University of Padova

Department of Management and Engineering

Ph.D. Program in Mechatronics and Product Innovation Engineering

XXXIV Cycle

Multisensor Measurement and Modeling of Human Motion

Coordinator: Prof. Daria Battini

Supervisors: Prof. Monica Reggiani
Prof. Emanuele Menegatti

Ph.D. Student: Mattia Guidolin

May 2022

Abstract

The analysis of human motion is a multifaceted topic with crucial applications in several fields. Medicine, rehabilitation, biomechanics, but also robotics, logistics, and, lately, an increasing number of industrial scenarios share the need to quantify how a person is moving.

Starting from laborious manual annotations on images or videos, the technologies that allow an accurate assessment of human motion have developed rapidly in the last few decades. The current gold standard makes use of highly accurate optoelectronic systems capable of tracking the three-dimensional positions of a set of retroreflective markers with submillimeter precision. When such markers are applied to specific body landmarks, the motion of a person can be inferred. To this end, the correct positioning of the markers is a key step to obtain accurate results. Therefore, such analyses are typically performed in dedicated laboratories by specialized personnel with in-depth knowledge of human anatomy.

Although this approach is suitable for medical applications, the advent of Industry 4.0 first, requiring intelligent networking among the entities of the production system by means of Cyber-Physical Production Systems (CPPS), and, more importantly, of Industry 5.0, shifting the focus from the technology to the well-being of the human operator, brought a new set of requirements for the assessment of human motion. In fact, to ensure safe and productive collaboration between human operators and robotic devices, the latter must be constantly aware of the people within their workspace. Thus, human motion needs to be measured in unconstrained environments, in real-time, and without impacting the person's dexterity. To this end, inertial and markerless motion capture (MoCap) are alternative technologies to estimate the pose of a person without the drawbacks of optoelectronic systems.

Inertial systems exploit multiple inertial measurement units (IMUs) placed in specific parts of the body. Such systems combine the estimated orientations of each sensor with prior knowledge of a model describing the analyzed subject to reconstruct their motion. The accuracy of inertial MoCap is approaching the one of optoelectronic systems, without the need for dedicated laboratories or specialized personnel. In fact, despite the fact that the user must wear several sensors, the placement is less strict with respect to the markers used in optoelectronic MoCap.

Markerless systems, on the other hand, rely on deep learning algorithms to estimate a person's pose exploiting one or multiple cameras, without requiring any sensor or marker on the body. However, the achievable accuracy is one order of magnitude lower than the one of the other systems at best.

The next critical advancement in human motion analysis will consist of developing accurate non-invasive systems. Ideally, a complete MoCap system should be able to provide highly accurate measurements, without requiring complex hardware setups nor hindering the freedom of movement in any way, and in real-time. None of the currently available systems can fulfill all these characteristics.

This work presents the research activities that I conducted during my Ph.D. in this direction. The main objective of my research was to provide novel tools and algorithms to maximize the motion estimation accuracy, support heterogeneous quantities measured/estimated by different sensing systems, and minimize the number of sensors required on the body. Such a complex objective required the definition of three macro levels in which I divided my work: the *Sensing* level, the *Tracking* level, and the *Modeling* level. The first level aims to seamlessly integrate different sensor typologies with the developed algorithms. The second focuses on maximizing the pose estimation accuracy when using a distributed network of sensors. Finally, the latter enables multimodal sensor fusion of heterogeneous data by referring all the measured quantities to a common underlying model of the human.

The *Sensing* level is the first block of my work. Within this level, I focused on analyzing, comparing, and selecting state-of-the-art technologies for the assessment of human motion. The main goal of this level is to provide a bridge between the raw measured quantities and the developed algorithms, independently of the typology of sensors being used. This was achieved by defining a common interface used to represent the poses of multiple people independently of the input source, the number of measured (or estimated) marker positions, and the number of measured (or estimated) IMU orientations. As a result, all the algorithms developed within the *Tracking* and *Modeling* levels can be used independently of the typology or number of sensors employed. This is of paramount importance in the direction of enabling human MoCap in unconstrained environments, allowing the system to be adapted to the specific user and scenario.

Within the *Tracking* level, I developed state-of-the-art tools to enable robust, accurate, real-time multi-person tracking in distributed sensor networks. In fact, typical challenges of single-camera markerless systems (e.g., limited fields of view and occlusions) can be overcome by exploiting a distributed camera network. However, this introduces other types of challenges, since the input data can be potentially noisy and/or introduce delays. Therefore, the overall problem was divided into four distinct subproblems.

As a result, four modules were developed within this level: the Tracker module, the Merger module, the Optimizer module, and the Filter module. The former allows to perform robust frame-by-frame tracking of multiple people detections obtained by a multi-sensor network. No assumptions are made about the typology and number of sensors, nor on the number of people being tracked. The Merger module is used, after the Tracker, to correctly fuse the (potentially partial) information obtained by each sensor into a unique enhanced description of the persons' poses. The Optimizer module introduces a description of the people being analyzed. It allows to solve a global optimization problem with the goal of minimizing the body segments length variability across consecutive frames. Finally, the Filter module enables smoothing of the estimated motion trajectories by filtering high-frequency noise. All the aforementioned operations are performed in real-time. The accuracy increase achieved by the proposed workflow was assessed in a newly acquired dataset that we plan to release in the near future.

Finally, at the *Modeling* level, the information received from either the *Sensing* or *Tracking* levels is used to drive a musculoskeletal model of the human in real-time. The current gold standard for measuring human motion in highly accurate biomechanical analyses, in fact, relies on the usage of a musculoskeletal model of the human to simulate motion via inverse kinematics optimizations. However, such analyses typically require complex setups (e.g., optoelectronic systems) and extended post-processing of the recorded motion data.

Within this level, protocols considered *de-facto* standards in the biomechanics field were successfully adapted to allow their use in different contexts. This required several modifications of the original tools used in biomechanical analyses to achieve real-time performance and enable inverse kinematics optimizations to be progressed independently of the sensing system used for the measurements. Indeed, optoelectronic MoCap, inertial MoCap, markerless MoCap, or any combination of the former can be used. This is a key feature to enable multimodal sensor fusion, since all the measured quantities refer to the same underlying model.

The extensive work on sensing, tracking, and modeling converged on the development of an open-source efficient, flexible, modular framework for real-time multi-sensor measurement and modeling of human motion. This required the selection of state-of-the-art tools from both the robotics and biomechanics communities. The communication between different sensors and algorithms is based on ROS, an established middleware considered the *de-facto* standard for the development of complex distributed robotics applications. The algorithms developed for real-time inverse kinematics optimizations, on the other hand, rely on OpenSim, a well-known library for biomechanical analyses that includes several already validated musculoskeletal models.

To conclude, the work developed within my Ph.D. research has the potential to push forward, via a multidisciplinary approach, the state-of-the-art on accurate unobtrusive assessment of human motion in unconstrained environments. Indeed, the proposed system has the ability to enable the development of a variety of emerging applications, where precise knowledge of the human pose in unpredictable environments, with a minimal number of on-body sensors, and in real-time, is a fundamental requirement.

Sommario

L'analisi del movimento umano rappresenta un argomento poliedrico con diverse applicazioni in molteplici campi. Medicina, riabilitazione, biomeccanica, ma anche robotica, logistica e, ultimamente, l'ambiente industriale, sono accomunati dalla necessità di quantificare come una persona si sta muovendo.

Inizialmente limitate ad annotazioni manuali su immagini o video, le tecnologie disponibili per valutare il movimento sono cresciute esponenzialmente negli ultimi decenni. Attualmente lo standard di riferimento in questo ambito consiste nell'uso di sistemi optoelettronici estremamente accurati, in grado di tracciare la posizione nello spazio di una serie di marcatori retroriflettenti con precisione submillimetrica. L'applicazione di tali marcatori in specifici punti del corpo umano permette di stimarne la posa. A tal fine, il corretto posizionamento dei marker è un requisito fondamentale per permettere di ottenere risultati accurati. Pertanto, tali analisi sono tipicamente condotte da personale specializzato, con una conoscenza approfondita dell'anatomia umana, in laboratori dedicati.

Sebbene questo approccio sia adatto alle applicazioni mediche, l'avvento dell'Industria 4.0, che richiede una comunicazione intelligente tra tutte le entità del sistema produttivo, e, soprattutto, dell'Industria 5.0, la quale ha spostato l'attenzione dalla tecnologia al benessere dell'operatore umano, ha introdotto una nuova serie di requisiti. Infatti, per garantire una collaborazione sicura ed efficace tra operatori umani e dispositivi robotici, questi ultimi devono essere costantemente informati delle persone all'interno del proprio spazio di lavoro. Di conseguenza, il movimento umano deve essere misurato in ambienti non vincolati, in tempo reale, e senza influire sulla libertà di movimento della persona. A tal fine, sistemi di motion capture (MoCap) inerziali e markerless rappresentano tecnologie alternative per ot-

tenere una stima della posa della persona senza le limitazioni tipiche dei sistemi optoelettronici.

I sistemi inerziali sfruttano una serie di sensori inerziali (IMU) posizionati in parti specifiche del corpo. Tali sistemi combinano le orientazioni stimate da ogni sensore con un modello cinematico del soggetto analizzato per ricostruirne il movimento. La precisione di questi sistemi si avvicina a quella dei sistemi optoelettronici, senza però la necessità di laboratori dedicati o personale specializzato. Infatti, nonostante richiedano all'utilizzatore di indossare diversi sensori, il posizionamento è meno rigoroso rispetto a quello dei marcatori utilizzati nei sistemi optoelettronici.

I sistemi markerless, invece, si basano su algoritmi di deep learning al fine di stimare la posa di una persona sfruttando una o più telecamere, senza richiedere alcun sensore o marcatore fisico sul corpo. Tuttavia, la precisione ottenibile è un ordine di grandezza inferiore rispetto agli altri sistemi, anche in condizioni ideali.

Il prossimo progresso fondamentale nell'analisi del movimento umano consisterà nello sviluppo di sistemi accurati ed allo stesso tempo non invasivi. Idealmente, un sistema completo dovrebbe essere in grado di fornire misurazioni estremamente accurate, in tempo reale, e senza richiedere complesse configurazioni hardware né ostacolare in alcun modo la libertà di movimento. Nessuno dei sistemi attualmente disponibili è in grado di soddisfare tutte queste caratteristiche.

Questo lavoro presenta le attività di ricerca che ho condotto in questa direzione durante il mio dottorato. Lo scopo principale della mia ricerca ha riguardato lo sviluppo di nuovi strumenti e algoritmi per massimizzare l'accuratezza della stima del movimento, supportare dati eterogenei misurati o stimati da diversi sistemi di motion capture, e minimizzare il numero di sensori richiesti sul corpo. Un obiettivo così complesso ha richiesto la definizione di tre macro livelli in cui ho suddiviso il mio lavoro: il livello di *Sensing*, il livello di *Tracking* e il livello di *Modeling*. Il primo livello ha lo scopo di integrare, senza soluzione di continuità, diverse tipologie di sensori con gli algoritmi sviluppati. Il secondo, invece, si concentra sulla massimizzazione dell'accuratezza della stima della posa attraverso l'utilizzo di una rete distribuita di sensori. Infine, l'ultimo livello consente la fusione di dati ottenuti da sensori eterogenei, associando tutte le grandezze misurate ad un unico modello comune dell'essere umano.

Il livello di *Sensing* rappresenta il blocco iniziale del mio lavoro. All'interno di questo livello mi sono concentrato sull'analisi, il confronto e la selezione di tecnologie all'avanguardia per la stima del movimento umano. L'obiettivo principale di questo livello è fornire un collegamento tra le grandezze fisiche direttamente misurate e gli algoritmi sviluppati per la stima della posa, indipendentemente dalla tipologia di sensore utilizzato. Ciò è stato possibile attraverso la definizione di un'interfaccia comune utilizzata per rappresentare la posa di più persone indipendentemente dalla sorgente di input, dal numero di marker, e dal numero di IMU. Di conseguenza, tutti gli algoritmi sviluppati all'interno dei livelli di *Tracking* e di *Modeling* possono essere utilizzati indipendentemente dalla tipologia o dal numero di sensori presenti. Questo è di fondamentale importanza al fine di consentire la stima del movimento umano in ambienti non vincolati, permettendo al sistema di adattarsi allo specifico utente e scenario.

All'interno del livello di *Tracking* ho sviluppato strumenti all'avanguardia per abilitare il tracciamento di più persone mediante l'utilizzo di reti di sensori distribuiti. Tale sistema è robusto, accurato e permette una stima della posa in tempo reale. Le principali difficoltà tipiche dei sistemi markerless basati su singola telecamera (ad esempio, campi visivi limitati e occlusioni) possono essere superate sfruttando una rete di telecamere. Tuttavia, ciò introduce nuove tipologie di problemi, in quanto i dati ricevuti da ogni sensore possono essere potenzialmente rumorosi e/o introdurre ritardi. Di conseguenza, il problema globale è stato suddiviso in quattro sottoproblemi distinti.

All'interno di questo livello sono stati sviluppati quattro moduli: il modulo Tracker, il modulo Merger, il modulo Optimizer e il modulo Filter. Il primo consente di eseguire il tracciamento frame-by-frame delle pose di più persone ottenute da una rete multisensore. Il sistema funziona indipendentemente sia dalla tipologia e dal numero di sensori, che dal numero di persone tracciate. Il modulo Merger viene utilizzato, a valle del Tracker, per fondere correttamente le informazioni (potenzialmente parziali) ottenute da ogni sensore in un'unica descrizione completa della scena. Il modulo Optimizer, invece, introduce una descrizione delle persone analizzate. Consente di risolvere un problema di ottimizzazione con l'obiettivo di ridurre al minimo la variabilità della lunghezza dei segmenti del corpo tra frame consecutivi. Infine, il modulo Filter permette il filtraggio delle traiettorie stimate, riducendo il rumore ad alta frequenza. Tutte le precedenti operazioni vengono eseguite in tempo reale. L'aumento di precisione ottenuto dal sistema proposto è stato valutato su di un dataset recentemente acquisito che prevediamo di rilasciare nel prossimo futuro.

Infine, nel livello di *Modeling*, le informazioni ricevute dai livelli di

Sensing o di *Tracking* vengono utilizzate per guidare un modello muscoloscheletrico dell'essere umano in tempo reale. L'attuale standard di riferimento per la misurazione del movimento umano in analisi biomeccaniche altamente accurate, infatti, si basa sull'uso di un modello muscoloscheletrico dell'essere umano per simulare il movimento tramite un approccio di cinematica inversa. Tuttavia, tali analisi richiedono tipicamente configurazioni complesse (ad esempio, sistemi optoelettronici) e un'estesa elaborazione dei dati grezzi registrati.

All'interno di questo livello, i protocolli considerati standard in ambiente biomeccanico sono stati adattati con successo per consentirne l'utilizzo in contesti diversi. Ciò ha richiesto diverse modifiche agli strumenti originali utilizzati in biomeccanica per ottenere prestazioni in tempo reale e consentire l'esecuzione di ottimizzazioni basate su cinematica inversa indipendentemente dal sistema di misurazione utilizzato. Il sistema sviluppato permette di utilizzare MoCap di tipo optoelettronico, inerziale, markerless, o qualsiasi combinazione dei precedenti. Questa è una caratteristica fondamentale per consentire la fusione multimodale di sensori eterogenei, poiché tutte le grandezze misurate si riferiscono ad un unico modello condiviso.

L'ampio lavoro all'interno dei tre livelli sopracitati ha portato allo sviluppo di un framework open-source efficiente, flessibile e modulare per la misurazione e modellazione multisensore in tempo reale del movimento umano. Ciò ha richiesto la selezione di strumenti all'avanguardia sia dall'ambito robotico che da quello biomeccanico. Infatti, la comunicazione tra diversi sensori e algoritmi è basata su ROS, un middleware consolidato attualmente considerato lo standard di riferimento per lo sviluppo di complesse applicazioni robotiche. Gli algoritmi sviluppati per la stima della cinematica in tempo reale, invece, si basano su OpenSim, una nota libreria per analisi biomeccaniche che offre numerosi modelli muscoloscheletrici già validati.

Per concludere, il lavoro sviluppato nell'ambito della mia ricerca di dottorato ha le potenzialità di fare avanzare lo stato dell'arte nell'ambito della stima non invasiva del movimento umano in ambienti non vincolati attraverso un approccio multidisciplinare. Il sistema proposto, infatti, consente lo sviluppo di una varietà di applicazioni emergenti in cui la conoscenza precisa della posa umana in ambienti imprevedibili, con un numero minimo di sensori sul corpo, e in tempo reale, è un requisito fondamentale.

List of Publications

[1] D. Battini, N. Berti, S. Finco, **M. Guidolin**, M. Reggiani, and L. Tagliapietra, "WEM-Platform: A Real-Time Platform for Full-Body Ergonomic Assessment and Feedback in Manufacturing and Logistics Systems", *Computers & Industrial Engineering*, vol. 164, p. 107 881, 2022.

[2] ——, "Real-Time Full Body Ergonomic Platform for Ergonomics Assessment and Fast Worker Training in Industrial Systems", in *21st Triennial Congress of the International Ergonomics Association (IEA)*, 2021.

[3] **M. Guidolin**, R. A. B. Petrea, R. Oboe, M. Reggiani, E. Menegatti, and L. Tagliapietra, "On the Accuracy of IMUs for Human Motion Tracking: A Comparative Evaluation", in *2021 IEEE International Conference on Mechatronics (ICM)*, IEEE, 2021, pp. 1–6.

[4] **M. Guidolin**, E. Menegatti, M. Reggiani, and L. Tagliapietra, "A ROS Driver for Xsens Wireless Inertial Measurement Unit Systems", in *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, IEEE, vol. 1, 2021, pp. 677–683.

[5] A. Malaguti, M. Carraro, **M. Guidolin**, L. Tagliapietra, E. Menegatti, and S. Ghidoni, "Real-Time Tracking-by-Detection of Human Motion in RGB-D Camera Networks", in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, IEEE, 2019, pp. 3198–3204.

[6] **M. Guidolin**, L. Tagliapietra, E. Menegatti, and M. Reggiani, "Hi-ROS: Open-Source Multi-Camera Sensor Fusion for Real-Time People Tracking", *Computer Vision and Image Understanding*, [submitted - under review].

[7] **M. Guidolin**, E. Menegatti, and M. Reggiani, “UNIPD-BPE: Synchronized RGB-D and Inertial Dataset for Multimodal Body Pose Estimation and Tracking”, *Data*, [submitted - under review]).

[8] N. Berti, S. Finco, **M. Guidolin**, M. Reggiani, and D. Battini, “Real-Time Postural Training Effects on Single and Multi-Person Ergonomic Risk Scores”, *IFAC-PapersOnLine*, [accepted].

[9] G. Nicola, L. Tagliapietra, **M. Guidolin**, S. Ghidoni, and N. Pedrocchi, “Feedback Motion Planning in Human-Robot Shared Workspace via Deep Reinforcement Learning”, *IEEE Transactions on Automation Science and Engineering*, [submitted - under review].

Contents

List of Figures

List of Tables

Acronyms

ABD	Abduction/Adduction
ADL	Activity of Daily Living
API	Application Programming Interface
AR	Action Recognition
BPE	Body Pose Estimation
CPPS	Cyber-Physical Production Systems
CV	Computer Vision
DE	Differential Evolution
DoF	Degree of Freedom
DRL	Deep Reinforcement Learning
ECG	Electrocardiogram
EEG	Electroencephalogram
EMG	Electromiography
FE	Flexion/Extension
FK	Forward Kinematics
FoV	Field of View
GRF	Ground Reaction Force
Hi-ROS	Human Interaction in ROS
HRI	Human-Robot Interaction
ID	Inverse Dynamics

IER	Internal/External Rotation
IK	Inverse Kinematics
IMU	Inertial Measurement Unit
IR	Infrared
ISB	International Society of Biomechanics
MB-IK	Marker-Based Inverse Kinematics
MEMS	Micro-Electro-Mechanical System
MOB-IK	Marker-and-Orientation-Based Inverse Kinematics
MoCap	Motion Capture
NIOSH	National Institute for Occupational Safety and Health
OB-IK	Orientation-Based Inverse Kinematics
OWAS	Ovako Working posture Assessment System
PERA	Postural Ergonomic Risk Assessment
P-RMSE	Percentage Root Mean Squared Error
REBA	Rapid Entire Body Assessment
RGB	Red-Green-Blue
RGB-D	Red-Green-Blue-Depth
RMSE	Root Mean Squared Error
RoM	Range of Motion
ROS	Robot Operating System
RULA	Rapid Upper Limb Assessment
SD	Standard Deviation
SDI	Strap Down Integration
SDK	Software Development Kit
STA	Soft Tissue Artifact
STS	Smart Textile System
T	Tracker
TKF	Tobit Kalman Filter
TMF	Tracker, Merger, Filter

TMOF	Tracker, Merger, Optimizer, Filter
ToF	Time of Flight
UML	Unified Modeling Language
WEM	Worker, Ergonomist, Manager
WLS	Weighted Least-Squares

1 Introduction

The study of human movement is an intriguing topic that has always impacted many and diverse disciplines. Such a complex subject had a deep impact on arts, as in the countless works of Leonardo da Vinci driven by his studies on human anatomy, photography, from the pioneering experiments of Eadweard Muybridge, considered the first attempts to objectively study the biomechanics of the human body, and science, with applications ranging from biomechanics, computer vision (CV), action recognition (AR), and human-robot interaction (HRI) [1].

Human movement involves multiple interacting anatomical systems (e.g., nerves, muscles, vision, etc.), and precise coordination between such systems and the limbs. The study of such a complex biomechanical system resulted, in the 20th century, in the development of a new discipline of science, namely, biomechanics [2]. At the same time, the increasing attention on this topic enabled the methods to extract kinematics information of the body pose from images to rapidly evolve in recent years. Starting from laborious manual annotations, today human movement is typically studied by means of automatic optoelectronic systems. Such technology exploits multiple cameras emitting invisible infrared light to infer the three-dimensional (3D) positions of passive retroreflective markers attached to the person's body [3].

The advent of Industry 4.0 first, requiring intelligent networking among the entities of the production system by means of Cyber-Physical Production Systems (CPPS), and, more importantly, of Industry 5.0, shifting the focus from the technology to the well-being of the human operator, contributed to extend the number of disciplines where knowledge of human motion is of paramount importance. In fact, to guarantee a safe and productive interaction between operators and robots, the latter need to be (1) aware of the task to perform, (2) aware of the human partner, (3) symbiotic in their interactions with humans, and (4) cooperative, to maximize both productivity and quality of life of the operator. Therefore, collaborative robots must be constantly aware of the humans approaching their workspace.

The industrial scenario defines three types of HRI:

- Coexistence: humans and robots share the same workspace, but execute completely separate tasks;
- Cooperation: humans and robots perform different tasks, but with a shared objective;
- Collaboration: humans and robots actively interact while executing complex tasks.

The first two scenarios need basic knowledge of the human, like their real-time centroid's position in space. However, the third scenario requires accurate measurement of the operator's pose and, possibly, of their intention. Such data must be available in real-time, and without requiring bulky hardware on the operator's body, not to hinder their movements [4], [5].

1.1 HUMAN MOTION ANALYSIS: A HISTORICAL OVERVIEW

The first biomechanical studies on human motion were enabled by photography and, specifically, by chronophotography. The word *chronophotography* describes several kinds of sequential photography that, at the end of the 19th century, allowed to represent motion by taking a sequence of pictures delayed by a fixed time [6].

The first and most known attempts on analyzing motion using chronophotography were from the contemporary experiments of the English photographer Eadweard Muybridge (1830 – 1904) and of the French scientist Etienne-Jules Marey (1830 – 1904). Muybridge's most famous work, The Horse in Motion (Figure 1.1) aimed to answer a popular biomechanical question of the period: is there a moment in which all the four hooves of a trotting horse are off the ground at the same time? Using an array of 12 cameras positioned beside a barn, Muybridge was able to take sequential shots of a galloping horse, each delayed by about 40 ms. The images allowed to answer the question, showing how indeed there is a time when all the four hooves are simultaneously off the ground. Muybridge extensively focused also on human motion (e.g., walking downstairs, boxing, walking of children, jumping, etc.), works that are considered the beginning of biomechanics [1], [2].

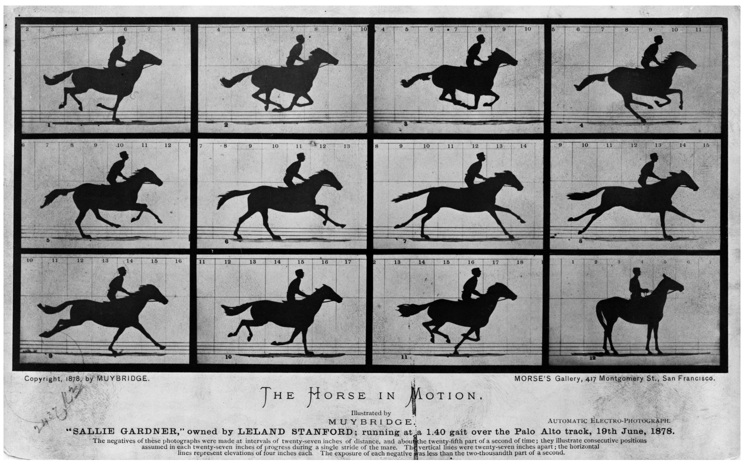

Copyright, 1878, by MUYBRIDGE.
MORSE'S Gallery, 417 Montgomery St., San Francisco.
THE HORSE IN MOTION.
Illustrated by
MUYBRIDGE.
AUTOMATIC ELECTRO-PHOTOGRAPH.
"SALLIE GARDNER," owned by LELAND STANFORD; running at a 1.40 gait over the Palo Alto track, 19th June, 1878.
The negatives of these photographs were made at intervals of twenty-seven inches of distance, and about the twenty-fifth part of a second of time; they illustrate consecutive positions assumed in each twenty-seven inches of progress during a single stride of the mare. The vertical lines were twenty-seven inches apart; the horizontal lines represent elevations of four inches each. The exposure of each negative was less than the two-thousandth part of a second.

Figure 1.1: Muybridge's The Horse in Motion, 1878.

While Muybridge is known as the pioneer in motion capturing, Marey's contribution to human motion analysis also deserves an honorable mention. In fact, he had the pioneering idea to introduce the use of markers on the subject's body. Such markers resulted in high-contrast dots or lines in the acquired images, allowing accurate identification of the body landmarks of interest (Figure 1.2). This idea will be the basis for the development, more than one century later, of marker-based optoelectronic systems, currently considered the gold standard in human motion analysis.

However, the interest in human motion goes way back in history, more than 2000 years before Muybridge and Marey's studies. The first evidence, in fact, dates back to the ancient Greek philosopher Aristotele (-383 to -321). His short text, Περὶ πορείας ζῴω [7], is the first known document on biomechanics. It describes the gait of various animals, including detailed observations on human motion, with geometric analyses of the limbs during the movement. In classical antiquity, in fact, human and animal motion patterns were typically analyzed together, by comparing the first to the latter.

Leonardo da Vinci (1452 – 1519) also showed a strong interest in human motion, dedicating a large number of his works to the study of human

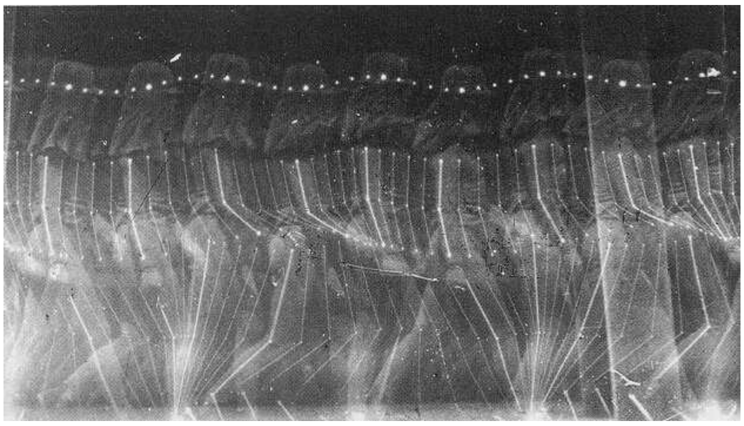


Figure 1.2: Marey's Walk, 1886.

anatomy. His sketchbooks contain, besides extremely detailed models of the human body, studies on kinematics trees of human motion, precursors of the kinematic chains used in modern biomechanical models. As an example, he analyzed the weight shifting during stair ascend and descend, concluding that a person's center of mass needs to always be on top of the center of the foot on which they are standing [2].

However, it was Giovanni Alfonso Borelli (1608 – 1679) the first person who applied the analytical methods developed by Galileo Galilei (1564 – 1642) to biology. He found that bones act as levers, while muscles behave following mathematical principles. Specifically, he was the first to understand that such levers magnify motion rather than force, so that the muscles need to produce a much higher force with respect to the external one acting on the body. For these reasons, he is often labeled as the father of biomechanics [8].

1.2 CURRENT STATE OF THE ART

Analyzing human motion consists of retrieving information that defines the pose of a person. Therefore, the final objective is the estimation of the angular values that describe each body joint at a specific time frame. This information is typically used in combination with a musculoskeletal model of the person being analyzed. These models are composed of rigid bodies representing the bones, connected by mechanical joints. Representations of the body muscles are included in the models, defined as the actuators needed to contract the limbs by providing the joint torques necessary to generate movement.

Musculoskeletal models are advanced tools for the assessment of human motion, enabling accurate analyses of the functional capacity of muscles and the design of specific surgical procedures, among others. A variety of models developed over the years are available in the literature. Each model can vary the number of degrees of freedom (DoF) and muscle-tendon actuators, and the mathematics used to simulate muscle activity. Models can be specialized for the analysis of specific movements (e.g., [9] for lower limbs, [10] for upper limbs) or allow an analysis of the full-body motion (e.g., [11]). The set of measured (or, more correctly, estimated) anatomical joint angles uniquely describes the subject's kinematics.

If the model incorporates body segment inertial parameters, it is possible to estimate the motion of the center of mass. Moreover, such data, together with external measurements of the forces acting on the body (e.g., ground reaction forces (GRF) measured using force plates placed on the ground), allow an estimation of the internal dynamics generating the movement [12]. These analyses are typically the result of inverse kinematics (IK) optimizations, followed by inverse dynamics (ID) optimizations, which both consider data describing a specific time frame. It is important to highlight that accurate estimation of the body kinematics is of paramount importance, since inaccuracies in the estimation of the kinematics will result in even larger errors for the estimation of body kinetics [13]. Finally, by concatenating the results through time (either kinematics, dynamics, or both), it is possible to reconstruct human motion.

Joint angles can be obtained using direct techniques (i.e., requiring devices affixed on the body) or indirect techniques (i.e., using vision-based systems) [3]. Direct methods, as the name suggests, are capable of directly measuring the joint angles. Therefore, a musculoskeletal model is not required. However, direct methods typically require bulky devices attached on the subject's links that connect the joint of interest. This typically limits their use on applications that do not require knowledge of the full-body kinematics (e.g., to measure the range of motion (RoM) of a specific joint).

Indirect methods, on the other hand, aim to estimate the anatomical joint angles by directly measuring different quantities (e.g., the movement of a set of markers applied on the body, or the orientation of an inertial measurement unit (IMU) attached to the limbs). In this case, to obtain information on the joint angles, such data must be applied to a model specifically scaled to represent the analyzed person. Although indirect methods are more complex, the technological progress in MoCap, together with advanced techniques developed in the latest decades, resulted in direct methods being considered as the gold standard for accurately assessing human motion.

As mentioned earlier, direct methods allow to directly measure the anatomical joint angles. This is typically achieved by using specifically designed goniometers. Traditional goniometers consist of a protractor with extending arms. They come in two forms: short arm, for smaller joints, such as wrist, elbow, and ankle, and long arm, for joints with longer levers, such as the knee and hip joints. To measure a joint angle, the fulcrum of the goniometer needs to be precisely aligned to the fulcrum of the joint, with the stationary arm aligned to the proximal body segment and the extending arm following the distal body segment [14]. Modern digital goniometers are equipped with an electronic scale that allows real-time streaming of the measured data, providing more accurate measurements compared to traditional universal goniometers [15].

Still, goniometers suffer from various limitations. First, to produce reliable measurements, the alignment of the goniometer on the body segments must be extremely accurate. This condition is far from being easily achievable, both due to soft tissue artifacts (STAs) and to inter-subject variability. Secondly, goniometers can be quite bulky and, therefore, interfere, both physically and psychologically, with the freedom of movement of the subject being analyzed. The popularity of these devices mainly depends on their simplicity of use, and from the advantage of providing direct measurements of the joint angles without requiring an underlying musculoskeletal model.

However, human motion analysis should ideally be able to provide accurate kinematic information, possibly in real-time, and without limiting the subject or influencing their motion [16]. In this regard, indirect techniques should be preferred to direct methods, since they are less intrusive, thus reducing the interference to the subject's movement [3]. However, indirect methods require a model of the human to estimate the anatomical joint angles from other quantitative measurements of motion acquired by means of a motion capture (MoCap) system. Although being more complex with respect to direct methods, the rapid advances in technology of the latest decades allowed these systems (and, specifically, optoelectronic MoCap) to gain the role of *de-facto* standard in human motion analysis. However, in recent years, attention has been shifting to alternative MoCap systems, whose accuracy is rapidly increasing.

MoCap systems can be divided into three categories: optoelectronic systems, requiring a set of retroreflective markers to be applied on the body, inertial systems, based on a chain of IMUs worn by the subject, and markerless systems, which do not require any sensor or marker on the body and rely on deep learning algorithms for the estimation of motion.

The current gold standard for MoCap systems consists of marker-based optoelectronic systems [17]. Optical motion analysis requires the estimation of the pose (which consists of position and orientation) of an object in space. This technology uses a calibrated network of high-performance infrared (IR) cameras to track the 3D location of several reflective markers attached to the body of the subject being analyzed [18]. However, an accurate estimation of a persons' whole-body pose is an extremely challenging problem, the human body being a complex, highly articulated entity prone to self-occlusions. Thus, the structure of the human body is typically simplified using a series of rigid bodies (bones) connected by mechanical rotational joints [3], [19]–[21]. A detailed description of the working principle and characteristics of these systems is given in Section 1.2.1.

When the line of sight between the cameras and the person cannot be guaranteed, or when motion is captured in unconstrained environments, wearable sensing based on IMUs is considered the most promising solution [22]. An IMU consists of a triaxial accelerometer and a triaxial gyroscope, used to measure linear accelerations and angular velocities with respect to a predefined rigid local frame. Modern IMUs often also include a triaxial magnetometer, to allow measurements of the Earth's magnetic field [23]. IMU-based systems have the advantage of being completely self-contained and independent of artificially generated sources (e.g., the IR light emitted from the cameras used in optoelectronic MoCap). The main limitation of such systems is represented by noise and errors in the raw measurements, which can lead to drifting phenomena in the estimation of the IMU orientation [24]. Despite the aforementioned limitations, IMUs are becoming a portable and cost-effective alternative to optical motion capture systems, and are currently being introduced also in clinical settings for functional movement quality assessments. A detailed description of the working principle and characteristics of inertial systems is given in

Section 1.2.2.

If it is not possible to sensorize the subject by applying markers or IMUs on their body, markerless MoCap is a promising alternative to the two aforementioned systems. Such technology exploits one or multiple cameras (which can capture image information, depth information, or both) to obtain a 3D view of the scene. The joint locations are estimated by exploiting deep learning algorithms (e.g., [25]–[28]) without the necessity of any dedicated sensor on the body. For this reason, markerless systems are by far the least invasive. The movement of the analyzed subject is not constrained, and the initial setup time is minimal. Markerless systems, although they can resolve the drawbacks typical of inertial and optoelectronic MoCap, suffer from several limitations. The number of tracked joints is usually limited, resulting in a coarser description of motion. Moreover, existing methods for estimating 3D human pose from markerless MoCap are at best one order of magnitude less accurate than optoelectronic systems [29]. A detailed description of the working principle and characteristics of markerless systems is addressed in Section 1.2.3.

1.2.1 Optoelectronic Systems

Optoelectronic systems are considered the gold standard for assessing human motion [17]. They are extensively adopted in the cinema and video game industries to model virtual characters, and in clinical analyses and rehabilitation for evaluating a patient's progression.

Optoelectronic systems rely on a set of high-speed IR cameras, typically placed on the perimeter of a dedicated laboratory, to synchronously record the same scene from multiple points of view. Each camera is equipped with an IR emitter to lighten the field of view. The light is reflected by multiple retroreflective markers that need to be applied to specific body landmarks. This allows to obtain high-contrast images, where the markers are seen as white dots on a black background. The size of each dot depends on the diameter of the marker, the resolution of the camera, and the distance between the marker and the camera's focal point. Typical marker diameters range from $1\,\mathrm{mm}$ to $12\,\mathrm{mm}$. While markers are highly reflective in the IR domain, sunlight also includes a strong component in this domain, introducing artifacts in the estimation of the marker positions. For this reason, optoelectronic systems typically allow the manual adjustment of specific software parameters to minimize the interference of sunlight and to avoid detections of phantom markers caused by reflections. However, optoelectronic systems are not suited for outdoor usage. They are typically restricted to dedicated laboratories, both because of their sensitivity to sunlight and because of the care required by the hardware setup.

The working principle of optoelectronic MoCap allows to retrieve the three-dimensional positions of the detected markers by exploiting stereophotogrammetry. First, the two-dimensional (2D) position of each marker is computed, in pixel coordinates, as the centroid of the white dot representing the marker in the acquired binary image. Then, the 2D marker coordinates computed by each camera are combined to estimate their 3D positions with respect to a common fixed reference frame via triangulation. This requires an accurate prior calibration of the relative poses of each camera with respect to the others, which must be performed before the motion acquisition. Such computation can nowadays be done in real-time. This can be achieved since the system extracts the 2D positions of all the markers seen by each camera and, then, limits the 3D reconstruction to this restricted set of points.

