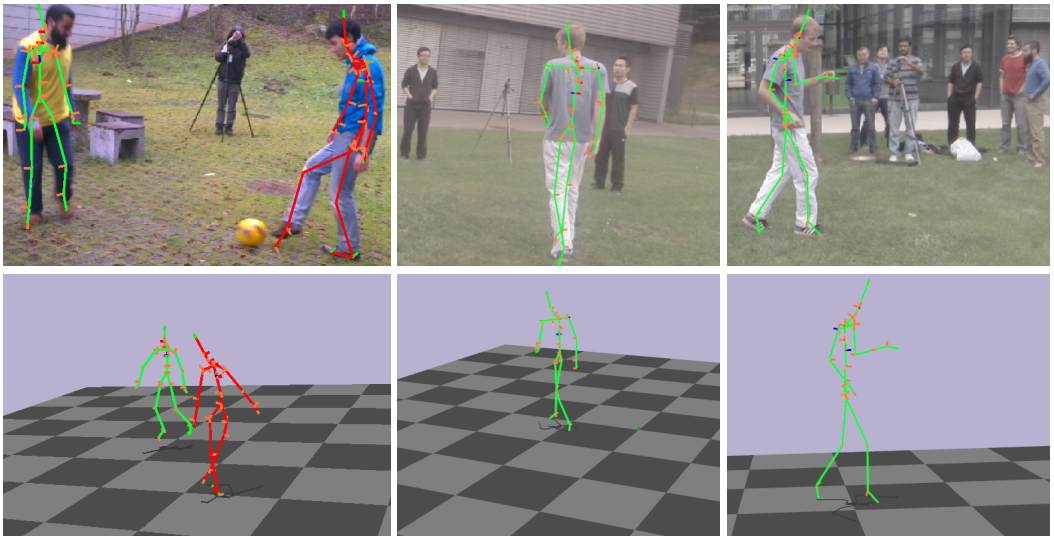
Figure 6.5: Qualitative analysis of tracking results. The tracking results are displayed as skeletons overlaid over two of the input images. (a) *Walk2*: The second row shows the accurately tracked skeleton in a camera view that is not used by the algorithm. (b) and (c) *Walk1*: The first row shows two frames from a static camera which captures the motion of another camera. The second row is the view of the moving camera. The tracked location of the moving camera (white rectangle) is overlaid on the static camera view. The green and red lines depict $x$ and $y$ axes of the camera orientation in the image plane. The moving camera location in the static view is highlighted with the red circles; see also the results video in [Elhayek *et al.* (2014b)].

to the similarity term can mislead tracking (bottom). This is avoided by actively detecting the occlusion and excluding the corresponding camera from similarity computation (top).

Finally, our first-order smoothness prior ($E_{Smooth}$; Eq. 6.4) prevents the camera or pose parameters from drifting quickly to implausible values, as observed in the second row of Fig. 6.9d (see Section 6.2.3).

### 6.4.2 Quantitative Evaluation

As it is difficult to obtain ground-truth values for real actor motion in an outdoor scenario, we performed a quantitative analysis of our tracking algorithm on synthetic



(a) (b) (c)

Figure 6.6: Examples of tracking. The tracking results are displayed as skeletons overlaid over the input images in the first row. (a) *Soccer2*, (b) *Walk4*, and (c) *Run*: The second row shows the tracked skeletons in views that do not correspond to any camera views

and studio data that were jointly recorded with a multi-view video and marker-based motion capture system.

**Synthetic data.** We rendered four sequences containing a single actor with several combinations of static and moving cameras. The synthetic datasets represent perfect conditions for our algorithm, i.e. they are free from noise and foreground and background are clearly separated. This allows us to exactly evaluate the accuracy of the optimization (Fig. 6.10). The motions of each camera are generated by combining different translations and rotations around the capture volume.

Visual inspection shows that our algorithm manages to correctly track the skeletal and camera poses in all synthetic sequences. As expected, numerical evaluation indicates that a higher number of moving cameras (from a fixed number of total cameras) increases the error, since the optimization becomes more difficult (Table 6.2). In this particular setup, at least 2 static cameras fix the global position of the actor accurately. Therefore, decreasing the number of static cameras from 5 to 3 does not affect the skeletal joint position accuracy on an absolute scale, however, it decreases the moving camera tracking accuracy. In general, one static camera is not sufficient to localize the absolute coordinates of the actor and the cameras completely. This leads to an unknown global transformation between our and the

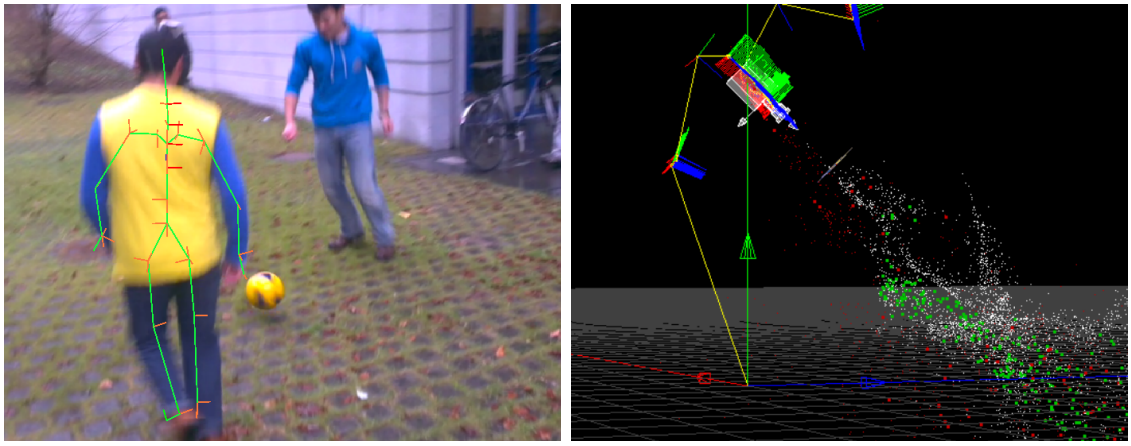


Figure 6.7: *Left:* Pose tracking on the *Soccer1* sequence viewed from a moving camera. *Right:* Tracking of the same camera using SfM. The estimated trajectory of the camera is displayed as a yellow line which is far from being smooth indicating the failure.

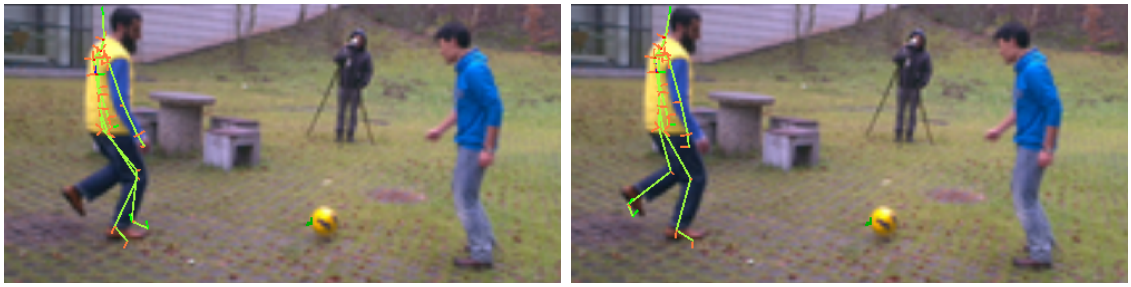


Figure 6.8: *Left:* Tracking *Soccer1* fails with only 3 static cameras. *Right:* With 2 additional moving cameras it succeeds.

ground truth coordinate frames which makes the absolute 3D coordinate error not meaningful. Thus, we use 2D joint position error.

### 6.4.3 Marker-based Quantitative Evaluation

An additional important contribution is a set of validations on sequences that were recorded inside a studio with both a multi-view video system and a frame-synchronized Phase-space marker-based motion capture system. The Phasespace system uses 2 active LED markers attached to the body of the performing actor in the center of the studio. The multi-view video system features cameras of $2048 \times 2048$ pixel resolution runnig at 25 fps. All images are effectively downsampled to a reso-

Table 6.2: Performance of the proposed algorithm for a synthetic scene with varying number of moving and static cameras. The skeletal pose error is measured on average over 65 predetermined skeletal joint position and over the entire frame range in the sequence. The 2D joint position error is measured in a 2D plan of a cameras which is not included in the optimization.

| # Moving cams. | 1 | 1 | 2 | | 3 | | |
|---|---|---|---|---|---|---|---|
| # Static cams. | 5 | 3 | 2 | | 1 | | |
| Average camera position error (cm) | 12.44 | 12.96 | 16.43 | 24.17 | 36.98 | 59.43 | 60.56 |
| Average camera view angle error (degree) | 2.88 | 3.09 | 2.8 | 2.62 | 5.31 | 5.89 | 11.37 |
| Average skeletal 2D joint position error (pixel) | 0.5636 | 0.5430 | 0.6532 | | 4.4346 | | |

lution of $256 \times 256$ before tracking. The tests in this section are performed using a discrete pose optimization algorithm that estimates a discrete set of pose parameters per time step, rather than our space-time optimizer. For a frame-synchronized video system, this yields better results. Further, this is the only way in which we can compare against the baseline method of [Stoll *et al.* (2011)], which also uses this discrete optimization strategy. In the sequences recorded with this setup, the person wears normal street clothing, and markers are attached on top. The specifics of each sequence in the set are explained in the following paragraphs that evaluate several facets of our new algorithm.

First, we want to demonstrate that several of our extensions of the pose fitting energy compared to [Stoll *et al.* (2011)] also lead to improved tracking accuracy over that baseline method when recording with static cameras only. The first sequence was recorded with 8 static video cameras and the marker system in studio lighting, is 150 frames long and shows the actor performing a walking motion. We consider the marker positions measured with the PhaseSpace system as a ground truth for evaluating the tracking accuracy. To this end, we need to identify the positions of the markers w.r.t the skeletal model tracked by our algorithm. We do this by describing each marker with an offset vector in a local frame of the nearest bone. Each such offset between a marker and one skeletal joint is estimated based on observing the offset vector between the marker and the joint on a set of correctly tracked frames, and keeping the average displacement. First, adding only our weighting in HSV color

(a) (b) (c) (d)

Figure 6.9: Importance of algorithmic components (*Soccer2*). The results of our algorithm (top) and alternatives constructed by replacing or removing a certain component, respectively: (a) the two-sided color similarity (6.3) is replaced by a one-sided similarity [Elhayek *et al.* (2012a); Stoll *et al.* (2011)], (b) the weighting in HSV color scheme is disabled (i.e., $W = I$ in Eq. 6.3), (c) the occlusion handling is disabled, (d) the smoothness term in the prior (6.4) is removed, see text for details.

space to the algorithm of [Stoll *et al.* (2011)] already decreases the average marker position error from 4.0 cm to 1.9 cm over the baseline method. If, in addition, we add the two-sided color similarity term (which is essential in case of moving cameras) we observe a further reduction in error to 1.4 cm. However, extending the energy from [Stoll *et al.* (2011)] with the two-sided term alone (i.e. without any weighting in HSV color space), may still lead to errors in bad lighting conditions (e.g. part of the actor is in shadow), because it penalizes dissimilar colors. An adaptive color model would be needed for that which we investigate in future work. This shows that several of our algorithmic extensions to the baseline fitting energy also benefit the case of static camera tracking and lead to a notable reduction in tracking error; see Fig. 6.11 and the results video in [Elhayek *et al.* (2014b)].