Ideally, to estimate the 3D position of a marker, two views are sufficient. However, since markers can be easily occluded during movement, a large number of cameras (usually ranging from 6 to 12) are typically adopted to increase the probability of having two (or more) cameras seeing each marker during the whole recording session. If more than two cameras are able to detect the same marker, redundancy allows to optimize the estimated 3D position and, therefore, to increase the overall system's accuracy [18], [30]–[32]. However, the number of markers should not be excessive, not to overly restrict natural movements. Increasing the number of markers can also result in close clusters, where the tracking of each specific marker becomes challenging [33].

The advancements in technology achieved in the latest decades, and the many competitors in this field (e.g., Vicon [34], Qualysis [35], and BTS [36]), allowed optoelectronic systems to reach an accuracy lower than $1\,\mathrm{mm}$ on the reconstruction of 3D marker coordinates [1]. However, reconstructing human motion introduces additional challenges that reduce the final accuracy.

One of the primary and, probably, the most important limitation of optoelectronic and, in general, indirect systems, is the necessity to assess skeletal movement from sensory systems applied to the body [21]. In this context, the model used to describe the human being consists of a set of rigid bodies (bones) connected by mechanical frictionless joints. Under these assumptions, at least three non-collinear markers are required to fully estimate the 6 degrees of freedom (DoF) of a segment (3 relating to the translation and 3 defining the orientation). Knowing the relative orientations of two linked segments, the three joint angles describing the joint state can be calculated. This is possible under the hypothesis that the relative positions between the markers and the segments to which they are attached remain constant during the motion. However, this hypothesis is generally not true.

In fact, skin movement relative to the underlying bone is a major source of errors that limit the accuracy of indirect systems. The errors introduced by this assumption produce artifacts in the reconstructed motion, known as STAs [18], [37], [38]. Soft tissue movement introduces systematic and random errors with frequencies similar to the ones of real movement, making them difficult to attenuate through data filtering. These errors alone can exceed $10\,\mathrm{mm}$ for the estimation of anatomical landmarks, leading to inaccuracies of $10°$ in the estimation of the joint angles [39]. However, these effects can be reduced using advanced techniques, such as increasing the number of markers attached to each segment, following specifically designed protocols for the positioning of the markers on the body, and exploiting IK optimizations for the analysis [31].

Furthermore, the cameras required by the system are highly delicate, and extremely small movements of the cameras after the calibration procedure can lead to large errors in the estimation of the positions of the markers. The correct attachment of markers on the subject's body is also a critical and intrusive procedure. It is a time-consuming task that requires extreme precision and in-depth knowledge of human anatomy, since small variations in the placement of the markers can induce large variations in the estimated joint angles [40], [41]. Thus, it depends on a specialized individual specifically trained to ensure the correct and consistent positioning of the markers. However, even in the best-case scenario, day-to-day and inter-tester variability in marker placement cannot be avoided, thus reducing the reliability of optoelectronic measurements, especially for motions in the transverse plane [42], [43].

1.2.2 Inertial Systems

One of the biggest challenges in motion tracking is having an accurate estimation with: (1) non-invasive sensors, not to influence their movements,

(2) non-limited workspaces, to allow measuring subjects in any environment, and (3) online, to enable real-time feedback and intervention [44]. As discussed in the previous section, while optoelectronic systems can provide accurate data, and online, they do require invasive marker placements and protected laboratories. When the line of sight between the sensors and the human cannot be ensured, or when motion is to be captured in large or outdoor spaces, wearable sensing based on IMUs is considered the most promising solution. In this context, chains of IMUs worn on the body can be used to overcome the necessity of a protected confined environment for motion assessment [22]. Moreover, the placement of such sensors on the body is extremely less strict with respect to optoelectronic MoCap, allowing for fast setup times without the necessity of specialized personnel.

An IMU is an electronic device that can be used to retrieve estimates of its orientation, expressed with respect to a fixed reference frame. Initially bulky and heavy, IMUs have been the subject of extensive research in navigation and aerospace for decades [45], [46]. In recent years, Micro-Electro-Mechanical Systems (MEMS) allowed the production of extremely small and lightweight IMUs, while maintaining their original characteristics. Such dramatic changes resulted in new disciplines taking into account the use of IMUs, such as robotics and human motion analysis [44], [47].

IMUs include a triaxial accelerometer and a triaxial gyroscope that are used to measure, respectively, linear accelerations and angular velocities [23]. However, the integration of the measured linear accelerations and angular velocities alone does not allow the estimation of an absolute heading angle [48]. For this reason, IMUs often incorporate an additional sensor, specifically, a triaxial magnetometer. The magnetometer allows to estimate the orientation with respect to an Earth-fixed frame, defined by the orthogonal directions of the gravitational and magnetic fields. To provide an estimation of the sensor's orientation, the measured quantities (linear accelerations, angular velocities, and magnetic fields) are typically fused by means of sensor fusion algorithms. Several algorithms are available in the literature, and they mostly rely on Kalman filters (e.g., [49]–[51]) or complementary filters (e.g., [52]–[54]). It is important to note that the orientation of an IMU is always an estimation resulting from measures taken in different domains. Furthermore, MEMS sensors are often noisy and their measurements include errors that can be grouped into two categories: bias errors (consisting of an unknown zero level) and gain errors (consisting of an unknown scale factor) [24]. For these reasons, IMUs suffer from integration drift phenomena. Small errors in the measurement of accelerations, angular velocities, and magnetic fields can progressively be integrated, resulting in increasing errors in the orientation estimation [55].

Systems based on IMUs, however, have the great advantage of being completely self-contained, so that the measurement entity is constrained neither in motion nor to any specific environment [53]. Moreover, multiple IMUs attached to different body parts of a person can be used to estimate the whole-body motion, based on each sensor's orientation. As in optoelectronic MoCap, the estimation of human motion is based on a musculoskeletal model of the person. In fact, the orientation information alone is not sufficient to describe a person's pose. However, by connecting each IMU to its correct segment in the model, it is possible to extract the required anatomical joint angles. Furthermore, if the estimated orientations are used to perform an IK optimization[1], the effects of drifting on the final results can be strongly reduced.

However, sensor drift is still an open challenge. Thus, capture sessions when using inertial MoCap should remain limited, requiring multiple recalibration of the devices when recording for prolonged periods [56]. To conclude, while such sensors still need to be worn by the user, their placement is much less strict with respect to the markers required by optoelectronic systems and can be placed on top of clothes without sensible drawbacks on the overall system's accuracy.

1.2.3 MARKERLESS SYSTEMS

The next critical advancement in human motion analysis will consist of developing accurate non-invasive systems. An accurate fully-automated, non-invasive, markerless, and sensorless approach would in fact provide a major breakthrough for research and analyses in a vast number of fields, as in medicine, rehabilitation, sports, industry, HRI. While inertial suits are a promising technology capable of achieving results similar to optical MoCap systems [57], they still require the subject to wear dedicated hardware on their body. Markerless MoCap systems, therefore, represent the latest trend aimed at overcoming such limitations.

Initially originated from the fields of CV and machine learning, such technology has recently sparked interest also in the biomechanics com-

[1]Detailed information on IK optimizations can be found in Section 5.3.

munity, mainly due to the leap in the achieved accuracy enabled by data-driven approaches. The ability to measure a person's kinematics without requiring any marker or sensor on the body, in fact, would highly reduce the required preparation time, while at the same time not influencing the recorded motions, both physically and psychologically. One of the biggest uncertainties, when using optoelectronic and inertial systems, lies in the impossibility to quantify how the idea of wearing sensors on the body impacts how the subject will perform the movements [21]. Although markerless MoCap accuracy is not comparable to optoelectronic systems, it still represents a valid alternative for a variety of applications where the application of sensors on the body is simply not feasible (e.g., for clinical analyses of young children, or to assess motion during particularly delicate tasks). In these contexts, the trade-off between accuracy and invasiveness would still lead towards markerless systems.

The estimation of human motion without the aid of any body-mounted sensor or marker has been an intensive research topic for decades. Many systems have been developed through the years (e.g., [58], [59], [60]), as the interest in such a field has gained much attention in the last decade, especially thanks to the development of portable, easy-to-use, and low-cost 3D MoCap systems (e.g., the Microsoft Kinect, Microsoft Corp., Redmond, WA, USA [61]). While such systems vary in the number and typology of cameras used, the number of tracked people, and their real-time performance, a common denominator of markerless MoCap is the ability to estimate the joint locations describing the human body, typically by exploiting deep learning algorithms (e.g., [27], [28], [62]), without the necessity of any marker on the body.

Despite the improvements achieved in recent years, accurate real-time assessment of human motion via markerless motion capture is still an open problem. Common challenges come from background clutters, varying lighting conditions, limited fields of view (FoVs), and occlusions. More importantly, the greatest challenge depends on the general difficulty of tracking the human body, a system characterized by a large number of DoFs and prone to self-occlusions.

For these reasons, although markerless systems can solve the drawbacks typical of optoelectronic and inertial MoCap, they still suffer from several limitations. The number of tracked joints is usually limited. At the same time, existing methods to estimate the 3D human pose from single images are still less accurate than other MoCap systems [29]. Despite efforts to increase the reliability of vision-based markerless systems (e.g., in [63], where the raw estimates of the poses are refined by fitting a hierarchical model of the human body having constrained link sizes that can only slowly vary in time), the accuracy is still one order of magnitude lower than the one achievable with optoelectronic systems ($\sim 6\,\mathrm{cm}$ [64] and $\sim 1\,\mathrm{mm}$ [65] respectively). For these reasons, markerless motion capture is typically preferred in applications that do not require a very precise estimation of human motion (e.g., in human-robot coexistence and human-robot cooperation). However, there is no general consensus on the minimum accuracy required by MoCap systems, as measurement errors can vary significantly depending on the experimental setup, the type of motion, and the human model adopted [3].

Markerless approaches can be divided into two categories: model-based methods, which include different degrees of a priori knowledge, and model-free methods, which do not constrain the raw estimates. Current state-of-the-art approaches for 2D estimation of body poses (i.e., using single images or videos as input) do not use human models [66]. In this context, in fact, the inclusion of a human model can be problematic, due to the nature of the estimated quantities, that are limited to 2D locations of specific body keypoints. On the other hand, 3D approaches tried to constrain the raw estimates by exploiting different typologies of a priori models. They are typically based either on pictorial structures ([67]) or on 3D meshes ([68], [69]).

However, such models are often too simplistic and generic, reducing the number of joints (and, therefore, of DoFs) to reduce the computational complexity of the detection. For this reason, such models cannot be used for accurate movement analyses [21].

1.3 RESEARCH OBJECTIVES

In the latest decades, robotic devices have gained a central role both in the industrial scenario, with the advent of Industry 4.0 first and, more importantly, Industry 5.0, and in everyday life, where exoskeletons are starting to help workers and rehabilitation patients to perform their tasks. However, active collaborations between humans and robots are still limited by the blindness of robotic devices to human behavior.

The biomechanics community has developed, through the years, advanced tools for accurately assessing human motion. However, such systems are typically used for delicate analyses carried out in protected laboratories. Moreover, many advanced techniques are not suited for real-time use, requiring extensive post-processing of pre-recorded motion data [70].

This work aims to take a step forward in the direction of adapting such technologies to everyday living and working environments. The idea is that the system should adjust to the specific user and scenario, and not the opposite. To address this important challenge, my research focused on the development of different algorithms and modules to interface heterogeneous sensors, improve the body pose estimation (BPE) accuracy when exploiting a multi-sensor network, and perform multi-person real-time IK optimizations based on highly accurate musculoskeletal models of the analyzed subjects. All the work in these three macro areas converged on the development of a modular framework, namely *Hi-ROS* (human interaction in ROS[2]), with the final goal of enabling accurate and real-time assessment of multiple people's motion independently of the sensing devices being used and of the environments where the acquisitions are progressed. Part of the framework[3] has already been released as open-source under the Apache v.2 license. Additional modules are planned to be added and released in the near future.

Such a complex research objective required the adoption of a multilevel approach. The idea consists of dividing a large problem (i.e., providing accurate real-time data describing human motion from multiple heterogeneous sensors) into smaller subsystems. Each subsystem is considered a black box, receiving one input and producing one output. Thus, a subsystem does not need to be aware of how its input data was produced, nor of how its output data will be used by other subsystems.

As a result, my work was conceptually divided into three main levels: the *Sensing* level, the *Tracking* level, and the *Modeling* level. Three main characteristics drove the definition of all the modules developed during my Ph.D. within these levels:

[2]ROS [71] (Robot Operating System) is a set of software libraries for the development of advanced distributed robotic systems. Its extensive use within the robotics community made it gain the role of *de-facto* standard in this field.

[3]The code will soon be publicly and freely available under the Apache v.2 license at `github.com/hiros-unipd`

1. the usage of standard interfaces for the communication between different sensors and algorithms;
2. the possibility to perform real-time data fusion among multiple sensors;
3. the inclusion of a musculoskeletal model of the analyzed persons, typically used in biomechanics.

1.3.1 SENSING LEVEL

The *Sensing* level represents the level at which direct measurements of physical quantities are acquired. Since the focus of my work is on providing reliable data describing the body pose of multiple people, such quantities primarily consist of marker positions and IMU orientations. Both quantities can be measured directly (i.e., by exploiting an optoelectronic system or an inertial suit) or estimated (i.e., via markerless MoCap).

The definition of standard interfaces permits to separate the specific sensor, and, to some extent, also the typology of sensors, from the information it is producing. At the same time, they allow to treat every module as a black box, with one input and one output. Thus, each module does not need to know how other modules behave, nor from which module its input data was produced. Moreover, standard interfaces allow to seamlessly swap different typologies of sensors (e.g., markerless estimated body keypoints, optoelectronic markers, IMUs) without requiring a specialization of the *Tracking* and *Modeling* levels. Indeed, both levels support any typology of input data, as well as combinations of data (e.g., it is possible to employ heterogeneous sensing systems and fuse their information). This was achieved by defining an efficient structure that can store both information of multiple people's poses, as well as information of raw sensor data (Section 2.4). Detailed information on the *Sensing* level design and features, together with an in-depth analysis on the most suitable typologies of IMUs for human motion estimation are reported in Chapter 3.

1.3.2 TRACKING LEVEL

The second level defined within my research aims to solve two major issues faced when multiple sensors are used to assess the motion of multiple interacting subjects. First, it ensures the temporal consistency of the estimated motion. The input from multiple sensors might, in fact, refer to different time frames, and communication (and/or computation) delays can indeed alter the order in which distributed data are received and fused. This can result in sensible jitter in the estimated poses and, in the worst-case scenario, in completely wrong estimations of the body poses. Thus, the *Tracking* level contains a module specifically designed to correctly order and track the raw measurements from a generic number of sensors, ensuring temporal consistency of motion. The module also enables robust real-time temporal tracking of multiple people's movements. In this way, it is possible to extract further information describing the motion, such as linear and angular velocities and accelerations of each body joint and link.

The *Tracking* level also allows to fuse body pose data obtained from multi-sensor networks (e.g., from a distributed camera network). In this way, it is possible to enhance the limited information that each sensor can retrieve individually (e.g., partial body poses estimated by each camera in a camera network can be merged in a single, augmented pose).

This level is conceptually divided into four modules:

1. robust frame-by-frame temporal tracking among the detections of each sensor in the network;
2. multi-sensor data merging to retrieve more complete information;
3. global optimization of the estimated poses to ensure consistency of the body dimensions;
4. real-time data smoothing of the estimated motion trajectories.

Detailed information on the *Tracking* level, together with the accuracy improvement achieved using the developed framework on a novel multimodal dataset that we recently acquired[4], are reported in Chapter 4.

1.3.3 Modeling Level

The standard interfaces implemented in the *Sensing* level allow to seamlessly represent heterogeneous sensor measurements. The modules within the *Tracking* level, on the other hand, allow multi-sensor fusion of multiple people's body poses obtained from a network of homogeneous sensors. However, a correct multimodal fusion of the measurements from heterogeneous sensors requires a further level of analysis. In fact, different MoCap systems produce diverse typologies of data, which are typically referred to unrelated underlying models. Thus, a direct fusion of their output would not produce meaningful information.

[4]The dataset, namely *UNIPD-BPE*, will be released under the Creative Commons CC0 license in conjunction with the publication of [72].

To enable the usage of heterogeneous sensors, all the measurements need to refer to a common model of the human. The *Modeling* level is specifically designed to empower the simultaneous use of multiple heterogeneous sensors. In fact, where a full-body inertial suit might be too bulky or expensive, and a large calibrated camera network too difficult to employ, an optimal subset of IMUs and cameras can represent the best trade-off between encumbrance, complexity, and accuracy.

For this reason, a key characteristic of my work is the possibility to use all the supported sensors' data to drive multiple musculoskeletal models (one per analyzed subject), in real-time. The use of a common model to which all data is referred is a key feature to enable the simultaneous usage of heterogeneous sensors. In this way, it is possible to exploit the advantages of multiple sensing systems, minimizing the effect of their disadvantages when used individually.

As for the *Sensing* and *Tracking* levels, also this level is designed to ensure maximum generality. Thus, the integrated musculoskeletal models are fully customizable. The number of body segments, joints, and DoFs can be chosen to comply with the requirements of any application. Moreover, while the core of my work focuses on enabling an active interaction between humans and robotic agents, the generality of the developed modules allows the system to estimate and track the motion of any typology of articulated objects.

The tool chosen for the development of the *Modeling* level is OpenSim [73], an open-source platform for the development of musculoskeletal models and for the simulation of human motion kinematics and dynamics. With almost 70,000 users, it is one of the most widely used software for biomechanical analyses [74]. OpenSim allows to define subject-specific models that include custom sets of markers and, recently, IMUs. However, the proposed IK algorithms used to simulate the body movement are designed for offline analyses of a single subject, expecting input data to be already acquired and properly post-processed before reconstructing the motion.

For this reason, the modules developed within the *Modeling* level required major modifications of the original OpenSim pipeline to enable real-time IK optimizations of multiple people's body poses. The first challenge to be faced required a redefinition of the original workflow used in OpenSim to allow real-time feed of motion data. Second, IK optimizations are typically computationally intensive. In fact, their complexity increases proportionally to the complexity of the model being used (i.e., the number of bones, links, muscles, and constraints included), the number of markers used, and the number of IMUs used. Thus, an efficient multi-threaded architecture was defined to achieve real-time performance, while also enabling multi-person analyses.

Detailed information on OpenSim's characteristics, together with the motivations that led to its selection for the integration of a musculoskeletal model of the human within the proposed framework, are reported in Section 2.2.2. The modifications allowing real-time IK optimizations based on OpenSim's tools and a preliminary test case in which marker positions and IMU orientations are simultaneously used to drive up to four full-body models, on the other hand, are reported in Chapter 5.

1.4 Thesis Organization

The rest of this thesis is organized as follows:

CHAPTER 2 describes the development of *Hi-ROS*, a modular framework for real-time assessment of multiple people's motion, where different typologies of sensors and BPE algorithms can be integrated in a plug-and-play fashion. Despite being the final result of my research, the structure of the developed framework is reported at the beginning of this dissertation since it allows to better clarify the structure of my work. After a brief introduction to the topic (Section 2.1), the tools chosen as a foundation for the framework are presented (Section 2.2). The first tool consists of ROS, a middleware that rapidly gained the role of *de-facto* standard for advanced distributed robotics applications. It provides high-level tools to simplify the development of complex robotics applications requiring efficient online communication among distributed sensors. The second tool is OpenSim, an open-source simulator widely used for biomechanics analyses. OpenSim supports the definition of custom musculoskeletal models and offers advanced APIs for IK and ID optimizations that can be based on marker positions or on IMU orientations. The chapter continues by analyzing the structure of the proposed framework and the role of each of the three levels in which my work was divided (Section 2.3). The custom-defined interfaces used for the communication among each module of the framework are then presented in Section 2.4. Finally, Section 2.5 draws the final remarks and concludes the chapter.

CHAPTER 3 describes the *Sensing* level. After a brief introduction (Section 3.1), a discussion on the supported sensors is presented. They are divided into vision-based MoCap (Section 3.2), where the measured quantities are the positions of a set of markers describing the pose assumed by the person, and inertial MoCap (Section 3.3), where motion is assessed by means of a set of IMUs worn on the body. Subsequently, the development of an efficient open-source driver to interface and synchronize multiple IMUs within *Hi-ROS* (and, consequently, ROS) is analyzed in Section 3.4. Section 3.5 reports an in-depth analysis of the most suited typologies of IMUs for human motion assessment, while Section 3.6 concludes the chapter.

CHAPTER 4 shifts the focus to the work I developed within the *Tracking* level. First, an introduction is proposed describing the main advantages enabled by the modules defined at this level (Section 4.1). Second, a newly acquired dataset that we plan to release in the near future is presented (Section 4.2). The dataset, namely *UNIPD-BPE*, contains synchronized RGB, depth, and inertial data recorded from five Microsoft Azure Kinect cameras [61] and two Xsens MVN Awinda suits (Xsens Technologies, Enschede, Netherlands) [75]. The dataset enables the development and testing of different BPE and tracking algorithms, as well as multimodal sensor fusion approaches, without the necessity of expensive hardware and bulky acquisition setups. Section 4.3 describes in detail the modules allowing for real-time temporal tracking of multiple people using a network of homogeneous sensors. The generality of the developed algorithms allows the user to select any subset of modules. This was made possible by exploiting the same common message structures for the communication between all the modules defined in the *Tracking* level. The same section also includes an in-depth validation of the proposed system by reporting the results obtained on the *UNIPD-BPE* dataset when tracking up to four interacting people. Finally, conclusions are drawn in Section 4.4.

CHAPTER 5 presents a detailed description of the *Modeling* level. After an initial introduction on the importance of employing a common model of the human (Section 5.1), an in-depth description of how a generic musculoskeletal model can be represented in *Hi-ROS* is proposed (Section 5.2). Then, Section 5.3 describes how the IK optimization problem is formalized when using a set of markers or IMUs as input. Section 5.4 presents the modifications required to achieve multi-person IK optimizations and multimodal sensor fusion, while ensuring real-time performance. A preliminary test case is included in which marker positions and IMU orientations are used simultaneously to drive up to four full-body musculoskeletal models. Both the marker positions and IMU orientations are estimated via markerless MoCap, without requiring any physical sensor or marker to be applied to the person's body. This choice allowed to stress the proposed system by including multiple people's body measurements, while effectively simulating the usage of heterogeneous sensors. In fact, by exploiting standard interfaces for the communication between modules, the developed IK solver algorithm is agnostic to the sensing systems used as input. Finally, Section 5.5 concludes the chapter summarizing all the work within this level.

CHAPTER 6 discusses three applications enabled by the *Hi-ROS* framework. The common denominator of these works resides in Industry 5.0 and, specifically, on HRI. After a brief introduction to the topic (Section 6.1), the first application is presented (Section 6.2). It focuses on feedback motion planning in human-robot shared workspaces via deep reinforcement learning (DRL) [76]. In this work, DRL is used to train a collaborative robot to re-plan its motion in real-time, based on the position of the operator. The proposed experiments required real-time knowledge of the operator's pose to guarantee their safety when entering the robot's workspace. In this regard, *Hi-ROS* was used to ensure robust real-time estimation and tracking of the operator during the entire duration of the experiments. The second work consists of the development and validation of a platform for the real-time assessment of workers' ergonomics (Section 6.3). The platform, named *WEM-Platform* (where WEM stands for Worker, Ergonomist, Manager), allows automatic calculation of various ergonomic indexes in real-time. The system is designed to support the use of inertial data, markerless data, or optoelectronic data as input. This was achieved by exploiting *Hi-ROS* for the assessment of the body poses. Section 6.4, finally, presents a system that is currently being implemented in the Logistics Laboratory of the University of Padova. The goal is to create a real-time control loop to give immediate feedback to a manual operator while performing a series of tasks, based on a set of control volumes that are built around specific positions on the workstation. In this regard, the location and pose of the operator are acquired in real-time by combining the feeds of multiple cameras, required to cover the complete working area. Section 6.5, then, concludes the chapter.

CHAPTER 7 draws the conclusions of this dissertation. It summarizes the aims of my research, the possible uses and applications that can benefit from *Hi-ROS*, the open research questions, and the future directions of this work.

2 Hi-ROS: A Modular Framework for Human-Robot Interaction

Part of the work presented in this chapter has been published as a scientific paper [77].
I have made a substantial and principal contribution in the conception and design of these studies, related software development, analyses and interpretations of the results, drafting, and critical revision of the final manuscripts.
Co-authors' permissions for the inclusion of the studies in this dissertation have been obtained.

2.1 Introduction

As described in the previous chapter, the main objective of my Ph.D. focused on the development of different algorithms and modules to interface heterogeneous sensors, enhance the body pose estimation precision when exploiting a multi-sensor network, and perform real-time IK optimizations based on highly accurate musculoskeletal models of the analyzed persons. All the work in these three macro areas resulted in the development of a modular framework, namely *Hi-ROS*, to enable accurate real-time human motion assessments in everyday living and working environments, without the necessity of a dedicated laboratory and complex setups. Thus, the framework offers efficient online communication between all the modules within the three levels defined in Section 1.3, supports the simultaneous use of multiple heterogeneous sensors, and allows the usage of their data to drive an accurate musculoskeletal model of the human, in real-time.

This complex goal could only be achieved by adopting a multidisciplinary approach. Specifically, the design and development of the different modules present in the framework required to borrow and join state-of-the-art tools from both the robotics and the biomechanics communities. The idea was guided by modern sensor fusion approaches where, by fusing information from multiple sensors, it is possible to maximize the measurement accuracy while minimizing the trade-offs. Similarly, by combining state-of-the-art tools developed by different scientific disciplines, analogous results can be achieved.

Efficient real-time communication between multiple distributed systems is a typical challenge of robotic applications. In fact, every robotic system requires interactions between sensors, actuators, and external agents (e.g., other robots or humans). Several robotic middlewares are available, all with the primary goal of enabling easy, robust, accurate, and online communication capabilities within heterogeneous agents. However, while the human is starting to enter the control loop [78]–[80], accurate assessment of human motion, ideally without requiring dedicated hardware or sensors on the body, and in real-time, is still an open challenge.

The biomechanics community, on the other hand, has decades of experience in the analysis of human motion. State-of-the-art tools rely on complex musculoskeletal models of the human body for simulating motion and estimating the internal state of the subject (e.g., the joint angles describing the body pose, the joint torques required to generate the motion, etc.). In this regard, the quality of the analysis depends both on the accuracy of the sensing device, and also on how precisely the model can describe the individual subject. While extensive validations are required for the definition of meaningful anatomical models, this approach allows to perform extremely accurate assessments of human motion, with errors lower than $2°$ in the estimation of the anatomical joint angles [3].

However, the typical workflow used in biomechanics analyses is not designed to comply with online applications. It requires motion data (typically the positions of multiple markers applied on the body captured by means of an optoelectronic system) to be recorded in a dedicated laboratory. Virtual markers are applied to the same locations in the model, and the pre-recorded marker positions are subsequently used in a simulation aiming to drive the model in the configuration that best matches the experimental data acquired at each time frame. Before running the simulation, raw data needs to be properly processed (e.g., by relabeling wrongly estimated markers and filtering noisy measurements). Moreover, the IK optimization used to simulate motion is a computationally intensive procedure. Therefore, such analyses are mostly performed offline.

Three main characteristics permeated the design of the algorithms I developed during my Ph.D., as well as the selection of the most appropriate tools to be used for the development of the modules within each level.

The first characteristic concerns modularity. In fact, one of the most important features to enable human motion assessment in unconstrained environments is the ability to easily integrate different sensors in a plug-and-play fashion. This required the definition of standard interfaces for the communication among all the modules developed within my work. The possibility to freely choose the most appropriate MoCap setup (both concerning the typology of sensors being used, as well as the number of sensors) is, in fact, a key aspect for enabling the system to be adaptable to any use case.

The second requirement lies in the real-time capabilities of the BPE. While offline analyses are important tools in a variety of fields, the modules I developed are even more powerful when used to provide real-time results, thus enabling several typologies of applications where online knowledge of human motion is a necessary requirement. Thus, efficiency is the second feature that guided both the selection of the most suitable tools and the design choices that shaped the development of *Hi-ROS*.

Finally, and more importantly, my Ph.D. research was driven by the strong belief that all the measured quantities need to refer to a common model of the human in order to be meaningful. As reported in Section 1.2, the most accurate biomechanical analyses of human motion rely on a musculoskeletal model of the person to describe their movement. Thus, the measured quantities are not directly used to estimate the motion (i.e., joint angles are not directly calculated from the orientations of the distal and proximal segments computed from the marker positions). Instead, they are used to simulate the motion on the model. The simulation aims to minimize the error between the measured marker positions (or IMU orientations) attached to the body and the corresponding virtual marker positions (or IMU orientations) in the model. This approach was shown to be more accurate with respect to a direct calculation of the pose, minimizing the impact of measurement noise and soft tissue artifacts [81].

The usage of an anatomical model also enables the use of different sensors to simultaneously measure heterogeneous quantities describing the same motion. This is a key feature for enabling multimodal sensor fusion, since all the data refer to the same model. At the same time, the use of heterogeneous sensors can contribute to the overall reduction of the complexity of the system. As an example, real-time assessment of workers' ergonomics is gaining more and more interest in the latest years [82]–[84]. In this context, the usage of optoelectronic MoCap is typically not feasible. However, markerless MoCap and inertial MoCap might also represent challenges. The first system might struggle to correctly estimate the body poses, since factories are typically extremely cluttered environments, resulting in strong occlusions in the FoVs of the cameras. On the other hand, the use of a full-body inertial suit might hinder the worker's movement and be uncomfortable to wear for extended periods of time. The optimal solution in this scenario involves a limited number of cameras to have an initial (possibly partial) estimate of the pose, coupled with a reduced set of IMUs on the most crucial body segments to overcome occlusions and increase the BPE accuracy.

The first two characteristics that drove my work (i.e., modularity and real-time performance) are typical requirements of advanced robotics applications. This fact led to the decision of exploiting a robotics middleware (specifically, ROS) for the communication between all the developed modules. The third characteristic (i.e., the inclusion of a musculoskeletal model of the human), on the other hand, required the usage of state-of-the-art tools and libraries borrowed from the biomechanics community (specifically, OpenSim). The main features of ROS and OpenSim and a detailed analysis of the motivations that drove their choice are reported in Section 2.2. Subsequently, Section 2.3 describes in detail the structure of the *Hi-ROS* framework that, as my research, was divided into three main levels: the *Sensing* level, the *Tracking* level, and the *Modeling* level. In-depth descriptions of each level's design and scope are reported. Finally, Section 2.4 concludes the chapter by analyzing the definition process of the message structures used for the communication between all the modules developed. The structures were designed with generality and efficiency in mind. They allow the storage of information on the body poses of any number of persons, as well as the positions and orientations of a generic set of markers and/or IMUs used for the motion assessment. Despite the defined structures being named *Skeleton Messages*, as their primary usage is to represent the pose of a person as a set of markers connected by links, their generality permits to describe the state of any type of articulated object.

2.2 Selection of the Tools

The proposed framework relies on two main tools: ROS, from the robotics community, and OpenSim, from the biomechanics community. They were chosen to maximize hardware support, flexibility of the developed modules, real-time performance, and achievable accuracy. Such requirements lead to the selection of the ROS middleware to handle the communication among modules and of the OpenSim library for the integration of a musculoskeletal model of the human. This section analyzes the main advantages enabled by the use of ROS (Section 2.2.1) and OpenSim (Section 2.2.2).

2.2.1 ROS – Robot Operating System

The first problem that was tackled during the development of the *Hi-ROS* framework concerned the communication between several typologies of sensors and different BPE and tracking algorithms, while at the same time allowing heterogeneous data to be used to drive a model of the human. To increase the complexity, a further fundamental requirement was to ensure real-time capabilities of the proposed framework. In this regard, the communication between the levels of the framework, but also between the modules within each level, needs to be efficient, reliable, and robust. Network delays, missing or partial data, or sensor failures can, in fact, result in catastrophic consequences. In the context of HRI, a delay in the measurement of the operator's pose can easily lead to dangerous situations in which the robot does not have sufficient time to properly react when needed. Missing or partial data can produce similar results, with the additional criticality of potentially leading to erroneous estimates of the motion. Thus, not only can measurements be delayed in time, but an incorrect estimation of motion might result in the robot performing potentially dangerous actions. Both of these problems are aggravated in the case of sensor or network failure.

Communication is therefore a key aspect to consider. The definition of standard interfaces is a necessary step in the direction of guaranteeing a safe interaction between humans and robotic devices. However, standard interfaces alone are not sufficient to overcome all of the aforementioned challenges. In fact, while standard interfaces are a necessary step to allow efficient exchange of information between nodes, reliability and robustness are not trivial requirements.

These factors led to the decision of exploiting a middleware for handling the communication between nodes. This choice provides several advantages. First, a middleware offers an already validated architecture to handle the communication among the different components of a distributed system, ensuring robustness and reliability. In addition, widely adopted middlewares can rely on the support of large communities of users. Thus, they are more likely to attract the attention of hardware manufacturers, which are more prone to release official drivers to interface their products within the middleware. In this way, the proposed framework gains even more flexibility, allowing the use of different sensing devices and different motion estimation algorithms, while maintaining compatibility with the modules developed in *Hi-ROS*.

Bakken *et al*. [85] defined a middleware as "a layer of software above the operating system but below the application program that provides a common programming abstraction across a distributed system". Middlewares are designed to manage the complexity and heterogeneity of the hardware, improve the quality and efficiency of the developed code, and simplify the development of large distributed systems. The main advantages of exploiting a middleware and, specifically, a robotics middleware, are software modularity, hardware abstraction, platform independence, and, thus, portability of the developed software. Hentout *et al.* [86] defined a list of requirements that an ideal robotics middleware should fulfill:

- *license*: it is preferable for robotics middlewares to be open-source and available free of charge;
- *operating system*: they should be multi-platform to enable their deployment on different operating systems;
- *programming languages*: usage of several programming languages should be supported;
- Usability: middlewares should be easy to use to simplify the development of complex distributed applications;
- *transparency*: middlewares should provide an abstraction layer to hide hardware heterogeneity and complexity;
- *communication*: communication libraries offering synchronous and asynchronous mechanisms should be present;
- *efficiency*: robotics middlewares should allow efficient use of the hardware components to enable real-time processing;

- *documentation*: middlewares should include extensive documentation and tutorials for explaining their use and spur their adoption.

Several robotics middlewares were developed in the latest decades (e.g., CORBA [87], OROCOS [88], Miro [89], Player/Stage [90], RT [91], YARP [92], OpenRAVE [93], ROS [71]), each with slightly different characteristics. Many comparisons have been attempted through the years ([86], [94], [95]), typically focusing on the open-source nature of the projects, the support for distributed architectures, the offered hardware interfaces, the number of included high-level algorithms, the simulation capabilities, and the support for real-time applications.

However, while such information can be fundamental for the choice of the most suitable middleware in the short term, when a project is expected to be carried out over extended periods of time (e.g., several years), different aspects acquire increased importance. A critical factor, in this regard, is the broadness of use of the middleware. In fact, middlewares that can count on a large community of users are more likely to be actively supported in the long term. Moreover, in open-source middlewares, a high number of users and developers directly translates into a high number of developed and validated modules. Similarly, hardware manufacturers will be more prone to release the required drivers for a widely used middleware, rather than for a relatively unadopted one. These factors led to the selection of ROS as the most suitable middleware for the development of the proposed framework.

ROS is an open-source meta-operating system for robotic applications. It fulfills most of the requirements of a middleware, including hardware abstraction, low-level device control, implementation of commonly used algorithms, runtime communication between processes, and package management [96]. The development of ROS began in 2006 at the Kenneth Salisbury's Robotics Laboratory of Stanford University from the personal work of Eric Berger and Keenan Wyrobek. After receiving the first funding from internal university programs, the project raised the interest of Willow Garage, a robotics research laboratory and technology incubator. Willow Garage funded and managed the development of ROS from 2007 to 2013, the year in which Willow Garage was shut down. In the same year, the newly created Open Source Robotics Foundation (which changed its name to Open Robotics in 2017) took the lead in the ROS development. As of today, ROS is the *de-facto* standard for robotics applications, with over 200,000 users worldwide.

ROS can be defined as a peer-to-peer network of concurrent processes. Each component of the network is represented as a ROS *node*. Thus, any process that performs some kind of computation or task is a node in ROS. Multiple nodes can run concurrently on the same machine or be distributed in a network. In ROS, each node is independent, can have different inputs and outputs, and can communicate at runtime with other nodes by passing *messages* which are sent and received through dedicated *topics*. This allows to separate a complex task into a series of simpler subtasks, where each subtask consists of a ROS node or even a series of nodes.

A ROS message can contain any type of data structure. Similar to C++ structs, any number of primitive types (e.g., int, float, bool, etc.) or nested structures (e.g., arrays, but also any custom-defined message) are supported. Messages are sent and received through specific topics based on publish/subscribe semantics. Topics are named buses over which nodes can exchange messages. A single topic can have any number of concurrent publishers and subscribers. In general, nodes do not need to know to whom they are sending messages, or which node published the messages they are receiving. In fact, communication in ROS is designed to decouple the production of information from its consumption. A node will simply publish the produced messages through a topic. At the same time, any other node that requires such data will subscribe to the appropriate topic.

The peer-to-peer nature of ROS requires a mechanism to allow nodes to locate one another. This role is fulfilled by the so-called ROS *master*. The master provides name registration services and lookup of all the active nodes, updated at runtime. Thus, it provides a dynamic allocation of connections. Moreover, the master is responsible for managing a *parameter server*. The parameter server is a shared, multi-variate dictionary that can be accessed by all the nodes via network APIs to store and retrieve parameters at runtime. ROS also offers additional tools (e.g., packages, services, bags). The interested reader is referred to the ROS wiki [96].

2.2.2 OpenSim

As for robotic middlewares, several software platforms are available for analyzing motion through the use of a musculoskeletal model of the human. The most used in the biomechanics field are Anybody [97], BoB [98], SIMM [99], and OpenSim [73]. Many comparisons can be found in the literature ([100]–[103]) with the aim of evaluating the precision of the models proposed by each system, the results of the IK and ID optimizations, and the ease of use. However, the main characteristics that guided the selection of the most suitable platform for the development of the proposed framework were the typologies of supported models, the open-source nature of the project, and the technical implementation of the simulation algorithms. As a result, the final choice fell on OpenSim, a freely available open-source software developed and supported by Stanford University. It allows users to create, share, and analyze musculoskeletal models, supporting both static and dynamic simulations of movement. With almost 70,000 users and 4,000 research papers citations, it is one of the most widely used software for biomechanical analyses [74].

As mentioned above, OpenSim was chosen for multiple reasons. First, it already offers several musculoskeletal models of the human developed by the biomechanics community. Such models can be specifically designed for localized analyses (e.g., for assessing the shoulder RoM, or gait analyses) or describe the full-body (e.g., the model developed by Rajagopal *et al.* [11]). Most importantly, all the proposed models are validated to ensure their correctness. Nevertheless, all OpenSim models can be freely modified if needed. The number of body segments, joints, and DoFs can be altered to meet the requirements of any application.

The second reason is the open-source nature of the project. OpenSim is a free library with almost 70,000 users worldwide. The open-source approach given to the project from its early stages encourages the development of third-party extensions that constantly enable new functionalities. Moreover, the complete source code of OpenSim is publicly and freely available under the Apache v.2 license. This allows to freely dig into the implementation of all the offered tools and to modify parts of the code to comply with particularly challenging requirements. At the same time, open-source software promotes collaboration among users and developers, allowing to identify possible bugs in the code, as well as to share and propose modifications aiming to improve and refine the developed algorithms.

Finally, the last and most technical reason concerns the OpenSim implementation. OpenSim, in fact, is built upon Simbody [104], a well-known library for large-scale mechanical modeling of any system that can be represented as bodies interconnected by joints, acted upon by forces, and restricted by constraints. Both Simbody and OpenSim are entirely written in C++, one of the most efficient high-level programming languages. This is extremely important, especially in the context of employing such libraries for the real-time assessment of human motion. In fact, while the analyses performed using OpenSim are not specifically designed for real-time applications, its architecture makes it a good candidate for usage as a base, with the required modifications, in online applications.

The typical workflow in OpenSim can be divided into five separate steps:

1. *Defining the model.* The first component of any analysis in OpenSim is the model to be used. Although being primarily used for musculoskeletal simulations, OpenSim models can represent any system of rigid bodies connected by frictionless joints that are acted upon by forces to produce motion. In musculoskeletal models, such bodies represent the geometry and inertial properties of all the human body segments. Joints, on the other hand, describe the articulations that connect bodies to form a kinematic chain. A model can also include internal forces from muscles activation and external forces from interaction with the environment. While a large number of already validated models are available within OpenSim, one of the major goals of the project is the possibility to freely create and share with the community any typology of models (e.g., [105], where the author developed a musculoskeletal model of the hindlimb and pes of Deinonychus).
2. *Importing pre-recorded experimental data.* Experimental data used in OpenSim is usually pre-recorded in a clinical laboratory and properly processed before use. Typical data include marker trajectories from optical MoCap, GRF measured using force platforms, and electromyography (EMG) to measure the muscles' activity. The recorded data might need to be converted before usage in specific file formats supported by OpenSim.
3. *Scaling the model.* Scaling is one of the most important steps in biomechanical analyses. Scaling a generic musculoskeletal model means modifying its anthropometry, physical dimensions, and mass properties to match those of the specific subjects who are being analyzed. Precise scaling of the model is a key factor for maximizing the accuracy of IK and ID solvers. This step is necessary since the majority of musculoskeletal models are based on measurements of multiple cadaver specimens, to represent the average human. However, the

scaling step is not required if the adopted model is already subject-specific.

4. *Simulating the movement.* Once data is fed to the system and the model is properly scaled to match the measurements, it is possible to proceed with the simulation of the movement. Inverse methods are used to estimate the state of the model (i.e., joint angles, coordinates, joint moments, muscle activity, etc.) at each time frame. First, an IK optimization allows to estimate joint angles and coordinates of the model. For each time frame of recorded motion, such data is calculated as the coordinate values that position the model in a configuration that best matches the experimental measurements. This is achieved by solving a weighted least-squares (WLS) problem with the goal of minimizing the distance between the experimental markers data and the corresponding marker positions defined in the model. Then, if external forces were measured, it is possible to combine such data with the IK results to estimate the kinetics of a musculoskeletal model. ID optimizations determine the internal forces and joint torques that caused the motion. Therefore, in this case, three actors are required: measurements of the external forces (e.g., by using force platforms), joints accelerations (estimated by double integration of the estimated joint positions), and the mass properties of the bodies in the model.
5. *Analyzing the results.* The last step allows to dig into the details of a simulation. Several analyses are available:
   - *Body Kinematics*. Reports the kinematics state (i.e., positions, orientations, linear and angular velocities, linear and angular accelerations) of specified bodies during the whole simulation.
   - *Point Kinematics*. Reports similar data, but referred to any point defined local to a specific body.
   - *Muscle Analysis*. Reports internal muscle attributes (e.g., fiber length and velocity, active- and passive-fiber force, etc.) during the whole simulation.
   - *Induced Acceleration*. Computes the accelerations caused by specific forces acting on the model, allowing to assess the contribution of individual muscle forces for the motion generation.
   - *Force Reporter*. Reports all the forces acting on the model, both measured during the motion and estimated by the ID optimization.

As is clear from the aforementioned workflow, OpenSim was specifically designed for offline analyses. The motion data used as input needs to already be recorded and properly processed (e.g., by relabeling wrongly estimated markers, or filtering the raw measurements) before running the simulations. For this reason, the original workflow, as well as the implementation of different algorithms, required substantial modifications to enable real-time usage of the system within the proposed framework. Such modifications are described in detail in Section 5.4.

2.3 FRAMEWORK STRUCTURE

All the modules developed within the *Hi-ROS* framework can be conceptually split into the three levels in which I divided my research: the *Sensing* level, the *Tracking* level, and the *Modeling* level. Such levels contain algorithms to allow real-time measurements of heterogeneous quantities, robust multi-sensor fusion and tracking, and modeling of the human. Figure 2.1 shows a schematic representation of the developed framework. Different interactions between levels are possible, as the communication among modules is based on a unique common interface.

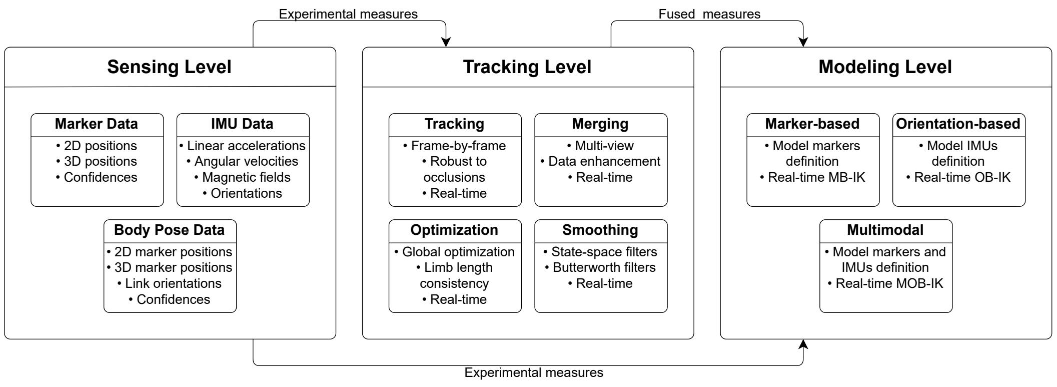


Figure 2.1: Proposed framework structure. All the possible connections between levels, here represented as black arrows, rely on a unique common interface.

The *Sensing* level (Section 2.3.1) can act as input for both the *Tracking* level (Section 2.3.2) and the *Modeling* level (Section 2.3.3). This allows the proposed framework to be extremely flexible. When using multiple homogeneous sensors to record the motion of multiple persons, the *Tracking* level contains powerful modules to perform a multi-sensor fusion of the partial measurements of each sensor and real-time temporal tracking of the estimated body poses. It also allows to increase the pose estimation accuracy, by identifying wrong estimates and merging the partial data measured by each individual sensor. However, different applications might only rely on a single sensor (e.g., a single (red-green-blue-depth) RGB-D camera for estimating a person's pose via markerless MoCap), without requiring any type of data fusion. Thus, the *Modeling* level was designed to support both the direct output of the *Sensing* level and the enhanced data obtained by exploiting the *Tracking* level. In this regard, the modules developed within the *Modeling* level support any typology of input data. Marker positions (either measured by an optoelectronic system or estimated via markerless MoCap) and link orientations (either measured by an inertial system or estimated via markerless MoCap) can be used, in any combination, to drive a common subject-specific musculoskeletal model.

2.3.1 Sensing Level

The *Sensing* level is the foundation of the developed framework. It is defined as the level at which physical quantities are acquired by means of heterogeneous sensors. Thus, it contains the modules required to communicate with different types of hardware. The *Sensing* level is crucial to fulfill, within the proposed framework, one of the main requirements that drove the development of robotics middlewares: hardware transparency. As in robotics middlewares transparency allows to hide hardware heterogeneity and complexity, the *Sensing* level provides an analogous abstraction layer within *Hi-ROS*. Different types of sensors, but also different MoCap systems, seamlessly communicate with the *Tracking* and *Modeling* levels via common standard interfaces.

At the current stage, the proposed framework is designed to support any typology of sensors capable of measuring body kinematics. However, the selected tools allow to easily integrate new typologies of sensors (e.g., force platforms, EMG sensors), if needed. Currently, three main categories of input data are supported: marker data, IMU data, and body pose data (Figure 2.1).

Marker data include position, linear velocity, linear acceleration, and confidence. Positions, velocities, and accelerations can either be 2D or 3D. While 2D information is not sufficient to allow model-based IK optimizations, the choice of supporting 2D data was primarily driven to ensure generality of the framework and to allow usage of other modules that do not strictly require 3D data as input. The confidence field, finally, is used to describe how well a specific marker is being tracked or estimated.

IMU data include orientation, angular velocity, angular acceleration, confidence, and, possibly, information on the distal and proximal markers connected by the link where the IMU is positioned. The described sensor can either be part of an IMU chain or be used individually. Similarly to markers, the confidence field allows to quantify the quality of the orientation estimation.

Finally, body pose data contain information useful for describing a person's motion. It is designed to support body poses described by means of:

- multiple marker positions measured via optical MoCap;
- multiple marker positions estimated via markerless MoCap;
- multiple link orientations measured via inertial MoCap;
- multiple link orientations estimated via markerless MoCap;
- any combination of the aforementioned quantities.

The adoption of standard interfaces allows to treat every module of the framework as a black box, with one input and one output. Thus, any module within the *Tracking* and *Modeling* levels can be used regardless of the input source. As an example, marker-based IK optimizations (MB-IK) can be run independently of the MoCap system being used, the number of markers employed, and the capture frequency. Similarly, orientation-based IK optimizations (OB-IK) do not rely on a specific manufacturer of IMUs, nor on a predefined number of sensors. Finally, standard interfaces allow to seamlessly combine marker positions and IMU orientations in a single enhanced representation of the body pose.

A discussion on the supported sensors within *Hi-ROS*, with a case study aimed at determining the most accurate typologies of IMUs for human motion tracking, are reported in Chapter 3.

2.3.2 TRACKING LEVEL

The second level of this framework aims to fuse multi-sensor measurements while ensuring temporal consistency of the estimated motion. It is designed to take as input multiple people's poses estimated by exploiting a network of homogeneous sensors. No assumptions are made about the number, typology, and synchronization of the sensors being used, the number of people being analyzed, and the algorithm used for the estimation of the body poses. The *Tracking* level includes a series of modules to ensure temporal consistency of the measurements obtained from a distributed network and allow robust real-time multi-people tracking. Moreover, by fusing body pose data obtained from multiple sensors, it is possible to enhance the partial information that each sensor can retrieve individually, detect possible incorrect estimates, and increase the overall system's accuracy.

The *Tracking* level is divided into four modules:

1. robust frame-by-frame temporal tracking among the detections of each sensor in the network;
2. multi-sensor data merging to retrieve more complete information;
3. global optimization of each subject's limb lengths throughout the whole experiment;
4. real-time data smoothing of the estimated motion trajectories.

The temporal tracking module takes as input multiple body poses estimated from homogeneous sensors and assigns a unique ID to each person that is kept in time. It ensures the temporal consistency of all the measurements in the network, as well as of the motion trajectory of each detected person. The merging block, on the other hand, performs a fusion of the poses estimated by different sensors that describe the same motion. That is, if redundant body poses are available, the module is in charge of determining the sensors where the pose is estimated more accurately, discarding possible outliers, and, finally, fusing the clean data. The optimization module allows to perform a global optimization on the estimated poses to ensure consistent dimensions of the body segments throughout the whole acquisition. This node is particularly useful for markerless MoCap systems, where the detected keypoints describing the estimated body poses can lead to varying limb lengths, depending on the quality of the input data and on the precision of the BPE algorithm. Finally, the data smoothing module allows to perform real-time filtering of any typology of data supported by the framework (i.e., marker positions, IMU orientations, but also full-body poses). Both the filter to be used and the desired cutoff frequency can be freely selected by the user, depending on the typology of movements being performed and on the application requirements.

Detailed information on the *Tracking* level, together with the accuracy improvement achieved using *Hi-ROS* on a novel multimodal dataset that we recently acquired, are reported in Chapter 4.

2.3.3 MODELING LEVEL

As discussed in the introduction of this dissertation, a meaningful fusion of heterogeneous data needs to rely on a common model of the human. In fact, since different sensing systems rely on distinct and, possibly, conflicting underlying models, a direct fusion of the measured quantities defining a person's body pose is prone to lead to erroneous results. A key feature of the proposed framework is the possibility to fuse heterogeneous measurements from any number of markers and IMUs for the estimation of human motion. This can be achieved because all measurements refer to a common musculoskeletal model of the human.

The *Modeling* level is specifically designed to enable the simultaneous use of multiple sensors. The inputs of this level can be either raw measurements acquired within the *Sensing* level or the refined results of the *Tracking* level. The goal of the *Modeling* level is to provide accurate estimates of human motion by exploiting state-of-the-art tools borrowed from the biomechanics field, while at the same time ensuring modularity and real-time performance. In this context, modularity is achieved by exploiting the same common structures for representing both the required input data, and the output motion estimated by means of an IK optimization. Real-time capabilities, on the other hand, required sensible modifications of the original OpenSim workflow. While the definition of the model and, to some extent, its scaling, do not invalidate the possibility to perform real-time assessments, the handling of input data and the solver architecture are not suited for online applications.

In fact, experimental data cannot be pre-recorded, but online communication between the sensing system and the IK solver needs to be established. Similarly, smoothing of the raw measured trajectories, as well as robust tracking of the detected poses must be automatized. To achieve that, the *Tracking* level can be used as an intermediate step between the *Sensing* level and the *Modeling* level. Finally, simulation and analysis of the movement require to be optimized to comply both with the online feed of data, and to minimize the delay introduced by the IK computation time to process each frame. This required the development of a multi-threaded architecture, where the input frames (i.e., the data describing the positions of markers and/or orientations of links) are concurrently consumed by multiple threads. In-depth details on the *Modeling* level, as well as on the aforementioned modifications, will be given in Chapter 5.

The developed modules support real-time IK optimizations based on marker positions (MB-IK), IMU orientations (OB-IK), and combinations of markers and IMUs (marker-and-orientation-based IK, MOB-IK). By enabling MOB-IK, sensor fusion is made possible for different combinations of sensory inputs. This allows, as an example, to obtain accurate motion data by exploiting a single camera, coupled with any markerless BPE algorithm, and a limited set of IMUs to overcome possible occlusions. The main advantages of this configuration are the simplicity and cost-effectiveness of the required hardware, while also maximizing the achievable accuracy.

2.4 Definition of the Standard Interfaces

Information describing the pose of multiple persons in *Hi-ROS* is based on the concept of a *skeleton group*. Let a skeleton group ($\mathcal{SG}$) be defined as the set of skeletons in the scene at time t, expressed with respect to a generic reference frame $\mathcal{F}$:

$$\mathcal{SG}_t^{\mathcal{F}} = \{\mathcal{S}_{t,n}^{\mathcal{F}} \mid n \in [0, N[\} \tag{2.1}$$

where N is the total number of persons.

Each skeleton ($\mathcal{S}$) is defined as:

$$\mathcal{S}_{t,n}^{\mathcal{F}} = \{\mathbf{m}_{t,n,p}^{\mathcal{F}},\ \mathbf{l}_{t,n,q}^{\mathcal{F}} \mid p \in [0, P[,\ q \in [0, Q[\} \tag{2.2}$$

It is formed by a set of markers $\mathbf{m}_{t,n,p}^{\mathcal{F}}$ and a set of links $\mathbf{l}_{t,n,q}^{\mathcal{F}}$. P is the total number of markers defining the skeleton, while Q is the total number of links connecting pairs of markers. Each marker $\mathbf{m}_{t,n,p}^{\mathcal{F}}$ can contain information on its position, linear velocity, and linear acceleration expressed with respect to $\mathcal{F}$, as well as a confidence value describing how well it is being detected by the system. Each link $\mathbf{l}_{t,n,q}^{\mathcal{F}}$, on the other hand, can contain information on its orientation, angular velocity, and angular acceleration expressed with respect to $\mathcal{F}$, its confidence (if available), and the IDs of the proximal and distal markers connected by the link. The generality of the definition permits to have markers that are not connected by any link, as well as links that do not connect any marker. This allows to exploit the same interfaces also to represent raw measured quantities (i.e., marker positions and IMU orientations).

All information describing the poses of multiple people is stored in custom-defined ROS message structures. Such messages are used for communication among the different levels of the framework, as well as among submodules. The messages are designed to ensure generality: no assumptions are made on the number of markers and/or IMUs defining a pose, nor on the type of skeleton being used. In fact, the proposed tools are not limited to describing human motion only, but, depending on the BPE algorithm being used, can potentially track any articulated object or animal. Each message includes the 3D positions of the measured/estimated markers defining each persons' body, as well as the orientations of its links (defined as the segments connecting pairs of markers).

Starting at the highest level, the *SkeletonGroup* message contains two fields:

- *header*: header including publication time and reference frame the data refer to;
- *skeletons[]*: vector of *Skeleton*s ($\mathcal{S}$).

Thus, a skeleton group contains all the information describing the poses of multiple persons referred to a single time frame, specified by the *header* field.

The *Skeleton* message contains nine fields:

- *id*: ID of the person (if available);
- *src_time*: timestamp of the input data;
- *src_frame*: reference frame of the input data;
- *max_markers*: maximum number of markers that can be present in the skeleton;
- *max_links*: maximum number of links that can be present in the skeleton;
- *confidence*: confidence of the estimation (if available);
- *bounding_box*: smallest bounding box confining all the markers (if available);
- *markers[]*: vector of *Marker*s ($\mathbf{m}$);
- *links[]*: vector of *Link*s ($\mathbf{l}$).

Each *Skeleton* is composed of a series of markers connected by links. Markers can represent either a person's estimated joint centers or actual markers applied on the body, while links allow to define the body hierarchy. The *src_time* and *src_frame* fields store information on the input source used for the pose estimation (e.g., the source time of the input image and the corresponding camera reference frame, respectively). Finally, the *bounding_box* field allows to store information describing the smallest rectangular parallelepiped containing all the markers.

*Marker*s and *Link*s are defined as follows. The *Marker* message consists of four fields:

- *id*: ID of the marker;
- *name*: name of the marker (if available);
- *confidence*: confidence of the estimation (if available);
- *center*: *KinematicState* of the marker's center.

The *Link* message is similar to the *Marker* message, but also includes information on the parent and child marker IDs that form the link. Therefore, it consists of six fields:

- *id*: ID of the link;
- *name*: name of the link (if available);
- *parent_marker*: ID of the parent marker connected by the link;
- *child_marker*: ID of the child marker connected by the link;
- *confidence*: confidence of the estimation (if available);
- *center*: *KinematicState* of the link's center.

Virtual links can also be used. A virtual link does not necessarily correspond to a human body link, and consists of a pure orientation. Thus, it does not need to connect two specific markers. The advantage of using virtual links allows to seamlessly store raw IMU data without requiring the definition of new message structures.

To conclude, a *KinematicState* is defined as:

- *pose*: position and orientation of the object;
- *velocity*: linear and angular velocities of the object;
- *acceleration*: linear and angular accelerations of the object.

The *KinematicState* can either be used to describe a marker's position (thereby leaving the *orientation* field empty), a link's orientation (thereby leaving the *position* field empty), or a system of reference (using both the *position* and *orientation* fields). Figure 2.2 shows the proposed hierarchy to represent the poses of a group of persons in *Hi-ROS*.

2.5 Conclusions

This chapter described the design and development of *Hi-ROS*, an open-source modular framework for real-time assessment of multiple person's motion supporting different typologies of sensing systems. Despite being the final result of my research, the analysis of the framework's structure allows to better clarify the architecture and the components developed during my Ph.D.

After introducing the problem and the three levels in which I decided to divide my work, the two main tools used for the development of *Hi-ROS*

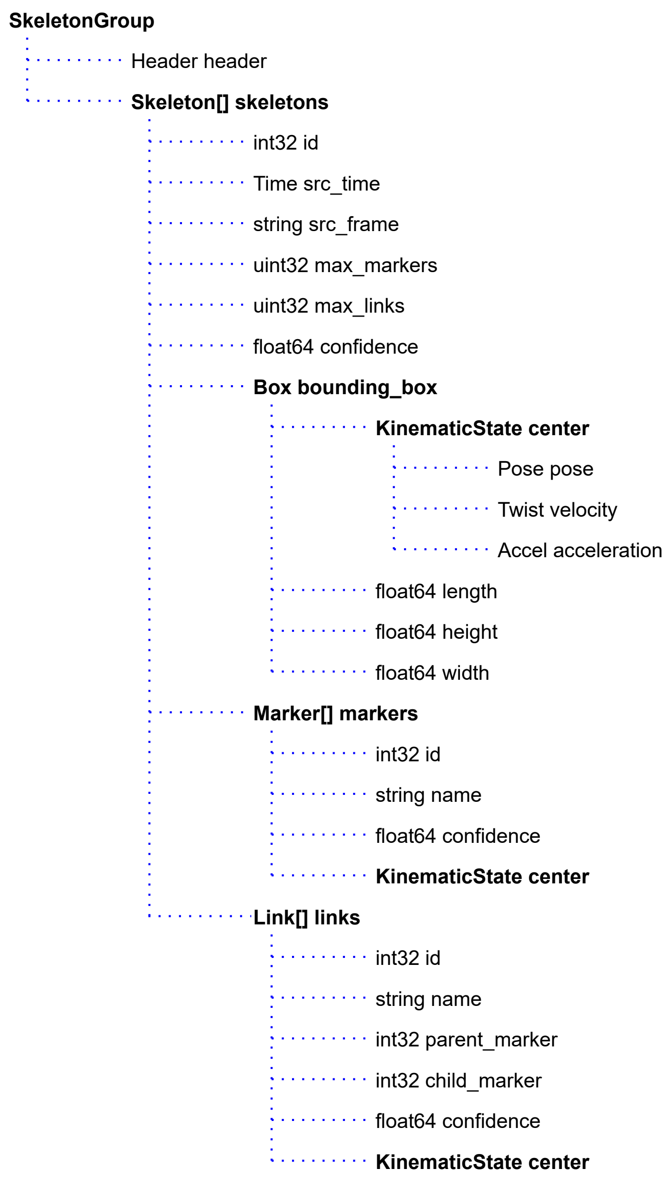


Figure 2.2: *SkeletonGroup* message organization. The picture reports the definition and hierarchy of the custom messages used for the communication between the different nodes of the framework. The bold text represents custom-defined messages, while the plain text indicates standard ROS messages.

were presented. The first tool was ROS, a middleware that is currently considered the *de-facto* standard for distributed robotic applications. In this work, ROS was used to enable efficient, reliable, and robust real-time communication between all the modules defined within each level. Moreover, due to the wide adoption of ROS, a large number of sensors are already supported, either officially or by means of custom-developed drivers. As a result, the proposed framework gains even more flexibility, allowing the use of the developed modules on a variety of sensing devices.

The second tool used in my research was borrowed from the biomechanics community. It consisted of OpenSim, a well-known library for biomechanical analyses. OpenSim includes several already validated musculoskeletal models of the human, as well as efficient libraries to perform accurate IK optimizations to simulate motion. The open-source nature of the project allows to freely modify any model and to dig into each algorithm's implementation. The tools offered by OpenSim were the basis for enabling real-time multi-person IK optimizations fed by the data measured (or estimated) by heterogeneous sensors.

The chapter then continued analyzing the structure of the proposed framework, divided into the *Sensing* level, the *Tracking* level, and the *Modeling* level. First, an introduction on the sensors supported within the *Sensing* level is proposed, together with their different measured/estimated physical quantities. They consist of marker positions (either measured by an optoelectronic system or estimated by markerless MoCap) and IMU orientations (either measured by inertial systems or estimated by markerless MoCap).

The dissertation proceeded by shifting the focus on real-time multi-person motion analysis in distributed networks of homogeneous sensors. The *Tracking* level, in fact, was defined to include all my work on this topic. It consists of four modules allowing to (1) perform robust temporal tracking of the detections obtained by each sensor, (2) merge multi-sensor data to produce an enhanced description of motion, (3) perform a global optimization to ensure consistent limb lengths during the whole acquisition, and (4) filter high-frequency noise of the estimated trajectories. All the computation is in real-time, independently of the number of sensors being used and of the number of people in the scene.

Finally, within the *Modeling* level, I presented my work to enable multi-modal real-time IK optimizations of multiple people's poses. This required several modifications to the typical workflow used in biomechanical analyses. As a result, the developed system is not dependent on the specific hardware used for motion assessment. Moreover, heterogeneous quantities (i.e., marker positions and IMU orientations) can be combined to concurrently drive the motion of a common musculoskeletal model, in real-time. Finally, multiple people can be analyzed simultaneously, either on the same machine or by exploiting a distributed network of PCs.

The chapter concludes with a description of the standard interfaces defined within *Hi-ROS*. They allow to efficiently represent the poses of multiple people, described by a series of markers connected by links. Such structures are designed to describe any typology of skeleton, as well as the raw acquired data. This is a key feature to ensure modularity of the proposed framework and to enable multimodal sensor fusion at the *Modeling* level.

3 Sensing Level

Part of the work presented in this chapter has been published as scientific papers [106], [107].
I have made a substantial and principal contribution in the conception and design of these studies, related software development, analyses and interpretations of the results, drafting, and critical revision of the final manuscripts.
Co-authors' permissions for the inclusion of the studies in this dissertation have been obtained.

3.1 Introduction

The *Sensing* level is the first level defined within my work. Its objective is to provide a bridge between the hardware used to measure the physical quantities describing motion and the algorithms used to enhance the motion estimation accuracy and to drive a musculoskeletal model of the human. As presented in Section 1.2, three typologies of MoCap systems can be used for the estimation of the human pose: optoelectronic systems, inertial systems, and markerless systems. The measured quantities (or estimated, in the case of markerless MoCap) consist of the 3D positions of a set of markers placed on the body and the orientations estimated by a set of IMUs worn by the subject.

It is important to note that the remainder of this dissertation will primarily focus on markerless and inertial MoCap. This choice depends on two main reasons. The first resides in the high-level goal of my Ph.D. The developed framework, in fact, aims to provide accurate motion analysis in everyday living and working environments, while also providing real-time results. In this regard, human motion needs to be assessed without the necessity of complex and cumbersome setups. Thus, optoelectronic MoCap should not be the preferred method, since it typically requires delicate hardware that limits its usage to dedicated confined laboratories.

Second, while optoelectronic and markerless systems technologies are certainly different, the quantities they measure (or estimate) are semantically identical. In fact, they both describe the pose of a person as a set of 3D marker positions placed in specific parts of the body. Thus, all the work on markerless MoCap developed in this thesis can easily be used also in contexts adopting optoelectronic systems.

The remainder of this chapter describes the main characteristics of the typologies of sensing systems supported by the proposed framework. They are divided into vision-based MoCap (Section 3.2), with a focus on the different technologies that allow the acquisition of RGB-D data) and inertial MoCap (Section 3.3), with a focus on the design and development of an open-source driver to allow real-time streaming and synchronization of data from multiple IMUs (Section 3.4). Finally, Section 3.5 analyzes the characteristics of three IMUs belonging to different market categories, with the aim of assessing to what extent they can be used for human motion assessment. In-depth results on their static and dynamic performance, as well as the impact of drifting phenomena, are reported. The frequency and amplitude values imposed for the dynamic assessments were specifically selected to provide good coverage in the bandwidth characterizing the majority of human movements.

3.2 OPTICAL MOTION CAPTURE

Optical MoCap systems estimate the human pose by relying on data captured from one or multiple cameras. As analyzed in Section 1.2, optical motion capture can be divided into two categories: optoelectronic systems, which measure the 3D positions of a set of retroreflective markers via triangulation, and markerless systems, which estimate the positions of multiple body keypoints without requiring any marker or sensor worn on the body.

Since the first require delicate hardware, long setup times, and specialized personnel, their use is typically confined to dedicated laboratories. For this reason, the rest of this section will focus on markerless MoCap and, specifically, on the different typologies of cameras that can be used for single- and multi-camera BPE and tracking.

The cameras used in markerless MoCap can be divided into two main families, depending on whether depth information is produced or not. RGB (red-green-blue) cameras provide a visualization of the scene by including color information for each pixel of an image. Depth cameras, on the other hand, produce a depth map of the scene describing the distance between each pixel and the camera's focal point. When a camera is capable of sensing both RGB and depth, it is referred to as RGB-D camera. Three main families of technologies are typically used to generate depth information: stereo vision (Section 3.2.1), structured light (Section 3.2.2), and time of flight, usually referred to as ToF (Section 3.2.3).

3.2.1 Stereo Vision

The driving principle of stereo vision mimics the stereoscopic setup of human vision to perceive depth [108], [109]. Stereo vision is used to infer sparse or dense depth maps of the scene by identifying matching pixels in the images acquired by two or more cameras that capture the same scene from slightly different angles. Knowing the relative pose of each camera with respect to the others and the disparity of the pixels in the images, it is possible to infer the 3D depth of each pixel from the 2D positions using standard triangulation techniques [110]. Stereo vision systems intrinsically provide RGB images, thus belonging to the family of RGB-D cameras.

Passive systems rely on the detection of a set of common features in the images. The main advantage of these systems is the possibility of being deployed both indoors and outdoors. However, passive stereo vision suffers from unreliable depth estimations in regions that suffer from a low number of detected features. In fact, automatic estimation and measurement of accurate inter-image correspondences are complex tasks. The limited measurement accuracy for the depth estimation, coupled with the computationally intensive signal processing required, hinder the usability of stereo vision systems in real-time setups [111]. As a result, the aforementioned difficulties led to the development of active stereo vision systems.

Active systems employ a light source (typically in the IR domain) to project pseudo-random patterns on the scene. This allows to increase the number of detectable features and, thus, the accuracy in the depth estimation. The second advantage of active systems relies on their robustness to varying lighting conditions. Finally, the randomness of the projected patterns allows the use of multi-camera setups, since different patterns do not interfere with each other [112].

3.2.2 Structured Light

Structured light is a different technology that also belongs to the family of active methods [113]. Structured light cameras, in fact, infer depth information by projecting a known IR light pattern (typically dots or stripes) on the scene. Differently from active stereo vision, however, in this case, a single camera is needed to sense the scene. The depth information is estimated based on the variation of known feature points in the sensed pattern projected on the scene. The camera, whose pose with respect to the IR projector must be known, calculates the difference between the original projected pattern and the distorted pattern observed. This allows to reconstruct the 3D coordinates of the detected feature points, using a method similar to the one used in stereo vision techniques [114].

Common challenges of these systems come from transparent and highly reflective objects. While the projected pattern is not distorted by transparent objects, thus not allowing a measurement of the distance, the opposite result is obtained with highly reflective bodies. In the last case, in fact, the pattern is excessively distorted, producing wrong estimates in the depth estimation. Another problem arises when using multiple cameras with overlapping FoVs. When the projected pattern is emitted at similar wavelengths, multiple patterns can compete with each other, causing interference. One possible solution requires hardware synchronization of all the cameras in the network, allowing to introduce a fixed controlled delay between each camera's projector.

Despite the aforementioned limitations, these cameras allow to achieve high accuracy, especially when the distance between the objects and the camera is not excessive. However, because of the strong components in the IR domain of sunlight, structured light sensors do not permit outdoor usage.

3.2.3 Time of Flight

Finally, ToF sensors calculate the distance between a point and the camera by measuring the phase delay in the IR light reflected from the objects in the scene [115]–[117]. This technique requires an emitter, used to send the light, and a receiver, to measure the time in which the light is reflected back. Typically, they are combined in a single device.

The light coming from the emitter is diverged to illuminate the whole FoV, and the time of flight of the reflected light is measured using a 2D array of photodiodes combined with time-to-digital converters or with time-to-amplitude circuitry [118]. Then, each pixel's depth value is estimated by measuring the phase shift between the incident light and the reflected light. The range of a ToF camera can be calculated as $D = c/2f$, where D is the depth, f is the modulation frequency, and c is the speed of light [119].

As in structured light systems, light is typically emitted in the IR spectrum. Thus, the same limitations with respect to outdoors usability apply. Furthermore, like structured light systems, ToF cameras are sensitive to interference from other cameras emitting at a similar wavelength. The advantages of these systems are the high achievable accuracy, and the possibility to retrieve depth information from surfaces with little to no textures.

As described in Section 2.2, the communication between several typologies of sensors and all the algorithms developed within my Ph.D. relies on ROS. ROS represents the data captured by any typology of cameras (either RGB or RGB-D, and independently of the technology used for the depth estimation) using a unique standard message structure. This allows to seamlessly swap hardware without requiring any modification on the software that relies on its measurements. At the same time, this enables the setup of camera networks using different brands and typologies of sensors.

Furthermore, ROS is widely adopted in the robotics community. For this reason, ROS drivers to interface a large number of cameras are freely available (either developed by manufacturers or by third-party developers)[1]. In this way, the algorithms developed within my Ph.D. gain even more flexibility, allowing the use of multiple different sensing devices, as well as different BPE algorithms, while maintaining compatibility with the modules defined within *Hi-ROS*. Figure 3.1 shows three RGB-D cameras supported in ROS and, consequently, in *Hi-ROS*, each exploiting different technologies for the estimation of the depth map.

An in-depth analysis of the work developed on markerless MoCap and, specifically, on the real-time fusion and tracking of multiple people's poses obtained by exploiting a generic network of homogeneous sensors,

[1]A (non-comprehensive) list of supported sensors is reported in [120].

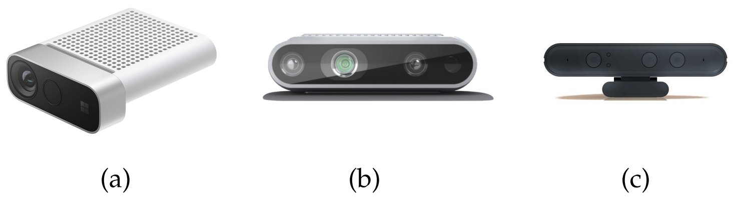


Figure 3.1: Microsoft Azure Kinect [61] (a), Intel RealSense D435 [121] (b), Orbbec Astra Mini [122] (c) RGB-D cameras. The first exploits a ToF sensor for the depth estimation, the second active IR stereo vision, and the third structured light.

is presented in Section 4.3 of this dissertation.

3.3 Inertial Motion Capture

When the line of sight between the sensor and the human cannot be ensured, or when motion needs to be captured in large or outdoor spaces, IMU-based wearable sensing is considered the most promising solution [22]. IMUs are devices able to detect motion continuously and are suited for online human motion tracking, requiring neither invasive sensors nor constrained workspaces.

An IMU consists of a triaxial accelerometer and a triaxial gyroscope used to measure linear accelerations, including gravity, and angular velocities, with respect to a predefined and rigidly associated local frame [23]. IMUs may also integrate an extra triaxial magnetometer to obtain an absolute estimate of the heading angle. The 3D orientation of the IMU with respect to a global coordinate system can be estimated through sensor fusion of the sensors' measurements, by exploiting well-known state-of-the-art algorithms, such as complementary filters [123] or Madgwick's filter [53].

IMU-based systems have the advantage of being completely self-contained and independent of artificially generated sources. The measurement entity, in fact, is unconstrained either in motion or in environmental characteristics [53]. Moreover, the recent integration of onboard MEMS sensors introduced potential advantages in terms of cost, size, weight, and energy consumption [124].

MEMS IMUs are small, energy-efficient, and low-cost. Unfortunately, low-cost MEMS are usually noisy and their measurements include errors that can be grouped into two categories: bias errors (consisting of an unknown zero level) and gain errors (consisting of an unknown scale factor) [24]. Despite the fact that a large number of recent works aim at minimizing such effects (e.g., [125]–[127]), completely drift-free orientation estimation is still an open problem. Nonetheless, IMUs are becoming a portable and cost-effective alternative to optical motion capture systems, and are currently being introduced also in clinical settings for functional movement quality assessing trials. Inertial-based systems are also very attractive for online motion analyses, being able to reach high update rates (up to hundreds of Hz) with limited computational resources.

IMUs firmly attached to a human body segment can provide an estimate of its kinematics, either directly [128] or through global optimization processes, such as IK [129]. In rehabilitation environments, inertial systems have been used to capture real-world knee RoM for total knee arthroplasty patients. The method proposed in [130] represents a great step forward in the monitoring of the patients by continuously examining the RoM of the knee, using two IMUs rigidly attached to the leg. Finally, in [131] a self-developed low-cost IMU was used to implement a motion analysis system applied in a clinical protocol for gait analysis. The accuracy, consistency, and repeatability were validated by exploiting a robotic system and a motion capture system as a ground truth.