We now further show that even in studio conditions, the static algorithm [Elhayek *et al.* (2012a); Stoll *et al.* (2011)] fails with moving cameras. To confirm this fact and to evaluate the camera tracking accuracy of our algorithm, we recorded a second in studio sequence with 3 moving and 5 static cameras. The sequence is 500 frames long. Our reference for accuracy comparison are the motion capture markers on the body. Our SfM based tracking of the moving cameras may contain errors, and thus yield reprojection errors in the moving cameras. Therefore, the 2D positions

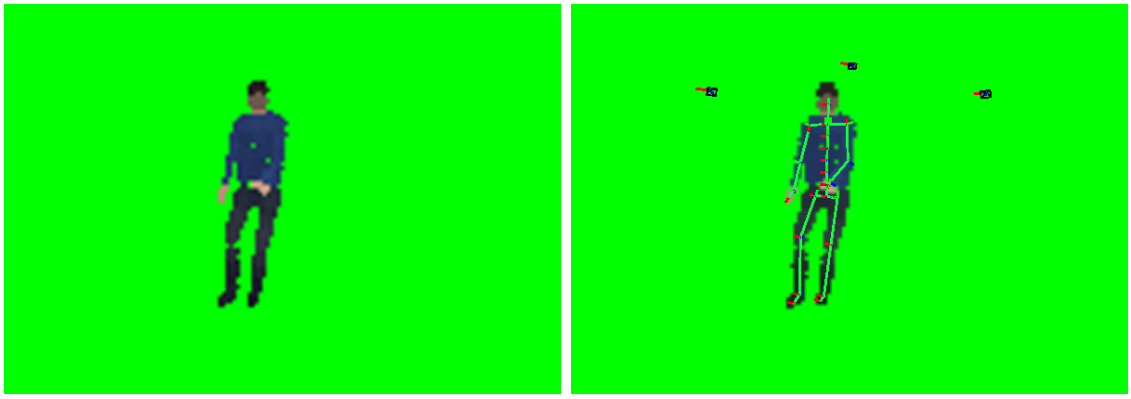


Figure 6.10: *Left:* an example frame from the synthetic sequence. *Right:* tracking result with estimated locations and orientations of three moving cameras overlaid on the frame.

of two markers in one moving camera view were annotated manually in a range of 100 frames as ground truth. We use the 2D distance in the image planes of that camera between the respective body markers tracked by our algorithm and the ground truth to assess accuracy. The average error of [Stoll *et al.* (2011)] is 25.9 pixels which reflects its failure to track this sequence. In contrast, our algorithm achieves an average of 1.8 pixels as it tracked that sequence much more reliably; see Fig. 6.12 and results video in [Elhayek *et al.* (2014b)].

We further tracked the three moving cameras using a SfM algorithm [Thormählen *et al.* (2008)] and landmarks in the studio background. It failed to track two of the three cameras because their motion consist of only rotation and small translation. This further shows the power of our algorithm which works with any type of motion in the cameras, but also means that we cannot quantitatively compare the tracking accuracy of these two cameras obtained with our algorithm against ground truth. We used the correct SfM tracking of the third camera as a ground truth to evaluate our camera tracking accuracy. The average camera position error is 16.4 (cm) and the average difference in angle in viewing direction between ground truth and our tracked solution is 13.4 degrees. This also shows quantitatively that the camera tracking is of good quality.

## 6.5 Discussion

This Chapter presents an algorithm for marker-less human motion capture with moving and unsynchronized cameras that requires only minimal user interaction.

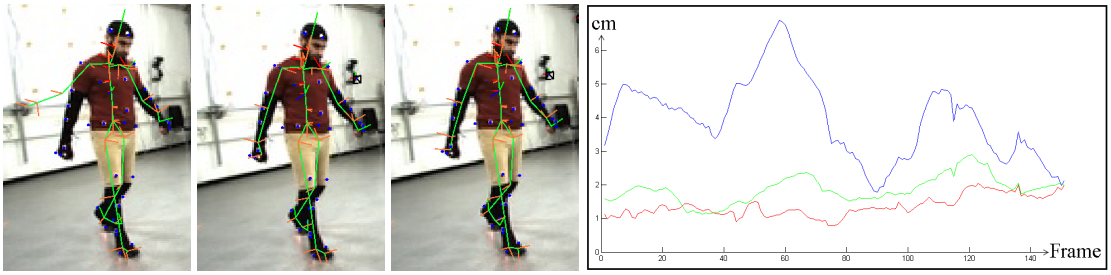



Figure 6.11: Quantitative evaluation of algorithmic components (*Walk5*). Tracking result of (a) [Stoll *et al.* (2011)]; average error 4.0 (cm). The blue dots correspond to the markers positions, (b) our weighting in HSV color scheme with [Stoll *et al.* (2011)]; average error 1.9 (cm), (c) both weighting in HSV scheme and two-sided similarity with [Stoll *et al.* (2011)]; average error 1.4 (cm), (d) the plot of the markers positions error per frame where the blue, green and red correspond to (a), (b) and (c); respectively.

Unlike existing approaches for skeletal tracking with moving cameras, our algorithm does not require any additional hardware and succeeds on even highly dynamic and cluttered scenes and for a more general range of camera motion where feature-based camera calibration fails. Furthermore, our algorithm enables accurate full-body outdoor motion tracking of one or several actors who perform non-trivial motion. This is made possible by a new energy functional that simultaneously models camera and skeletal pose parameters in a space-temporally consistent way based on the appearance of tracked actors. We demonstrated the starkly improved performance and application range of our algorithm relative to a baseline method it originated from both quantitatively and qualitatively in an extensive set of experiments. In this context we further contribute with one of the first evaluation datasets for video-based pose tracking with moving cameras that features ground truth marker-based pose data, as well as ground truth motion data of non-stationary cameras. We believe that our technique is a step towards bridging the gap between complex and expensive capture studios and unconstrained outdoor motion capture, such as on-set tracking, which is essential in many computer graphics and computer vision applications.

As the estimation of the camera motion parameters is based only on a small sample of the 3D space (i.e., the position and pose of the actor), resulting camera paths can be less accurate then with SfM approaches. The uncertainty is large in the camera's viewing direction (and becomes more pronounced with large focal

Figure 6.12: Comparison with [Stoll *et al.* (2011)] on a moving and static cameras sequence (*Walk 6*). Four sample frames of the tracking result from one moving camera view. *Top:* [Stoll *et al.* (2011)]; average error 25.9 (pixel). *Bottom:* Our algorithm; average error 1.8 (pixel).

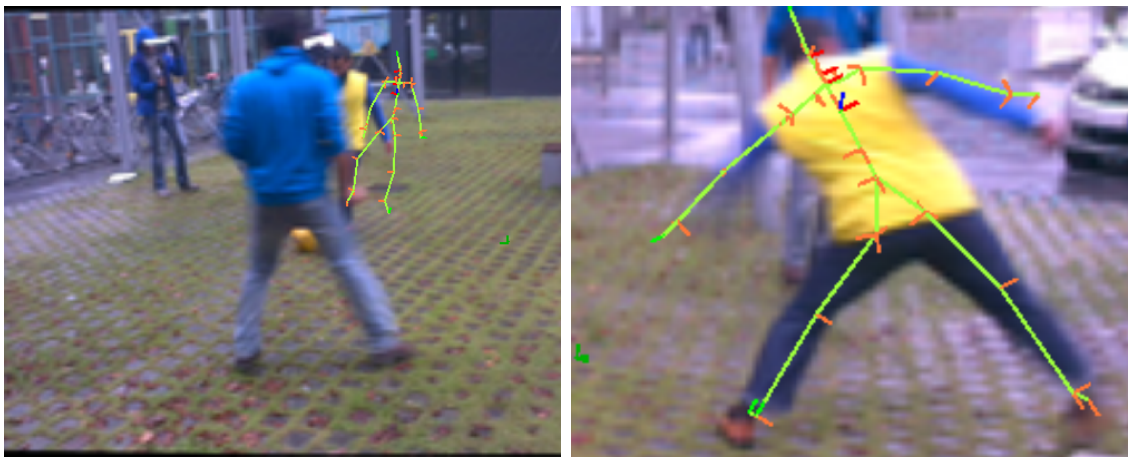


Figure 6.13: Failure cases. *Left:* Moving camera does not recover after a long occlusion. *Right:* Inaccurate arms tracking because of the motion blur.

lengths), as small changes in the distance of the camera to the performer have only little influence on the appearance of the model. However, our quantitative evaluation shows that the obtained accuracy is still good under these more challenging conditions. Also, our method successfully tracks both camera and human motion in scenes where traditional SfM methods would fail as demonstrated in the results video in [Elhayek *et al.* (2014b)] and the quantitative experiments reported earlier. For some scenes, we could include image features as additional evidences into our energy function to increase the stability of the tracking.

Our algorithm requires the tuning of four hyper-parameters: $\lambda_1$ and $\lambda_2$ for controlling the contribution of the prior on the final energy and $T_o$ and $T_n$ for occlusion detection. We chose their values through experiments and kept them fixed for all results.

Although our algorithm produced reasonable tracking results even in challenging environments, failure cases remain. Figure 6.13 exemplifies specific directions where future improvement is desired: Our occlusion handling strategy relies on the linear extrapolation of camera motions during the occlusion. This may fail when the camera motion is highly nonlinear, which is likely for long occlusions as shown in Fig. 6.13 left. For this case, a more expensive global optimization could be exercised for recovering from the occlusion. Figure 6.13 right shows an example of tracking failure (in the left arm) due to strong motion blur.

In the future, synergies between motion deblurring and tracking shall be ex-

plored. Occlusion of body parts in many camera views can lead to tracking errors. Solutions to this problem deserve further investigation. Our occlusion detection scheme for cameras can also be used in detecting skeletal pose tracking failures (Section 6.3): When there is more than one camera undergoing occlusion, this indicates a likely skeletal pose error, and a global pose optimization, such as particle filtering, could be initiated to recover from it.

In practice, the tracking accuracy of the algorithm proposed in this chapter decreases with input filmed using less than five cameras. Therefore, in Chapter 7, we propose a novel marker-less motion capture algorithm which achieves high tacking accuracy from input filmed with as few as two cameras.

# Chapter 7

# Motion Capture with a Low Number of Cameras

In Chapter 5, we have demonstrated that marker-less skeletal motion tracking is also feasible in a less controlled studio setting; i.e. with unsynchronized cameras. Moreover, we have presented in Chapter 6 a method for capturing the skeletal motions of humans using a sparse set of potentially moving cameras in an uncontrolled outdoor environment in front of more general backgrounds where foreground segmentation is hard. Commonly these methods rely on a kinematic skeleton model with attached shape proxies, and they track the motion by optimizing an alignment metric between model and images in terms of the joint angles. Formulating and optimizing this usually highly non-convex energy is difficult. Global optimization of the pose is computationally expensive, and thus local methods are often used for efficiency, at the price of risking convergence to a wrong pose. With a sufficiently high number of cameras ($\geq 8$), however, efficient high accuracy marker-less tracking is feasible with local pose optimizers. Unfortunately, this strategy starts to break badly if only $2-3$ cameras are available, even when recording simple scenes inside a studio.