While the aforementioned works focused on the analysis of specific joint angles, full-body inertial MoCap requires the usage of several IMUs attached to each body segment. The estimated orientations are then used as input for an IK optimization to estimate the pose of the analyzed person [57].

However, the usage of multiple (and, possibly, wireless) sensors introduces additional challenges. In fact, the measurements obtained by the sensors might refer to different time frames, and communication (and/or computation) delays can indeed alter the order in which distributed data are received and combined. Therefore, accurate synchronization procedures are required to ensure that each time frame contains consistent data from all the sensors required by the MoCap system.

3.4 AN EFFICIENT OPEN-SOURCE DRIVER FOR XSENS WIRELESS INERTIAL MEASUREMENT UNIT SYSTEMS

One of the most widely adopted IMU systems, both for its reliability and accuracy, is the MTw Awinda wireless human motion tracker by Xsens. Figure 3.2 shows an MTw Awinda tracker with its default local coordinate reference system.

The built-in 3D orientation estimation filter is specifically tuned for the range of frequencies typical to human motion and can provide online estimates of the sensor's orientation up to 120 Hz. Data from multiple trackers connected to the same master PC are time-synchronized within 10 µs [132]. Typical applications where the MTw Awinda can be used are ergonomics, rehabilitation, biomedical analyses, virtual reality, human-machine interaction, and robotics.

Such applications, however, are increasingly demanding for real-time and, possibly, open-source solutions to capture human motion. To this aim, this section describes the development of an efficient open-source C++ driver to interface and synchronize multiple Xsens MTw Awinda trackers with ROS. The driver[2] has been released as open-source and free of charge under the Apache v.2 license.

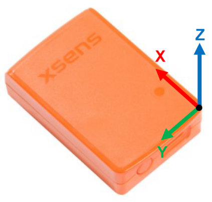


Figure 3.2: Xsens MTw Awinda and its default local coordinate reference system (source: [132]).

Although other implementations of a ROS driver for the Xsens MTw Awinda are available on GitHub ([133], [134]), they all suffer from major limitations. Specifically, the first one depends on legacy software from Xsens that is no longer available, while the second is a mere variation of one of Xsens SDK examples, adapted to publish ROS messages. Moreover, the driver requires Ubuntu 16.04 LTS and ROS Kinetic, which are both 6 years

[2]The code is publicly and freely available under the Apache v.2 license at `github.com/hiros-unipd/xsens_mtw_wrapper`

old at the time of writing, and incorrectly assumes that the time a packet is received corresponds to the actual time the packet's orientation refers to. As we will analyze in Section 3.4.1.4, this assumption introduces a series of problems when dealing with the Xsens MTw Awinda IMUs.

The driver proposed in this work was developed to solve all the aforementioned limitations. In fact, both ROS Melodic (Ubuntu 18.04 LTS) and ROS Noetic (Ubuntu 20.04 LTS) are supported. The driver supports the maximum number of trackers allowed by the Xsens SDK (i.e., 20 trackers) connected to the same master PC, and allows to directly stream, through one or multiple configurable ROS topics, the data obtained from each sensor (raw accelerations, angular velocities, magnetic fields, and estimated orientations) up to 120 Hz.

Furthermore, the absolute timestamp is calculated for each packet, which is a feature not directly available within the Xsens SDK. The messages sent through the network are based on ROS standard messages and comply with the ROS conventions [135]. This allows the developed driver to directly interface with any ROS package that supports ROS standard messages.

Moreover, a synchronization procedure is implemented to guarantee that no data from a single tracker are missing from each time frame. This is required when the data are used to compute human motion exploiting an IK procedure.

By interfacing the MTw Awinda trackers and streaming the sensor readings to the ROS network, this driver pushes forward the development of applications and tools for a variety of human-robot interaction tasks where online knowledge of the human pose is crucial.

3.4.1 System Design

The proposed driver consists of four main classes:

- *WirelessMasterCallback*: manage the connection between the trackers and the wireless master connected to the PC (Section 3.4.1.1);
- *MtwCallback*: manage the callbacks of each MTw Awinda tracker (Section 3.4.1.2);
- *Synchronizer*: synchronize data among different trackers to avoid time frames with missing packets (Section 3.4.1.3);
- *Wrapper*: the actual driver implementation (Section 3.4.1.4).

Figure 3.3 shows the Unified Modeling Language (UML) diagram describing the proposed driver.

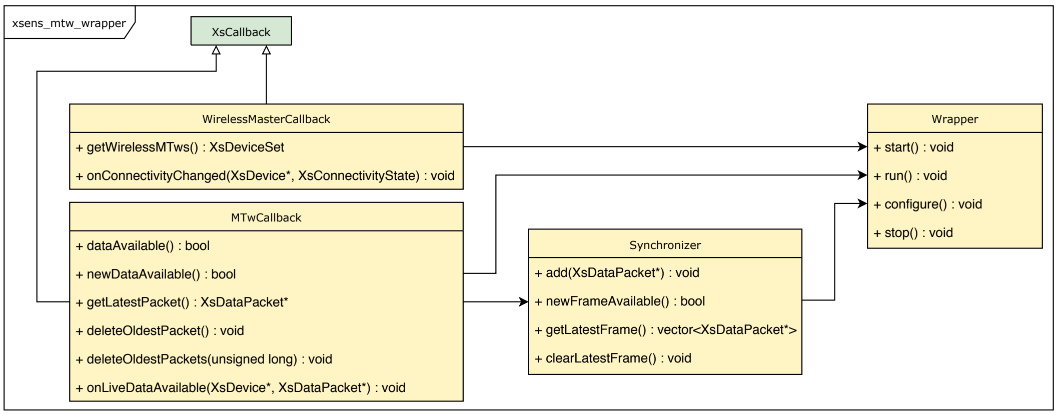

xsens_mtw_wrapper
XsCallback
WirelessMasterCallback
+ getWirelessMTws() : XsDeviceSet
+ onConnectivityChanged(XsDevice*, XsConnectivityState) : void
MTwCallback
+ dataAvailable() : bool
+ newDataAvailable() : bool
+ getLatestPacket() : XsDataPacket*
+ deleteOldestPacket() : void
+ deleteOldestPackets(unsigned long) : void
+ onLiveDataAvailable(XsDevice*, XsDataPacket*) : void
Synchronizer
+ add(XsDataPacket*) : void
+ newFrameAvailable() : bool
+ getLatestFrame() : vector<XsDataPacket*>
+ clearLatestFrame() : void
Wrapper
+ start() : void
+ run() : void
+ configure() : void
+ stop() : void

Figure 3.3: UML diagram of the developed driver. The green color indicates that the class is part of the Xsens SDK.

3.4.1.1 *WirelessMasterCallback*

Class *WirelessMasterCallback* inherits from Xsens' *XsCallback* class, which is used to manage the callbacks of both the wireless master and the single trackers. It manages the connection and disconnection of trackers on the master receiver and allows to retrieve the list of connected trackers.

3.4.1.2 *MtwCallback*

Class *MtwCallback* also inherits from Xsens' *XsCallback* class. It manages a circular buffer for each tracker which is filled each time a new packet is available. The maximum size is limited to 300 packets. This means that at the lowest possible frequency ($40\,\mathrm{Hz}$) the buffer can store up to $7.5\,\mathrm{s}$ of data, while at the maximum possible frequency ($120\,\mathrm{Hz}$) the buffer can store up to $2.5\,\mathrm{s}$ of data.

In normal conditions, the buffer is never filled, since the data are automatically deleted after being consumed by the *Synchronizer*, as explained in detail in the next section. If an anomalous increase in the number of packets in the buffer occurs, the condition is promptly reported to the user. This is a symptom of connection issues between the sensor and the master receiver and should be avoided by the user by acting on the connection parameters of the device.

3.4.1.3 Synchronizer

The Xsens communication protocol ensures highly accurate time-synchronized data sampling (within $10\,\mu s$) in all the trackers connected to the same master. However, some packets can get lost during the transmission.

As an example, assume that two trackers are connected to the same master: mtw_0 and mtw_1. The following situation can occur: mtw_0 provides packets relative to frames 100, 101, 102, 103, while mtw_1 provides packets relative to frames 100 and 103. In this case, we have two partial frames (101 and 102) that only contain data from tracker mtw_0. Partial frames are potentially destructive for tracking algorithms and might cause inconsistencies and side effects. Therefore, a policy to handle them should always be selected by the user depending on its application requirements.

For instance, if the user is interested in performing an IK optimization on the data gathered from the IMUs, most of the available tools do not work if partial frames occur. While in offline applications such frames can be easily handled by removing them or filling the missing data during the post-processing phase, in online applications this is not feasible. For this reason, an online synchronization procedure must be implemented to ensure that each time frame contains data from all the trackers.

Class *Synchronizer* is responsible for avoiding partial frames. Currently, the driver supports two different synchronization policies: *fillPartialFrames* and *skipPartialFrames*.

Policy *fillPartialFrames* allows to fill the missing packets by estimating their values using linear interpolation. Taking the same example as before as a reference, with this policy, the output would be:

- frame 100: {mtw_0: pkt_{100}, mtw_1: pkt_{100}};
- frame 101: {mtw_0: pkt_{101}, mtw_1: $\hat{pkt}_{101}$};
- frame 102: {mtw_0: pkt_{102}, mtw_1: $\hat{pkt}_{102}$};
- frame 103: {mtw_0: pkt_{103}, mtw_1: pkt_{103}}.

Notice that frames 101 and 102 contain packets pkt_{101} and pkt_{102} relative to mtw_0 and the interpolated values $\hat{pkt}_{101}$ and $\hat{pkt}_{102}$ computed from packets pkt_{100} and pkt_{103} relative to mtw_1, since the real packets are missing. The missing packets' orientations are calculated by exploiting spherical linear interpolation, while data from accelerometer, gyroscope, and magnetometer use linear interpolation.

On the other hand, policy *skipPartialFrames* removes the frames where one or more packets are missing. Taking the same example as before as a reference, with this policy the output would be:

- frame 100: {mtw_0: pkt_{100}, mtw_1: pkt_{100}};
- frame 103: {mtw_0: pkt_{103}, mtw_1: pkt_{103}}.

Notice that frames 101 and 102 are skipped, because the corresponding packets from mtw_1 are missing.

The two synchronization policies have different trade-offs. Policy *fillPartialFrames* allows to obtain higher update rates, close to the nominal one. The number of skipped frames is minimized, but some frames can contain data that do not correspond to the real measurements. This synchronization policy should be used when a constant update rate is required, or when the movements to be analyzed are not excessively fast.

Policy *skipPartialFrames* guarantees that the data from each published frame are the most recent one, as well as that the data of each tracker in the same time frame refer to exactly that frame. However, in this case, the actual frequency can be slightly lower than the nominal one. Each time a tracker loses one packet, the full time-frame (i.e., the data from all the other trackers referring to that frame) is in fact discarded. This synchronization policy should be used when a constant update rate is not strictly required, while it is crucial to only have the measured data in each time-frame.

The choice of which synchronization policy should be used is up to the user. If there is no interest in synchronizing data among the trackers, the synchronization can be turned off. Further details on other parameters that the user can modify are given in Section 3.4.1.5.

3.4.1.4 Wrapper

Class *Wrapper* implements the connection, synchronization, and communication between the trackers and ROS. It configures both the wrapper's settings and the trackers' settings, tries to connect to all the available trackers, and performs an initial synchronization between the first packets of each tracker.

The connection of the MTw trackers to the master consists of the following steps:

1. construct an *XsControl* object required to manage the connection between the wireless master and the PC;
2. scan all the USB ports on the PC and try to find an attached wireless master;
3. open the port where the master has been found;
4. construct an *XsDevice* object that refers to the wireless master;
5. enable the configuration mode on the master;
6. attach a *WirelessMasterCallback* object to the master;
7. set the desired update rate;
8. set the desired radio channel.

Once the master has been configured, a callback handler for each available tracker is attached to the master in order to be able to retrieve the data packets. It is now possible to enter measurement mode, where each tracker begins to stream its packets.

Since the Xsens MTw Awinda trackers include a triaxial magnetometer, their orientation is estimated taking into account the magnetometer readings, which might be noisy in environments affected by magnetic disturbances. To make the trackers behave like pure IMUs, it is possible to reset their initial orientation. It is worth noticing that the onboard filter that estimates the orientation will still use the magnetometer, since Xsens does not give the possibility to modify any parameter of the filter.

Finally, before starting to publish the desired messages through ROS topics, the initial packets from each tracker are synchronized. This procedure is necessary to assign the correct absolute timestamp to each packet. In fact, the MTw Awinda trackers, unlike other Xsens trackers (e.g., the MTi series [136]) do not provide an absolute timestamp with each received packet. It is possible to calculate the relative timestamp between two packets by knowing the update rate and the packet IDs, but direct information on the absolute timestamps is not available. Furthermore, the Xsens communication protocol tries to minimize the number of packet losses, but it does not guarantee that packets arrive at a constant rate.

For example, let us have an update rate of $100\,\mathrm{Hz}$, which corresponds to a $\delta t = 10\,\mathrm{ms}$ between two consecutive packets. The following situation is typical: a train of n packets arrives at the same time from one tracker, then wait for $n \cdot \delta t\ \mathrm{s}$, then the next n packets arrive. By knowing the initial packet ID and its absolute timestamp, it is possible to assign the correct absolute timestamps to all the other packets. However, this information is not directly available. Since Xsens does not provide official information on the delay between the actual reading and the arrival of the packet on the master, we assume that, if n packets arrive at the same time t', t' will correspond to the absolute timestamp relative to the latest packet.

The synchronization procedure stores the initial packets from each connected tracker, finds the latest packet ID from each train of packets, and assigns the correct timestamp to it. From this point on, the absolute timestamp of each packet is calculated with the following formula:

$$t_i = T_0 + (pkt_i - PKT_0)/f_N \tag{3.1}$$

where t_i is the absolute timestamp of packet i with packet ID pkt_i, T_0 is the absolute timestamp of the initial packet with packet ID PKT_0, and f_N is the nominal update rate selected for the sensors.

Notice that T_0 and PKT_0 are the same among all sensors. This is valid because the Xsens protocol guarantees that the same packet ID among different sensors connected to the same master is related to the same time-frame. With this information, it is now possible to start streaming each sensor's data through the ROS network.

The driver loops through each connected tracker and finds if a new data packet is available. If the user chose not to synchronize data between trackers, the absolute timestamp is assigned to the packet, and it is published immediately. If the user chose to synchronize data between trackers, when a new packet arrives, it is added to the *Synchronizer*'s buffer. When the *Synchronizer* identifies the presence of a full time-frame, the data relative to the frame are published. The loop is repeated until a stopping request is received, and an appropriate shutdown procedure is executed.

The current version of the driver is based on the latest Xsens SDK version supporting the MTw Awinda trackers (i.e., Xsens MT SDK v. 4.6.0) and is freely available at `github.com/hiros-unipd/xsens_mtw_wrapper`.

3.4.1.5 Parameters

Table 3.1 contains all the parameters that the user can set by modifying file *launch/custom_configuration_example.launch*.

Table 3.1: List of parameters that the user can set in the proposed driver.

Parameter	Description
`xsens_mtw_node_required`	Set if the other ROS nodes on the PC should be killed when the driver is killed
`node_name`	Set the name of the ROS node
`tf_prefix`	Set a prefix to avoid conflicts in the ROS transforms
`desired_update_rate`	Set the desired update rate
`desired_radio_channel`	Set the desired radio channel
`reset_initial_orientation`	Set if the initial orientation should be reset (IMU-like behavior) or not (MIMU behavior)
`enable_custom_labeling`	Set if custom labels should be assigned to the trackers. The custom labels can be defined in file *config/sensor_labels.yaml*
`synchronize`	Set if data should be synchronized
`sync_policy`	Set the synchronization policy (`fillPartialFrames` or `skipPartialFrames`)
`publish_mimu_array`	Set if a single topic containing all the sensor readings should be published or if a series of topics for each sensor should be published
`publish_imu`	Set if the IMU data (accelerometer, magnetometer, and orientation as quaternion) should be published
`publish_mag`	Set if the magnetometer data should be published
`publish_euler`	Set if the orientation as Euler angles (roll, pitch, yaw) should be published
`publish_free_acceleration`	Set if the free acceleration should be published
`publish_pressure`	Set if the pressure should be published
`publish_tf`	Set if the orientation as ROS transform should be published

3.4.2 Experiments

A series of experiments have been conducted to assess the performance of the developed driver. Specifically, the focus has been on the evaluation of the computational time required to process each packet, and on the real update rate that can be achieved, which depends on the number of lost packets. The driver has been tested on an Intel Core i7-7500U CPU @ 2.70 GHz laptop with 8 GB of RAM.

3.4.2.1 *Analysis of the Computational Time*

The average computational time required to process one packet has been evaluated by varying the number of connected trackers, the update rate, and the synchronization policy. The total time required to process a single packet is the sum of the times required to acquire the packet from the tracker, publish the packet, delete the packet from the buffer, and, if a synchronization policy is chosen, synchronize the data among the trackers:

$$t_{tot} = t_{acq} + t_{pub} + t_{del}\ (+\ t_{sync}) \tag{3.2}$$

The number of connected trackers and the desired update rate do not affect the average time required to process one packet. The only relevant parameter, in this case, is the synchronization policy. The results are reported in Table 3.2.

Table 3.2: Computational times required to process one packet with the implemented synchronization policies. The last row reports the ratio between the total time required to process one packet and the average time between two consecutive packets in the worst-case scenario (120 Hz update rate). Data are presented as mean (SD).

Policy	No Synchronization	Skip Partial Frames	Fill Partial Frames
Acquire	1.90 (0.27) µs	2.10 (0.22) µs	2.15 (0.36) µs
Publish	6.57 (0.93) µs	6.48 (1.22) µs	6.43 (1.07) µs
Delete	1.27 (0.13) µs	1.27 (0.12) µs	2.13 (0.14) µs
Synchronize	-	0.27 (0.05) µs	0.63 (0.09) µs
Total	*9.74 (1.26) µs*	*10.12 (1.48) µs*	*11.34 (1.46) µs*
(%)	*0.117 (0.015) %*	*0.121 (0.018) %*	*0.136 (0.018) %*

We can notice how the average times to acquire and publish one packet do not depend on the synchronization policy and are roughly equal to 2 µs and 6.5 µs respectively. On the other hand, the average time required to delete one packet from the buffer is higher when using the *fillPartialFrames* policy (2.1 µs versus 1.3 µs). This can be explained by the fact that the management of the *Synchronizer*'s internal buffer is more complex with the *fillPartialFrames* policy. The deletion of a packet is not as straightforward as with the other policy or when no policy is set. Some packets, in fact, cannot be deleted after being published if the next packet is missing. This check determines an increase in the average time required to delete a packet.

The average time required to synchronize the data between packets is also higher when the *fillPartialFrames* policy is set ($0.6\,\mu s$ versus $0.3\,\mu s$). In general, the overhead required to synchronize the data is approximately $0.3\,\mu s$ when using the *skipPartialFrames* policy, and $1.6\,\mu s$ when using the *fillPartialFrames* policy. This can be explained by the fact that the first policy only has to check if the latest time frame contains data from all the trackers and, if any packet is missing, it can free its buffer. On the other hand, the latter policy must store more data in its buffer, since it needs to perform an interpolation based on the older packets when a missing one is detected. In the worst-case scenario (that is, with policy *fillPartialFrames*), the average time required to process one packet is approximately $11.3\,\mu s$.

The last row of Table 3.2 shows the ratio between the total time required to process one packet and the average time between two consecutive packets when the update rate is set to $120\,\mathrm{Hz}$, which corresponds to a δt of $8.33\,\mathrm{ms}$. We can see that the computational time required by the driver, even when the most computationally intensive synchronization procedure is set, is negligible ($0.14\,\%$ of the δt).

3.4.2.2 Analysis of the Lost Packets

The necessity of a synchronization procedure comes from the fact that some packets can get lost during the communication with the master. The cause of the lost packets does not depend on the driver and appears to be related to the distance between the trackers and the master.

To assess the impact of the lost packets on the performance of the tracker, we varied the number of trackers, the update rate, the synchronization policy, and the distance between the trackers and the master receiver. The trackers have been kept still at some defined distances ($0.1\,\mathrm{m}$, $2\,\mathrm{m}$, $4\,\mathrm{m}$), without occlusions between the trackers and the master.

The number of connected trackers, the desired update rate, and the synchronization policy does not affect the average number of lost packets. In this case, the only relevant parameter is the distance between the trackers and the master receiver. The results are reported in Table 3.3.

We can see how the relative distance between the trackers and the master receiver affects the number of lost packets. When the trackers are close to the master receiver, the number of lost packets is close to 0. When the dis-

Table 3.3: Number of lost packets at different distances between the trackers and the master PC.

Distance	**Lost Packets**	**Received Packets**	**% Lost**
0.1 m	10	22767	0.04 %
2 m	474	22722	2.04 %
4 m	477	22736	2.05 %

tance increases above 2 m a plateau can be observed, where approximately 2 % of the packets are lost.

Three scenarios are possible, depending on the synchronization policy set. Without a synchronization policy, some trackers might be publishing at their nominal rate, while other trackers might be publishing at a slower rate, due to the number of packets that are lost in the communication with the master. In this case, some time frames will have missing data. By averaging the update rates of all the trackers, the result should be close to 98 % of the nominal rate. If the *skipPartialFrames* policy is selected, the overall update rate will not be higher than the update rate of the slowest tracker, but all the published data will refer to the same time frame, without any partial frame. Finally, if the *fillPartialFrames* policy is selected, the overall update rate will be close to the nominal one, even when some packets are lost, by estimating missing data when necessary. In this case, to have a decrease in the update rate, all the trackers connected to the master should miss the data relative to the same time frame.

3.4.3 FINAL REMARKS

This section presented an efficient open-source C++ driver to interface the Xsens MTw Awinda trackers with ROS. The key features of the developed driver are:

1. Possibility to publish data up to 120 Hz through a single ROS topic containing the readings from all the trackers, or through multiple ROS topics for each sensor;
2. Assignment of an accurate absolute timestamp to each packet, which is a feature not available within the Xsens SDK;
3. Online availability of both the raw accelerometers, gyroscopes, and magnetometers data, as well as each tracker's estimated orientation;

4. Integration of two different synchronization policies, which are needed to avoid partial frames, minimize the number of lost packets, and guarantee that the data from different trackers are correctly synchronized.

This last point is necessary for applications that require a constant update rate, or a strict time synchronization among the trackers. The synchronization policy can be set by the user and has a negligible impact on the performance of the system. In fact, the overhead of the worst-case scenario (that is, with policy *fillPartialFrames*) is equal to $1.6\,\mu s$, with the total time required to process a single packet of $11.3\,\mu s$. Using the ROS middleware, the proposed driver enables a variety of applications where online knowledge of the human pose is required, with a focus on HRI applications, where ROS is the *de-facto* standard.

Experiments have been performed to assess the performance of the system, both with respect to the computational time required to process each packet, and with respect to the number of packets lost during the wireless communication with the master. The results show that the driver is efficient (the total time required to process a single packet corresponds to approximately $0.14\,\%$ of the time between two consecutive packets) and the packet losses negligible ($2\,\%$ of the packets are lost at a distance of $4\,\mathrm{m}$ from the master receiver).

Furthermore, the extensive evaluation of the performance of Xsens MTw Awinda IMUs reported in [107] and based on the driver proposed in this section, confirmed the effectiveness of the developed solution to augment the sensors' provided information with reliable timestamps.

3.5 On the Accuracy of IMUs for Human Motion Tracking

As stated in the previous sections, IMUs are nowadays becoming a portable and cost-effective alternative to optical motion capture systems, and are currently being introduced also in clinical settings for functional movement quality assessing trials.

However, despite the increased interest in the topic, it is still quite difficult to understand the performance quality and limitations of many available mass-market IMUs. It is not uncommon to deal with poorly documented technical specifications, especially in terms of static/dynamic accuracy of the proprietary filters or the integrated sensor characteristics. Moreover, the testing setup and working conditions are not always clearly detailed. Consequently, identifying which IMU device is the most suitable for a specific application is non-trivial.

The authors of [137], for example, implemented a new extended Kalman filter-based algorithm demonstrating better accuracy under dynamic conditions than Xsens' proprietary filter, which is optimized to track movements in a human-compatible range. The performance of the proposed method has been evaluated using an optoelectronic motion capture system as ground truth.

In [138], the authors analyze the performance of another IMU, the MbientLab MetaMotionR, with respect to a Vicon motion capture system, using a motorized gimbal to produce some predefined movements.

In this section, a study focused on the analysis of the performance of three different well-established mass-market IMUs is reported. All the IMUs used are comparable in terms of size and weight, although their market value belongs to different price ranges: the ultra-low-cost InvenSense MPU-9250 ($\sim$10€), the low-mid-cost MbientLab MetaMotionR ($\sim$90€), and the high-cost Xsens MTw Awinda ($\sim$400€). The first IMU only provides the raw data measured from the integrated sensors (triaxial gyroscope, accelerometer, and magnetometer), while the other two also integrate proprietary filters for orientation estimation.

To assess the performances of both raw data and orientation estimates, Madgwick's filter is used as an unbiased means of comparison. The experimental protocol follows the one proposed in [128], employing a direct drive servomotor, instead of a robotic manipulator, to move a custom-made 3D printed socket where the IMUs are firmly attached. This allows simulating dynamic movements with frequencies and amplitudes compatible with human motion, evaluating orientation tracking errors under a broad set of controlled and repeatable conditions. It is useful to note that this robotic approach is tailored to highlight the performance of the IMU devices, since there is no source of error except for the ones affecting the servomotor control, which are found to be negligible.

3.5.1 MATERIALS

In this section, a brief description of each IMU used in the experiments is provided, focusing on the technological and constructional specifications of both integrated sensors and orientation estimation algorithms. Figure 3.4

shows the IMUs used in the experiments with their default local coordinate reference systems, while the most relevant specifications are resumed in Table 3.4.

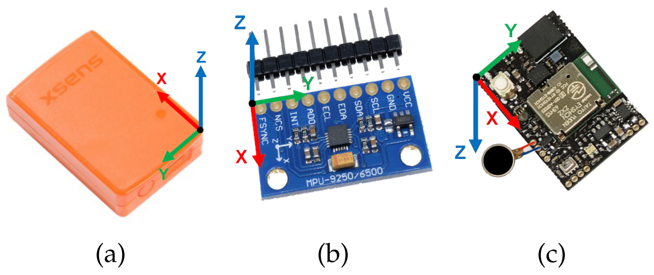


Figure 3.4: Xsens MTw Awinda (a), InvenSense MPU-9250 (b), MbientLab MetaMotionR (c) and their default local coordinate reference systems.

Table 3.4: Specifications of the IMUs used in the experiments (sources: [132], [139], [140]).

Specifications	**MTw Awinda**	**MPU-9250**	**MetaMotionR**
Interface	Wireless 2.4 GHz	I^2C - SPI	Bluetooth LTE 2.4 GHz
Maximum update rate	120 Hz	100 kHz - 20 MHz	100 Hz
Price/Unit [€]	∼400	∼10	∼90
Power supply	LiPo battery	2.4-3.6 V cabled pinout	Li-ion battery
Acc. measurement range [g]	±16	±2–16	±2–16
Gyro. measurement range [°/s]	±2000	±250–2000	±125–2000
Mag. measurement range [μT]	±190	±4800	±1300
Acc. nonlinearity [%FS]	0.5	0.5	0.5
Gyro. nonlinearity [%FS]	0.1	0.1	0.1
Mag. nonlinearity [%FS]	0.1	N/A	1
Acc. rate noise spectral density [$\mu g/\sqrt{\text{Hz}}$]	200	300	300
Gyro. rate noise spectral density [$°/s/\sqrt{\text{Hz}}$]	0.01	0.01	0.007
Mag. rate noise spectral density [$\mu T/\sqrt{\text{Hz}}$]	0.02	N/A	N/A

3.5.1.1 Xsens MTw Awinda

MTw Overview. The MTw (2016) is a miniature wireless IMU incorporating a 3D accelerometer, a 3D gyroscope, a 3D magnetometer, and a barometer. An embedded processor handles sampling, buffering, calibration, integration, and wireless data transmission. Data transmission is implemented through a patented radio protocol, achieving time synchronization of up to 20 MTws across the wireless network within $10\,\mu s$. The communication is managed by the Awinda Master station, receiving synchronized data at up to $120\,Hz$. The raw sensor measurements are internally calibrated and sampled at $1\,kHz$, exploiting Strap Down Integration (SDI) for orientation and velocity increments computation. Specific information on integrated sensors is not provided, except for those listed in Table 3.4.

Proprietary Filter. Xsens' 3 DoF Kalman Filter for human motion (*XKF3hm*) is used to estimate the 3D orientation of the IMU in real-time. 3D inertial data and measured magnetic field are efficiently fused directly on-board. The continuous integration using SDI captures movements that are short-term accurate, high-bandwidth, and, due to carefully made assumptions on the motion dynamics and the sensor characteristics, provides drift-free orientation estimates. Moreover, long-term accelerations may cause performance degradation with respect to roll/pitch angles. The manufacturers claim that the *XKF3hm* provides the most accurate heading estimate available, thanks to the accurate dead-reckoning provided by the SDI and the implementation of practical application knowledge.

Summarizing, considering roll/pitch static and dynamic accuracies, values of $0.5°$ and $0.75°$ RMS are stated, while $1°$ and $1.5°$ RMS for heading angles. Although the stated performances are remarkable, the user is provided with little to no information regarding operational conditions or bandwidth ranges where these values can be considered reliable.

3.5.1.2 InvenSense MPU-9250

MPU-9250 Overview. Three 16-bit analog-to-digital converters are available to digitize the output of each sensor. At the time the product was developed, the manufacturers stated that it was the world's smallest 9-axis motion tracking device.

Each sensor features user-programmable full-scale ranges (Table 3.4) to adjust performances to different applications' characteristics. The communication with the registers can be performed using both I^{2}C and SPI protocols, the latter being suited for fast applications given its maximum operating speed of $20\,\mathrm{MHz}$.

An important feature is the presence of an embedded Digital Motion Processor, which can acquire and process inertial and magnetometer data from the sensors. The device integrates neither wireless communication features nor batteries for power supply, therefore, it needs to be cabled to a voltage source (2.4-$3.6\,\mathrm{V}$). The IMU is, however, designed to be low-power-consuming, as it runs with a $3.5\,\mathrm{mA}$ operating current when all 9 motion sensing axes are enabled.

Drift issues are strongly reduced by the minimal cross-axis sensitivity between the sensors. The device does not integrate a filter for orientation estimation, and no information on the accuracy achievable when employing a specific algorithm is available. Therefore, in this work, only Madgwick's filter will be used, on the raw sensor data, to compare the achievable accuracy with the other IMUs.

3.5.1.3 MbientLab MetaMotionR

MetaMotionR Overview. The MetaMotionR (MMR) (2016) is a wearable IMU device that includes a BMI160 (3-axis, 16-bit digital accelerometer and gyroscope), and a BMM150 (3-axis geomagnetic sensor), along with temperature, barometer, and luminosity sensors. It relies on the Bosch 9-axis sensor fusion software for orientation estimation.

Everything is integrated on a PCB, powered by a $100\,\mathrm{mA\,h}$ lithium-ion $3.7\,\mathrm{V}$ battery. The board uses a Bluetooth Low Energy communication protocol, able to stream data up to $100\,\mathrm{Hz}$. Both BMI160 and BMM150 feature user-programmable full-scale ranges (Table 3.4). In full operation mode, the low power consumption is typically $925\,\mu\mathrm{A}$ for the BMI160 and $0.5\,\mathrm{mA}$ for the BMM150.

Proprietary Filter. There are four available onboard sensor fusion algorithms to estimate IMU orientation. For fair comparison and analysis, the *NDOF* (9 DoFs) fusion mode was chosen for the experiments, where the absolute orientation is calculated from accelerometer, gyroscope, and magnetometer data, with a maximum transmission rate of $100\,\mathrm{Hz}$.

The algorithm was designed for human motion tracking; therefore, long accelerations over long periods of time can cause its performance to deteriorate. It automatically performs a background calibration of all sensors, although the manufacturers suggest periodically performing a manual recalibration. Pitch and roll angle drifts are compensated by sensing the Earth's gravity, and heading drift is compensated by sensing the Earth's magnetic field. Moreover, when the device is in motion, the algorithm ignores the accelerometer data and relies only on the gyroscope for the estimation of the pitch and roll angles. Likewise, if the magnetometer data are detected to be distorted, they will be ignored.

The manufacturers claim that the bias stability is close to the maximum stability allowed for a consumer electronic device, considering the low cost of the device. Lastly, the sensor fusion algorithm can automatically detect if the device is standing still, stopping the integration of the gyroscope data to prevent drift. As for the MTws, the user is not provided with detailed information on dynamical operational conditions, bandwidth limitations, and calibration accuracies.

3.5.2 EXPERIMENTS

3.5.2.1 Experimental Setup

The experimental setup used for this study is shown in Figure 3.5. It consists of the set of commercial IMUs introduced in Section 3.5.1, a custom-made 3D printed socket where the IMUs are firmly attached, and a direct drive servomotor (SGMCS-02BDC41, Yaskawa). The motor drive (Servopack SGDV-2R8A01B, Yaskawa) is interfaced to a National Instruments DAQ board for motor control and encoder reading. The servomotor is controlled via Simulink, with an accurately tuned PID controller, capable of following the provided reference signal with a maximum error of 0.1°.

The IMUs are interfaced with ROS using internally developed C++/Python real-time drivers (i.e., [106] for Xsens IMUs, unpublished for the others). These allow to directly stream, on single or multiple configurable ROS topics, the data obtained from each sensor (raw accelerations, angular velocities, magnetic fields, and estimated orientations) up to 120 Hz. Furthermore, the absolute timestamp is calculated for each packet received from the IMUs, to retrieve consistent synchronized measurements. All the software is run on a single desktop PC (Alienware Aurora R8, Dell) with Ubuntu 18.04 LTS for data acquisition and processing.

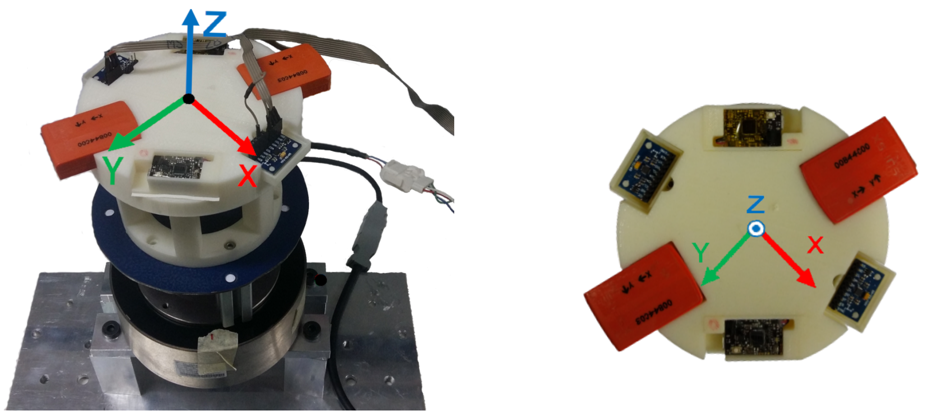


Figure 3.5: Experimental setup. (a) Complete setup including direct drive servomotor, customized socket, and IMUs used for data collection, (b) top view.

3.5.2.2 Experimental Protocol

The experimental protocol follows the one proposed in [128]. It consists of static and dynamic validations, precisely under stationary conditions and during sinusoidal rotations around the following axes, defined with respect to a global coordinate reference system (Figure 3.5):

- $\overrightarrow{a_1} = [0,\ 0,\ 1]^T$ (horizontal)
- $\overrightarrow{a_2} = [1,\ 0,\ 0]^T$ (vertical)
- $\overrightarrow{a_3} = [\sqrt{2}/2,\ 0,\ \sqrt{2}/2]^T$ (45°)

These configurations allow to assess IMU performances, respectively, during attitude, heading, and mixed contemporary attitude and heading movements. The static performance test comprises a pure static evaluation, in which the set of IMUs is kept stationary for $10\,\mathrm{min}$ to test the performance of both proprietary and Madgwick's filters in terms of divergence from a still position.

Regarding the dynamic evaluation, a set of sinusoidal movements was generated combining 7 different frequencies (0.18, 0.32, 0.56, 1, 1.78, 3.16 and $5.62\,\mathrm{Hz}$) and 5 different amplitudes (± 3, ± 6, ± 9, ± 12 and ± 18°), for a total of 35 experiments. Both the frequency and the amplitude values were selected to provide good coverage in the bandwidth characterizing the majority of human movements [128]. Each experiment consists of a

$40\,\mathrm{s}$ sinusoidal movement, of which the first and last $10\,\mathrm{s}$ are exponentially smoothed to avoid abrupt movements. The experiment then ends with $20\,\mathrm{s}$ of rest period after the movement is completed. This set of 35 experiments is repeated 5 times for each axis configuration, to ensure dataset consistency.

3.5.3 RESULTS

Before performing the experiments, each IMU sensor was calibrated following the procedures suggested by the manufacturers, to achieve optimal performance with the proprietary filters, and to collect consistent measurements. Moreover, Madgwick's filter was tuned accordingly, ensuring the minimal orientation error on all IMU data. Since no significant differences were observed for the results of the 3 configurations defined in Section 3.5.2.1, the discussion focuses on the $45°$ configuration, reflecting all the other cases.

The metric chosen to quantify the accuracy of each IMU is the orientation error in the quaternion space, defined as:

$$\boldsymbol{\Phi}(\boldsymbol{q}_A, \boldsymbol{q}_B) = 2 \arccos\left(\boldsymbol{q}_A \cdot \boldsymbol{q}_B\right) \tag{3.3}$$

where $\boldsymbol{\Phi}$ defines the length of the shortest path (i.e., a geodesic) connecting the two quaternions $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ on the 4-dimensional hyper-sphere where they are defined.

To fairly compare the performance over the whole set of experiments, for each test we computed the RMSE as $\sqrt{\sum_{t=1}^{N} \boldsymbol{\Phi}(\boldsymbol{q}_{exp,t}, \boldsymbol{q}_{gt,t})^2 / N}$, where $\boldsymbol{q}_{exp,t}$ and $\boldsymbol{q}_{gt,t}$ represent the orientation obtained by each IMU and the ground truth orientation, at time t, respectively. We then normalized such value with respect to the semi-amplitude of the sine movement and finally expressed it as a percentage (P-RMSE).

The obtained P-RMSE values are reported in Figure 3.6 and Figure 3.7, while the RMS errors were averaged over the whole set of experiments in Table 3.5. Experiments with a P-RMSE greater than $50\,\%$ were considered unacceptable and thus discarded, concluding that tracking could not be achieved under these conditions.

Static Accuracy. To validate the static accuracy, the IMUs were kept still for $10\,\mathrm{min}$ with the servomotor turned off, to avoid any source of distur-

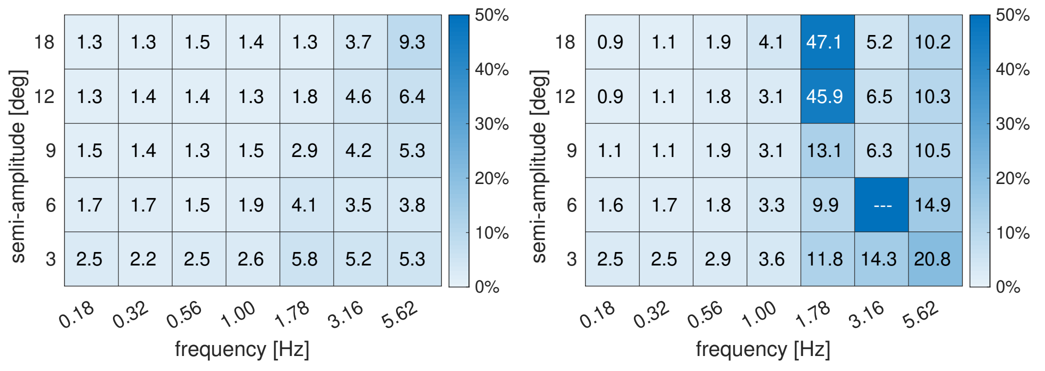


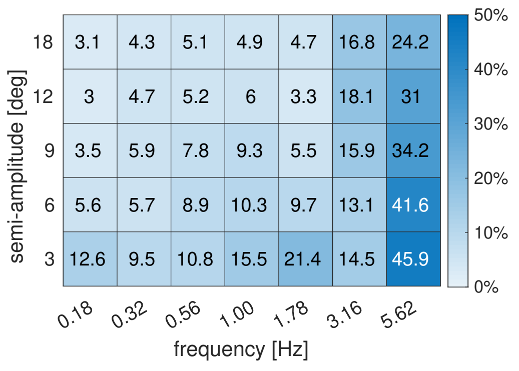
Figure 3.6: Percentual RMS orientation errors (P-RMSE) for the $45°$ configuration, averaged among the different trials of each sine amplitude-frequency combination. Proprietary filters of (a) MTw and (b) MMR. P-RMSE values higher than $50\,\%$ are discarded.

Table 3.5: Average orientation errors (RMSE $\pm$ SD) for each setup configuration (°).

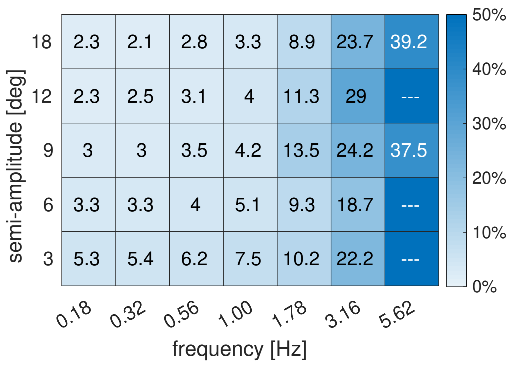
IMU	**Experiment**	**Proprietary Filter**	**Madgwick's Filter**
MMR	Horizontal	0.47 ± 0.22	0.83 ± 0.39
	Vertical	0.97 ± 0.46	0.90 ± 0.42
	45°	0.83 ± 0.28	1.04 ± 0.49
	Overall	*0.76 ± 0.32*	*0.92 ± 0.44*
MPU	Horizontal	—	1.68 ± 0.73
	Vertical	—	1.64 ± 0.70
	45°	—	1.01 ± 0.41
	Overall	—	*1.44 ± 0.62*
MTw	Horizontal	0.27 ± 0.09	1.29 ± 0.49
	Vertical	0.25 ± 0.09	1.12 ± 0.45
	45°	0.26 ± 0.09	1.05 ± 0.39
	Overall	*0.26 ± 0.09*	*1.16 ± 0.44*

bance. The experiments started after the orientation transitory ended and convergence to a constant value was achieved.

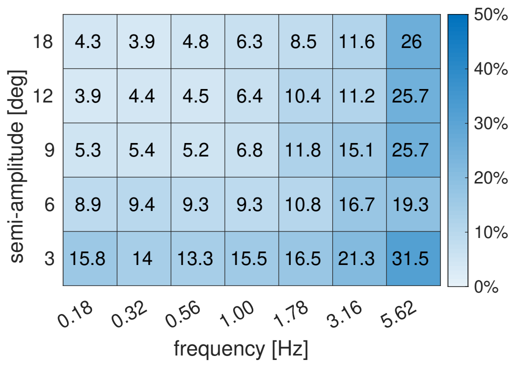
The drift values computed under these conditions highlight negligible drifts for all the considered filters, with magnitudes always lower than $2.2 \times 10^{-4}\,°/\mathrm{s}$. No significant differences can be observed between Madgwick's filters (MMR drift = $1.99 \times 10^{-4}\,°/\mathrm{s}$, MPU drift = $1.87 \times 10^{-4}\,°/\mathrm{s}$, MTw drift



(a)



(b)



(c)

Figure 3.7: Percentual RMS orientation errors (P-RMSE) for the $45°$ configuration, averaged among the different trials of each sine amplitude-frequency combination. Madgwick's filter on (a) MTw, (b) MMR and (c) MPU. P-RMSE values higher than $50\,\%$ are discarded.

= $1.88 \times 10^{-4}\,°/\mathrm{s}$), showing that the sensor measurements of all IMUs are consistent and reliable.

Regarding the proprietary filters, we observed the lowest drift on the MMR ($3.57 \times 10^{-5}\,°/\mathrm{s}$), since the MMR filter detects still positions and stops integrating the gyroscope measurements, keeping the orientation locked until movement is recognized. In this way, all the typical sources of errors that cause orientation estimates to drift are avoided. To conclude, the MTw's drift is $2.16 \times 10^{-4}\,°/\mathrm{s}$, a value comparable to the ones obtained through Madgwick's filter.

Dynamic Accuracy. The results obtained during the dynamic accuracy assessment were constantly affected by drift, motivating the authors to perform further experiments to identify the cause of such phenomena. The performed investigation characterized the drift as constant in the entire set of experiments, caused by the vibrations of the controlled motor, having a frequency content that is way out of the experiments' range (over $50\,\mathrm{Hz}$ with an assessment bandwidth lower than $6\,\mathrm{Hz}$). These findings motivated the choice of compensating the drift, being caused by external sources due to the characteristics of the experimental setup.

The ground of truth is directly computed from the encoder readings by compensating the initial offset between each IMU's coordinate system and the socket's one. For each experiment, the P-RMSE is computed on the $20\,\mathrm{s}$ of pure sinusoidal motion.

As can be seen in Figure 3.6 and Figure 3.7, the MTw proprietary filter provides the best results in dynamic conditions, with a P-RMSE always lower than $10\,\%$, demonstrating to be the most adequate for high-precision human motion tracking. The computed average drift of $2.61 \times 10^{-4}\,{}^{\circ}/\mathrm{s}$, which is almost equal to the one under static conditions, highlights their robustness to external disturbances and vibrations.

The MMR proprietary filter estimates turned out to be deeply affected by drift in the dynamic experiments, with a drift in the heading estimates of $\sim 0.7\,{}^{\circ}/\mathrm{min}$. A possible explanation for this behavior could be found in the documentation, where the manufacturers state that if the magnetometer readings are detected to be distorted, they are ignored and the estimate relies only on the gyroscope and accelerometer. This hypothesis is further confirmed by the much lower drift affecting orientations estimated by Madgwick's filter on the same raw data. This solution is nevertheless suitable for relatively slow movement applications (e.g., in motion rehabilitation).

Furthermore, a significant performance drop can be observed, independently from the adopted configuration, in the following experiments: $\langle 1.78\,\mathrm{Hz},\ 12^{\circ}\rangle$, $\langle 1.78\,\mathrm{Hz},\ 18^{\circ}\rangle$, and $\langle 3.16\,\mathrm{Hz},\ 6^{\circ}\rangle$. The authors explain this drop as a consequence of an incorrect internal estimate of the gravity vector, caused by measured free-body linear accelerations comparable with gravity ones. In the subsequent experiments, where accelerations are even higher, probably the internal filter disables the accelerometers, causing a lighter performance degradation than under the conditions mentioned above (Figure 3.6).

Taking into account the estimates provided by the Madgwick's filter applied to the raw data measured from the three IMUs (Figure 3.7), no significant differences can be observed for motion frequencies below $1\,\mathrm{Hz}$. Moreover, in this range of frequencies, no one outperforms the proprietary filters.

For motion frequencies above $1\,\mathrm{Hz}$, the MPU performs slightly better than its competitors but, again, the differences are minimal. At the highest frequency ($5.62\,\mathrm{Hz}$) the P-RMSE is always higher than $20\,\%$, but this lack of accuracy could be due to the low sampling rate of the raw data made available from the IMUs. Indeed, while internally raw data are sampled and used to estimate the orientation at high frequency, the same data are made available at much lower rates (e.g., for the MTw the internal sampling rate is $1\,\mathrm{kHz}$, while raw data are made available externally only at a maximum rate of $120\,\mathrm{Hz}$).

Finally, drift assessment under dynamic conditions highlights that the MTw and the MMR outperform the MPU (drifting respectively of $5.61 \times 10^{-4}\,°/\mathrm{s}$, $6.62 \times 10^{-4}\,°/\mathrm{s}$, and $2.93 \times 10^{-3}\,°/\mathrm{s}$). Moreover, while for the latter the drift is one order of magnitude higher than under static conditions, for MTw and MMR it is almost the same, showing poor robustness to vibrations of the MPU.

3.5.4 FINAL REMARKS

This work investigated the potentialities and limits of three IMUs, belonging to different market segments, for human motion tracking applications. In particular, in increasing order of cost, InvenSense MPU-9250, MbientLab MetaMotionR, and Xsens MTw Awinda.

A direct driver servomotor moves, following a sinusoidal reference, a custom-made 3D printed socket where the IMUs are firmly attached, ensuring repeatability and accuracy. Amplitudes and frequencies of the sinusoidal motion are varied within a range compatible with human motion characteristics. Orientation estimates are compared with the encoder-provided ground truth in terms of both absolute and percentual RMSE (P-RMSE). Furthermore, both static and dynamic drifts are assessed. To compare the performances of the integrated sensors, an open-source Madgwick's filter implementation has been employed.

The obtained results show no significant drift on orientation estimates for all the tested IMUs under static conditions and for the MTw under dynamic conditions. On the contrary, limitations are shown in terms of sensitivity to vibrations, for the MPU, and to magnetic field distortions combined with accelerations comparable with the gravity one, for the MMR.

Under dynamic conditions, the MTw's proprietary filter outperforms the competitors during medium and fast movements (i.e., frequencies higher than 1 Hz). Instead, below that threshold, the performance of proprietary filters is almost the same for MTw and MMR. In contrast, no significant difference arises from the comparison of orientation estimates obtained by applying the Madgwick's filter to the raw data coming from the different IMUs, highlighting similar performances for all the integrated sensors.

3.6 Conclusions

This chapter described my research within the *Sensing* level. The dissertation began with an analysis of the different typologies of sensors used in the MoCap systems supported in the proposed work. Due to the limitations of optoelectronic systems, which typically confine their use to specialized laboratories, the discussion primarily focused on markerless and inertial MoCap. Thus, two categories of sensors can be identified: RGB/RGB-D cameras (depending on whether depth information is produced or not) and IMUs.

First, the different technologies allowing to estimate the depth map of a scene using one or multiple RGB-D cameras were analyzed. They can be divided into stereo vision, structured light, and ToF sensors, depending on the physical phenomenon exploited for the depth estimation.

The focus then shifted to IMUs. While in markerless MoCap a single camera can be sufficient for basic assessments of motion, inertial MoCap always requires multiple IMUs to be worn on the body of the analyzed subject. Therefore, the chapter continued by presenting an efficient implementation of a driver allowing to interface multiple Xsens MTw Awinda IMUs with ROS and enabling real-time streaming of their data. The driver also includes a novel synchronization algorithm to ensure that each time frame contains consistent data from all the connected IMUs, avoiding possible partial frames. This is particularly useful when using such data to perform an IK optimization.

The analysis then shifted to investigating the performance and limits of three different IMU manufacturers for human motion tracking. A series of experiments were carefully designed to mimic the amplitudes and frequencies typical of human movement. The experimental setup consisted of three IMUs (i.e., Xsens MTw Awinda, MbientLab MetaMotionR, and InvenSense MPU-9250) firmly attached to a direct drive servomotor controlled to follow specific sinusoidal references. The ground truth was directly computed from the encoder readings by compensating the initial offset of each IMU's coordinate system. The obtained results demonstrated how Xsens outperforms the other IMUs at the highest frequencies, while at the lowest frequencies all sensors perform equally. Thus, while expensive sensors allow for more accurate tracking of fast motion, for slow movements (e.g., in a rehabilitation context), cheaper sensors are also adequate.

4 Tracking Level

Part of the work presented in this chapter has been submitted as scientific papers [72], [77].
I have made a substantial and principal contribution in the conception and design of these studies, related software development, analyses and interpretations of the results, drafting, and critical revision of the final manuscripts.
Co-authors' permissions for the inclusion of the studies in this dissertation have been obtained.

4.1 Introduction

As introduced in Section 1.3, the *Tracking* level unifies my work focused on accurately tracking multiple people by fusing the data obtained from multiple sensors in real-time. It is designed to take as input multiple people's poses estimated by exploiting a generic network of homogeneous sensors. No assumptions are made about the number, typology, and synchronization of the sensors being used, the number of people being analyzed, and the algorithm used for the estimation of the body poses.

The *Tracking* level aims at overcoming several problems arising when a distributed network of sensors (e.g., a camera network) is used to assess human motion from multiple viewpoints. Common challenges come from background clutters, limited FoVs, occlusions (due to the environment, but also self-occlusions of the human body), and the general difficulty of tracking the human body, a system characterized by a large number of DoFs and prone to self-occlusions.

One possible solution to reduce the impact of the aforementioned limitations consists of exploiting a distributed camera network to acquire data of the same scene from multiple viewpoints. By fusing the partial information coming from each camera, it is possible to increase stability, accuracy, and reduce occlusions, allowing to obtain stable 3D reconstructions of the subjects' movements.

However, the usage of a multi-camera network introduces additional levels of complexity. First, different sensors might perceive different numbers of people on the scene. In fact, a person visible to a camera might be fully occluded in another camera's FoV. Additionally, one (or more) sensors might fail to accurately estimate the motion of a person. Thus, contrasting information can be present on the network.

Synchronization is a second important problem. Data obtained from a distributed system might refer to different time frames. Moreover, the central node in charge of fusing the data from each sensor might receive the packets in the wrong order (e.g., the computation and/or communication times from one of the sensors might be higher with respect to another, resulting in older packets arriving after more recent ones).

Furthermore, the information obtained from each sensor might refer to different reference frames. Even when data refer to a unique global reference frame, the calibration (i.e., the set of poses describing the relative positions among all the sensors in the network) might not be perfect. Thus, the same marker describing the same person might be placed in slightly different positions when expressed with respect to the global reference frame. This can result in sensible jitter in the estimated poses and, in the worst-case scenario, in completely wrong estimations of the body poses.

Finally, a typical problem in markerless MoCap is the possibility of having varying body dimensions across consecutive frames. In fact, the detected markers describing the pose of a person can be noisy, due to several factors. The quality of the hardware and of the BPE algorithm being used, how cluttered the specific environment is, the typology of poses that are assumed, and the number of interacting people, are all factors affecting the quality of the estimation. This can result in the estimation of unrealistic poses, both regarding the values of the anatomical joint angles and the lengths of the body segments.

My work on this topic tackles the aforementioned challenges by dividing the overall problem (i.e., providing accurate real-time body poses of multiple people from a generic network of sensors) into four subproblems. In this regard, four modules were developed to enable:

1. robust tracking of the different poses obtained by each sensor

(Skeleton Tracker module, Section 4.3.2.1);
2. accurate fusion of the body poses estimated by all the sensors in the network, after detecting possible wrong estimates (Skeleton Merger module, Section 4.3.2.2);
3. optimization of the estimated body poses to ensure consistency of the body dimensions and fix of possible outliers (Skeleton Optimizer, Section 4.3.2.3);
4. online smoothing of the motion trajectories by filtering high-frequency noise (Skeleton Filter module, Section 4.3.2.4).

The computation of each of the aforementioned modules is in real-time[1].

To assess the performance of the proposed workflow, an extensive dataset of movements was acquired. Both single-person sequences, containing data from 15 participants performing a set of 12 activities of daily living (ADLs), and multi-person sequences, including 7 different actions with two to four persons interacting in a confined area, were recorded. The multi-person sequences were specifically designed to challenge multi-people tracking algorithms. Strong occlusions (both partial and full-body) and multiple people exiting and reentering the cameras' FoVs are present.

The dataset, namely *UNIPD-BPE*, includes 13.3 h of high definition RGB and depth data (corresponding to over 1 400 000 frames) recorded by a calibrated RGB-D camera network of five synchronized Azure Kinect cameras. Furthermore, 3 h of inertial MoCap poses are obtained by exploiting highly accurate Xsens MVN Awinda full-body suits, corresponding to a total of over 600 000 frames recorded by each of the 17 IMUs used by each suit. All the recorded data will be released under the Creative Commons CC0 license in conjunction with the publication of [72].

To this end, Section 4.2 describes in detail all the recorded sequences available within the *UNIPD-BPE* dataset, as well as the experimental protocol used for the acquisitions and synchronization among all the sensors used. Finally, Section 4.3 proposes an in-depth analysis of the workflow proposed for real-time multi-sensor and multi-people BPE and tracking. The section concludes by presenting and discussing the results obtained by exploiting the *Hi-ROS* framework on the *UNIPD-BPE* dataset.

[1]Detailed analyses of the proposed system's accuracy, robustness, and real-time performance are reported in Section 4.3.3

4.2 UNIPD-BPE: SYNCHRONIZED RGB-D AND INERTIAL DATA FOR MULTIMODAL BODY POSE ESTIMATION AND TRACKING

As presented in Section 1.2, highly accurate human motion analysis relies on optoelectronic systems that track small retroreflective markers attached to the subject's body. These systems, although extremely accurate, are characterized by high costs and complex setups. Such characteristics constrain their use to specific applications that are confined in a dedicated laboratory (e.g., clinical analyses or animation industry MoCap). However, real-time human pose estimation could benefit a variety of fields, ranging from HRI, industry, autonomous driving, surveillance, and telerehabilitation. In such contexts, the deployment of optoelectronic systems is usually not feasible, and markerless analyses are a promising tool to address this issue.

Markerless BPE has been a topic of intensive research for decades in the CV community. Despite the improvements achieved in the latest years thanks to the advances enabled by data-driven approaches [25], [28], [141], [142], the accurate assessment of human motion without relying on any sensor or marker attached to the body is still an open challenge. Limited FoVs of the cameras and occlusions due to the environment, but also self-occlusions of the human body, limit the accuracy of such systems. One possible solution to reduce the impact of the aforementioned limitations consists of exploiting a distributed camera network to acquire data of the same scene from multiple viewpoints. By fusing the partial information obtained from each camera, it is possible to reduce the effect of occlusions and, at the same time, increase the overall system's accuracy.

In recent years, the development of portable and easy-to-use low-cost 3D cameras (e.g., the Microsoft Kinect) has further pushed the interest in markerless BPE [143]–[146]. The main advantage of these devices is the possibility to retrieve real-time synchronized RGB and depth data of the scene, up to 30 Hz. However, despite the widespread use of such sensors and the variety of available human motion datasets, only a small number of public datasets include RGB-D data and even less offer multiple calibrated RGB-D views. In fact, to the best of the author's knowledge, a comprehensive dataset including complex scenes with multiple people, RGB and depth data from a significant calibrated RGB-D camera network, together with ground truth body poses for all the recorded sequences, is still missing. All the most used markerless MoCap datasets (either focused on BPE or on action recognition) lack at least one of the aforementioned features.

HumanEva [147] is one of the first and most used datasets recorded for benchmarking markerless human pose estimation algorithms. The dataset includes six ADLs recorded from four different actors using four grayscale cameras, three RGB cameras, and a marker-based optoelectronic system as a ground truth. No information on the depth of the scene is available, and each sequence only involves a single person.

Human3.6M [148], on the other hand, offers depth data of the scene using a single ToF sensor. Also in this case, ground truth poses are acquired via marker-based MoCap, while visual data are recorded using four RGB cameras. The dataset includes a predefined set of 16 ADLs performed by 11 actors. Even in this case, no interactions among subjects are available.

Our previous work, the *IAS-Lab Action Dataset* [149], was one of the first to include RGB-D sensors in the acquisition setup. This dataset consists of 15 ADLs performed by 12 people. RGB and depth data are provided, as well as the persons' body poses estimated by exploiting a markerless BPE algorithm. However, data are recorded using a single Kinect v1 camera. Additionally, no ground-truth poses are available, nor are sequences with multiple people.

Berkeley MHAD [150] is one of the first datasets to include accelerometers in the acquisition setup. Eleven ADLs performed by 12 actors are recorded using marker-based MoCap, 12 RGB cameras, two Kinect v1 cameras, and six accelerometers. However, similarly to the previous works, the focus is on estimating single persons' actions, and no interactions are taken into account.

TUM Shelf [67] is among the most used datasets for benchmarking markerless BPE algorithms. It includes five RGB cameras to record a group of four people disassembling a shelf. Severe occlusions and unbounded motion of the persons are the main challenges of this dataset. However, since no other sensing devices are involved, the dataset offers only sparse manually annotated poses as a ground truth. The same authors also released the *TUM Campus* dataset [67]. The particularity of this dataset is that it is captured outdoors. The recorded scenes depict three people interacting on campus grounds. Similar to *TUM Shelf*, only three RGB cameras are used. Thus, the same limitations apply.

CMU Panoptic [151] is a large-scale dataset that includes 480 VGA cameras, 31 HD cameras, and 10 Kinect v2 cameras. A variety of actions (including both single-person and multi-person activities) are recorded inside a custom-built dome accommodating all the hardware. However, since vision is the only modality used to retrieve data, the recorded poses are only computed via triangulation based on a 2D BPE algorithm that runs on each camera, without any external ground truth.

Another public dataset including multiple depth views is the *NTU RGB+D* dataset [152]. Forty subjects are recorded performing a set of 60 actions that include ADLs, mutual activities, and health-related movements. The sensors used to extract the persons' poses are three Kinect v2 cameras. However, since the focus is on the validation of action recognition algorithms, no ground-truth poses are provided, but only labels indicating the type of actions being performed.

All the aforementioned datasets mainly focused on vision, including markerless and marker-based MoCap. *UTD-MHAD* [153], on the other hand, introduced the use of one IMU, in conjunction with a Kinect v1 camera. Eight subjects are individually recorded while performing a set of 27 predefined actions ranging from sports, hand gestures, ADLs, and training exercises. Similarly to the previous work, however, the focus is on action recognition. Thus, the available ground truth is limited to manually annotated labels describing the actions being performed.

Total Capture [154] is a widely used dataset and one of the first to introduce the usage of a full-body inertial suit consisting of 13 IMUs, alongside eight RGB cameras and marker-based MoCap. Five subjects are recorded performing a set of five actions selected from RoM activities, walking, acting, running, and freestyle. Ground-truth poses are computed via marker-based MoCap. However, the dataset does not include interactions among subjects, and no information on the depth of the scene is available.

AndyData-lab [155], similarly to the previous work, includes data from marker-based MoCap, a full-body inertial suit, two RGB cameras, while also adding finger pressure sensors. Since this work focuses on human motion analysis in industrial settings, 13 subjects are recorded while performing six industrial tasks, including screwing at different heights and manipulating loads. As in the previous work, neither interactions among subjects nor information on the depth of the scene are available.

Finally, *Human4D* [156] includes data from an optoelectronic system and four Intel RealSense RGB-D cameras (Intel Corp., Santa Clara, CA, USA). Four actors are recorded, both individually and in pairs, while performing a set of 14 single-person ADLs and five two-person activities in a professional MoCap studio. Ground-truth poses are collected via marker-based MoCap, and both RGB and depth recordings of the scene are available. However, during the recordings, all actors needed to wear a full-body black suit to accommodate the body markers required by the optoelectronic system during the entire trial. These artificial clothes can hinder the performance of RGB-based markerless BPE algorithms, potentially decreasing their accuracy, since they do not constitute a realistic scenario.

This section presents the University of Padova Body Pose Estimation dataset (*UNIPD-BPE*), an extensive dataset for multi-sensor BPE containing a large number of single-person and multi-person sequences with up to four people interacting. Full-body poses, as well as raw data from each sensor, are recorded both by means of a calibrated network with five RGB-D cameras (i.e., Microsoft Azure Kinect, Microsoft Corp., Redmond, WA, USA) and by exploiting up to two highly accurate full-body inertial suits (i.e., Xsens MVN Awinda, Xsens Technologies, Enschede, Netherlands). All recorded data will be released under the Creative Commons CC0 license in conjunction with the publication of [72].

The Azure Kinect is the latest RGB-D camera developed by Microsoft, with improved performance compared to the previous model (i.e., the Kinect v2). As demonstrated in [157], the Azure Kinect SD is reduced by more than $50\,\%$ with respect to the Kinect v2, while also achieving a depth estimation error lower than $11\,\mathrm{mm}$. For these reasons, the Azure Kinect is a promising device with a wide range of uses including object recognition, people tracking and detection, and human-computer interaction.

This dataset is the first to include high-definition RGB, depth, and BPE data from five calibrated Azure Kinect cameras. Videos and point clouds are recorded both at a resolution of $1920\,\mathrm{x}1080$ pixels @ $30\,\mathrm{Hz}$, and $640\,\mathrm{x}576$ pixels @ $30\,\mathrm{Hz}$ (native resolution of the depth sensor). Moreover, all subjects' body poses are estimated via markerless MoCap by exploiting the Azure Kinect Body Tracking SDK [158], offering baseline data to develop and benchmark different BPE and tracking algorithms. The high number of cameras allows to assess the impact of different camera network configurations on the accuracy achieved by markerless BPE algorithms, while the high-resolution recordings allow to quantify how different image resolutions can impact a specific algorithm.

The *UNIPD-BPE* dataset also contains full-body inertial MoCap data, collected by up to two Xsens MVN Awinda suits. Each suit consists of 17 MTw Awinda trackers, including a 3-axis gyroscope, a 3-axis accelerometer, and a 3-axis magnetometer. As demonstrated in [107], these sensors are extremely accurate for inertial BPE. Each tracker has a dynamic accuracy of $0.75\,°\mathrm{RMS}$ for roll and pitch, and $1.5\,°\mathrm{RMS}$ for the heading estimation, constituting a flexible and reliable tool for capturing human motion [159].

The proposed dataset includes both the raw data from each tracker, and detailed data describing each subject's body kinematics, computed by exploiting the MVN Analyze software. Such software combines the data of all motion trackers with a biomechanical model of the human, allowing to obtain an accurate and drift-free estimate of the body pose [57]. The hardware/software combination used on this work allowed to record raw IMU data (estimated orientation, angular velocities, linear accelerations, magnetic fields) for all the trackers required by each suit @ $60\,\mathrm{Hz}$, as well as 3D positions, orientations, velocities, accelerations of the 23 segments defining the Xsens biomechanical model, anatomical joint angles of 22 joints plus 6 additional joint angles targeted to ergonomic analyses, and the body center of mass location throughout all the sequences.

No optoelectronic data are included in this dataset because the required markers attached to the body are highly reflective, resulting in a strong distortion in the depth of the Kinects, and, consequently, in a poor estimation of the body pose. While it is possible to properly synchronize the two systems to avoid interference, this solution still degrades the Azure Kinect's performance. Therefore, to ensure maximum accuracy of the recorded markerless data, we chose to employ an inertial MoCap system in place of the optoelectronic one. The software used for the estimation of the body poses (Xsens MVN Analyze), coupled with the chosen hardware (Xsens MVN Awinda), allows us to obtain an accuracy comparable to state-of-the-art optoelectronic systems, as demonstrated in [57].

All the cameras and inertial suits used in this work are hardware synchronized (Section 4.2.2.4), while the relative poses of each camera with respect to the inertial reference frame are calibrated before each sequence to ensure maximum overlap of the two sensing systems outputs. The proposed setup allowed to record synchronized 3D poses of the persons on the scene both via Xsens' IK algorithm (inertial MoCap) and by exploiting the Azure Kinect Body tracking SDK (markerless MoCap), simultaneously. The additional raw data (RGB, depth, camera network configuration) allow the user to assess the performance of any custom markerless MoCap algorithm (based on RGB, depth, or both). Further analyses can be progressed by varying the number of cameras being used and/or their resolution and frame rate. Moreover, raw angular velocities, linear accelerations, magnetic fields, and orientations from each IMU allow to develop and test multimodal BPE approaches focused on merging visual and inertial data. Finally, the precise body dimensions of each subject required by MVN Analyze are provided. They include body height, weight, and segment lengths measured before the beginning of a recording session. They were used to scale the Xsens biomechanical model, and also constitute a ground truth for assessing the markerless BPE accuracy on estimating each subject's body dimensions.

The recorded sequences include 15 participants performing a set of 12 ADLs (e.g., walking, sitting, and jogging). The actions were chosen to present different challenges to BPE algorithms, including different movement speeds, self-occlusions, and complex body poses. Moreover, multi-person sequences, with up to four people performing a set of seven different actions, are provided. Such sequences offer challenging scenarios where multiple self-occluded persons move and interact in a restricted space. They allow assessing the accuracy of multi-person tracking algorithms, focused on maintaining frame-by-frame consistent IDs of each detected person. To this end, the proposed dataset has already been used to validate our previous work, describing a real-time open-source framework for multi-camera multi-person tracking [77]. A total of 13.3 h (over 1 400 000 frames) of RGB, depth, and markerless BPE data from five RGB-D cameras are present in the dataset, while the inertial MoCap system allowed to record 3 h (over 600 000 frames) of human poses, corresponding to 51.2 h of raw IMU data from all the sensors used in each suit.

4.2.1 Data Description

The *UNIPD-BPE* dataset contains: (1) high definition videos and point clouds from each RGB-D camera, (2) positions, orientations, and confidences of the body joints estimated via markerless MoCap, (3) raw IMU data from each tracker used in the inertial suits, (4) full-body kinematics and anatomical joint angles obtained via inertial MoCap. Table 4.1 summarizes all available data, while Sections 4.2.1.1 and 4.2.1.2 describe in detail the recordings obtained by each RGB-D camera and by the inertial suits, respectively.

4.2.1.1 Microsoft Azure Kinect

The camera network used in this work consists of 5 Azure Kinect cameras (labeled *k01*, *k02*, *k03*, *k04*, *k05*). Details on the spatial configuration of the sensors can be found in Section 4.2.2.1. Each camera includes a 1 MP ToF

Table 4.1: Content of the *UNIPD-BPE* dataset.

Source	**Typology**	**Details**
Calibration	Transforms	Relative poses among cameras
Camera network	Video	1920x1080 pixels @ 30 Hz (native resolution)
	Video	640x576 pixels @ 30 Hz (reprojection on the depth)
	Depth	640x576 pixels @ 30 Hz (native resolution)
	Depth	1920x1080 pixels @ 30 Hz (reprojection on the RGB)
	BPE	3D positions, orientations, confidences of 32 joints
Inertial suit	IMU data	Orientations and raw IMU data @ 60 Hz
	BPE	3D positions, orientations, velocities, accelerations of 23 segments
	Joint angles	Anatomical joint angles of 22 joints
	Center of mass	3D position of the person's center of mass

depth sensor, a 12 MP CMOS rolling shutter RGB sensor, a 6-DoF IMU, and a 7-microphone circular array. A factory calibration process provides intrinsic and extrinsic calibrations of the sensors.

The *UNIPD-BPE* dataset contains the following data, captured from each of the 5 cameras:

- video recordings (1920x1080 pixels @ 30 Hz (native resolution) and 640x576 pixels @ 30 Hz (reprojected on the depth));
- depth recordings (1920x1080 pixels @ 30 Hz (reprojected on the RGB) and 640x576 pixels @ 30 Hz (native resolution));
- 3D positions, orientations, confidences of 32 body joints defined in the Azure Kinect Body Tracking SDK model (Section 4.2.3.1).

Data are recorded at the maximum frame rate allowed by the system. The video resolution was chosen to provide high-definition captures, while also maintaining the dataset size manageable.

4.2.1.2 Xsens MVN Awinda

The Xsens MVN Awinda suit used in this work consists of 17 MTw Awinda trackers placed on the head, chest, shoulders, upper arms, forearms, hands, pelvis, thighs, shanks, and feet. Each tracker includes a 3-axis gyroscope, a 3-axis accelerometer, a 3-axis magnetometer, and has a dynamic accuracy of 0.75 °RMS for roll and pitch, and 1.5 °RMS for the heading estimation [159].

Before each sequence, the body model used to estimate the motion was specifically scaled to each participant's characteristics. All subjects' body dimensions and general information (sex, age, weight, height) are annotated in dedicated files included in the dataset.

The *UNIPD-BPE* dataset contains the following data, captured for up to two subjects simultaneously:

- orientations, angular velocities, linear accelerations, magnetic fields of 17 MTw Awinda trackers @ 60 Hz;
- 3D positions, orientations, linear and angular velocities, linear and angular accelerations of 23 body segments defined in the Xsens MVN Analyze model (Section 4.2.3.2);
- anatomical joint angles (flexion/extension, abduction/adduction, internal/external rotation) of 22 body joints, plus 6 additional joint angles calculated for ergonomic analyses;
- 3D position of the body center of mass.

Data are recorded at 60 Hz (maximum frame rate allowed by the system) using the Xsens MVN Analyze software (version 2021.0.1).

4.2.1.3 Dataset Structure

A total of 13.3 h of RGB, depth, and markerless BPE data are present in the dataset, corresponding to over 1 400 000 frames obtained from a calibrated network with five RGB-D cameras. The inertial suits, on the other hand, allowed to record 3 h of inertial MoCap data, corresponding to a total of over 600 000 frames recorded by each of the 17 IMUs used by every suit. Figure 4.1 shows an example frame of the available data recorded during a walking sequence.

The dataset is divided into two folders: *single_person*, containing all the sequences where a single subject is recorded, and *multi_person*, containing all the sequences with multiple subjects.

Single-Person Sequences. To make the data easily accessible, the single-person sequences are organized as follows. The *single_person* folder contains the data recorded from 15 subjects performing the 12 actions described in Table 4.2, with four repetitions each. Thus, it contains 15 folders, named *sbj<xx>*, where *<xx>* indicates the subject's ID. Each *sbj<xx>* folder contains the data recorded by the cameras, the inertial MoCap data, and a *yaml* file (named *sbj<xx>_info.yaml*) including the subject's ID, sex, age, weight, and the body dimensions used for inertial BPE.

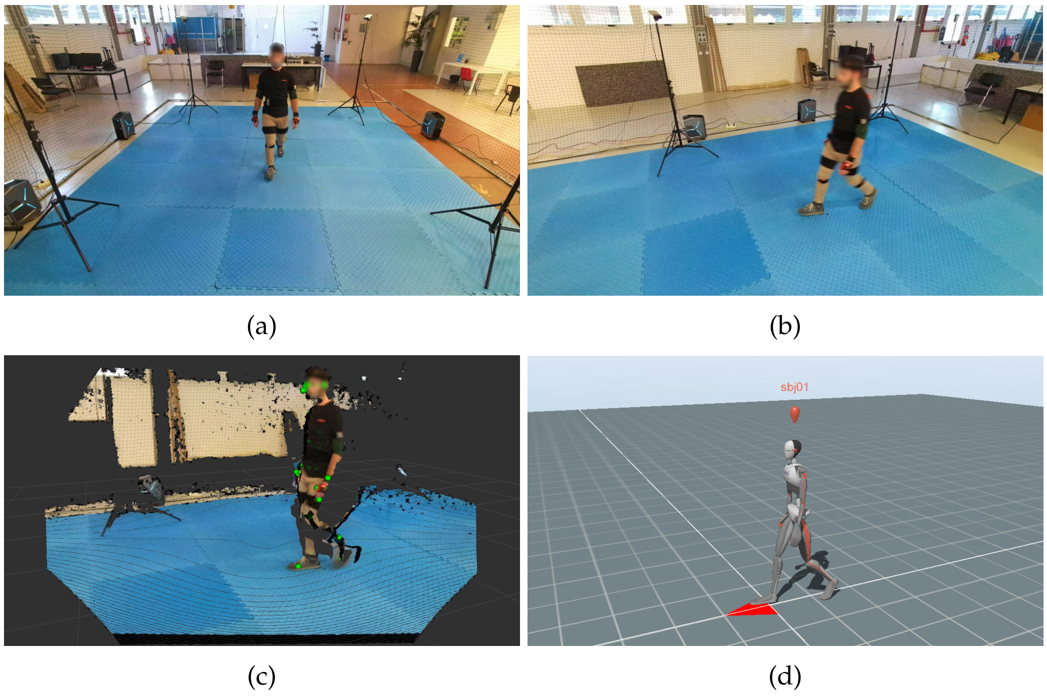


Figure 4.1: Sample data during a walking sequence: (a) RGB frame from *k01*, (b) RGB frame from *k02*, (c) depth and markerless pose estimation from *k02*, (d) inertial pose estimation from MVN Analyze.