In a separate strand of work, researchers developed learning-based discriminative methods for body part detection in a single image. Since part detection alone is often unreliable, it is often combined with higher-level graphical models, such as pictorial structures [Andriluka *et al.* (2009)], to improve robustness of 2D part or joint localization. Recently, these 2D pose estimation methods were extended to the multi-view case, yielding 3D joint positions from a set of images taken at the same time step [Belagiannis *et al.* (2014)]. Detection-based pose estimation can compute joint locations from a low number of images taken under very general conditions.

However, accuracy of estimated joint locations is comparably low, mainly due to the uncertainty in the part detections, and pose computation is far from real-time. Also, the approaches merely deliver joint positions, not articulated joint angles, and results on video exhibit notable jitter.

This chapter describes a new method to fuse marker-less skeletal motion tracking with body part detections from a convolutional network (ConvNet) for efficient and accurate marker-less capture of articulated skeleton motion of several subjects in general scenes, indoors and outdoors, even from input filmed with as few as two cameras. Through fusion, the individual strengths of either strategy are fruitfully enforced and individual weaknesses compensated. The core contribution is a new way to combine evidences from a discriminative ConvNet-based monocular joint detector [Tompson *et al.* (2014a)] with a model-based articulated pose estimation framework [Stoll *et al.* (2011)]. This is done by a new weighted sampling from a pose posterior distribution, guided by the articulated skeleton model that employs part detection likelihoods. This yields likely joint positions in the image with reduced positional uncertainty, which are used as additional constraints in a pose optimization energy. The result is one of the first algorithms to capture temporally stable, fully articulated joint angles from as little as 2-3 cameras, also of multiple actors in front of moving backgrounds.

We tested our algorithm on challenging indoor and outdoor sequences filmed with different video and mobile phone cameras, on which model-based tracking alone fails. The high accuracy of our method is shown through quantitative comparison against marker-based motion capture, marker-less tracking with many cameras, and detection-based 3D pose estimation methods. Our approach can also be applied in settings where other approaches for pose estimation with a low number of sensors, that are based on depth cameras [Baak *et al.* (2011)] or inertial sensors [Pons-Moll *et al.* (2011)], are hard or impossible to utilize, e.g. outdoors. The accuracy and stability of our method is achieved by carefully and cleverly combining all input information (i.e. 2D detections, the pose of the previous frame, several views, the 3D-model, and camera calibration). For instance, our method provides 1) strategies to select the correct scale of the ConvNet; 2) strategies to avoid tracking failure by weighting the final contribution of each estimate and by limiting the search space; 3) a new term which carefully integrates the body part detections from all cameras. The work presented in this chapter was published in [Elhayek *et al.* (2015a)].

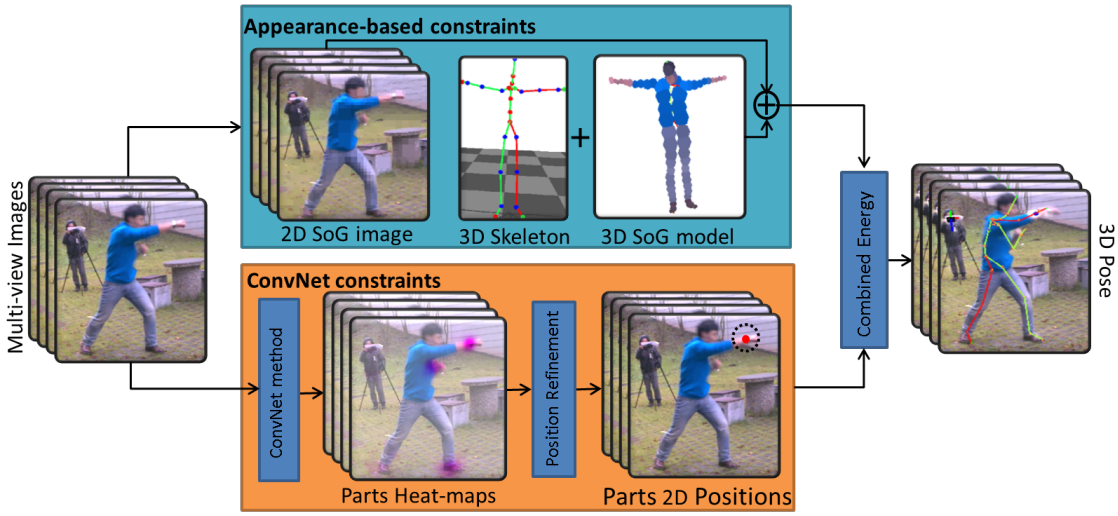



Figure 7.1: Overview of our approach: Our pose optimization scheme combines constraints from an appearance-based 3D model-to-image similarity with ConvNet joint detection constraints to compute articulated joint angles at each time frame.

## 7.1 Method Overview

The input to the approach proposed in this chapter are multi-view video sequences of a scene, yielding $n$ frames $I = I_1^c, ..., I_n^c$ for each static and calibrated camera $c \in C$. Cameras can be of varying types and resolution, but run synchronized at the same frame rate.

We model each human in the scene with an articulated skeleton, comprising of 24 bones and 25 joints. Joint angles and global pose are parameterized through 48 pose parameters $\Theta$, represented as twists. Later, for 13 of the joints - mostly in the extremities - ConvNet detection constraints are computed as part of our fused tracker. In addition, 72 isotropic Gaussian functions are attached to the bones, with each Gaussian's position in space (mean) being controlled by the nearest bone. Each Gaussian is assigned a color, too. This yields an approximate 3D Sum of Gaussians (SoG) representation of an actor's shape; refer to Section 3.2.1 for more details. In parallel, each input image is subdivided into regions of constant color using fast quad-tree clustering, and to each region a 2D Gaussian is fitted; see Section 3.2.2 for more details. Before tracking commences, the bone lengths and the Gaussians need to be initialized to match each tracked actor. Depending on the type of sequence (recorded by us or not), we employ an automatic initialization scheme by optimizing

bone lengths, as described in Section 3.2.1. If initialization poses were not captured, the model is manually initialized on the first frame of multi-view video.

The baseline generative model-based marker-less motion capture approach by Stoll *et al.* (2011) and our extensions in this thesis use the aforementioned scene representation and estimate pose by optimizing a color- and shape-based model-to-image similarity energy in $\Theta$. This smooth and analytically differentiable energy can be optimized efficiently, which results in full articulated joint angles at state-of-the-art accuracy if enough cameras (typically $\geq 8$) are available, and if the scene is reasonably controlled i.e. well-lit and with little lighting variations. These methods quickly fail, however, if the number of cameras falls below five, and if - in addition - scenes are recorded outdoors, with stronger appearance changes, with multiple people in the scene, and with more dynamics and cluttered scene backgrounds.

To make this model-based tracking strategy scale to the latter more challenging conditions, we propose in this chapter a new way to incorporate into the pose optimization additional evidence from a machine learning approach for joint localization in images based on ConvNets. ConvNet-based joint detection [Tompson *et al.* (2014a)] shows state-of-the-art accuracy for locating joints in single images, even in challenging and cluttered outdoor scenes. However, computed joint likelihood heat-maps are rather coarse, with notable uncertainty, and many false positive detections; see Fig. 7.2. Extracting reliable joint position constraints for pose optimization directly from these detections is difficult.

To handle these uncertainties, we propose a model-guided probabilistic approach that extracts the most likely joint locations in the multi-view images from the uncertain ConvNet detections. To this end, the pose posterior for the next frame is approximated by importance sampling with weights from the detection likelihood in the images. Here, the pose prior is modeled reliably based on articulated motion extrapolation from the previous time step's final pose estimate. From the sampled posterior, the most likely image location for each joint is computed, which is then incorporated as constraint into the pose optimization energy, see Section 7.3. In conjunction, this yields a new pose energy to be optimized for each time frame of multi-view video.

$$E(\Theta) = w_{col}E_{col}(\Theta) + w_{BP}E_{BP}(\Theta) - w_l E_{lim}(\Theta) - w_a E_{acc}(\Theta) \tag{7.1}$$

where $E_{col}(\Theta)$ is a color- and shape-based similarity term between the projected body model and the images (Section 7.2), $E_{BP}(\Theta)$ denotes the ConvNet detection

term (Section 7.3), and $w_{col}$ and $w_{BP}$ control their weights. $E_{lim}(\Theta)$ enforces joint limits, and $E_{acc}(\Theta)$ is a smoothness term penalizing too strong accelerations [Stoll *et al.* (2011)]. The weights $w_{col} = 1$, $w_{BP} = 5$, $w_l = 0.1$ and $w_a = 0.05$ were found experimentally and are kept constant in all experiments.

This new energy remains to be smooth and analytically differentiable, and can thus be optimized efficiently using standard gradient ascent method, initialized with the previous time step's extrapolated pose. ConvNet detections can be computed faster too. By optimizing this new energy, we can track fully articulated joint angles at state-of-the-art accuracy on challenging scenes with as few as two cameras. The outline of the processing pipeline of the approach proposed in this chapter is illustrated in Fig. 7.1.

## 7.2 Appearance-based Similarity Term

The appearance-based similarity term $E_{col}$ measures the overlap between a 3D model and the 2D SoG images for the images of each camera $c$ as defined in (Eq. 3.7). Therefore, the final appearance similarity term computed over all cameras reads as follows:

$$E_{col}(\Theta) = \sum_{c \in C} \sum_{j \in \mathscr{K}_{I^c}} \min \left( \left( \sum_{i \in \Psi(\mathscr{K}_m(\Theta, c))} E_{ij} \right), E_{ii} \right), \tag{7.2}$$

where $E_{ij}$ is the similarity between a pair of Gaussians $\mathscr{B}_i$ and $\mathscr{B}_j$ given their colors as defined in (Eq. 3.4). We use the same occlusion approximation of [Stoll *et al.* (2011)]. This prevents projected 3D Gaussians from contributing multiple times in (Eq. 3.2); see Section 3.2 for details.

## 7.3 ConvNet Detection Term

We employ a ConvNet-based localization approach [Tompson *et al.* (2014a)] to compute for each of the $n_{prt} = 13$ joints $j$ in the arms, legs and head a Heat-map image $H_{j,c}$ for each camera view $c$ at the current time step. This approach achieves state of the art results on several public benchmarks and is formulated as a Convolutional Network [LeCun *et al.* (1998a)] to infer the location of the 13 joints in monocular RGB images; please see Section 3.3 for a summary of this approach. A common problem in discriminative body part detection methods is the separation between

Figure 7.2: Refinement of the Body Part Detections using the pose posterior. **Left:** Overlay of the heat-map for the right ankle joint over the input image. **Middle:** sampling from pose posterior around the rough 2D position $p_{j,c}^{init}$ (black dots). **Right:** The final refined location of the body part $d_{j,c}$ (blue dot).

front-back and left-right of the body anatomy because of the different camera positions. This problem becomes more complicated in case of 3D pose estimation of multiple humans, given similar body parts of different humans in each view. A major advantage of the ConvNet detections for 3D human pose estimation is that they do not suffer from this front/back ambiguity. The detector has been trained to differentiate left and right limb joints effectively. We attribute this to their high discriminative capacity, efficient use of shared (high-level) convolutional features, learned invariance to input image transformations, and large input image context. We do not explicitly train the ConvNet on frames used in our work, but use a net pre-trained on the MPII Human Pose Dataset [Andriluka *et al.* (2014)], which consists of 28,821 training annotations of people in a wide variety of poses and static scenes. Note that training on our own sequences (or sequences similar to ours) may increase accuracy even further.

We employ a weighted sampling from a pose posterior guided by the kinematic model to extract the most likely 2D joint locations $d_{j,c}$ in each image from the uncertain likelihood maps (Section 7.3.1). These are used as additional constraints in the pose optimization energy (Section 7.3.2).

### 7.3.1 Refining Joint Detections

The joint detection likelihoods in $H_{j,c}^{s}$ exhibit notable positional uncertainty, false positives, and close-by multiple detections in multi-person scenes, Fig. 7.3 (Left).

We therefore propose a new scheme to extract the most likely location $d_{j,c}$ of each joint in each camera view (and for each tracked person if multiple people are in the scene), given the history of tracked articulated poses. Our approach is motivated by weighted sampling from the 3D pose posterior distribution $P(D|\Theta)$.

$$P(\Theta|D) \propto P(D|\Theta)P(\Theta) \; . \tag{7.3}$$

Here, $D$ is short for the image evidence. The likelihood $P(D|\Theta)$ is represented by the ConvNet responses in the image plane. The pose prior $P(\Theta)$ is modeled by building a Gaussian pose distribution with a mean centred around the pose $\Theta_0^t$ predicted from the previous time steps, as follows:

$$\Theta_0^t = \Theta^{t-1} + \alpha(\Theta^{t-1} - \Theta^{t-2}) \; . \tag{7.4}$$

where $\alpha = 0.5$ for all sequences. In practice, we compute for each joint the most likely location $d_{j,c}$ by weighted sampling from the posterior. Instead of working on all joints and images simultaneously, we simplify the process, assuming statistical independence, and thus reduce the number of samples needed by performing the computation for each image and joint separately. First, an extrapolated projected mean 2D location of $j$ in $c$ is computed $p_{j,c}^{init}$ by projecting to joint location in pose $\Theta_0^t$ into the image. Then we sample $N = 250$ 2D pixel locations $p$ from a 2D-Gaussian distribution with mean $\mu = p_{j,c}^{init}$ and $\sigma = 20$ pixel. This can be considered a per-joint approximation of the posterior $P(\Theta)$ from Eq (7.3), projected into the image. Fig. 7.2 illustrates this process.

For each sample $p$ we compute a weight $w(p)$

$$w(p) = \begin{cases} H_{j,c}^q(p) & H_{j,c}^q(p) > H_{th} \\ 0 & H_{j,c}^q(p) \leq H_{th} \end{cases} \tag{7.5}$$

where we set $H_{th} = 0.25$. The final assumed position of the joint $d_{j,c}$ is calculated as the average location of the weighted pose posterior samples

$$d_{j,c} = \sum_{i=1}^{N} p_i * w(p_i). \tag{7.6}$$

The latter step can be considered as finding the mode of the weighted samples drawn from the posterior $P(\Theta|D)$ using the ConvNet responses as likelihood. As a result, $d_{j,c}$ is an accurate estimate of the actual 2D position of the body part. Note that the size of the person in the image may vary significantly over time and across camera views. To cope with this, the scale $q$ of the heat-map at which detections are computed best is automatically selected for each camera, joint, and time step as part of the computation of $d_{j,c}$. Specifically, $q$ is the scale $s$ at which in a $50 \times 50$ pixel neighborhood around $p_{j,c}^{init}$ the highest detection likelihood was found.

In case more than one body part of the same class (e.g. left wrist) are close to each other in one of the views, for instance if there are multiple actors in the scene (see Fig 7.3(Right)), the value $d_{j,c}$ can be wrongly found as the middle between the two detections. Since the heat-map value at $d_{j,c}$ is comparably low in the middle between two parts, such erroneous detections (e.g. with two nearby people in one view) can also be filtered out by the above weighting with a minimum threshold.

### 7.3.2 Detection Term

Our ConvNet joint detection term measures the similarity between a given pose $\Theta$ of our body model and the refined 2D body part locations. Since the body model pose lies in the 3D space and the refined 2D body part locations are detected on the image, we first need to project the 3D joint positions defined by $\Theta$ into the respective camera image plane using the projection operator $\Psi_c$ of camera $c$. We incorporate the detected joint locations $d_{j,c}$ into the SoG model-based pose optimization framework by adding the following term to (Eq. 7.1):

$$E_{BP}(\Theta) = \sum_{c \in C} \sum_{j=1}^{n_{prt}} w(d_{j,c}) \exp\left( -\frac{\|\mathbf{\Psi_i}(\mathbf{l_j}(\mathbf{\Theta})) - \mathbf{d_{j,c}}\|^2}{\sigma^2} \right). \tag{7.7}$$

Here, $w(d_{j,c})$ is a weight for a constraint computed as the detection likelihood of the most likely image location $d_{j,c}$; i.e. $w(d_{j,c})$ is the heat-map value at $d_{j,c}$. $\mathbf{l}_j(\Theta)$ is the 3D joint position of $j$ if the model strikes pose $\Theta$.

## 7.4 Experiments and Results

We evaluated our algorithm on six real world sequences, which we recorded in an uncontrolled outdoor scenario with varying complexity. The sequences vary in the numbers and identities of actors to track, the existence and number of moving objects

Figure 7.3: **Left:** Joint detection likelihoods for the right ankle in the heat-maps $H_{j,c}$ exhibit notable positional uncertainty, and there are many false positives and close-by multiple detections. **Right:** Even though two body parts for the same class (i.e. lower wrist) are close to each other in the images, our approach is able to correctly estimate their individual locations.

in the background, and the lighting conditions (i.e. some body parts lit and some in shadow). Cameras differ in the types (from cell phones to vision cameras), the frame resolutions, and the frame rates. By quad-tree decomposition, all images are effectively downsampled to a small resolution used in the generative energy (i.e. blob frame resolution). For the joint detection computation, the full resolution images are used and four heat-maps, with different scales for the subject, are generated. Please note that all cameras are frame synchronized. In particular, the cell phone cameras and the GoPro cameras are synchronized using the recorded audio up to one frame's accuracy. Moreover, we recorded additional sequences in a studio for marker-based or marker-less quantitative evaluation of skeletal motion tracking. The ground truth of the *Soccer* sequence was computed based on manual annotation of the 2D body part locations in each view. The ground truth of the $Marker$ sequence was acquired with a marker-based motion capture system and the ground truth of $Run1$ was

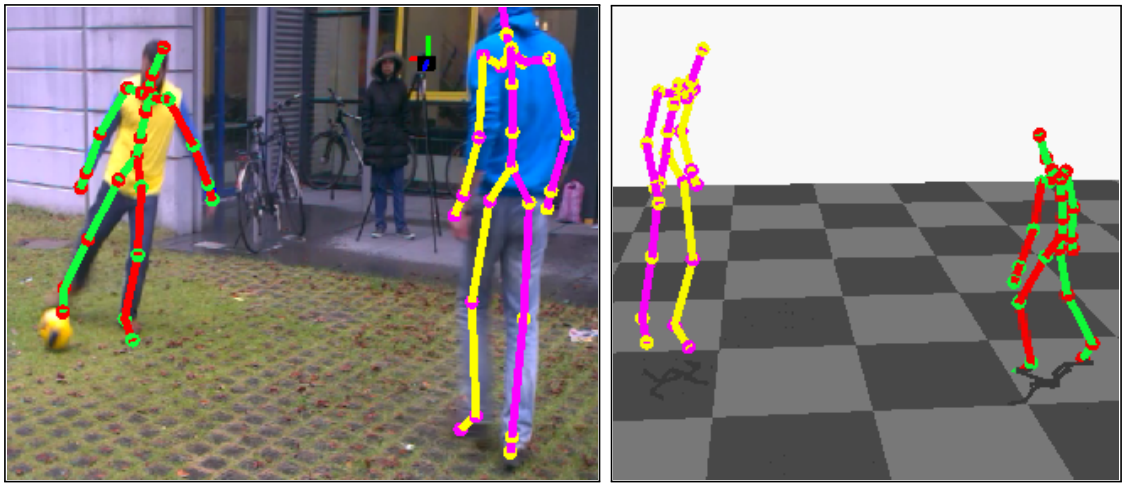


Figure 7.4: Our ConvNet-based marker-less motion capture algorithm reconstructs joint angles of multiple people performing complex motions in outdoor settings, such as in this scene recorded with only three mobile phones: (left) 3D pose overlaid with one camera view, (right) 3D visualization of captured skeletons.

estimated based on marker-less tracking with a dense setup (i.e. 11 cameras) using a variant of [Stoll *et al.* (2011)]. Table 7.1 summarizes the specifications of each sequence. Apart from body model initialization, which requires the user to apply a few strokes to background segment the images of four actor poses (see Section 3.2), tracking is fully-automatic. Further, the run-time of our algorithm depends on the number of cameras and actors, and the complexity of the scene, e.g. the number of Guassians needed in 2D. For a single actor and three cameras (e.g. the Walk sequence from the HumanEva dataset [Sigal *et al.* (2010)]), our algorithm takes around 1.186$s$ for processing a single frame.

**Qualitative Results** Figures 7.4 and 7.5 show example poses tracked from outdoor sequences with our approach. Please see also the video in [Elhayek *et al.* (2015b)] for additional results. Our algorithm successfully estimates the pose parameters of the actors in challenging outdoor sequences with two or three cameras. In particular, our algorithm successfully tracks the two actors in *Soccer* and *Juggling*, who often occlude each other, it tracks the actors in highly cluttered scenes ($Walk2$, $Run2$) - each of which contains many moving people in the background, and it performs well in a sequence with strong lighting variations ($Walk1$). All of these sequences are challenging to previous methods.

Table 7.1: Specification of each sequence evaluated by our approach.

| Sequence | *Soccer* | *Kickbox* | *Marker* | *Run*1 | *Run*2 | *Walk*1 | *Walk*2 | *Juggling* |
|---|---|---|---|---|---|---|---|---|
| Num. of Cams. | 3 | 3 | 2 | 3 | 2 | 3 | 3 | 4 |
| Num. of Frames | 300 | 300 | 500 | 1000 | 600 | 600 | 210 | 300 |
| Frame Rates | 23.8 | 23.8 | 25 | 35 | 30 | 60 | 30 | 30 |
| Camera Types | cell-phone (*HTC One X*) | | PhaseSpace Vision Camera | | GoPro | | | |
| Input Frame Resol. | 1280x720 | | 256x256 | 1296x972 | 1280x720 | | | |
| Blob Frame Resol. | 160x90 | | 256x256 | 160x90 | 240x135 | | | |
| Tracked Subjects | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 2 |
| Moving Backgroud | No | No | No | No | Yes | Yes | Yes | Yes |