The recorded data are stored in six subfolders: five folders containing the camera network data, named after the convention *k<yy>*, where *<yy>* indicates the camera's ID, and one additional folder containing the inertial suit data, named *xsens*. Each *k<yy>* folder contains four repetitions for each of the 12 actions, resulting in 48 files (one per sequence), named following the convention *sbj<xx>_<action_name><zz>.bag*, where *<zz>* indicates the recorded repetition. For each recorded sequence, the *xsens* folder contains three sets of files, named *sbj<xx>_<action_name>-<zz>.(mvnx | bvh | c3d)*. Each single-person action has an average duration of approximately 13 s. The complete list of recorded actions is reported in Table 4.2.

The *bag* file format indicates a bag file, commonly used in ROS [71] to store ROS message data. This format was chosen since it allows to store and distribute heterogeneous streams of synchronized data. By using *bag*

Table 4.2: List of actions performed during single-person sequences.

Index	Action Name	Description
0	t_pose	T-pose to be used for calibration purposes
1	n_pose	N-pose to be used for calibration purposes
2	walk	Walking at self-selected speed
3	squat	Squatting
4	bend	Bending down
5	sit	Sitting on a chair
6	jog	Jogging in place
7	jump	Jumping in place
8	cross_arms	Crossing arms
9	point	Pointing to different directions
10	wave	Waving hands
11	throw	Pretending to throw an object

files, it is also possible to play the recorded data simulating a real-time acquisition. Additionally, the content of a bag file can be exported in different formats by exploiting one of the many open-source tools developed by the ROS community (e.g., [160]). ROS bags, in fact, play an important role in ROS, and a variety of tools have been written to allow storage, processing, analysis, and visualization of the stored data.

All the *bag* files in this dataset contain RGB captures and depth point clouds from each camera, information on the camera network calibration, positions, orientations, and confidences of each participant's joints estimated via markerless MoCap.

The *mvnx* extension (MVN Open XML format) refers to Xsens' proprietary format. It is a human-readable XML format that can be imported into various software programs, including MATLAB and Microsoft Excel. This format contains information on sensor data, segment kinematics, and joint angles, as well as the subject's body dimensions. The *bvh* format (BioVision Hierarchical data) embeds captured motion data in ASCII format and is typically used in animation applications. It requires a hierarchical structure, such that only relative joint angles can be exported into this file format. Finally, *c3d* (Coordinate 3D) is a format used in optical systems and only contains 3D point coordinates. Therefore, the stored data are limited to the bony landmarks calculated from the estimated virtual marker set.

Multi-Person Sequences. Multi-person sequences include the seven actions described in Table 4.3, repeated with two, three, and four people simultaneously on the scene. The only exception is the action labeled *eight*, where the people are walking forming an eight, which required the presence of four people. The actions were selected to challenge different aspects typical of markerless BPE. As a result, there are different actions where multiple people are in close proximity, with partial and/or full occlusions, and with people exiting and reentering the scene.

Table 4.3: List of actions performed during multi-person sequences.

Index	Action Name	Description
0	static	Static poses to be used for calibration purposes
1	free_static	Free movements while remaining in the same place
2	free_dynamic	Free movements while changing positions
3	circle	Walk in a circle
4	cross	Switch positions while walking in a circle
5	in_out	Enter and exit from the cameras' FoVs
6	eight	Walk forming an eight

The *multi_person* folder contains data recorded from all the sequences including multiple subjects. It contains three folders, named *<xx>people*, where *xx* indicates the number of subjects present in each sequence. Similarly to the *single_person* sequences, each folder contains a *yaml* file (named *<xx>people_info.yaml*), the data recorded by the cameras, and the inertial MoCap data. In this case, however, the *yaml* file stores the IDs of all the subjects on the scene. At the beginning of each sequence, in fact, all the participants stand in front of the master camera (*k01*). To allow for the correct assignment of each subject's body dimensions, the *yaml* file contains the IDs of all the participants ordered from left to right, as seen by the master camera. The body dimensions can be retrieved by accessing the corresponding *sbj<zz>_info.yaml* file, where <zz> indicates the ID assigned for the single-person sequences.

The recorded data are stored in six subfolders: five folders containing the camera network data, named after the convention *k<yy>*, where *<yy>* indicates the camera's ID, and one additional folder containing the inertial suit data, named *xsens*. Each *k<yy>* folder can contain six or seven files (depending on the number of people interacting), named following the convention *<xx>people_<action_name>.bag*. Each *bag* file includes the same typology of data recorded for single-person sequences. In this case, however, no repetitions are available, since the focus is on providing relevant data for the assessment of multi-person skeletal tracking, and being each sequence the summation of the actions performed by multiple people simultaneously. For each recorded sequence, the *xsens* folder contains six sets of files, named *<xx>people_<action_name>_sbj<yy>.(mvnx | bvh | c3d)*, and *<xx>people_<action_name>_sbj<zz>.(mvnx | bvh | c3d)*, being inertial data available for up to two subjects simultaneously. Each multi-person sequence has an average duration of approximately 27.5 s. The complete list of recorded actions is reported in Table 4.3.

4.2.2 Methods

This section describes the experimental setup, the methodology used, and the characteristics of the participants. All data were recorded in a laboratory environment, to allow accurate calibration of the RGB-D camera network and proper alignment of markerless and inertial MoCap.

4.2.2.1 Experimental Setup

The experimental setup (Figure 4.2) includes five RGB-D cameras and up to two full-body inertial suits. Each camera is connected to a dedicated desktop PC, while the IMUs communicate wirelessly to a receiver (Awinda station) connected to a PC that acts as a master. All PCs are connected to the same local network. Software time synchronization among PCs is obtained using the NTP protocol [161], whereas sensors synchronization is performed by exploiting the onboard hardware offered by the two sensing systems. More details on hardware synchronization are reported in Section 4.2.2.4.

The cameras are placed at a height of 2 m, in the configuration shown in Figure 4.3. They are approximately placed in a circle with a radius of 3 m. This allows to cover an area of approximately 4 x 4 m where most cameras have full visibility of the persons in the scene. The pose of each camera with respect to a common global reference frame was estimated prior to the recordings using an internally developed calibration algorithm.

The recorded data were acquired using the Microsoft Azure Kinect ROS Driver[2] under ROS Noetic (Ubuntu 20.04 LTS). The driver allows to publish each person's detected poses as standard ROS messages. However, it does not include information on the detection confidence. For this reason, the driver has been customized to also include information on the estimated

[2]`github.com/microsoft/Azure_Kinect_ROS_Driver`

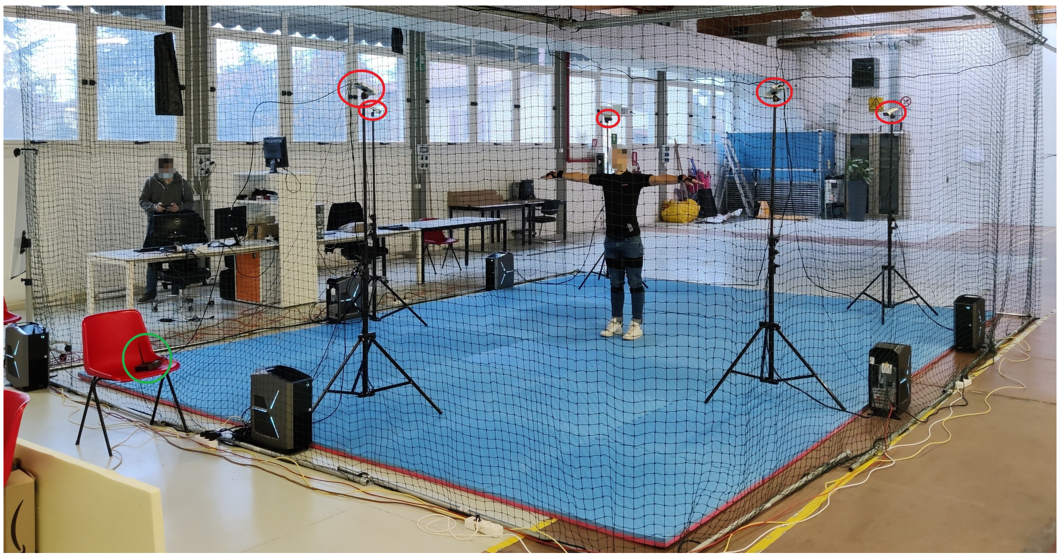


Figure 4.2: Experimental setup used for the acquisition of the *UNIPD-BPE* dataset. The five RGB-D cameras are highlighted in red, while the inertial suits' master receiver is highlighted in green.

joints' confidence in the messages. The mapping between the confidence levels assigned by the markerless BPE algorithm and the corresponding values stored in the messages is reported in Table 4.4.

Table 4.4: Azure Kinect Body Tracking SDK confidence mapping.

Confidence Level	**Description**	**Confidence Value**
NONE	The joint is out of range (too far from depth camera)	0
LOW	The joint is not observed (likely due to occlusion), predicted joint pose	1
MEDIUM	Medium confidence in joint pose	2
HIGH	High confidence in joint pose	3

4.2.2.2 Participants

A total of 15 participants were recruited for data collection (11 men, 4 women). The average age was 23.7 ± 2.7 years (min: 21 years, max: 29 years), the average weight 65.8 ± 12.7 kg (min: 48 kg, max: 92 kg), and the average

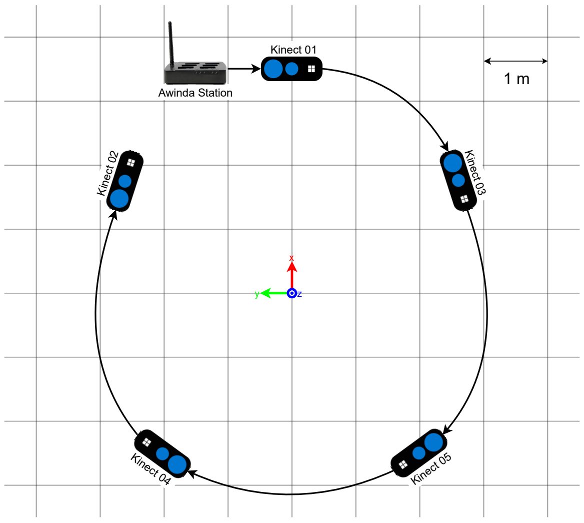


Figure 4.3: Spatial organization of the sensors used during the acquisitions. The axes define the global reference frame of both the camera network and the inertial suits, while the arrow lines indicate the cabled connections required for the hardware synchronization.

height $1.75 \pm 0.11\,\mathrm{m}$ (min: $1.57\,\mathrm{m}$, max: $1.98\,\mathrm{m}$). All participants gave written informed consent before data collection. Table 4.5 shows in detail each participant's characteristics. The ID assigned to each subject is the same for all experiments. Also in the sequences where multiple people interact, persons' IDs correspond to the ones used in their individual sequence.

4.2.2.3 *Acquisition Protocol*

Before each session, the 17 MTw Awinda trackers were placed on the participants' head, chest, shoulders, upper arms, forearms, hands, pelvis, thighs, shanks, and feet, following the Xsens protocol. The body model used for the motion estimation was then specifically scaled to each participant's characteristics. MVN Analyze was configured in the *Single level* scenario, since all tasks were executed on a fixed-level ground, without elevation changes. The system was then calibrated with the *N-pose and walk* procedure, and the world frame aligned with the camera network's global reference frame. To maximize the overlap between markerless and inertial BPE, the suit's world frame was realigned to the cameras' global frame before each sequence recording.

Table 4.5: Characteristics of the participants.

ID	Sex	Age [years]	Weight [kg]	Height [m]
01	m	22	72	1.75
02	m	22	55	1.70
03	m	21	60	1.65
04	m	22	92	1.90
05	m	21	84	1.81
06	m	22	63	1.72
07	m	22	75	1.98
08	m	22	78	1.88
09	f	23	48	1.65
10	m	28	68	1.75
11	f	23	52	1.57
12	m	27	72	1.75
13	f	29	56	1.70
14	f	24	48	1.58
15	m	27	64	1.79

For single-person sequences, the participants were asked to perform one of the actions described in Table 4.2 while facing a different cardinal direction in each repetition. Except for walking, where the start and end positions were fixed, the participants had maximum freedom regarding how to perform the actions.

Multi-person sequences include the actions reported in Table 4.3, each performed with a varying number of subjects ranging from two to four. Inertial data are recorded for up to two subjects per sequence simultaneously. Sensors placement and software configuration are the same as for single-person sequences.

4.2.2.4 Time Synchronization

This section describes the synchronization procedure followed for the acquisition of the dataset. In fact, since the dataset includes information from heterogeneous sources and a distributed camera network, all sensors must be time-synchronized.

Each Azure Kinect camera includes two synchronization ports (Sync in and Sync out). In this work, all cameras are synchronized through a daisy-chain configuration (Figure 4.3). To avoid interference among infrared projectors, the captures were offset from each other by $160\,\mu s$, as suggested in Microsoft's documentation. Therefore, the maximum delay between two cameras in the network is equal to $640\,\mu\mathrm{s}$, which is negligible with respect to the maximum frame rate of $30\,\mathrm{Hz}$ (< $2\,\%$ of the δt between two consecutive frames).

The Xsens MTw Awinda station includes four synchronization ports (two Sync in and two Sync out). In this work, the Awinda station was used as a master device to synchronize inertial and markerless MoCap. A custom cable was built to allow the Awinda station to send synchronization pulses to the master Kinect (*k01* in Figure 4.3). The chosen configuration allowed Xsens to properly synchronize the Kinect cameras by sending a triggering signal when a recording session is started. Thus, the *Start recording* command in MVN Analyze also triggered the streaming of the camera network data (RGB frames, depth frames, and markerless body tracking).

4.2.3 Body Joint Definitions and Hierarchy

4.2.3.1 *Microsoft Azure Kinect Body Tracking*

The Microsoft Body Tracking SDK allows to process Azure Kinect captures to generate body tracking results. A skeleton includes 32 joints. Each connection (bone) links the parent joint with a child joint. As demonstrated in [162], the mean joint estimation error has an average value of $8\,\mathrm{mm}$ and a standard deviation of $6\,\mathrm{mm}$ in static conditions. Table 4.6 lists the joint connections. Additional information can be found in [163].

4.2.3.2 *Xsens MVN Analyze*

The Xsens MVN Analyze software features a scalable biomechanical model and offers real-time 3D animation, graphs, and data streaming. A skeleton includes 23 segments connected by 22 joints. As demonstrated in [57], the inertial body poses show a RMSE lower than $5°$ for the estimation of the anatomical joint angles in the sagittal plane. Table 4.7 contains the list of the body segments defining a skeleton, its joints, and the trackers used to estimate human motion. Additional information can be found in [164].

4.2.4 Final Remarks

This section presented *UNIPD-BPE*, an extensive dataset for single- and multi-person BPE. Single-person sequences include 15 participants performing a set of 12 ADLs, while multi-person sequences include seven actions with two to four people interacting in a confined area.

Table 4.6: Azure Kinect Body Tracking joint definitions and hierarchy (source: [163]).

ID	Joint Name	Parent Joint	ID	Joint Name	Parent Joint
0	PELVIS	-	16	HANDTIP_RIGHT	HAND_RIGHT
1	SPINE_NAVAL	PELVIS	17	THUMB_RIGHT	WRIST_RIGHT
2	SPINE_CHEST	SPINE_NAVAL	18	HIP_LEFT	PELVIS
3	NECK	SPINE_CHEST	19	KNEE_LEFT	HIP_LEFT
4	CLAVICLE_LEFT	SPINE_CHEST	20	ANKLE_LEFT	KNEE_LEFT
5	SHOULDER_LEFT	CLAVICLE_LEFT	21	FOOT_LEFT	ANKLE_LEFT
6	ELBOW_LEFT	SHOULDER_LEFT	22	HIP_RIGHT	PELVIS
7	WRIST_LEFT	ELBOW_LEFT	23	KNEE_RIGHT	HIP_RIGHT
8	HAND_LEFT	WRIST_LEFT	24	ANKLE_RIGHT	KNEE_RIGHT
9	HANDTIP_LEFT	HAND_LEFT	25	FOOT_RIGHT	ANKLE_RIGHT
10	THUMB_LEFT	WRIST_LEFT	26	HEAD	NECK
11	CLAVICLE_RIGHT	SPINE_CHEST	27	NOSE	HEAD
12	SHOULDER_RIGHT	CLAVICLE_RIGHT	28	EYE_LEFT	HEAD
13	ELBOW_RIGHT	SHOULDER_RIGHT	29	EAR_LEFT	HEAD
14	WRIST_RIGHT	ELBOW_RIGHT	30	EYE_RIGHT	HEAD
15	HAND_RIGHT	WRIST_RIGHT	31	EAR_RIGHT	HEAD

The dataset includes $13.3\,\mathrm{h}$ of high definition RGB and depth data (corresponding to over $1\,400\,000$ frames) recorded by a calibrated RGB-D camera network of five synchronized Azure Kinect cameras, as well as each subject's full-body poses estimated using the Azure Kinect Body Tracking SDK. This allows to assess the impact of exploiting different numbers and/or configurations of cameras on the accuracy achieved by markerless BPE algorithms. The provided markerless body poses can be used as a baseline, while the raw recorded data (RGB, depth, and camera network configuration) allow the dataset user to assess the performance and accuracy of any custom markerless BPE algorithm (based on RGB, depth, or both).

Furthermore, $3\,\mathrm{h}$ of inertial MoCap poses are obtained by exploiting highly accurate Xsens MVN Awinda full-body suits, corresponding to a total of over $600\,000$ frames recorded by each of the 17 IMUs used by every suit. All sensors are hardware-synchronized, with the Xsens MVN Awinda system acting as a master to trigger the acquisitions. The relative poses of each camera with respect to the inertial reference frame are accurately calibrated before each sequence to ensure the best overlap of the two systems' outputs. This allows inertial MoCap estimates to be used to further investigate the accuracy of different markerless BPE algorithms. Since the raw IMU data are also available, the dataset can also be used to develop novel sensor fusion algorithms, aiming at improving the performance of both markerless MoCap, by increasing the achievable accuracy, and inertial

Table 4.7: Xsens MVN joint definitions and hierarchy (source: [164]).

ID	Segment Label	Tracker	Joint
0	Pelvis	Pelvis	jL5S1
1	L5	T8	jL4L3
2	L3	Head	jL1T12
3	T12	RightShoulder	jT9T8
4	T8	RightUpperArm	jT1C7
5	Neck	RightForeArm	jC1Head
6	Head	RightHand	jRightC7Shoulder
7	Right Shoulder	LeftShoulder	jRightShoulder
8	Right Upper Arm	LeftUpperArm	jRightElbow
9	Right Forearm	LeftForeArm	jRightWrist
10	Right Hand	LeftHand	jLeftC7Shoulder
11	Left Shoulder	RightUpperLeg	jLeftShoulder
12	Left Upper Arm	RightLowerLeg	jLeftElbow
13	Left Forearm	RightFoot	jLeftWrist
14	Left Hand	LeftUpperLeg	jRightHip
15	Right Upper Leg	LeftLowerLeg	jRightKnee
16	Right Lower Leg	LeftFoot	jRightAnkle
17	Right Foot	-	jRightBallFoot
18	Right Toe	-	jLeftHip
19	Left Upper Leg	-	jLeftKnee
20	Left Lower Leg	-	jLeftAnkle
21	Left Foot	-	jLeftBallFoot
22	Left Toe	-	-

MoCap, by limiting possible drifting phenomena.

The multi-person sequences offer challenging scenarios where multiple partially occluded persons move and interact in a restricted space. This allows to investigate the performance of multi-person tracking algorithms, both regarding the accuracy of the pose estimation in cluttered environments, and the ability to maintain frame-by-frame consistent IDs of each detected person in the scene.

The proposed dataset also presents some limitations. Due to the hardware used in the RGB-D camera network, no optoelectronic data could be included. This would offer an additional source of information, allowing also to assess the accuracy of inertial MoCap. Moreover, the main focus of the dataset is on the validation of different BPE algorithms. As a result, all recordings were acquired in a laboratory environment, with a limited amount of background clutter, to ensure the best overlap between markerless and inertial body poses.

To conclude, the *UNIPD-BPE* dataset aims to push forward the development of markerless BPE and tracking algorithms, enabling a variety of applications where unobtrusive accurate knowledge of human motion is of paramount importance. The dataset in fact includes data both for single-person RGB- and depth-based human motion estimation, for multi-person BPE and tracking, and for visual and inertial sensor fusion. The high-definition videos and point clouds, recorded by five calibrated and synchronized RGB-D cameras, allow simulating a variety of different scenarios (e.g., a pure RGB camera network, a pure depth camera network, an uncalibrated camera network, etc.). Finally, the included markerless and inertial body poses are useful for the development and testing of different multimodal sensor fusion and people tracking algorithms, without the necessity of expensive hardware and bulky acquisition setups.

4.3 OPEN-SOURCE MULTI-CAMERA SENSOR FUSION FOR REAL-TIME PEOPLE TRACKING

The ability to recognize and track human movements is of paramount importance in diverse fields, such as surveillance, HRI, telerehabilitation, and autonomous driving. Moreover, new emerging trends in Industry 4.0 and, recently, Industry 5.0, such as the use of cobots to support workers in their tasks ([165], [166]) and the active monitoring of operators to provide online ergonomic feedback ([167], [168]), require real-time knowledge of the operators' poses. The usage of inertial suits or optoelectronic systems to assess motion might not be viable in such contexts due to their high costs and complex setups, leaving markerless MoCap as the only feasible option.

The estimation of human motion without the aid of any body-mounted sensor or marker has been an intensive research topic for decades. Despite the improvements achieved in the latest years thanks to machine learning and deep learning approaches ([25], [141], [142]), real-time accurate assessment of human motion via markerless MoCap remains an open problem. Common challenges come from background clutters, limited FoVs, occlusions, and the general difficulty of tracking the human body, a system characterized by a large number of degrees of freedom and prone to self-occlusions.

One promising technology to mitigate such issues is to exploit a distributed network of RGB-D cameras that can acquire colored point clouds.

By fusing the partial information coming from each camera, it is possible to increase stability, accuracy, and reduce occlusions, allowing to obtain stable 3D reconstructions of the subjects' movements. Thanks to the latest performance improvements for markerless body pose detectors (e.g., [28]), in combination with relatively high frame rates in recent RGB-D cameras, simpler tracking approaches can achieve high accuracies without the overhead required by more sophisticated tracking algorithms [169]. In this context, a promising approach is to take advantage of the tracking-by-detection paradigm. Each camera is considered an independent detector that extracts information on the poses of the people being seen. Then, a single tracker takes as input all the detections and estimates the full trajectories of each person's movements via data association.

This section presents my research focused on real-time accurate assessment of human motion in multi-camera networks. The developed modules offer a series of advanced tools to enhance accuracy and robustness of markerless BPE when exploiting a calibrated camera network. The proposed workflow was designed with four major features in mind: (1) maximum accuracy of the estimated body poses, (2) simplicity of use, (3) real-time performance, and (4) generality, requiring the least possible number of assumptions. Thus, no assumptions are made about the typology or number of sensors that are used, nor about the detection algorithm that extracts the 3D poses of the persons.

All the developed modules[3] will be released under the Apache v.2 license in conjunction with the publication of [77], and have the potential of putting the basis to allow accurate human pose estimation and tracking, without the necessity of any sensor or marker on the body that might hinder a person's movements, and in real-time.

The proposed workflow consists of four main tools: a *Skeleton Tracker* node (Section 4.3.2.1), a *Skeleton Merger* node (Section 4.3.2.2), a *Skeleton Optimizer* node (Section 4.3.2.3), and a *Skeleton Filter* node (Section 4.3.2.4). Such nodes allow to accurately track all the people in the scene by merging the data obtained from each detector, optimizing the resulting poses, and filtering noisy estimates to obtain smooth trajectories, all in real-time. Figure 4.4 shows an example output of the proposed system while tracking four people.

[3]The code will soon be publicly and freely available under the Apache v.2 license at `github.com/hiros-unipd`

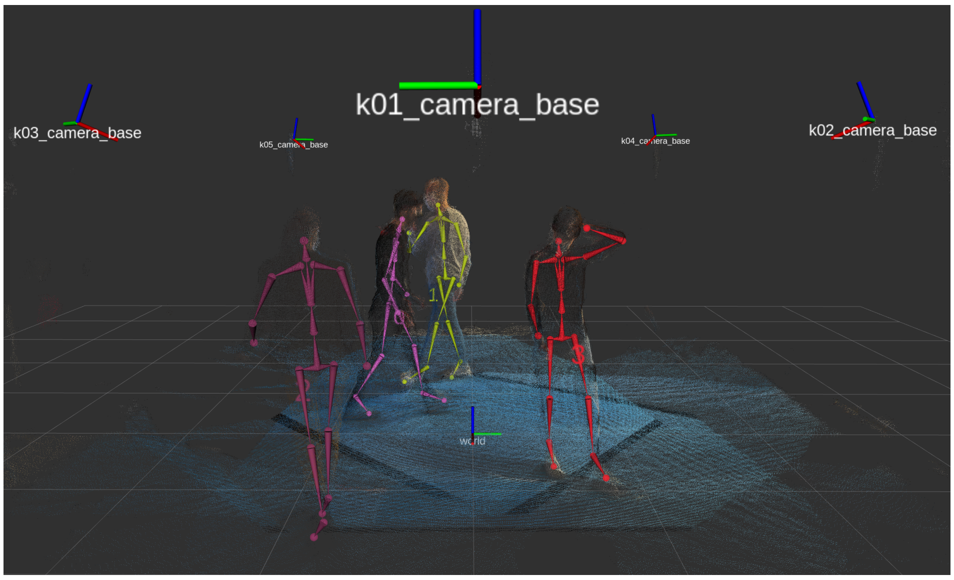


Figure 4.4: Output of the proposed system while tracking four people simultaneously. The input detections are obtained by exploiting the Microsoft Azure Kinect Body Tracking SDK.

Accuracy, robustness, and real-time performance of the system were evaluated on a public dataset ([72]). The dataset includes multiple sequences with different numbers of people interacting. The data provided include raw RGB and depth recordings obtained from a network composed of five synchronized Azure Kinect cameras, as well as accurate measurements of the participants' full-body kinematics via inertial MoCap. Due to the specific hardware and software used during the acquisitions, inertial body poses are estimated with an accuracy comparable to that of state-of-the-art marker-based optoelectronic systems, as demonstrated in [57]. Finally, static pose recordings of the participants and person-specific files containing their measured body dimensions are also provided, allowing our system to properly calibrate the optimization node.

4.3.1 Related Work

The release in recent years of affordable low-cost RGB-D sensors, like the Microsoft Kinect cameras, together with the growing importance of knowing operators' poses in multiple fields, brought increasing attention to markerless human BPE and tracking. This resulted in various works that focused on providing reliable estimates of human pose in real-time (e.g., [28], [170]–[173]). While single-camera human body tracking has been made easily accessible by exploiting such tools, an easy-to-use accurate framework for real-time multi-camera multi-person skeletal fusion and tracking is still missing.

Multi-camera people tracking systems can be divided into two groups: works that focus on leveraging 2D information to obtain 3D body poses and works that rely on RGB-D camera networks to increase the accuracy of the single camera 3D detections.

4.3.1.1 *Estimation of 3D Poses from 2D Data*

Most of the works that fall into this category utilize a calibrated network of RGB cameras to extract 2D body poses of the persons in the same scene from different viewpoints. Knowing the pose of each camera with respect to a global reference frame, it is possible to extract the 3D body poses based on triangulation. However, when dealing with multiple people, a prior tracking step is required to match the correct 2D points extracted by each camera with the correct 3D track. That is, multiple detections obtained from different cameras, but referring to the same person, need to be correctly associated, allowing to separate the sets of keypoints describing each subject.

Chen *et al.* [174] proposed an iterative process for estimating 3D poses from 2D body joints obtained by multiple cameras in real-time. Each camera stream is considered independent and does not require to be synchronized with the other ones. For this reason, the authors proposed an enhanced triangulation algorithm that does not require the 2D points of each view to be acquired simultaneously. Multi-person tracking is achieved by solving a weighted bipartite graph matching problem, where its affinity matrix is computed both from 2D and 3D geometric correspondences between each detection and track.

Reddy *et al.* [175] also presented a method for 3D pose estimation and tracking from an arbitrary number of camera feeds. The proposed system used a 4D convolutional neural network to produce a short time description of the people to be tracked. Tracking, in this case, is achieved by maximizing the total score of a cost matrix, where each element is computed as the inner product between pairs of descriptors. No information on the computation time is provided.

Chu *et al.* [176] tackled the same problem by taking advantage of temporal consistency to match the 2D poses obtained during each time frame with the previously estimated 3D skeletons. The affinity of a 2D detection to a 3D track is measured with the geometric constraints of the projection difference between the 2D pose and the reprojection of the 3D pose in the specific camera view. Once the affinity matrix is computed, the corresponding weighted bipartite problem is solved in real-time by exploiting the Hungarian algorithm. The authors also included a preprocessing joints filter step that allows removing outliers by computing the distance between each joint's epipolar line and the corresponding point and comparing it to a predefined threshold.

The main focus of all previous works is on computing accurate 3D estimates of the human pose. However, none of them considers the articulated object being tracked as a person. Including prior information of the body being tracked (e.g., by requiring a person's limb lengths not to vary between consecutive frames) can be a key feature to further increase markerless MoCap accuracy.

In this regard, Bultmann *et al.* [177] presented a novel method for real-time estimation of 3D human poses from a multi-camera setup. The 3D poses are recovered from 2D joints based on triangulation and refined by exploiting a body model that incorporates prior knowledge of the human skeleton. Multiple person detections are associated across camera views based on the epipolar distance of their joints. Then, each 3D skeleton is further optimized by exploiting prior information on the typical bone lengths of the human skeleton. However, such information is not considered as person-specific; thus, it does not take into account significant differences that can be present among different persons' body proportions (e.g., a child's proportions greatly differ with respect to an adult's ones).

4.3.1.2 Enhancement of 3D Poses from 3D Data

Works that rely on RGB-D camera networks do not require triangulation to compute the 3D poses of the persons in the scene. In this case, the main benefit from using multiple cameras is the lower sensibility of the whole system to occlusions. This allows the achievable accuracy to be increased, since missing information from one sensor might be available from another.

In this context, Moon *et al.* [178] used multiple Kinect sensors to correct inaccurate tracking data from a single camera. The developed system exploits a Kalman filter for real-time 3D human skeleton tracking, where the measurement noise vector is adjusted according to the reliability of each detection. In this case, however, the focus is on data fusion only, since the system only supports a single person's body tracking.

Kadkhodamohammadi *et al.* [179] also proposed a 3D human pose estimation system from multi-view synchronized RGB-D images that do not require prior knowledge of the number of persons in the scene. The multi-view fusion is solved as an iterative process where, for each time frame, the two closest skeletons that do not originate from the same view are merged until no pair of merging candidates are left. Then, a multi-view energy function is used to drive the body parts towards their optimal locations. The energy function takes into account how much the estimated body lengths differ from the average lengths computed during a training phase. However, such average lengths are computed across their entire training dataset, thus not being person-specific and suffering from the same limitations as [177]. Additionally, no information is provided on the computation time required by the optimization step.

Liu *et al.* [180], on the other hand, presented a work on human action recognition using a distributed calibrated RGB-D camera network. Even though the focus is not on multi-camera tracking, the authors proposed the usage of an information-weighted consensus filter to fuse the 3D skeleton detected by each camera. However, no tracking is performed, since the system only supports a single person.

Ryselis *et al.* [181] also exploited multiple Kinect cameras to monitor key performance indicators in physical training. To achieve higher accuracy, the authors used three Kinect devices to provide complete spatial coverage of the subject. Data fusion in this case is accomplished by calculating the averages of the joint coordinates estimated by each camera projected in a global reference frame. Also in this case, however, no tracking is performed, since the system only supports the analysis of a single subject.

Zhou *et al.* [182] proposed to integrate a Tobit Kalman filter (TKF) and a differential evolution (DE) algorithm to improve the accuracy of human motion estimation. The TKF allows ensuring kinematically admissible poses, while the DE optimization aims at minimizing the bone length variability, where the reference lengths are obtained from an initial application of the TKF. However, the system supports a single Kinect camera and only allows the analysis of a single person's pose. No information on the computation times required by the system is provided.

A general framework for multi-RGB-D camera systems was proposed by Fender *et al.* [183]. The framework can stream and process multiple

RGB, depth, and skeleton streams in real-time. The authors claim to perform skeleton fusion based on distance. To disambiguate the cases in which the skeletons are properly detected with respect to those in which the skeleton is rotated $180°$, body estimation is combined with face tracking. Since skeletal tracking is not the main focus of the framework, no other details are provided. The framework was planned to be released as open-source, but, to the best of the author's knowledge, it is still not available to the public.

Finally, in our previous work ([63]), we presented an improved version of the *OpenPTrack*'s skeletal tracking ([60]), an open-source software for real-time multi-camera people tracking in RGB-D camera networks. The tracking algorithm was enhanced with an improved version of the Kalman filter to ensure better temporal consistency of the body poses and an adaptation mechanism to avoid fast changes in the skeletons' limb lengths. However, the system is dependent on an internally developed human pose estimation algorithm and only supports specific sensors.

The literature review shows that, although markerless people tracking has been tackled by exploiting a variety of approaches, most works suffer from a series of limitations. Among all, real-time performance is typically omitted, very few open-source implementations are available, and additional constraints due to the body characteristics are often missing. In this work, we propose a system that aims to overcome such limitations. The proposed system is capable of tracking multiple people in real-time, without requiring any assumptions on the number, typology, and frame rate of the cameras being used. Information from multiple cameras allows for the detection of errors in the raw estimated poses and an enhancement of the overall system accuracy, strongly reducing the impact of occlusions. Moreover, information on each person's body dimensions can be fed to the system to ensure anatomically correct estimates of the poses. Finally, all the developed tools will soon be released as open-source under the Apache v.2 license in conjunction with the publication of [77].

4.3.2 SYSTEM DESIGN

The tools presented in this work aim at providing robust and reliable 3D tracking of human motion in real-time via markerless MoCap. To this end, the proposed system requires a network of N cameras that act as detectors. No assumptions are made about the number, typology, and frame rate of the cameras. The only requirement is the extrinsic calibration of the camera network. That is, after defining a global reference frame $\mathcal{G}$, the transform $\boldsymbol{T}_{\mathcal{C}_i}^{\mathcal{G}}$ between each i-th camera $\mathcal{C}_i$ and such frame $\mathcal{G}$ must be known.

Each camera represents an independent detector, which is in charge of performing markerless 3D BPE of the people in the scene. No assumptions are made about the algorithm used. The only requirement to be able to use a specific detector in the *Hi-ROS* framework is a bridge in charge of publishing the detected skeletons as a *SkeletonGroup* message (Section 2.4).

The proposed multi-sensor fusion workflow consists of four main tools:

1. A *Skeleton Tracker* node (Section 4.3.2.1) implementing robust skeletal tracking of the detections coming from each camera. This node ensures temporal consistency of the detected skeletons, by assigning each person a unique ID that is retained in time.
2. A *Skeleton Merger* node (Section 4.3.2.2) taking as input the tracked skeletons computed by the *Skeleton Tracker* and performing a fusion of the poses seen by all the cameras during the same time frame. The tool is designed to reduce the flickering obtained when data from a distributed camera network are merged. The node can properly handle and merge partial skeletons, as long as at least one common keypoint is available.
3. A *Skeleton Optimizer* node (Section 4.3.2.3) providing a global optimization on each person's poses. After an initial calibration of the body dimensions, this tool ensures limb length consistency throughout the whole experiment, also allowing to detect and remove possible outliers. This node introduces kinematic constraints on the tracked body poses, after integrating prior information of the persons' body dimensions in the system.
4. A *Skeleton Filter* node (Section 4.3.2.4) containing an efficient real-time state-space filter that can be used in cascade to any of the previous nodes. This node requires each skeleton to already have a unique ID (i.e., some prior form of frame-by-frame tracking must be performed), in order to obtain the temporal evolution of each person's pose. The filter's cutoff frequency can be manually tuned to obtain smoother trajectories of the detected keypoints, addressing the needs of different applications.

Figure 4.5 shows an overview of the proposed system. The white blocks represent a generic number of detectors, each running a BPE algorithm based on its camera's detections. Then, each detection is expressed as a

SkeletonGroup and fed to the *Skeleton Tracker*. The tracked skeletons can be used as is or fed to *Merger*, *Optimizer*, or *Filter*. The merged skeletons can, in turn, be the input to the *Optimizer*, or to the *Filter*, if the optimization is not needed by the application.

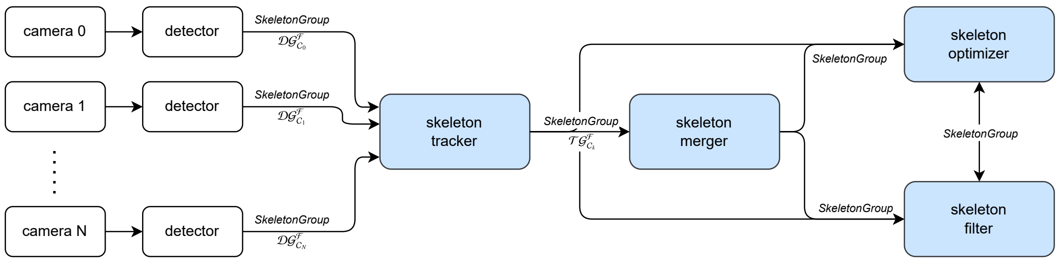


Figure 4.5: Overview of the proposed system. The white blocks are independent of *Hi-ROS*, while the blue blocks represent the tools offered by the proposed system. *Merger*, *Optimizer*, and *Filter* are not mandatory, allowing for different workflows.