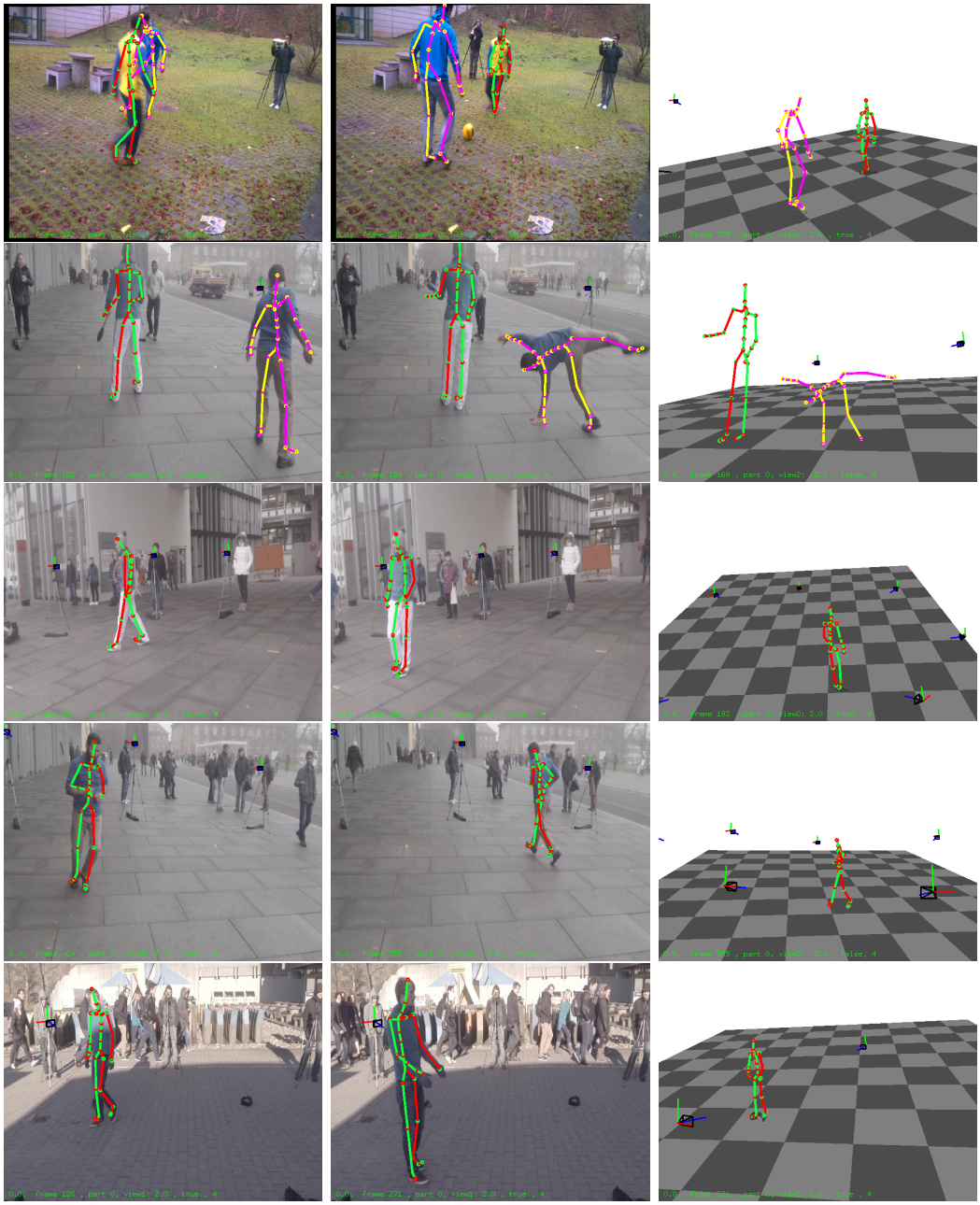


Figure 7.5: Qualitative results: From top to bottom particular frames for the *Soccer*, *Juggling*, *Walk*2, *Run*2 and *Walk*1 sequences recorded with only 2-3 cameras. For each sequence, from left to right, 3D pose overlaid with the input camera views for two frames and 3D visualizations of the captured skeletons.

Table 7.2: Average overlap of the 3D SoG models against an input view (not used for tracking).

| Sequence | **gen** (Gen. term only) [Stoll *et al.* (2011)] | **disc** (Discr. term only) | **gen+disc** (Combined energy) | Max. overlap |
|---|---|---|---|---|
| *Soccer* | 43.58 | 46.83 | **46.84** | 47.62 |
| *Juggling* | 58.48 | 60.72 | **62.87** | * |
| *Marker* | 49.33 | 46.99 | **54.17** | 60.58 |
| *Run*1 | 47.04 | 52.86 | **53.23** | 53.58 |
| *Run*2 | 17.93 | 55.96 | **55.98** | * |
| *Walk*1 | 31.78 | 54.16 | **54.77** | * |
| *Walk*2 | 34.12 | 34.96 | **35.52** | * |
| *Kickbox* | 57.07 | 58.01 | **59.32** | * |

**Quantitative Results** We evaluated the importance of each term of our combined energy function and compared our method against state-of-the-art multi-view and 3D body part detection methods. We evaluated the results of three variations of our approach: **gen** neglecting the ConvNet detection term (i.e. $w_{BP} = 0$ in Eq. 7.1), **disc** neglecting the Appearance-based Similarity term (i.e. $w_{col} = 0$ in Eq. 7.1), and **gen+disc**, our full combined energy (i.e. $w_{BP} = 5$ and $w_{col} = 1$). Please note that **gen** is similar to applying the generative marker-less motion capture method proposed by Stoll *et al.* (2011).

In Table 7.2, we calculated the average overlap of the 3D SoG models against one additional input view not used for tracking for each sequence. This value is calculated using the $E_{col}$ (Eq. 3.6) considering only the additional camera view. A higher number indicates that the reconstructed pose (and model) matches better the input image. As can be seen in Fig. 7.6, even small improvements in the overlap value translate to great improves in the tracking, e.g. hands and feet. The results in the table show that our combined method achieves higher accuracy than applying [Stoll *et al.* (2011)] or only applying the ConvNet detection term. Please note that Max. Overlap is the average overlap of the 3D SoG models, defined by the ground truth model parameters. The method proposed in Stoll *et al.* (2011) is used as ground truth for some sequences. However, it failed even with many cameras for outdoor sequences (marked with * in the table). Fig. 7.6 shows the visual improvements of our solution. As shown in the images, by combining both energy terms, we are able to better reconstruct the positions of the hands and feet.

We also compared the individual components of our approach in terms of the average 3D joint position reconstruction error over time. Table 7.3 summarizes the comparison for the sequences that possess ground truth 3D joint positions (obtained with different methods depending on the sequence). Fig. 7.7(top) shows the plot of the 3D joint position reconstruction error over time for sequence *Marker* for all three variants. Fig. 7.7(bottom) shows visual results for each variant. As seen in the images, our combined approach (**gen+disc**) is able to reconstruct the pose of the subject more accurately. Note that with a small camera setup (only 2-3 cameras), our approach is able to reach a similar level of accuracy achieved by a dense multi-view approach in controllable indoor scenes.

Figure 7.6: Particular frame for the *Juggling* sequence. From left to right, comparison between **gen+disc**, **disc** and **gen**, respectively. The individual strengths of both strategies are fruitfully enforced in our combined energy, which allows more accurate estimation of the positions of hands and feet.

**Comparisons** We evaluated our approach using the Boxing and Walking sequence from the HumanEva benchmark [Sigal *et al.* (2010)] and compared the results against Sigal *et al.* (2012), Amin *et al.* (2013) and Bel (2014). Table 7.4 summarizes the comparison results. As seen in the table, Amin *et al.* (2013) shows very low average error but we also achieve similar results using our hybrid approach, outperforming the other methods. However, it is also important to consider the motion reconstruction quality over time and not only the average 3D joint position error. In the results video in [Elhayek *et al.* (2015b)], it is shown that our method presents a better temporal reconstruction when compared to Amin *et al.* (2013). Our results are more stable, presenting a good temporal coherent reconstruction over time. In constrast, [Amin *et al.* (2013)] shows a considerable amount of jittering and wrong detections (i.e. jumps) over time.

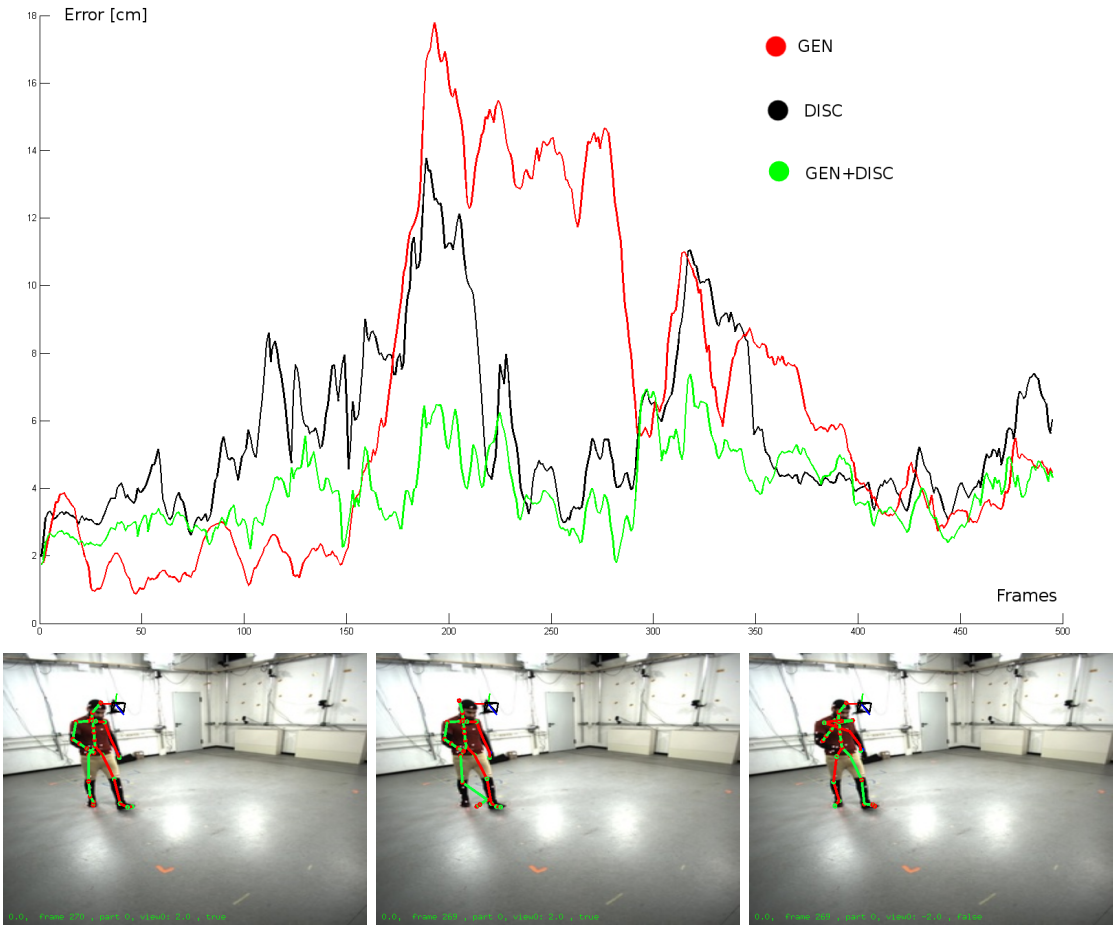



Figure 7.7: (top) Plot showing the average 3D joint position reconstruction error for sequence *Marker* using 2 input cameras only. (bottom) Visual results for variants **gen+disc**, **disc** and **gen**, respectively. Note that the correct reconstruction of the pose (e.g. hands and feet) is only possible with the combined terms in the energy function (**gen+disc**).

Figure 7.8 plots the average 3D joint position error for the Box sequence for both approaches. Note that our approach presents a constant error level, with a lower variance. We believe that part of this error is coming mostly from a different initial joint configuration in our approach, i.e. bone lengths and joint locations. In constrast to Amin *et al.* (2013), we do not train our model on the HumanEva dataset. Figure 7.9 shows our skeleton and the ground truth joint positions overlaid in the input images for all three camera views for the Box and Walk sequences. Note

that for the error calculation we only use the skeleton joints that exist in HumanEva, which is less than the total number of joints our standard skeleton contains. In the figure, the red and green joints are our reconstructed joints, and the blue and pink joints are the ground truth information. Note that although our joint positions match the real underlying human skeleton better, our skeleton configuration (i.e. bone lengths and joint locations) is not the same as in the HumanEva skeleton.

We argue that our increased 3D joint position error value is partly due to a mismatch between the dimensions of our skeleton and the dimensions of the HumanEva skeleton up to a constant offset. Also, please note that the marker positions in HumanEva (on the surface of the actor) are not identical to joint positions (inside the body) which causes an offset anyways. We believe that with this observation and the high temporal stability of our approach, our results are of higher quality.

## 7.5 Discussion

In this chapter, we presented a novel and robust marker-less human motion capture algorithm that tracks articulated joint motion with only 2-3 cameras. By fusing the 2D body part detections, estimated from a ConvNet-based joint detection algorithm, into a generative model-based tracking algorithm, based on the Sums of Gaussians framework, our system is able to deliver high tracking accuracy in challenging outdoor environments with only 2-3 cameras. Our method also works successfully when there is strong background motion (many people moving in the background), when very strong illumination changes are happening or when the human subject performs complex motions. By comparing against sequences recorded in controlled environments or recorded with many cameras, we also demonstrated that our system is able to achieve state-of-the-art accuracy despite a reduced number of cameras.

However, our method is subject to a few limitations. Currently, we can not track sequences with moving cameras. Nevertheless, we believe it is feasible to extend this

Table 7.3: Average 3D joint position error [cm].

| Sequence | *Soccer* | *Marker* | *Run*1 |
|---|---|---|---|
| **gen** (Gen. term only) | 13.93 | 6.39 | 13.50 |
| **disc** (Discr. term only) | **3.79** | 5.69 | 6.11 |
| **gen+disc** (Comb. energy) | 3.95 | **3.92** | **5.84** |

Table 7.4: Average 3D joint position error for the HumanEva Walk and Box sequences.

| Sequence | Walk [cm] | Box [cm] |
|---|---|---|
| Amin *et al.* (2013) | 5.45 | 4.77 |
| Sigal *et al.* (2012) | 8.97 | - |
| Bel (2014) | 6.83 | 6.27 |
| Our approach | 6.65 | 6.00 |

approach to work with moving cameras by combining our outdoor motion capture algorithm in Chapter 6 with our ConvNet detection term (Eq. 7.7). With the current method motion tracking with a single camera view is not feasible because several body-parts are allows occluded in each frames. Also, the frame rate of the camera needs to be adequate to handle the speed of the recorded motion. For example, if fast motions are captured with a lower frame rate, we might not be able to track the sequence accurately, as shown in Fig. 7.10 for the Kickbox sequence, recorded at 23.8fps. However, this is also a common problem with approaches relying on a dense camera setup. Unlike purely generative methods, our approach is still able to recover from tracking errors, even with such fast motion, and it can work correctly with higher frame rate cameras. Our approach works well even for challenging sequences like the juggling, which contains a cartwheel motion. However, for more complex motions, it might be necessary to re-train the ConvNet-based method for improving results.

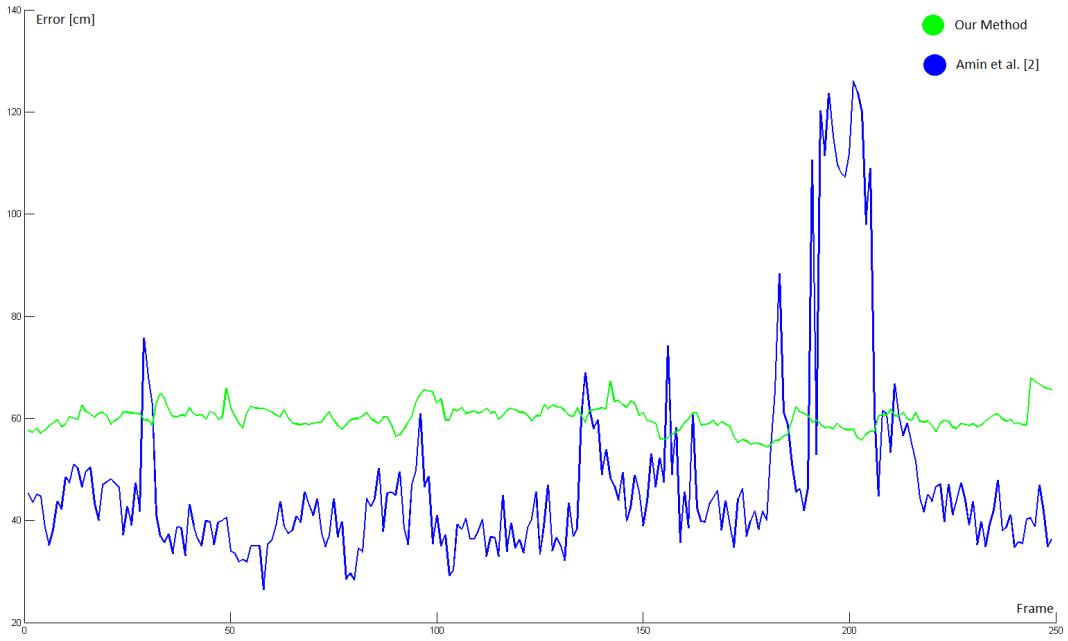



Figure 7.8: Plot showing the average 3D joint position error for the Box sequence from the HumanEva dataset using our approach (green curve) and Amin *et al.* (2013) (blue curve).

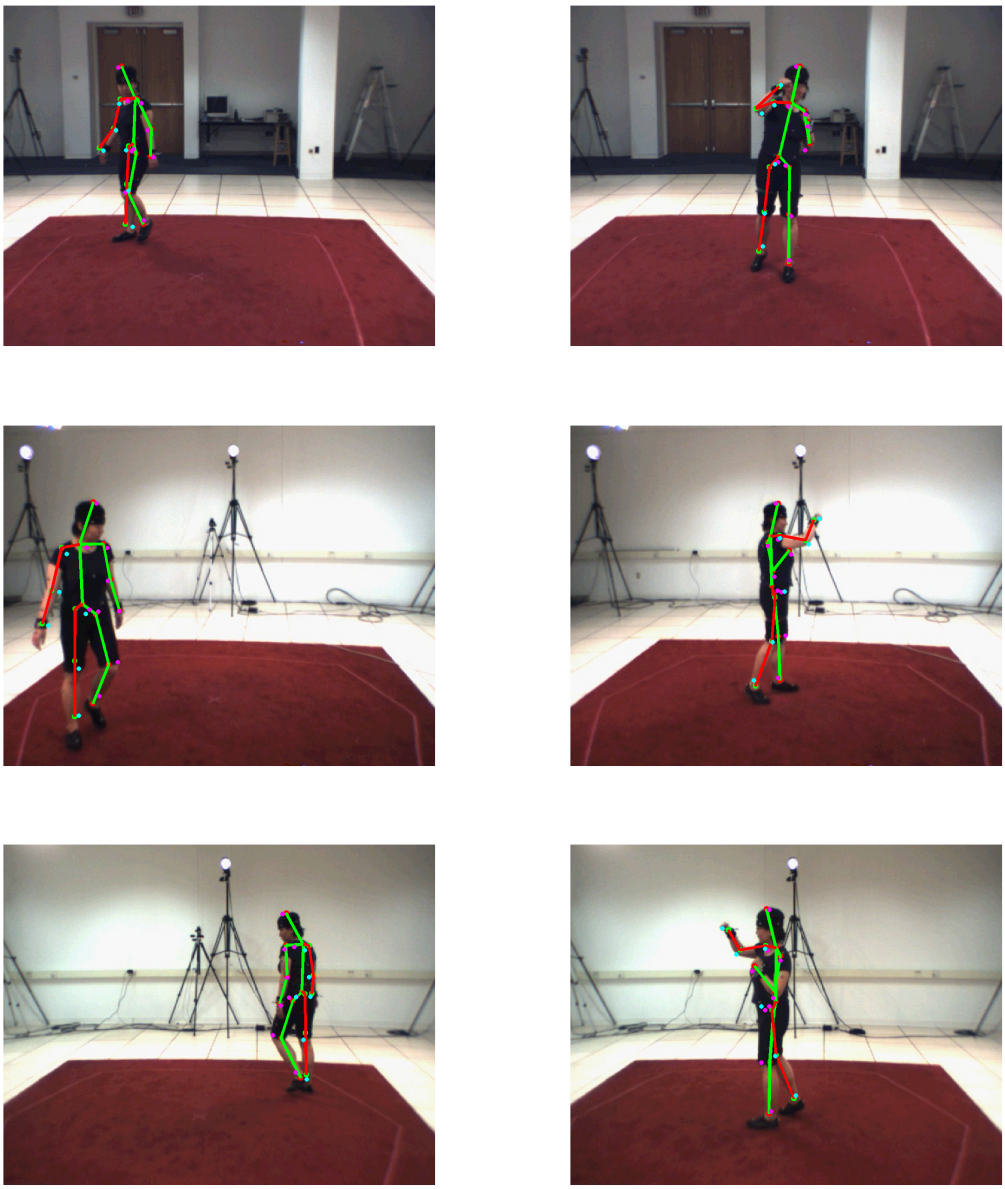


Figure 7.9: Differences between our initial skeleton configuration and the HumanEva skeleton configuration (our ground truth) - Box (left column) and Walk (right column) sequences - can cause an increase in our 3D average joint position error. In the figures, the red and green joints are our reconstructions and the blue and pink joints are the ground truth positions.

Figure 7.10: Fast motions recorded with a lower frame rate (23.8fps) generate blurred images, which makes it hard for our method to correctly track the foot with only 3 cameras.

# Chapter 8

# Conclusions and Future Work

In this thesis, we proposed novel approaches for generalizing the marker-less human motion-capture setup. Our algorithms succeed in general scenes with unsynchronized, moving and sparse multi-camera setups. Simplifying the complex and expensive human motion-capture setup opens up the benefits of human motion-capture to a wide range of industries. To this end, we proposed four algorithms which can be adopted in many practical applications to achieve similar performance as complex motion capture studios with a few cheap consumer-grade cameras (e.g. mobile-phone or GoPro) even in uncontrolled outdoor scenes. We proposed an optical multi-video synchronization method that achieves subframe accuracy in general scenes in Chapter 4. We introduced a spatio-temporal motion-capture method that works with unsynchronized cameras in Chapter 5. The proposed algorithm in Chapter 6 allows motion-capture to be performed with moving cameras, even in front of cluttered and dynamic backgrounds. Finally, our method in Chapter 7 allows to achieve high motion-capture accuracy with a low number of cameras.

As a result of the strong relation between the four methods proposed in this thesis, we consider them as four consecutive steps toward high-quality human motion-capture with few unsynchronized handheld cameras. In particular, the method proposed in Chapter 4 estimates multi-video synchronization parameters while the method in Chapter 5 uses these parameters to achieve optimal motion-capture accuracy with unsynchronized cameras. However, the second method fails with moving cameras, which is resolved by the method proposed in Chapter 6. Finally, the method in Chapter 7 works with a low number of cameras, which is a limitation of the third method.