The following sections analyze in detail the four tools developed within the proposed workflow: *Skeleton Tracker* (Section 4.3.2.1), *Skeleton Merger* (Section 4.3.2.2), *Skeleton Optimizer* (Section 4.3.2.3), and *Skeleton Filter* (Section 4.3.2.4).

4.3.2.1 Skeleton Tracker

The *Skeleton Tracker* is in charge of taking as input all the camera's detected poses and associating each skeleton to the correct track. This is achieved by solving an assignment problem, where each element of the cost matrix is computed as the distance between all possible pairs of detected skeletons at time t and tracked skeletons at time $t-1$.

Definitions. Let a skeleton group ($\mathcal{SG}$) be defined as the set of skeletons in the scene at time t, expressed with respect to a generic reference frame $\mathcal{F}$:

$$\mathcal{SG}^{\mathcal{F}}_t = \{\mathcal{S}^{\mathcal{F}}_{t,n} \mid n \in [0, N[\} \tag{4.1}$$

where N is the total number of people detected.

Each skeleton ($\mathcal{S}$) is defined as:

$$\mathcal{S}^{\mathcal{F}}_{t,n} = \{\mathbf{m}^{\mathcal{F}}_{t,n,p},\ \mathbf{l}^{\mathcal{F}}_{t,n,q} \mid p \in [0, P[,\ q \in [0, Q[\} \tag{4.2}$$

where $\mathbf{m}^{\mathcal{F}}_{t,n,p}$ is a set of markers and $\mathbf{l}^{\mathcal{F}}_{t,n,q}$ a set of links. P is the total number of markers defining the skeleton, while Q is the total number of links connecting pairs of markers. Figure 4.6 shows a representation of a possible skeleton consisting of 21 markers and 20 links.

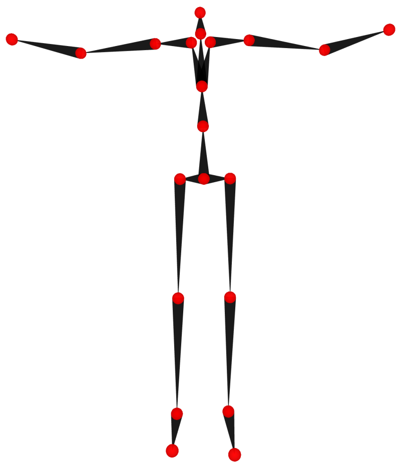


Figure 4.6: Example of a Skeleton formed by 21 markers (red) connected by 20 links (black).

A skeleton group can represent either the output of a detector (detection group) or the output of the *Skeleton Tracker* (track group).

To this end, a detection group ($\mathcal{DG}$) is defined as:

$$\mathcal{DG}^{\mathcal{F}}_{\mathcal{C}_k,t} = \{\mathcal{D}^{\mathcal{F}}_{\mathcal{C}_k,t,j} \mid j \in [0, J[\} \tag{4.3}$$

where $\mathcal{D}^{\mathcal{F}}_{\mathcal{C}_k,t,j}$ defines the j-th skeleton detected by the k-th camera $\mathcal{C}_k$, and J is the total number of detected skeletons.

Similarly, a track group ($\mathcal{TG}$) is defined as:

$$\mathcal{TG}^{\mathcal{F}}_{\mathcal{C}_k,t} = \{\mathcal{T}^{\mathcal{F}}_{\mathcal{C}_k,t,i} \mid i \in [0, I[\} \tag{4.4}$$

where $\mathcal{T}^{\mathcal{F}}_{\mathcal{C}_k,t,i}$ defines the i-th tracked skeleton having as source camera $\mathcal{C}_k$, and I is the total number of tracked skeletons.

Tracking Algorithm. Each time the *Tracker* receives a detection group $\mathcal{DG}^{\mathcal{F}}_{\mathcal{C}_k,t}$ from a camera $\mathcal{C}_k$, it checks the reference frame $\mathcal{F}$ where the data are expressed. If $\mathcal{F}$ differs from the global reference frame $\mathcal{G}$, then the *Tracker* computes the rigid transformation $\boldsymbol{T}_{\mathcal{F}}^{\mathcal{G}}$ to express all the data in the global reference frame:

$$\begin{aligned} \mathcal{DG}_{\mathcal{C}_k,t}^{\mathcal{G}} &= \boldsymbol{T}_{\mathcal{F}}^{\mathcal{G}} \cdot \mathcal{DG}_{\mathcal{C}_k,t}^{\mathcal{F}} \\ &= \{\mathcal{D}_{\mathcal{C}_k,t,j}^{\mathcal{G}} = \boldsymbol{T}_{\mathcal{F}}^{\mathcal{G}} \cdot \mathcal{D}_{\mathcal{C}_k,t,j}^{\mathcal{F}} \mid j \in [0, J[\} \end{aligned} \tag{4.5}$$

The transformation matrices $\boldsymbol{T}_{\mathcal{C}_k}^{\mathcal{G}}$, where $\mathcal{C}_k$ indicates each camera's local reference frame, are the result of a camera network calibration procedure.

Since in the remainder of the discussion all data will be expressed with respect to a common global reference frame, the apex $\mathcal{G}$ will be omitted to simplify the notation.

Each time a detection group $\mathcal{DG}_{\mathcal{C}_k,t}$ is received, the *Tracker* updates the latest available track group $\mathcal{TG}_{\mathcal{C}_k,t-1}$ accordingly. Then, the updated track group $\mathcal{TG}_{\mathcal{C}_k,t}$ is kept in memory to be used when the next detection group $\mathcal{DG}_{\mathcal{C}_k,t+1}$ arrives. The update process is divided into three parts:

1. update the detected tracks (skeletons that are present both in $\mathcal{DG}_{\mathcal{C}_k,t}$ and in $\mathcal{TG}_{\mathcal{C}_k,t-1}$)
2. add new tracks (skeletons that are present in $\mathcal{DG}_{\mathcal{C}_k,t}$ but were not present in $\mathcal{TG}_{\mathcal{C}_k,t-1}$)
3. remove unassociated tracks (skeletons that were present in $\mathcal{TG}_{\mathcal{C}_k,t-1}$ and are not present in $\mathcal{DG}_{\mathcal{C}_k,t}$).

The association between detections and tracks is achieved by finding the optimal solution to an assignment problem, where the cost matrix represents the distance between each detection $\mathcal{D}_{\mathcal{C}_k,t,j}$ and each track projection $\hat{\mathcal{T}}_{\mathcal{C}_k,t,i}$. The cost matrix $\boldsymbol{\Delta}_t \in \mathbb{R}^{I \times J}$ is defined as:

$$\boldsymbol{\Delta}_t = \{\delta_{t,ij} = w_l \cdot \delta_{t,ij}^{\mathbf{l}} + w_a \cdot \delta_{t,ij}^{\mathbf{a}} \mid i \in [0, I[, \ j \in [0, J[\} \tag{4.6}$$

where $\delta_{t,ij}^{\mathbf{l}}$ represents the linear distance between the marker positions to associate the i-th track with the j-th detection, $\delta_{t,ij}^{\mathbf{a}}$ the angular distance between the link orientations, w_l and w_a two weights that allow to balance the two contributions, I the total number of tracks at frame $t-1$, and J the total number of detections at frame t. Therefore, δ_t is a function of both the state at time t and at time $t-1$:

$$\delta_{t,ij} = f(\mathcal{T}_{\mathcal{C}_k,t-1,i}, \mathcal{D}_{\mathcal{C}_k,t,j}) = f(\hat{\mathcal{T}}_{\mathcal{C}_k,t,i}, \mathcal{D}_{\mathcal{C}_k,t,j}) \tag{4.7}$$

where $\hat{\mathcal{T}}_{\mathcal{C}_k,t,i}$ defines the predicted pose of the i-th track at time t exploiting a constant velocity model.

The linear distances are calculated as:

$$\delta^{\mathbf{l}}_{ij} = \frac{1}{N^\alpha} \sum_{k \in \mathcal{K}} w_k \left(\|\hat{\mathbf{m}}_{i,k} - \mathbf{m}_{j,k}\| + w_v \cdot \|\boldsymbol{v}_{i,k} - \boldsymbol{v}_{j,k}\| \right) \tag{4.8}$$

where $\mathcal{K}$ are the markers in common between the i-th track and the j-th detection, $\hat{\mathbf{m}}_{i,k}$ is the k-th marker's position of the predicted track, $\mathbf{m}_{j,k}$ the corresponding marker's position of the detection, $\boldsymbol{v}_{i,k}$ the last available velocity of the k-th marker of the track, $\boldsymbol{v}_{j,k}$ the corresponding velocity of the detection, w_k the weight associated with the marker, and w_v the velocity weight with respect to the position weight. N represents the number of markers in common, and its exponent $\alpha \geq 1$ allows to prefer track-detection matches that have a higher number of markers in common. The distance between two markers $\|\hat{\mathbf{m}}_{i,k} - \mathbf{m}_{j,k}\|$ is calculated as the Euclidean distance between their positions.

Similarly, the angular distances are calculated as:

$$\delta^{\mathbf{a}}_{ij} = \frac{1}{N^\alpha} \sum_{k \in \mathcal{K}} w_k \left(\|\hat{\mathbf{l}}_{i,k} - \mathbf{l}_{j,k}\| + w_\omega \cdot \|\boldsymbol{\omega}_{i,k} - \boldsymbol{\omega}_{j,k}\| \right) \tag{4.9}$$

where $\mathcal{K}$, in this case, represents the links in common between the i-th track and the j-th detection, $\hat{\mathbf{l}}_{i,k}$ is the k-th link's orientation of the predicted track, $\mathbf{l}_{j,k}$ the corresponding link's orientation of the detection, $\boldsymbol{\omega}_{i,k}$ the last available angular velocity of the k-th link of the track, $\boldsymbol{\omega}_{j,k}$ the corresponding angular velocity of the detection, w_k the weight associated with the link, and w_ω the velocity weight with respect to the position weight. In this case, N represents the number of links in common, and its exponent $\alpha \geq 1$ allows to prefer track-detection matches that have a higher number of links in common. The distance between two links $\|\hat{\mathbf{l}}_{i,k} - \mathbf{l}_{j,k}\|$ is calculated as the angle between the two quaternions defining their orientations along the shortest path:

$$dist(q_0, q_1) = 2\arccos(|q_0 \cdot q_1|) \tag{4.10}$$

where q_0 and q_1 represent the two quaternions describing the links' orientations.

Each marker/link weight is composed of two terms:

$$w_k = w_{\mathbf{m}/\mathbf{l}_{j,k}} \cdot w_{\mathbf{v}_{i,k}} \tag{4.11}$$

where $w_{\mathbf{m}/\mathbf{l}_{j,k}} = \kappa_{\mathbf{m}/\mathbf{l}_{j,k}}$ if the markers/links confidences are used ($\kappa_{\mathbf{m}/\mathbf{l}_{j,k}}$ represents the confidence of the k-th marker/link of detection j), 1 otherwise; $w_{\mathbf{v}_{i,k}} = (1 + \|\boldsymbol{v}_{i,k}\|)^{-\gamma}$, with $\gamma \in]0, 1[$, if the markers/links velocities are used, 1 otherwise. In this way, the faster a track's marker/link is moving and the lower its distance from the detection is weighted (see Figure 4.7). This choice allows not to penalize too much the computation of the distances during fast movements.

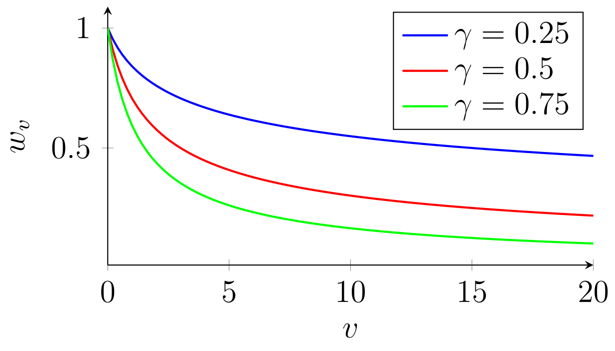


Figure 4.7: Velocity weight plot for different values of γ. Velocity expressed in $\mathrm{m/s}$ for positions, $\mathrm{rad/s}$ for orientations.

Once the cost matrix $\boldsymbol{\Delta}_t$ is filled, the Munkres algorithm [184] computes the optimal set of associations between detections and tracks to minimize the total cost (that is, the overall distance between detections and tracks). At this point, the previous tracks $\mathcal{TG}_{\mathcal{C}_k,t-1}$ are updated with the new data from the detections $\mathcal{DG}_{\mathcal{C}_k,t}$. Velocities and accelerations are updated accordingly. If some detections were not associated with any track (e.g., the number of detections at time t is greater than the number of tracks at time $t-1$, or some detections had a too high distance with respect to any track), then new tracks are initialized. Finally, if some tracks were not associated with any detection for a certain period of time t_m (e.g., the number of tracks at time $t - t_m$ is greater than the number of detections at time t, or some tracks had a too high distance with respect to any detection), then those tracks are deleted. This allows to maintain the correct IDs also in case of brief full-body occlusions.

4.3.2.2 Skeleton Merger

Small errors in the calibration of the camera network, as well as uncertainties in the estimation of body poses, can result in differences in the detected poses depending on the camera and orientation in which a person is seen. By concatenating the detections from each camera without some form of filtering, non-negligible flickering can be seen in the computed tracks. Kalman or generic low-pass filters are typically used in cascade to skeletal tracking to obtain smoother trajectories. However, due to the noisy nature of the input data, this requires extremely low cutoff frequencies to be set, thus not allowing to correctly track fast movements. To limit the flickering phenomena without penalizing fast movements, a possible solution is to merge the tracks describing the same person's pose seen by different cameras.

The *Skeleton Merger* node is in charge of fusing the tracked skeletons obtained by the *Skeleton Tracker* node. Each received track group $\mathcal{TG}^{\mathcal{G}}_{\mathcal{C}_k,t}$ is expressed with respect to the same global reference frame $\mathcal{G}$, and contains the set of skeletons detected by camera $\mathcal{C}_k$ at time t. This node does not require the cameras to be time-synchronized. In fact, the node determines the closest set of detections from each detector in time. Hardware time synchronization can lead to more accurate tracked poses, but it is not strictly required in the proposed system.

Since the cameras are not required to be synchronized, the merged skeletons can belong to slightly different time frames. First, the average source time $\bar{t}$ is calculated. Then the i-th merged track is computed as:

$$\overline{\mathcal{T}}_{\bar{t},i} = \frac{\sum\limits_{k \in \mathcal{K}} \gamma_{\mathcal{T}_{\mathcal{C}_k,t,i}} \cdot \mathcal{T}_{\mathcal{C}_k,t,i}}{\sum\limits_{k \in \mathcal{K}} \gamma_{\mathcal{T}_{\mathcal{C}_k,t,i}}} \tag{4.12}$$

where $\mathcal{T}_{\mathcal{C}_k,t,i}$ is the i-th track detected by camera $\mathcal{C}_k$ at time t, $\gamma_{\mathcal{T}_{\mathcal{C}_k,t,i}}$ its confidence (if provided by the detector), and $\mathcal{K}$ the set of cameras that can see the track.

However, before computing the merged skeletons, a procedure to detect outlier markers and/or links is implemented, based on a voting procedure among all detectors. First, possible flipped detections are identified by comparing the pelvis orientations. The detections from each camera are split into groups. Each group contains detections where the distance between the pelvis orientations is lower than $\pi/2$, where the distance is calculated following Equation 4.10. If more than one group is present, it is assumed that the detections from the largest group are correct, while the others are flipped.

After removing the flipped detections, it is possible to detect outlier estimates coming from one or more cameras for each marker and/or link by following an analogous voting procedure. The orientation distances of the links are calculated following Equation 4.10, while the position distances of the markers are computed using the Euclidean distance between each pair of markers.

To fuse the skeletons estimated from different cameras, the *Skeleton Merger* node constantly maintains a buffer of track groups (which is cleaned each time a merged skeleton is computed). Merging is triggered if at least one of the following conditions is met:

1. If the buffer contains a track group with the same source (i.e., the same detector) as the one from the newly received track group, then all the elements in the buffer should be merged, before adding the new track group;
2. If the delta time between the oldest element in the buffer and the newly received track group is greater than a maximum delta, then all the elements in the buffer should be merged, before adding the new track group. The maximum acceptable delta can be set by the user (it should typically be set equal to the cameras output rate);
3. If, after adding the new track group to the buffer, the number of tracks to merge is equal to the number of detectors, then the next track group will necessarily belong to a new time frame, and thus all the elements in the buffer should be merged.

4.3.2.3 Skeleton Optimizer

The *Skeleton Optimizer* node allows to further detect and fix outliers present, and to optimize joint positions/orientations to ensure limb length consistency of the tracked skeletons, after a prior calibration of the tracked persons. Therefore, the node can be divided into three parts: Calibration, Outliers Detection and Fix, and, finally, Optimization.

Calibration. The calibration procedure aims to compute the nominal set of limb lengths for a specific track $\mathcal{T}_i$ of a specific person:

$$\overline{\mathcal{L}}_i = \{\bar{l}_{i,j} \mid j \in [0, J[\} \tag{4.13}$$

where J is the total number of links defining a skeleton. The procedure is manually triggered and stores multiple frames of the tracked body poses in a buffer. Then the average link lengths $\bar{l}_{i,j}$ are computed, as well as the corresponding SDs $\sigma_{l_{i,j}}$. The SDs $\sigma_{l_{i,j}}$ are an indication of the calibration quality. If one or more SDs are higher than a maximum threshold, then the calibration quality is considered too low, and a new calibration should be performed. Once proper calibration data are computed for a person, it is possible to save the results for future use (e.g., in a following session involving the same subject). Then, such data can be loaded without the necessity to recalibrate the same person twice.

Outliers Detection and Fix. If calibration data are present, it is possible to further detect and fix outliers present in the tracked poses. A typical problem in markerless body tracking is the possibility to have non-negligible link length variability between consecutive time frames. Knowing the correct body dimensions of the tracked person makes it is possible to detect the markers that are being estimated in a wrong position.

For each tracked skeleton $\mathcal{T}_{t,i}$ where calibration data are present, the length of each link is compared with the correct one obtained from the calibration. If such a difference is greater than a maximum threshold, then the distal marker forming the link is considered an outlier. If an outlier is detected, the system attempts to overwrite the position of the distal marker with the position of the same marker estimated in the previous frame. This is useful to minimize missing data between consecutive frames. If such a marker was not present in the previous frame, then the marker is discarded.

Optimization. Once the highest sources of error are removed (i.e., the markers considered as outliers), the whole skeleton undergoes an optimization procedure to minimize limb length variability. Such optimization aims at minimizing an energy cost that takes into account both the limb length variability and the overall marker/link distances.

For each pair of markers $\boldsymbol{m}_{j,p}$, $\boldsymbol{m}_{j,d}$ that form a link $\boldsymbol{l}_j$, the energy cost of the length error is defined as:

$$E_{j,l} = (\|\boldsymbol{m}^*_{j,p} - \boldsymbol{m}^*_{j,d}\| - \bar{l}_j)^2 \tag{4.14}$$

where $\boldsymbol{m}^*_{j,p}$ and $\boldsymbol{m}^*_{j,d}$ are the positions of the markers after optimization, and $\bar{l}_j$ is the length of the j-th link computed during the calibration.

The energy cost of the marker position errors is defined as:

$$E_{j,p} = \|\boldsymbol{m}^*_{j,p} - \boldsymbol{m}_{j,p}\|^2 + \|\boldsymbol{m}^*_{j,d} - \boldsymbol{m}_{j,d}\|^2 \tag{4.15}$$

Finally, the energy cost of the link orientation errors is defined following Equation 4.10 as:

$$E_{j,o} = dist(\boldsymbol{l}_j^*, \boldsymbol{l}_j)^2 \tag{4.16}$$

where $\boldsymbol{l}_j^*$ is the orientation of the link after optimization.

The total energy for each pair of markers forming a link is defined as:

$$E_j = \sqrt{w_{j,l} \cdot E_{j,l} + w_{j,p} \cdot E_{j,p} + w_{j,o} \cdot E_{j,o}} \tag{4.17}$$

The three weights used are then defined as:

$$\begin{aligned} w_{j,l} &= \frac{\left| \|\boldsymbol{m}_{j,p} - \boldsymbol{m}_{j,c}\| - \bar{l}_j \right|}{\overline{L}} \\ w_{j,p} &= \frac{1 - w_{j,l}}{2} \\ w_{j,o} &= 1 - w_{j,l} - w_{j,p} \end{aligned} \tag{4.18}$$

where $\overline{L}$ is the maximum acceptable length error (links with errors greater than $\overline{L}$ are considered outliers). In this way, the higher the length error, the stronger the optimization will be, while maintaining the original marker positions and link orientations when the link length error is small. Finally, the total energy cost relative to the whole skeleton consists of the summation of all the energies related to each link:

$$E = \sum_{j=0}^{J} E_j \tag{4.19}$$

The global optimization problem that allows the minimization of the total energy cost is solved by exploiting the Levenberg-Marquardt algorithm implemented in the *Ceres Solver* library [185].

4.3.2.4 *Skeleton Filter*

Highly accurate biomechanical analyses of human movement typically require a filtering step in a post-processing phase to enhance the raw acquired data. For this reason, the proposed system includes a state-of-the-art real-time state-space filter to allow smoothing of the tracked bodies' trajectories.

The filter included in this work is based on an open-source implementation released by Pizzolato *et al.* [186] under the Apache v.2 license. The filter has been adapted to be used on *SkeletonGroup* messages. Changes in the number of markers or links between consecutive frames are correctly handled, and a custom low-pass filter is implemented for the link orientations. The cutoff frequency can be freely selected. Typical cutoff frequencies in biomechanics range between 6-10 Hz [12].

4.3.3 Experiments

The accuracy, robustness, and real-time performance of the developed system were assessed on a public dataset [72]. The dataset includes RGB-D and markerless MoCap data of multiple sequences with up to four people interacting, recorded using a calibrated network consisting of five Azure Kinect cameras. Single-person sequences are characterized by specific ADLs (e.g., sitting, walking, jogging, waving hands, etc.), while multi-person sequences contain movements that led the participants to overlap multiple times during the experiments. The dataset also includes body poses estimated by exploiting full-body Xsens MVN Awinda inertial suits for up to two people simultaneously.

No optoelectronic data are present due to the fact that the required markers attached to the body are highly reflective in the infrared domain, resulting in a strong distortion in the Kinects' depth and, consequently, in a poor estimation of body poses. However, the software used for the estimation of the body poses via inertial MoCap (that is, Xsens MVN Analyze) allowed to obtain an accuracy comparable to that of state-of-the-art optoelectronic systems, as demonstrated in [57].

To assess the accuracy of the proposed system, the markerless MoCap data retrieved from the dataset were converted into *SkeletonGroup* messages and used as input for the *Skeleton Tracker* as depicted in Figure 4.5. We then compared the inertial body poses to the ones obtained by exploiting different configurations of the proposed nodes:

1. configuration T: using the *Skeleton Tracker* only, which purely combines the raw detections from each camera;
2. configuration TMF: using *Tracker*, *Merger*, and *Filter*, thus avoiding the limb length optimization step;
3. configuration TMOF: exploiting the full pipeline, consisting of *Tracker*, *Merger*, *Optimizer*, and *Filter*.

To compare the accuracies obtained by the three configurations analyzed, we calculated the RMSE and the SD between the estimated joint angles that define the pose of a person. Let $q_D^{\mathcal{G}}$ and $q_P^{\mathcal{G}}$ be two quaternions that describe, respectively, the orientations of a distal segment and a proximal segment, expressed with respect to the same reference frame $\mathcal{G}$. The rotation of the joint connecting the two segments can be calculated as:

$$q_D^P = q_P^{\mathcal{G}*} \otimes q_D^{\mathcal{G}} \tag{4.20}$$

where $\otimes$ indicates the quaternion multiplication, and * the complex conjugate. The joint angles are then obtained by computing the corresponding Euler angles, following the zxy sequence [187]. The International Society of Biomechanics (ISB) defines flexion/extension (FE) as the rotation about the z axis, abduction/adduction (ABD) about the x axis, and internal/external (IER) rotation about the y axis ([188], [189]).

Joint angles were preferred to keypoint positions due to the fact that a direct comparison between keypoint distances could lead to erroneous results. This depends on two main factors: a) the two systems (markerless MoCap and inertial MoCap) place their keypoints in different positions on the body, and b) inertial suits can suffer from drift phenomena, leading to increasing errors in the absolute positions of the analyzed persons over time. However, the impact of drifting on the estimation of the joint angles is negligible [57].

The dataset used in this work is split into two sections: single-person sequences and multi-person sequences. Single-person sequences include the following actions: bending, crossing arms, jogging, jumping, N-pose, pointing, sitting, squatting, T-pose, throwing, walking, and waving. Multi-person sequences, on the other hand, include the following actions: people walking in a circle, people crossing, walking forming and *eight*, free dynamic movements, free static movements, entering/exiting from the scene, and static poses. More details can be found in Section 4.2. Out of the 19 different actions, we decided to remove N-pose, T-pose, and static sequences from our analysis since they are present for calibration purposes, and focus the analysis on active movements only. Table 4.8 describes all the actions taken into account for the accuracy assessment of the proposed system.

For each recorded frame, we extracted the values of the following joint angles: left/right shoulder FE, left/right elbow FE, left/right hip FE, and left/right knee FE. The choice was driven by the fact that such angles are the among the most informative, being the ones with the highest ranges of motion during typical movements, and can describe well the pose of a person. Only FE is taken into account due to the fact that ABD and IER cannot be estimated with high accuracy even by dedicated systems. Similarly,

Table 4.8: List of single-person and multi-person actions used for the experiments.

N people	Action	Description
1	*walk*	Walking at self-selected speed
1	*squat*	Squatting
1	*bend*	Bending down
1	*sit*	Sitting on a chair
1	*jog*	Jogging in place
1	*jump*	Jumping in place
1	*cross_arms*	Crossing arms
1	*point*	Pointing to different directions
1	*wave*	Waving hands
1	*throw*	Pretending to throw an object
2 - 4	*free_static*	Free movements while in the same place
2 - 4	*free_dynamic*	Free movements while changing positions
2 - 4	*circle*	Walk in a circle
2 - 4	*cross*	Switch positions while walking in a circle
2 - 4	*in_out*	Enter and exit from the cameras field of view
4	*eight*	Walk forming an eight

the Azure Kinect Body Tracking SDK estimates of such angles are generally very noisy.

For each action (including one, two, three, or four people), we calculated the mean RMSE and the SD of all the eight joint angles taken into account. Table 4.10 details the obtained results. The table outlines the accuracies achieved using configuration T, TMF, and TMOF, as well as the improvement gains obtained when using TMF and TMOF with respect to pure tracking when compared to the results provided by inertial MoCap. The accuracy analysis is divided between single-person sequences (Section 4.3.3.1) and multi-person sequences (Section 4.3.3.2). The robustness of the proposed system is discussed in Section 4.3.3.3, while real-time performance is analyzed in Section 4.3.3.4.

4.3.3.1 Single-Person Sequences

The raw output of the Azure Kinect Body Tracking SDK shows a mean RMSE between the values of the anatomical joint angles computed from the estimated body poses and the ones calculated by MVN Analyze of $18.77 \pm 15.23°$ across all single-person sequences. The RMSE is reduced by

21.74 % when also including *Merger* and *Filter*, going from 18.77° to 14.69°. However, the highest and most important improvement in accuracy can be observed in the SD, with a decrease of 34.54 %, going from 15.23° to 9.97°. In fact, it is important to mention that some joint angles show a constant delta between inertial and markerless data. This can be noticed mainly on the upper part of the body and appears to be caused by a systematic shift in the positioning of the elbow keypoints from the Azure Kinect Body Tracking SDK algorithm used in the dataset. For this reason, better insight on the accuracy improvement obtained when exploiting the full proposed pipeline is obtained by comparing the SDs of the errors, rather than the RMSE. The SD can, in fact, be seen as the RMSE after compensating the shift, as well as a measurement of the signal noise. Thus, the SD allows to quantify the joint angles estimation errors after compensating such shift.

When also including the *Optimizer*, the improvement with respect to pure tracking is still of 21.74 % on the RMSE and 34.60 % on the SD, which is slightly better with respect to the TMF case. This behavior can be explained by two possible causes. The first is that the raw estimated body keypoints are already placed in positions such that the body dimensions do not change much between consecutive frames; thus, the effect of the *Optimizer* on the final poses is minimal. The second reason is that the link lengths do not directly influence the estimation of a joint angle. As seen in Equation 4.20, only the relative orientation of the distal and proximal links contributes to the final value of the angle. Thus, the accuracy of the estimated joint angles might remain substantially unchanged, while at the same time the estimated keypoint positions can be more consistent and respect the person's body dimensions throughout the whole experiment.

When analyzing specific actions, we can see that *walking* and *waving* are the most accurate, achieving an accuracy of 6.45° and 6.08°, and the ones with the highest error reductions (44.44 % and 53.65 % respectively). This means that, in these sequences, the impact of *Merger*, *Optimizer*, and *Filter* is maximized, allowing the correct detection and fix of flipped detections and outliers. *Pointing*, on the other hand, is the most challenging action. This can be explained by the fact that during the *pointing* sequences there is one arm that is constantly occluding important parts of the body, such as the chest and/or the shoulder, creating difficulties in the correct estimation of the pose and, at the same time, limiting the impact of outlier detection. It is worth noting that such poses are also critical with respect to inertial MoCap. In fact, the shoulder joint kinematics is typically the most difficult to estimate correctly [190]. Therefore, the highest errors in the *pointing* sequences may also be caused by errors in the poses estimated by the inertial suit. *Walking* and *waving*, on the other hand, are more accurately tracked by the camera network, being occlusions minimal.

As depicted in Figure 4.8, showing an example of joint angles estimated using the inertial suit and the complete pipeline (TMOF) during a *wave* sequence (shoulders and elbows) and during a *jog* sequence (hips and knees), the estimates of the two systems are extremely similar, even in challenging actions as jogging.

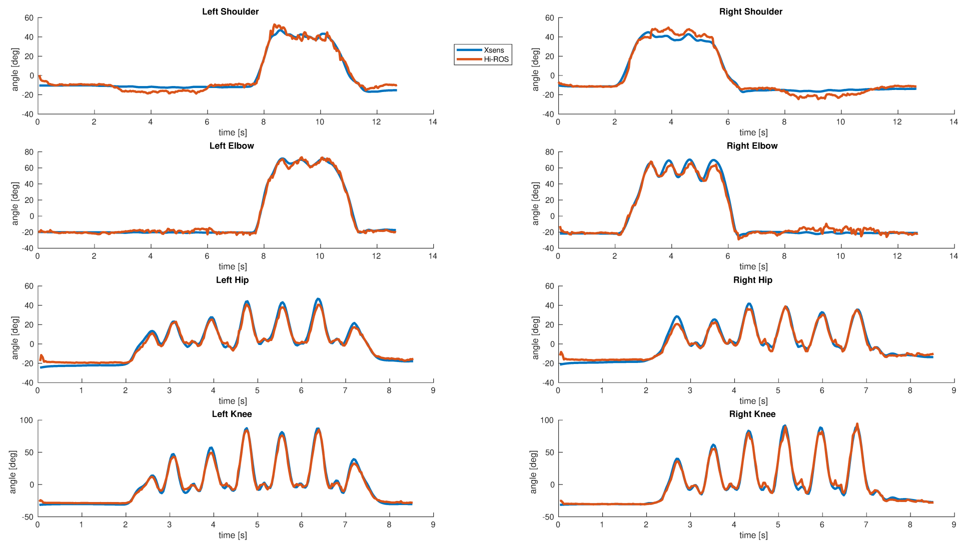


Figure 4.8: Joint angles calculated by Xsens MVN Analyze (in blue) and by the proposed system (in orange). Shoulder and elbow angles refer to a *wave* sequence, while hip and knee angles refer to a *jog* sequence.

Before discussing multi-person sequences, it is important to perform a more in-depth analysis of the *Skeleton Optimizer*. In fact, while its impact is overshadowed by the *Merger* when using state-of-the-art hardware and granting optimal views of the persons on the scene, the *Optimizer* is a powerful tool, designed to be used in the most challenging scenarios. When less accurate hardware is adopted, resulting in a noisy estimated depth, or if the tracked persons are far from the sensors, the estimated keypoint positions can lead to significantly varying limb lengths. Although the data recorded in the dataset do not suffer from these problems, the same cannot be ensured in different scenarios. When this is the case, the *Optimizer* allows to minimize the impact of erroneous estimates and to guarantee meaningful body poses even in the most challenging scenarios.

4.3.3.2 Multi-Person Sequences

Multi-person sequences show similar errors with respect to the single-person ones (Table 4.10). However, the average error for the sequences with two persons is even lower than the errors obtained in single-person sequences. The reason behind this can be explained by the fact that the increased difficulty arising from the higher number of people present in the scene is compensated for by the different typologies of actions being performed.

As depicted in Table 4.10, all three configurations show the same behavior, depending on the number of people in the scene, where the errors slightly increase as the number of persons in the scene increases. This was expected and is dependent on the fact that the more people are tracked in the same confined space, the more occlusions will be present. When a body is partially occluded, erroneous estimations of part of the body keypoints might occur.

Another source of error in markerless MoCap comes from the fact that it is easier for a detected skeleton to be completely occluded by others and, as a result, to be considered as a new person when it reappears. The Azure Kinect Body tracking SDK tends to spawn new skeletons in an incorrect pose, resulting in temporary errors during the first few frames. However, the *Tracker* is still able to assign the correct IDs even in these cases.

In multi-person sequences, the errors obtained from pure tracking with respect to inertial MoCap start from $16.74 \pm 13.38°$ when tracking two persons, increase to $17.39 \pm 13.94°$ when three persons are present, and reach $17.80 \pm 14.43°$ with four people in the scene. The same trend is obtained when using TMF and TMOF. The accuracy improvement with respect to raw tracking decreases as the number of people increases. With two people in the scene, the RMSE is reduced by $19.24\,\%$ when using TMF and by $18.88\,\%$ when using TMOF, while the SD is reduced by $31.84\,\%$ (TMF) and $31.32\,\%$ (TMOF). With three people in the scene, the RMSE is reduced by $15.76\,\%$ when using TMF and by $15.41\,\%$ when using TMOF, while the SD is reduced by $26.47\,\%$ (TMF) and $25.90\,\%$ (TMOF). Finally, with four people in the scene, the RMSE is reduced by $13.31\,\%$ when using TMF and by $13.03\,\%$ when using TMOF, while the SD is reduced by $20.51\,\%$ (TMF) and $20.03\,\%$ (TMOF). Here, the results of the *Optimizer* are slightly less accurate than those of the *Merger*. As in single-person sequences, the difference is negligible, since the different positions of the keypoints after the optimization do not directly impact the estimation of the joint angles.

The smaller improvement with respect to single-person sequences may be due to the fact that, overall, a lower number of Kinects are seeing the persons at each time frame. Although this difference does not affect the pure tracking accuracy (configuration T), which consists only of the association of the raw detections obtained by each camera, this affects *Merger* and *Optimizer*. In fact, outlier filtering can be less effective when a lower number of cameras can see the persons in the scene. However, the increase in accuracy is still significant, with a $20\,\%$ error reduction when using TMOF with respect to T in the worst-case scenario.

When analyzing single actions, in TMF and TMO we can notice a consistent behavior among the different sequences, where *free_static* is typically the most accurate action (with errors ranging from $8.05°$ to $10.43°$) and *in_out* is the least accurate one (with errors ranging from $11.66°$ to $12.08°$). This was expected since during *free_static* sequences people are standing in the same place during the whole experiment, whereas in *in_out* sequences people are constantly exiting and reentering the acquisition area. However, in the T configuration, the behavior is almost inverted. In this case, the *in_out* sequences are typically among the most accurate, while the highest errors occur in less demanding sequences, such as *free_static* and *free_dynamic*. A possible explanation for this is the presence of flipped detections. In fact, as introduced in Section 4.3.2, it is possible for one or more Kinects to detect a body pose flipped by $180°$. If such a person is moving, there is a high chance that the pose will be fixed after a few frames. However, if the person is standing still, the orientation of the detection might remain flipped throughout the whole experiment. In this sense, the fact that people are constantly moving in *in_out* sequences helps to reduce the impact of such flipped detections. On the contrary, in the *free_static* and *free_dynamic* sequences, this cannot happen. The same behavior is not present in the TMF and TMOF configurations. In fact, *Merger* and *Optimizer* are designed to correctly detect and remove both flipped detections and outlier markers/links.

4.3.3.3 Robustness

When tracking multiple people, it is important to be able to assign a consistent ID to each person throughout the whole experiment. Extreme proximity of two or more persons, as well as awkward poses, can cause tracking algorithms to switch the assigned IDs. In order to assess the robustness of the proposed system on assigning consistent IDs to all the tracked people, the output of multi-person sequences was manually checked. The tracked skeletons (and their IDs) were projected onto the point cloud obtained by merging the five Kinects (Figure 4.4). The correctness of each ID was then manually controlled for all the analyzed sequences.

The IDs were correctly assigned to each participant, independently of the number of people present in the scene and of the proximity of the tracked people. There are some cases, especially in *in_out* sequences, where some persons exit from the FoV of all the cameras for small windows of time. Then, when a person reenters the scene, the system gives them a new ID. However, even in these cases, the IDs are correctly assigned and kept by each track during the time windows in which they are visible to at least one camera. It is worth noticing that *cross* and *eight* sequences are extremely challenging in this regard, since people are constantly moving across each other, while also being partially occluded to one or more cameras. We can conclude that the proposed system is robust and allows to correctly track all the persons in the scene.