In Chapter 4 [Elhayek *et al.* (2012c)], we present a novel algorithm for temporally synchronizing multiple videos capturing the same dynamic scene. This algorithm

relies on general image features and it does not require explicitly tracking any specific object, making it applicable to general scenes with complex motion. To enable this, we contributed with a robust trajectory filtering and energy minimization framework based on RANSAC for the multi-camera case. Moreover, we propose a novel strategy for identifying an informative subset of video pairs which further improves the multi-camera synchronization performance and prevents the RANSAC algorithm from being biased by outliers.

In Chapter 5 [Elhayek *et al.* (2012a)], we present a new spatio-temporal method for marker-less motion capture. We reconstruct the pose and motion of a character from a multi-view video sequence without requiring the cameras to be synchronized, and without aligning captured frames in time. Unlike previous approaches that rely on synchronized input video, our method makes use of the additional temporal resolution to successfully track fast-moving actors with low frame rate cameras. It also enables setting up simpler and cheaper capture setups, as there is no need anymore for hardware-based synchronization and high-frame rate cameras. Moreover, by purposefully running cameras unsynchronized, we can even capture very fast motion at speeds that off-the-shelf cameras provide. However, this algorithm works only for static cameras inside a motion-capture studio.

In Chapter 6 [Elhayek *et al.* (2014a)], we present a method for capturing the skeletal motion of humans using a set of potentially moving cameras in an uncontrolled environment. This approach is able to track multiple people even in front of cluttered and dynamic backgrounds using unsynchronized cameras with varying image quality and frame rates where feature-based camera calibration, for example via structure-from-motion (SfM), fails.

In contrast to other outdoor motion-capture methods, this completely relies on optical information, and does not make use of additional sensor information such as depth images or inertial sensors. This method needs only minimal user interaction to accurately track full-body motion of one or several actors who perform non-trivial motion. We demonstrated the starkly improved performance and application range of this algorithm relative to the baseline method it originated from, both quantitatively and qualitatively, in an extensive set of experiments. In this context, we further contribute with one of the first evaluation datasets for video-based pose tracking from moving cameras that features ground-truth marker-based pose data, as well as ground-truth motion data of non-stationary cameras. The main limitation of this algorithm is that its accuracy decreases with input captured using fewer than five cameras.

In Chapter 7 [Elhayek *et al.* (2015a)], we presented a novel and robust marker-less human motion-capture algorithm that tracks articulated joint motion with only 2-3 cameras. By fusing the 2D body part detections, estimated from a ConvNet-based joint-detection algorithm, into a generative model-based tracking algorithm, based on the Sums of Gaussians framework, this algorithm is able to deliver high tracking accuracy in challenging outdoor environments with few cameras. This method also works successfully when there is strong background motion (many people moving in the background), when illumination is changing, or when the human subject performs complex motions. By comparing against sequences recorded in controlled environments or recorded with many cameras, we also demonstrated that this system is able to achieve state-of-the-art accuracy despite the reduced number of cameras.

We believe that these four algorithms are a step towards bridging the gap between complex and expensive capture studios and unconstrained outdoor motion-capture with simple setups, such as on-set tracking, which is essential in many computer graphics and computer vision applications. We feel that this advance will significantly increase the number of future human motion-capture applications in a wide range of industries.

As future work, we would like to investigate the use of a single RGB camera for marker-less human motion-capture. This is is not feasible with our algorithm in Chapter 7 because several body-parts are allows occluded in each frames. In the future, we hope to investigate approaches for extending the connectivity of the ConvNet-based joint detection algorithm in the time domain, to take advantage of local spatio-temporal information.

A common problem shared by our methods and the methods that rely on a dense camera setup is the difficulty in handling motion blur, which makes it hard to correctly track fast motion. Unlike purely generative methods such as the algorithms in Chapter 5 and 6, the ConvNet-based marker-less motion-capture approach in Chapter 7 is still able to recover from the tracking errors, even with fast motion. However, to avoid this problem, the frame rate of the camera needs to be adequate to handle the speed of the recorded motion. For example, if fast motions are captured with a lower frame rate, it might not be possible to track the sequence accurately, even with the approach in Chapter 7. Therefore, an interesting field of future research is to re-train the ConvNet-based method with data which contains motion blur. Moreover, synergies between motion deblurring and tracking shall be explored. The ConvNet-based approach in Chapter 7 works well even for challenging sequences. However, for more complex motions, it might fail. Thus, it may be necessary to re-train the ConvNet with more data which contains complex motions.

Currently, the process of estimating an actor-specific model still requires user input (i.e. manually segmenting multi-view images of example poses and manually initializing the pose parameters to roughly correspond to the initial poses). An interesting field of future research is how to automate this process.

Another area of future work is to find a better solution for the self-occlusion problem which can mislead the similarity function of the SoG tracker. This happens when projecting a 3D model onto a 2D image plane where several Gaussians, that are actually occluded, may be projected onto overlapping 2D positions. By handling the self-occlusions, we mean preventing overlapping projected 3D SoGs from contributing multiple times in the similarity function. So far, we used the simple approximation of the occlusion term proposed by Stoll *et al.* (2011), where we clamp the similarity to be at most the similarity of the image Gaussian with itself. We are convinced that it is possible to develop a better strategy for handling self-occlusions.

# References

(2014). *3D Pictorial Structures for Multiple Human Pose Estimation*. 104, 107

Amin, S., Andriluka, M., Rohrbach, M. & Schiele, B. (2013). Multi-view pictorial structures for 3d human pose estimation. In *BMVC*. 104, 105, 107, 108

Andriluka, M., Roth, S. & Schiele, B. (2009). Pictorial structures revisited: People detection and articulated pose estimation. In *CVPR*. 11, 91

Andriluka, M., Roth, S. & Schiele, B. (2010). Monocular 3d pose estimation and tracking by detection. In *CVPR*. 11

Andriluka, M., Pishchulin, L., Gehler, P. & Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE CVPR*. 28, 96

Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J. & Davis, J. (2005). Scape: shape completion and animation of people. *ACM Trans. on Graphics*, **24**, 408–416. 18

Baak, A., Müller, M., Bharaj, G., Seidel, H.P. & Theobalt, C. (2011). A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Proc. ICCV*, 1092–1099. 11, 14, 92

Balan, A., Sigal, L., Black, M., Davis, J. & Haussecker, H. (2007). Detailed human shape and pose from images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. 51

Ballan, L. & Cortelazzo, G.M. (2008). Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. In *3DPVT*. 18

## REFERENCES


Ballan, L., Brostow, G.J., Puwein, J. & Pollefeys, M. (2010). Unstructured video-based rendering: interactive exploration of casually captured videos. In *ACM SIGGRAPH*. 31

Belagiannis, V., Amin, S., Andriluka, M., Schiele, B., Navab, N. & Ilic, S. (2014). 3d pictorial structures for multiple human pose estimation. *CVPR, IEEE*. 91

Bo, L. & Sminchisescu, C. (2010). Twin gaussian processes for structured prediction. *Int'l Journal of Computer Vision*, **87**, 28–52. 10

Bourdev, L. & Malik, J. (2009). Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*. 11

Bray, M., Kohli, P. & Torr, P. (2006). Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In *ECCV*, 642–655. 12

Bregler, C., Malik, J. & Pullen, K. (2004). Twist based acquisition and tracking of animal and human kinematics. *Int'l Journal of Computer Vision*, **56**, 179–194. 18

Brox, T., Rosenhahn, B., Gall, J. & Cremers, D. (2010). Combined region and motion-based 3d tracking of rigid and articulated objects. *TPAMI*, **32**, 402–415. 12

Canton-Ferrer, C., Casas, J.R. & Pardàs, M. (2010). Marker-based human motion capture in multiview sequences. *EURASIP J. Adv. Signal Process*, **2010**, 73:1–73:11. 16

Carceroni, R., Padua, F., Santos, M. & Kutulakos, K. (2004). Linear sequence–to–sequence alignment. In *CVPR*. 51, 58

Caspi, Y., Simakov, D. & Irani, M. (2006). Feature-based sequence-to-sequence matching. *Int. J. Comput. Vision*, **68**, 53–64. 9, 10, 33, 36, 46

Chen, X. & Yuille, A. (2014). Articulated pose estimation by a graphical model with image dependent pairwise relations. *NIPS*. 14, 26

Dai, C., Zheng, Y. & Li, X. (2006). Subframe video synchronization via 3d phase correlation. In *Image Processing, 2006 IEEE International Conference on*. 9

Dantone, M., Gall, J., Leistner, C. & Gool., L.V. (2013). Human pose estimation using body parts dependent joint regressors. In *CVPR*. 11

Deutscher, J. & Reid, I. (2005). Articulated body motion capture by stochastic search. *IJCV*, **61**, 185–205. 10

Eichner, M. & Ferrari, V. (2009). Better appearance models for pictorial structures. In *BMVC*. 11

Elhayek, A., Stoll, C., Hasler, N., Kim, K.I., Seidel, H.P. & Theobaltl, C. (2012a). Spatio-temporal motion tracking with unsynchronized cameras. In *Proc. CVPR*, 1870–1877. 8, 50, 80, 85, 112

Elhayek, A., Stoll, C., Hasler, N., Kim, K.I., Seidel, H.P. & Theobaltl, C. (2012b). Video: Spatio-temporal motion tracking with unsynchronized cameras. Http://gvv.mpi-inf.mpg.de/GVV*projects.html*. 60, 61

Elhayek, A., Stoll, C., Kim, K., Seidel, H.P. & Theobalt, C. (2012c). Feature-based multi-video synchronization with subframe accuracy. In A. Pinz, T. Pock, H. Bischof & F. Leberl, eds., *Pattern Recognition*, vol. 7476 of *Lecture Notes in Computer Science*, 266–275, Springer Berlin Heidelberg. 8, 32, 111

Elhayek, A., Stoll, C., Kim, K.I. & Theobaltl, C. (2014a). Outdoor human motion capture by simultaneous optimization of pose and camera parameters. In *Proc. CGF*. 8, 69, 112

Elhayek, A., Stoll, C., Kim, K.I. & Theobaltl, C. (2014b). Video: Outdoor human motion capture by simultaneous optimization of pose and camera parameters. Http://gvv.mpi-inf.mpg.de/projects/outdoorsHMC/. 72, 80, 81, 85, 86, 89

Elhayek, A., Aguiar, E., Jain, A., Tompson, J., Pishchulin, L., Andriluka, M., Bregler, C., Schiele, B. & Theobalt, C. (2015a). Efficient convnet-based marker-less motion capture in general scenes with a low number of cameras. In *Proc. CVPR*. 8, 92, 113

Elhayek, A., Aguiar, E., Jain, A., Tompson, J., Pishchulin, L., Andriluka, M., Bregler, C., Schiele, B. & Theobalt, C. (2015b). Video: Efficient convnet-based marker-less motion capture in general scenes with a low number of cameras. Http://gvv.mpi-inf.mpg.de/GVV*projects.html*. 100, 104

## REFERENCES


Felzenszwalb, P.F. & Huttenlocher, D.P. (2005). Pictorial structures for object recognition. *IJCV'05*. 11

Fischler, M.A. & Bolles, R.C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, **24**, 381–395. 36

Fischler, M.A. & Elschlager, R.A. (1973). The representation and matching of pictorial structures. *IEEE Trans. Comput'73*. 11

Gall, J., Rosenhahn, B. & Seidel, H.P. (2008). Drift-free tracking of rigid and articulated objects. In *CVPR*. 12

Gall, J., Stoll, C., Aguiar, E.D., Rosenhahn, B., Theobalt, C. & Seidel, H.P. (2009). Motion capture using joint skeleton tracking and surface estimation. In *CVPR*. 51, 65

Gall, J., Rosenhahn, B., Brox, T. & Seidel, H.P. (2010). Optimization and filtering for human motion capture – a multi-layer framework. *IJCV*, **87**, 75–92. 10

Ganapathi, V., Plagemann, C., Koller, D. & Thrun, S. (2010). Real time motion capture using a single time-of-flight camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, 755–762. 11

Germann, M., Hornung, A., Keiser, R., Ziegler, R., Würmlin, S. & Gross, M. (2010). Articulated billboards for video-based rendering. *CGF (Proc. Eurographics)*, **29**, 585–594. 13

Gkioxari, G., Arbelaez, P., Bourdev, L. & Malik, J. (2013). Articulated pose estimation using discriminative armlet classifiers. In *CVPR*. 11

Gleicher, M. & Ferrier, N. (2002). Evaluating video-based motion capture. In *Computer Animation, 2002. Proceedings of*, 75–80, IEEE. 15

Hartley, R. & Zisserman, A. (2004a). *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edn. 39

Hartley, R.I. & Zisserman, A. (2004b). *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edn. 71

Hasler, N., Rosenhahn, B., Thormählen, T., Wand, M., Gall, J. & Seidel, H.P. (2009a). Markerless motion capture with unsynchronized moving cameras. In *IEEE Conference on Computer Vision and Pattern Recognition*. 4, 12, 13, 49, 50, 51, 58, 68, 70

Hasler, N., Rosenhahn, B., Thormählen, T., Wand, M., Gall, J. & Seidel, H.P. (2009b). Markerless motion capture with unsynchronized moving cameras. In *CVPR*. 10

Ilic, S. & Fua, P. (2006). Implicit meshes for surface reconstruction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **28**, 328 –333. 51

Ionescu, C., Li, F. & Sminchisescu, C. (2011). Latent structured models for human pose estimation. In *ICCV*. 11

Jain, A., Tompson, J., Andriluka, M., Taylor, G.W. & Bregler, C. (2013). Learning human pose estimation features with convolutional networks. *arXiv preprint arXiv:1312.7302*. 26

Jain, A., Tompson, J., Andriluka, M., Taylor, G. & Bregler, C. (2014a). Learning human pose estimation features with convolutional networks. In *ICLR*. 14

Jain, A., Tompson, J., LeCun, Y. & Bregler, C. (2014b). Modeep: A deep learning framework using motion features for human pose estimation. *ACCV*. 14, 27

Kirk, A.G., O'Brien, J.F. & Forsyth, D.A. (2005). Skeletal parameter estimation from optical motion capture data. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2005*, 782–788. 16

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998a). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**, 2278–2324. 26, 95

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998b). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**, 2278–2324. 28

Lee, C.S. & Elgammal, A. (2010). Coupled visual and kinematic manifold models for tracking. *Int'l Journal of Computer Vision*, **87**, 118–139. 10

## REFERENCES


Li, R., Tian, T.P., Sclaroff, S. & Yang, M.H. (2010). 3d human motion tracking with a coordinated mixture of factor analyzers. *Int'l Journal of Computer Vision*, **87**, 170–190. 10

Lowe, D. (2004a). Distinctive image features from scale-invariant keypoints. *IJCV*, **60**, 91–110. 10

Lowe, D.G. (2004b). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, **60**, 91–110. 35

Menache, A. (1999). *Understanding Motion Capture for Computer Animation*. Morgan Kaufmann, 2nd edn. 15

Meyer, B., Stich, T. & Pollefeys, M. (2008). Subframe temporal alignment of non-stationary cameras. In *BMVC*. 9

Meyer, B., Stich, T., Magnor, M. & Pollefeys, M. (2009). Subframe temporal alignment of non-stationary cameras. In *CVPR*. 51, 58

Moeslund, T., Hilton, A. & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, **104**, 90–126. 1, 10, 49, 67

Nagi, J., Ducatelle, F., Di Caro, G., Ciresan, D., Meier, U., Giusti, A., Nagi, F., Schmidhuber, J. & Gambardella, L. (2011). Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *Signal and Image Processing Applications (ICSIPA), 2011 IEEE International Conference on*, 342–347. 29

Nair, V. & Hinton, G.E. (2010). Rectified linear units improve restricted boltzmann machines. In *ICML*, 807–814, Omnipress. 28

Pádua, F.L.C., Carceroni, R.L., Santos, G.A.M.R. & Kutulakos, K.N. (2010). Linear sequence-to-sequence alignment. *IEEE Trans. Pattern Anal. Mach. Intell.*, **32**, 304–320. 33

Phasespace (2014). Phasespace. Http://www.phasespace.com/. 16

Pishchulin, L., Andriluka, M., Gehler, P. & Schiele, B. (2013a). Poselet conditioned pictorial structures. In *CVPR*. 11

Pishchulin, L., Andriluka, M., Gehler, P. & Schiele, B. (2013b). Strong appearance and expressive spatial models for human pose estimation. In *IEEE ICCV*. 11

Plankers, R. & Fua, P. (2003). Articulated soft objects for multiview shape and motion capture. *TPAMI*, **25**, 1182–1187. 18

Plänkers, R. & Fua, P. (2001). Tracking and modeling people in video sequences. *Comput. Image and Vision Understanding*, 285–302. 11

Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J. & Koch, R. (2004). Visual modeling with a hand-held camera. *IJCV*, **59**, 207–232. 5, 68

Pons-Moll, G., Baak, A., Gall, J., Leal-Taixe, L., Mueller, M., Seidel, H.P. & Rosenhahn, B. (2011). Outdoor human motion capture using inverse kinematics and von Mises-Fisher sampling. In *Proc. ICCV*, 1243–1250. 12, 92

Poppe, R. (2007). Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, **108**. 1, 10, 49, 67

Razavian, A.S., Azizpour, H., Sullivan, J. & Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, **abs/1403.6382**. 26

Roetenberg, D. (2006). *Inertial and magnetic sensing of human motion*. Ph.D. thesis, Enschede. 16

Sapp, B. & Taskar, B. (2013). Modec: Multimodal decomposable models for human pose estimation. In *CVPR*. 11

Schmaltz, C., Rosenhahn, B., Brox, T. & Weickert, J. (2011). Region-based pose tracking with occlusions using 3d models. *Machine Vision and Applications*, 1–21. 12

Shiratori, T., Park, H.S. & Hodgins, L.S.Y.S.J.K. (2011). Motion capture from body-mounted cameras. *ACM TOG (Proc. SIGGRAPH)*, **30**, 31:1–31:10. 13

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A. & Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, 1297–1304. 11

Shrestha, P., Weda, H., Barbieri, M. & Sekulovski, D. (2006). Synchronization of multiple video recordings based on still camera flashes. In *ACM Multimedia*. 10

Sidenbladh, H. & Black, M. (2003). Learning the statistics of people in images and video. *IJCV*, **54**, 183–209. 18

Sidenbladh, H., Black, M. & Sigal, L. (2002). Implicit probabilistic models of human motion for synthesis and tracking. In *ECCV*, vol. 1, 784–800. 51

Sigal, L., Balan, A. & Black, M. (2010). Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *Int'l Journal of Computer Vision*, **87**, 4–27. 1, 10, 11, 19, 49, 67, 100, 104

Sigal, L., Isard, M., Haussecker, H. & Black, M. (2012). Loose-limbed people: Estimating 3d human pose and motion using non-parametric belief propagation. *International Journal of Computer Vision*, **98**, 15–48. 104, 107

Sinha, S.N. & Pollefeys, M. (2004). Synchronization and calibration of camera networks from silhouettes. In *ICPR*. 9

Sminchisescu, C. & Telea, A. (2002). Human pose estimation from silhouettes. a consistent approach using distance level sets. In *In WSCG International Conference on Computer Graphics, Visualization and Computer Vision*. 6, 69

Sridhar, S., Oulasvirta, A. & Theobalt, C. (2013). Interactive markerless articulated hand motion tracking using rgb and depth data. In *IEEE ICCV*. 14

Stein, G.P. (1998). Tracking from multiple view points: Self-calibration of space and time. In *DARPA IU Workshop*, 521–527. 9

Stoll, C. (2009). *Template Based Shape Processing*. Ph.D. thesis, Max-Planck-Institut für Informatik, Germany. 16, 18

Stoll, C., Hasler, N., Gall, J., Seidel, H.P. & Theobalt, C. (2011). Fast articulated motion tracking using a sums of gaussians body model. In *IEEE International Conference on Computer Vision*. 7, 12, 15, 18, 19, 20, 21, 22, 24, 25, 31, 50, 53, 55, 58, 60, 65, 77, 79, 80, 84, 85, 86, 87, 88, 92, 94, 95, 100, 103, 114

Taigman, Y., Yang, M., Ranzato, M. & Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 1701–1708. 26

Taylor, J., Shotton, J., Sharp, T. & Fitzgibbon, A.W. (2012). The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 103–110. 11

Thormählen, T., Hasler, N., W, M. & peter Seidel, H. (2008). Merging of feature tracks for camera motion estimation from video. In *Proc. CVMP*, 43–50. 5, 68, 86

Tompson, J., Jain, A., LeCun, Y. & Bregler, C. (2014a). Join training of a convolutional network and a graphical model for human pose estimation. *NIPS*. 7, 14, 26, 27, 29, 92, 94, 95

Tompson, J., Stein, M., Lecun, Y. & Perlin, K. (2014b). Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.*, **33**, 169:1–169:10. 26

Tompson, J., Goroshin, R., Jain, A., LeCun, Y. & Bregler, C. (2015). Efficient object localization using convolutional networks. *CVPR*. 26, 29

Toshev, A. & Szegedy, C. (2014). Deeppose: Human pose estimation via deep neural networks. In *CVPR*. 14, 26

Vicon (2014). Vicon. Http://www.vicon.com. 16

Wedge, D., Huynh, D. & Kovesi, P. (2006). Motion guided video sequence synchronization. In *ACCV*. 3, 31

Wei, X. & Chai, J. (2010). VideoMocap: modeling physically realistic human motion from monocular video sequences. *ACM TOG (Proc. SIGGRAPH)*, **29**, 42:1–42:10. 11

Wei, X., Zhang, P. & Chai, J. (2012). Accurate realtime full-body motion capture using a single depth camera. *ACM Transactions on Graphics*, **31**, 188:1–188:12. 11

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. In *Adv. in Comput. Math.*, 389–396. 23, 58, 74

Wren, C., Azarbayejani, A., Darrell, T. & Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *TPAMI*, **19**, 780–785. 19

Wu, C., Stoll, C., Valgaerts, L. & Theobalt, C. (2013). On-set performance capture of multiple actors with a stereo camera. **32**, 161:1–161:11. 11

Yang, Y. & Ramanan, D. (2011). Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*. 11

Ye, G., Liu, Y., Hasler, N., Ji, X., Dai, Q. & Theobalt, C. (2012). Performance capture of interacting characters with handheld Kinects. In *Proc. ECCV*, 828–841. 13

Zeiler, M. & Fergus, R. (2014). Visualizing and understanding convolutional networks. In D. Fleet, T. Pajdla, B. Schiele & T. Tuytelaars, eds., *Computer Vision – ECCV 2014*, vol. 8689 of *Lecture Notes in Computer Science*, 818–833, Springer International Publishing. 26