4.3.3.4 Real-time Performance

The average computation time required to process a single frame was measured for all the nodes of the proposed system (i.e., tracking time, merging time, optimizing time, and filtering time). Table 4.9 reports the computation times obtained when varying the number of people tracked. The tests were run on an Intel Core i7-1165G7 CPU @ 2.80 GHz laptop with 16 GB of RAM.

As expected, the results show that the optimization is by far the most demanding task, ranging from 4.16 ms when a single person is being tracked to 8.33 ms when tracking four people. Tracking, merging, and filtering times are negligible, requiring respectively 0.19 ms, 0.55 ms, and 0.11 ms to track one person, and 0.86 ms, 0.88 ms, and 0.21 ms to track four people.

Table 4.9 also reports estimates of the computation times required to track more than four people. Such data are calculated by assuming a linear trend between the number of people being tracked and the computation time required by each node. It is important to note that the total time to process each frame is not equal to the sum of the computation times of all nodes. In fact, each node runs in parallel. Therefore, taking the four people

Table 4.9: Computation times required to process a single frame. Times to track one to four people are calculated on the dataset trials, while times to track five and ten people are estimated from the previous ones assuming a linear trend between the number of tracked people and the computation time required. Data are presented as mean (SD).

N people	**Tracker** [ms]	**Merger** [ms]	**Optimizer** [ms]	**Filter** [ms]
1	0.19 (0.05)	0.55 (0.08)	4.16 (0.95)	0.11 (0.03)
2	0.37 (0.14)	0.71 (0.28)	6.44 (2.13)	0.17 (0.06)
3	0.63 (0.26)	0.85 (0.43)	7.52 (3.10)	0.19 (0.08)
4	0.86 (0.39)	0.88 (0.55)	8.33 (3.76)	0.21 (0.10)
5	1.08	1.03	10.01	0.25
10	2.22	1.59	16.81	0.40

scenario as an example, the average time required to process each frame would be equal to $8.33\,\mathrm{ms}$ (the time required by the most computationally intensive node), which corresponds to a theoretical maximum frame rate of $\sim 120\,\mathrm{Hz}$. On the other hand, the average delay between input detections and output tracks corresponds to the sum of the times of all the nodes. In this case, it would be equal to $10.28\,\mathrm{ms}$.

Finally, since the usage of *Merger*, *Optimizer*, and *Filter* is optional, faster performance can be obtained by removing the *Optimizer*, with the downside of not guaranteeing consistent limb lengths during the experiments. We can see that when tracking ten people, the computation time required by the *Optimizer* is $16.81\,\mathrm{ms}$, corresponding to a maximum frame rate of $\sim 59\,\mathrm{Hz}$. However, without the *Optimizer*, the computation time required for each frame would decrease to $2.22\,\mathrm{ms}$, allowing to reach a maximum frequency of $\sim 450\,\mathrm{Hz}$.

4.3.4 Final Remarks

This section analyzed the modules I developed during my Ph.D. within the *Tracking* level. The proposed system offers a series of tools to perform real-time multi-camera multi-person tracking, consisting of four modules:

1. a *Skeleton Tracker* node to perform pure frame-by-frame tracking of the detected body poses;
2. a *Skeleton Merger* node to fuse detections from multiple cameras and

detect possible outliers;

3. a *Skeleton Optimizer* node to perform a global optimization to ensure consistency of each person's body dimensions;
4. a *Skeleton Filter* node to perform real-time filtering of the estimated markers poses and links orientations.

All the developed nodes communicate with each other by means of efficient custom-defined messages (Section 2.4).

The accuracy, robustness, and real-time performance of the proposed system were assessed on a public dataset ([72]). The dataset includes a variety of sequences with up to four people interacting, recorded by exploiting a calibrated camera network consisting of five Azure Kinect cameras. Body poses are captured for up to two people simultaneously using an accurate inertial MoCap suit (Xsens MVN Awinda).

The results show that, by exploiting the proposed system, it is possible to reduce BPE errors by up to 34.60 % when compared to a pure tracking-by-detection approach. When increasing the number of people in the scene, the system is still capable of reducing the estimation errors by 31.84 % with two persons, by 26.47 % with three persons, and by 20.51 % with four persons. Robustness analyses show that correct IDs are assigned to each tracked person in all the dataset sequences. Even during periods of close proximity of two or more people, no ID switch is reported.

Real-time performance was also analyzed by reporting the computational time required by each node to process a frame. The complete pipeline is able to produce real-time results, reaching 240 Hz when tracking a single person, or 120 Hz when tracking four people. Finally, the *Optimizer* shows a computation time ranging from 4.16 to 8.33 ms, since multiple iterations are typically needed to converge to the optimal solution.

The analyses performed suffer from two limitations. The first one is the possibility of having inertial body kinematics data only for up to two persons simultaneously. This is a limitation of the dataset, where the inertial suits were not available for more people. For this reason, the results reported in Table 4.10 for multi-person sequences are always obtained by comparing the joint angles of two subjects only. In sequences with more than two people interacting, it is not possible to know the inertial poses of the other persons. Another limitation depends on the nature of the body poses estimated by exploiting inertial suits. By using inertial suits, in fact, it is not possible to directly compare the Euclidean distances between the keypoints estimated using the two MoCap systems. Consequently, all the analyses were conducted on the joint angles describing the poses, rather than on the estimated body keypoints.

Future work aims to exploit raw inertial data (i.e., raw IMU orientations), together with markerless MoCap, as input to drive a common musculoskeletal model of the persons being analyzed. This should allow to further improve the system's accuracy, while also minimizing the drifting phenomena that are typical of pure inertial MoCap systems. This approach will also allow to directly compare both the estimated joint angles, as well as the estimated body keypoints, since they are defined on the same shared model.

4.4 Conclusions

This chapter presented my research within the *Tracking* level. The contribution in this regard was twofold. In fact, the main focus was on presenting novel algorithms for real-time multi-person BPE and tracking in distributed networks of homogeneous sensors. However, to properly assess the performance of the proposed system, an extensive dataset of movements was acquired.

The dataset, namely *UNIPD-BPE*, contains synchronized RGB-D and inertial data describing the motion of up to four interacting people. The acquisition setup included five Microsoft Azure Kinect cameras, used to record high definition videos, depth, and markerless BPE data from multiple viewpoints, and two Xsens MVN Awinda full-body inertial suits to provide accurate poses of the subjects, as well as raw data from the 17 IMUs required by each suit. The dataset contains both single- and multi-person sequences. Single-person sequences include 15 participants performing a set of 12 ADLs, while multi-person sequences include seven actions performed by up to four interacting people. The recorded sequences aim to push forward the development of markerless BPE and tracking algorithms, as well as multimodal sensor fusion approaches, enabling a variety of applications where accurate and unobtrusive knowledge of human motion is of paramount importance.

After presenting the *UNIPD-BPE* dataset, the chapter analyzed the four modules developed within this level, allowing real-time temporal tracking of multiple people using a network of homogeneous sensors. They consist of: (1) a *Skeleton Tracker* to perform pure frame-by-frame tracking of the detected persons; (2) a *Skeleton Merger* to fuse the detections from multiple cameras, resulting in an enhanced description of the body poses; (3) a *Skeleton Optimizer* to perform a global optimization to ensure consistency of each person's body dimensions; and (4) a *Skeleton Filter* to perform real-time smoothing of the estimated marker positions and link orientations. No assumptions are made about the number of sensors that are used, nor about the number of people who are tracked. Moreover, the generality of the developed algorithms allows the user to select different subsets of modules, depending on the specific application requirements.

The results obtained in the *UNIPD-BPE* dataset showed how the proposed system is robust, being able to correctly assign consistent IDs to each tracked person in all the dataset sequences. At the same time, the developed modules allow to reduce BPE errors by up to $35\,\%$ when compared to a pure tracking-by-detection approach. Finally, all the proposed algorithms are efficient, ensuring real-time performance even when large numbers of people are present in the scene. The results clearly showed how the developed system has the potential of putting the basis to allow accurate and robust multi-person pose estimation and tracking, without the necessity of any sensor or marker on the body, and in real-time.

Table 4.10: Framework accuracy when using the *Tracker* only (T), *Tracker*, *Merger*, and *Filter* (TMF), and *Tracker*, *Merger*, *Optimizer*, and *Filter* (TMOF). Data are presented as RMSE (SD).

N people	**Action**	**T [°]**	**TMF [°]**	**TMOF [°]**	**Δ T-TMF [%]**	**Δ T-TMOF [%]**
1	bend	17.98 (13.48)	14.39 (8.77)	14.36 (8.76)	-19.99% (-34.92%)	-20.13% (-34.99%)
1	cross_arms	17.70 (13.53)	13.82 (8.44)	13.80 (8.46)	-21.91% (-37.64%)	-22.05% (-37.51%)
1	jog	19.15 (17.45)	14.39 (12.20)	14.40 (12.19)	-24.89% (-30.08%)	-24.81% (-30.12%)
1	jump	21.60 (18.76)	16.40 (13.00)	16.44 (13.05)	-24.10% (-30.72%)	-23.90% (-30.43%)
1	point	24.39 (20.14)	21.13 (15.94)	21.21 (16.02)	-13.36% (-20.86%)	-13.02% (-20.46%)
1	sit	16.52 (13.37)	12.40 (8.38)	12.40 (8.28)	-24.98% (-37.26%)	-24.98% (-38.08%)
1	squat	17.31 (14.76)	14.42 (11.31)	14.28 (11.10)	-16.73% (-23.41%)	-17.53% (-24.82%)
1	throw	19.58 (16.11)	14.25 (9.14)	14.26 (9.16)	-27.24% (-43.26%)	-27.16% (-43.15%)
1	walk	16.15 (11.61)	12.98 (6.45)	12.98 (6.45)	-19.61% (-44.44%)	-19.61% (-44.44%)
1	wave	17.36 (13.12)	12.70 (6.08)	12.75 (6.16)	-26.83% (-53.65%)	-26.55% (-53.09%)
1	**mean**	**18.77 (15.23)**	**14.69 (9.97)**	**14.69 (9.96)**	**-21.74% (-34.54%)**	**-21.74% (-34.60%)**
2	circle	16.88 (12.93)	14.00 (8.50)	14.10 (8.62)	-17.06% (-34.27%)	-16.47% (-33.34%)
2	cross	17.00 (13.41)	13.90 (8.96)	13.99 (9.04)	-18.24% (-33.23%)	-17.71% (-32.60%)
2	free_dynamic	16.66 (13.59)	13.05 (8.42)	13.07 (8.49)	-21.68% (-38.03%)	-21.54% (-37.51%)
2	free_static	16.58 (13.91)	12.31 (8.05)	12.35 (8.10)	-25.76% (-42.15%)	-25.51% (-41.77%)
2	in_out	16.56 (13.04)	14.32 (11.66)	14.39 (11.72)	-13.50% (-10.54%)	-13.08% (-10.09%)
2	**mean**	**16.74 (13.38)**	**13.52 (9.12)**	**13.58 (9.19)**	**-19.24% (-31.84%)**	**-18.88% (-31.32%)**
3	circle	16.81 (13.21)	14.22 (9.17)	14.28 (9.28)	-15.40% (-30.62%)	-15.02% (-29.71%)
3	cross	17.67 (14.04)	15.68 (11.28)	15.82 (11.44)	-11.24% (-19.65%)	-10.45% (-18.50%)
3	free_dynamic	18.31 (15.28)	13.79 (9.54)	13.85 (9.64)	-24.68% (-37.55%)	-24.36% (-36.89%)
3	free_static	17.59 (13.21)	14.87 (8.99)	14.78 (8.94)	-15.44% (-31.91%)	-15.98% (-32.31%)
3	in_out	16.59 (13.97)	14.71 (12.27)	14.81 (12.35)	-11.33% (-12.21%)	-10.76% (-11.64%)
3	**mean**	**17.39 (13.94)**	**14.65 (10.25)**	**14.71 (10.33)**	**-15.76% (-26.47%)**	**-15.41% (-25.90%)**
4	circle	18.48 (14.66)	16.51 (12.19)	16.69 (12.37)	-10.67% (-16.83%)	-9.71% (-15.62%)
4	cross	17.91 (14.22)	15.76 (11.60)	15.84 (11.73)	-12.03% (-18.45%)	-11.59% (-17.54%)
4	eight	16.49 (13.03)	14.82 (10.50)	14.95 (10.64)	-10.13% (-19.38%)	-9.32% (-18.36%)
4	free_dynamic	18.96 (16.29)	15.41 (12.04)	15.44 (12.09)	-18.75% (-26.11%)	-18.57% (-25.80%)
4	free_static	17.61 (14.30)	14.53 (10.43)	14.43 (10.35)	-17.51% (-27.12%)	-18.05% (-27.65%)
4	in_out	17.35 (14.08)	15.57 (12.08)	15.55 (12.06)	-10.27% (-14.16%)	-10.36% (-14.36%)
4	**mean**	**17.80 (14.43)**	**15.43 (11.47)**	**15.48 (11.54)**	**-13.31% (-20.51%)**	**-13.03% (-20.03%)**

5 Modeling Level

5.1 Introduction

The last part of my research aimed at bringing together all the work defined within the *Sensing* and *Tracking* levels. The *Sensing* level (Chapter 3) allowed to exploit different typologies of sensors for the measurement of human motion. The standard interfaces used for the communication among all the modules are defined within this level. The *Tracking* level (Chapter 4), on the other hand, focused on multi-sensor fusion among a network of homogeneous sensors (e.g., a camera network). The developed modules allow to ensure temporal consistency of the data obtained from a distributed sensing system while accurately tracking multiple people in real-time. Additional modules are proposed for enhancing the measured motion by merging partial information from different sensors, optimizing the body poses to respect anthropometric constraints, and performing real-time smoothing of the motion trajectories.

However, to fuse data obtained from heterogeneous sources (e.g., marker positions estimated by optoelectronic systems and IMU orientations), all of the measured quantities need to refer to a common underlying model of the subject being analyzed. Thus, the third and last level of my research is the *Modeling* level. The work within this level aims to adapt state-of-the-art biomechanics techniques to allow their use in everyday living and working environments. As a result, the *Modeling* level allows representing motion by means of a musculoskeletal model of the human. The contribution of this level is twofold.

First, using the estimated body poses[1] to drive a musculoskeletal model should allow to increase the achievable accuracy, minimizing the impact of diverse sources of errors inherent in all the three classes of MoCap systems. In fact, errors due to STAs, typical of optoelectronic systems, can be strongly reduced by using the measured marker positions to simulate motion via an IK procedure [3]. Similarly, drifting phenomena can lead to increasing errors in the orientation estimation of an IMU. However, by employing multiple IMUs to simulate motion in a musculoskeletal model, the impact of drifting in the final estimation of the pose can be minimized, allowing for more stable results in the long term [191]. Finally, markerless systems suffer from estimation noise. Moreover, markerless BPE might provide wrong estimations of the body keypoints describing a person's pose, caused by challenging lighting conditions, cluttered environments, or unconventional body poses [192]. In this regard, the use of a model allows to ensure consistency in the estimated body dimensions, as well as to minimize the impact of a wrong keypoint detection on the full-body pose.

Second, the integration of a musculoskeletal model is a key feature that enables the simultaneous use of heterogeneous sensors for the assessment of human motion. In fact, my research was driven by the strong idea that proper multimodal sensor fusion must be performed at the *Modeling* level. Different sensors, MoCap systems, BPE algorithms typically rely on unrelated underlying models (if any). Therefore, a direct fusion of the measured quantities defining a person's body pose is prone to lead to erroneous results. However, by properly defining a unique musculoskeletal model that includes all the measured quantities (i.e., measured/estimated marker positions and measured/estimated IMU orientations), it is possible to actively use all the available data, independently of their source, with a common final goal.

The remainder of the chapter analyzes the main characteristics of the musculoskeletal models used in this work (Section 5.2). While real-time accurate assessment of human motion is the primary goal of my Ph.D., the generality of the selected tools and of the implementations allow to represent and track the motion of any articulated object. The chosen approach relies on an IK optimization with the goal of driving the model

[1]Such poses can either be the ones directly measured within the *Sensing* level or the enhanced poses obtained from multi-sensor networks. In fact, the *Measuring* level can communicate with both the *Sensing* and *Tracking* levels (Figure 2.1).

in the configuration that best matches the experimental data at each time frame. Section 5.3 describes the mathematical principles behind marker- and orientation-based IK optimizations. Finally, Section 5.4 analyzes the modified workflow and the implementation choices to enable real-time multi-person MOB-IK optimizations, allowing to use both markers and IMUs jointly. Preliminary results on the performance achieved using the set of marker positions and link orientations estimated by the Azure Kinect Body Tracking SDK on a full-body musculoskeletal model with 32 DoFs are included.

5.2 Definition of the Model

As analyzed in Section 2.2, OpenSim was the final choice for the representation of a musculoskeletal model in my research. An OpenSim model can represent humans, animals, or, generally, any articulated object. It relies on the concept of components, where each component of the model corresponds to a specific part of the described physical system. Such components can be combined to generate or simulate movement. Seven classes of components cooperate for the definition of a model: reference frames, bodies, joints, constraints, contact geometry, markers, and controllers.

Bodies and *joints* allow to describe a model's skeletal system. Bodies represent rigid segments (each with its own *reference frame*), while joints define how pairs of bodies can move with respect to each other. In this regard, *constraints* can be used to limit a joint's RoM. Forces, either measured or generated by a *controller*, permit to describe motion dynamics. Muscles are modeled as specialized forces that act at multiple points in the connected bodies. Finally, a model can be associated with a specific *contact geometry*, as well as with a generic set of *markers* defined locally to any rigid body.

A short description of the principal components that define any OpenSim model will be presented. Not to deviate from the main subject of this dissertation, only the components used to define the model's skeleton and, thus, to perform an IK optimization (i.e., bodies, joints, constraints, and markers) will be analyzed. For additional information on the other components, the interested reader is referred to the OpenSim documentation [193].

Bodies. Bodies are the primary building block of an OpenSim model. They are used to represent rigid entities in the model that must be connected to other bodies by means of a joint. Thus, each body owns the joint that connects it to its parent body. Figure 5.1 shows a representation of two generic bodies P and C in OpenSim, connected by a joint J. Mathematically, a body in the model represents a moving reference frame with a certain center of mass and inertia. Its motion depends on the motion of its parent body and on the properties of the joint that connects them. A body can include several properties (e.g., its unique name, a mesh file representing its geometry, its mass, the position of its center of mass, and an inertia). Finally, each model automatically defines a special body: the Ground. The Ground body is the root of the model and represents the global reference frame in which the experimental data are expressed.

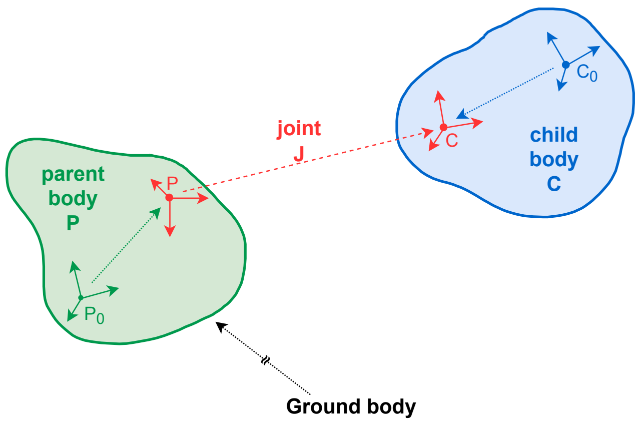


Figure 5.1: Representation of two bodies in OpenSim, described by their reference frames P_0 and C_0, connected by a joint J. The joint describes the relationship between two additional reference frames P and C, defined local to the respective bodies. In the example, the parent body is connected to the Ground body.

Joints. Joints are used to describe the relationships between pairs of bodies. Specifically, a joint defines the kinematic relationship between two generic frames affixed to the parent and child bodies. This is possible by defining a constant transform between the joint frame and the body frame. Several types of joints are available:

- *Weld Joint*: no DoFs, parent and child bodies are fused together;
- *Pin Joint*: one rotation DoF;
- *Slider Joint*: one translation DoF;

- *Ball Joint*: three rotation DoFs;
- *Ellipsoid Joint*: three rotation DoFs and coupled translations such that the movement of the child body traces an ellipsoid about the joint center;
- *Free Joint*: three rotation DoFs and three translation DoFs;
- *Custom Joint*: user-specific DoFs and constraints. Custom joints allow replicating all the aforementioned classes of joints, while also introducing kinematic constraints.

Constraints. Constraints are used in OpenSim to limit the movement between two adjacent bodies. Three different typologies of constraints are supported for the definition of a joint: *Point constraints*, *Weld constraints*, and *Coordinate coupler constraints*. Point constraints are used to fix a point described with respect to two bodies. Thus, no relative rotations between the two bodies are allowed. Weld constraints, on the other hand, fix both the relative locations and orientations of two bodies. Thus, no relative motion is allowed. Finally, coordinate coupler constraints allow the user to define custom constraints. They relate the generalized coordinates of a joint to any other coordinate in the model. Thus, the user must provide a function describing the relation between the two coordinates.

Markers. Markers are a necessary component to perform IK simulations of motion in the model. OpenSim allows the definition of a virtual set of markers in the model. The virtual marker set must match the experimental marker set used for the data acquisition. Markers can be defined local to any body in the model. Thus, a marker is described by its unique name, the body on which is attached, and its location on the body. Multiple markers can be attached to the same body.

Several already defined and validated models are freely available within OpenSim. While the majority describe the human body (or specific parts of the body), the generality of the aforementioned components allowed the community to design a variety of different models, not necessarily limited to human motion analysis. Moreover, the open-source nature of the project results in the possibility to freely alter any available model, if needed.

This is a key feature of OpenSim, since the creation of new accurate human models *ex nihilo* is a complex task that requires in-depth knowledge of human anatomy. Furthermore, new models should be thoroughly validated (possibly with large groups of subjects) to assess their correctness in describing human motion. For these reasons, the majority of the studies on human motion rely on already validated models.

However, generic models cannot be directly used to describe a person's movement. In fact, their body dimensions are typically calculated from the averaged anatomical measurements of multiple cadaver specimens [11], [194], [195]. Thus, before analyzing motion, a generic model requires to be scaled.

Scaling is used to alter the anthropometric characteristics of a model to match a specific subject. It is typically performed by comparing experimental marker data recorded during a static trial to the virtual markers defined in the generic model. A static trial consists of a short recording where the subject is standing in a known static pose. Scaling is then performed by comparing the measured distances of pairs of markers and the corresponding distances in the model. The dimension of the body segment in the model where the pair of markers is placed is linearly scaled such that the distance between the virtual markers matches the measured one. The scaling factors computed in this step are also used to scale mass and inertia, as well as any element attached to the body segments (e.g., muscles and wrapping objects).

5.3 Inverse Kinematics for Biomechanical Modeling

In robotics, the IK problem consists of determining the joint variables of a kinematic chain that correspond to a given pose of the end-effector. As opposed to forward kinematics (FK), where the pose of the end-effector is uniquely determined by the state of each joint, IK introduces several criticalities. The equations are in general non-linear. Thus, a closed-form solution might not be possible. For this reason, multiple solutions may exist (potentially an infinite number of solutions). It is also possible to have no admissible solutions. Nonetheless, the IK problem is of crucial importance to transform a desired end-effector motion specification into the corresponding required joint space motions [196].

Similarly, the IK tools developed in OpenSim take as input the experimental data describing a certain motion and, for each time frame, compute the joint angle values that position the model in the configuration that best matches the experimental data. In this regard, the best match consists of the pose that minimizes the sum of the WLS errors between the virtual experimental markers, IMUs, and/or coordinates in the model.

OpenSim requires three input files to solve IK problems (Figure 5.2). First, a model file describing the analyzed subject must be provided. A model consists of an *.osim* file, which follows the XML syntax rules with a specific grammar, where all its properties are stored. When using generic models, it is crucial to scale the model before the analysis to generate a subject-specific version of the model. The second input consists of a *.trc* file containing all the experimental data recorded during the motion. Finally, an *.xml* setup file specifying the settings to use for the IK is the third and last required input. It contains the weights to be assigned to each marker/IMU, the specific model to be used, the path of the file containing the trial's recordings, the time range over which the IK problem will be solved, and the desired accuracy of the solution (i.e., the number of significant digits to which the solution can be trusted).

The output of an IK optimization is a motion file (*.mot*) containing the generalized coordinates (i.e., the estimated anatomical joint angles and/or translations) describing the model's motion at each recorded time frame.

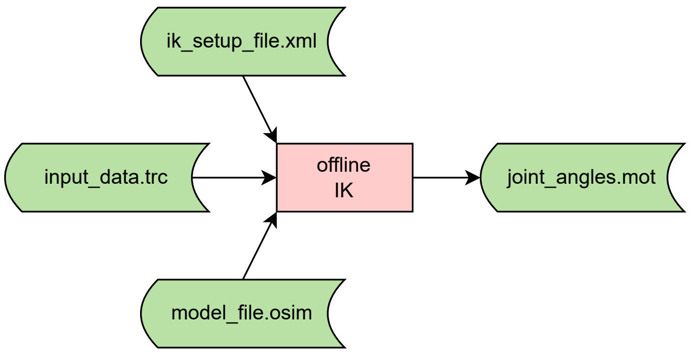


Figure 5.2: Input and output files required for an IK optimization in OpenSim. The green blocks represent files, while the red block represents an offline operation.

Typically, IK analyses are driven by experimental marker data measured by means of highly accurate optoelectronic MoCap systems. However, as discussed in Section 1.2, such systems require delicate hardware and trained personnel. This typically limits their use in specialized laboratory environments.

Due to the latest advances in IMU technology, wearable sensing based on a chain of IMUs worn on the body is considered a valid alternative to optical MoCap [22]. Thus, OpenSim recently introduced the support for orientation-based IK analyses [191] that exploit inertial data instead of markers, inspired by the preceding work of Tagliapietra *et al.* [129].

The rest of this section analyzes the mathematical principles of MB-IK optimizations (Section 5.3.1) and OB-IK optimizations (Section 5.3.2). The modifications required to allow the usage of OpenSim's musculoskeletal models within *Hi-ROS*, achieve real-time performance, enable the simultaneous use of markers and IMUs, and perform multi-person IK optimizations are discussed in Section 5.4.

5.3.1 Marker-Based Inverse Kinematics

The mathematical problem defined in IK optimizations requires two inputs, and a third optional one.

The first input is the description of a model representing the analyzed entity. As discussed in the previous section, human motion analyses typically rely on generic anatomical models. Thus, proper scaling is required to match the model characteristics to those of the specific analyzed subject.

The second input in IK optimizations is the data collected during the trial. In the case of MB-IK, such data consists of a set of trajectories expressing the motion of multiple retroreflective markers applied on the body.

Finally, it is possible to include in the analysis an optional set of known coordinate values. They can represent a subset of joint angles obtained by exploiting an external system (e.g., from built-in MoCap IK), or by using direct techniques (e.g., by using a goniometer). Thus, the angular values assumed by such joints, while contributing to the definition of the body pose, need not to be altered by the optimizer.

The aforementioned inputs are fed to the IK solver, which aims to estimate the set on joint angles that position the model in the admissible configuration (i.e., among the configurations that comply with all the defined model constraints) that best matches the experimental data. This consists of minimizing the overall distance between each experimental marker's position and the corresponding virtual marker on the model.

Figure 5.3 shows an example model to describe full-body motions. The model, created by Rajagopal *et al.* [11], has 23 bodies (not counting ground), 23 joints, and 32 DoFs. Since the usage of any model for IK optimizations requires the definition of a set of markers, 32 markers were placed on the model. Their locations correspond to the body keypoints estimated by the Azure Kinect Body Tracking SDK.

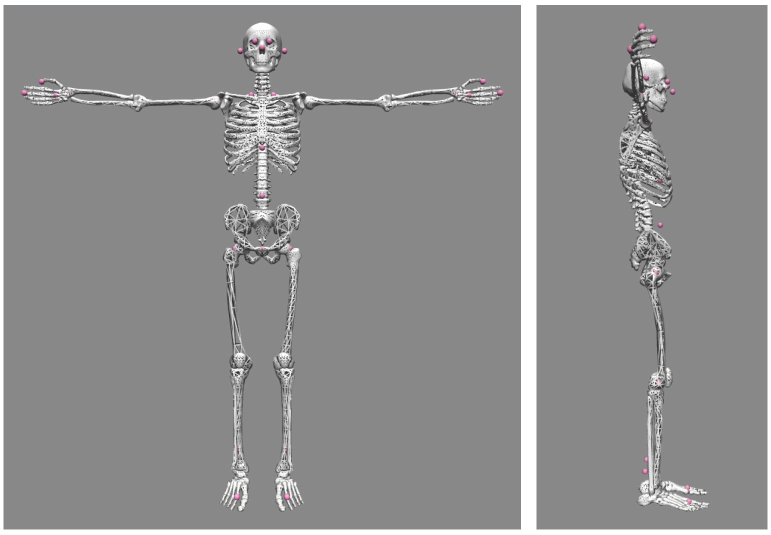


Figure 5.3: Full-body model including the marker positions estimated via markerless MoCap by the Azure Kinect Body Tracking SDK. Markers are represented as pink spheres. (a) Frontal view, (b) side view.

It is important to note that the IK tools solve a separate optimization problem for each recorded time frame. Thus, the poses assumed in older frames do not influence the results of the following frames.[2] The design choices allowing real-time performance, which will be described in Section 5.4, directly depended on the frame-by-frame nature of the optimization.

Mathematically, MB-IK is formalized as the solution of a WLS problem,

[2]Actually, the solution at the previous time frame can be used as an initial guess for the optimization of the current frame. While this might allow for faster convergence to the optimal solution, the estimated joint angles do not directly depend on the previous pose.

with the goal of minimizing the following cost function at each time step:

$$f_M(\mathbf{q}) = \frac{\sum_{i\in\mathcal{M}} w_i \|\mathbf{x}_i^{exp} - \mathbf{x}_i(\mathbf{q})\|^2}{\sum_{i\in\mathcal{M}} w_i} + \frac{\sum_{j\in\mathcal{C}} w_j (q_j^{exp} - q_j)^2}{\sum_{j\in\mathcal{C}} w_j} \tag{5.1}$$

where $\mathbf{q}$ is the vector of generalized coordinates (i.e., the joint angles describing the pose), $\mathbf{x}_i^{exp}$ the experimental position of the i-th marker among the set of markers $\mathcal{M}$, $\mathbf{x}_i(\mathbf{q})$ the position of the corresponding model marker depending on $\mathbf{q}$, q_j^{exp} the a priori value of the j-th coordinate among a set of coordinates $\mathcal{C}$, and q_j the corresponding value obtained from the optimization. Finally, w_i and w_j are weighting coefficients associated with each i-th marker and j-th known coordinate, respectively. Thus, in Equation 5.1, the first term represents the marker errors, while the second term accounts for the known coordinate errors.

Marker errors consist of the squared Euclidean distance between the measured marker position and the one assumed by the corresponding marker in the model when it is positioned using the generalized coordinates computed by the optimization. The weight w_i is used to specify how strongly the i-th marker's error should contribute to the total cost.

On the other hand, coordinate errors are defined as the squared difference between the experimental coordinate values and the joint angles computed by the IK optimization. Similarly to markers, each j-th coordinate has a weight w_j associated with it to represent how strongly that coordinate's error should be minimized.

5.3.2 ORIENTATION-BASED INVERSE KINEMATICS

A similar approach can be used to estimate motion by exploiting a set of IMUs on the body, in place of the markers used in MB-IK.

However, in this case, additional challenges arise. First, by only measuring orientations, it is not possible to scale a model to represent a specific subject's anthropometric characteristics. This does not represent a problem for the estimation of the anatomical joint angles, since the length of two adjacent rigid bodies does not affect their orientation. However, while the estimated pose will correctly describe the subject's motion, the underlying model will not represent the specific person (i.e., segment and muscle lengths, mass, and inertia will have incorrect values).

Second, each IMU's orientation needs to be correctly mapped to the orientation of the corresponding virtual IMU in the model. Thus, a procedure is required to compute a constant offset between the raw experimental orientation measured[3] by an IMU and the orientation of the corresponding virtual IMU. The virtual IMUs of the model are then offset by this constant value before running the IK optimization. To provide correct results, the subject must assume a known static pose during this initial procedure. While scaling is optional, the offset estimation is a required and important step before proceeding with the OB-IK. Imprecise offset values can, in fact, lead to extremely inaccurate estimates of the motion.

Before analyzing the mathematical formulation of the OB-IK problem, it is important to clarify how a virtual IMU can be defined in the model. The careful reader probably noticed that the components of a model presented in Section 5.2 do not include IMUs.

In OpenSim, a virtual IMU is represented by a *Physical Offset Frame*. Physical offset frames can be attached to any body of the model, and they specify a constant transform between the object they are representing and the body frame on which they are attached. To this end, the offset estimation procedure is used to determine the orientation that defines this transform. Figure 5.4 shows the same model depicted in Figure 5.3 where, instead of placing markers, 18 virtual IMUs are defined. To visualize their positions and orientations in the model, they are represented here as orange cuboids.

As in MB-IK, OB-IK also solves a separate optimization problem for each recorded time frame. The WLS problem to be solved, in this case, aims at minimizing the distance between each experimental orientation and the corresponding virtual orientation in the model. Such a distance is quantified based on the angle value described using the axis-angle representation of a 3D orientation.

Let $R^{\mathcal{G}}_{E_i}$ be the rotation matrix describing the orientation of the i-th experimental IMU and $R^{\mathcal{G}}_{V_i}$ the rotation matrix of the corresponding virtual IMU in the model, both expressed with respect to the same global reference frame $\mathcal{G}$. The relative orientation of V_i with respect to E_i is computed as:

$$R^{E_i}_{V_i} = (R^{\mathcal{G}}_{E_i})^T \cdot R^{\mathcal{G}}_{V_i} \tag{5.2}$$

[3]It is important to note that, in this context, the word "measure" is being abused. While IMUs can only *estimate* their orientation, based on direct measures of linear accelerations, angular velocities, and, possibly, magnetic fields, their primary use is, indeed, to provide orientation data.

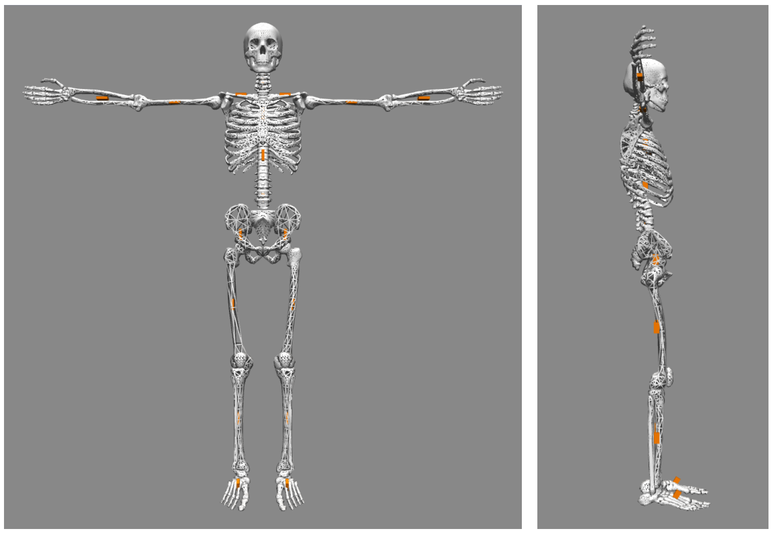

(a) (b)

Figure 5.4: Full-body model including the link orientations estimated via markerless MoCap by the Azure Kinect Body Tracking SDK. Orientations are used as fictional IMUs, here represented as orange cuboids. (a) Frontal view, (b) side view.

The angle θ_i describing the rotation in the axis-angle representation of $R_{V_i}^{E_i}$ is defined as:

$$\theta_i = \arccos\left(\frac{Tr(R_{V_i}^{E_i}) - 1}{2}\right) \tag{5.3}$$

where the $Tr(\cdot)$ operator indicates the trace of a matrix.

Then, the optimization is formalized as the solution of a WLS problem, with the goal of minimizing the following cost function at each time step:

$$f_O(\mathbf{q}) = \frac{\sum_{i \in \mathcal{I}} w_i \cdot \theta_i(\mathbf{q})^2}{\sum_{i \in \mathcal{I}} w_i} \tag{5.4}$$

where $\mathbf{q}$ is the vector of generalized coordinates (i.e., the joint angles describing the pose), $\theta_i(\mathbf{q})$ the distance between the i-th experimental IMU and the corresponding virtual IMU among the set of IMUs $\mathcal{I}$, and w_i the weight used to specify how strongly the i-th IMU's error should contribute to the total cost.

5.4 Real-time Marker-and-Orientation-Based Inverse Kinematics

This section analyzes the design and implementation of the multi-threaded architecture enabling real-time IK optimizations within *Hi-ROS*. First, the mathematical principles behind multimodal IK are presented (Section 5.4.1). The cost functions used in classical MB-IK and OB-IK optimizations were combined to allow the simultaneous use of marker positions and IMU orientations to drive a common musculoskeletal model.

Then, Section 5.4.2 describes the methodology used to achieve real-time performance, enable multimodal sensor fusion of heterogeneous quantities, and support multi-person IK optimizations. Finally, a case study exploiting a set of marker positions and link orientations estimated via markerless MoCap is proposed. The BPE algorithm used consists of the Azure Kinect Body Tracking SDK, since it is able to estimate both the positions of a set of keypoints describing multiple people's poses, and the orientations of their links. Such orientations are then used as fictional IMUs applied on specific body segments. This choice was made to stress the developed system by including a high number of markers and IMUs in the model used for the simulation of motion, as well as to provide test cases with multiple interacting people.

5.4.1 Multimodal Inverse Kinematics

The introduction of a musculoskeletal model of the human within the *Modeling* level required relevant modifications to the original methodology used in OpenSim (Section 5.3). First, to enable multimodal sensor fusion, the possibility of simultaneously using marker and orientation data on the same model is required. This can be achieved by defining a cost function that includes both the terms of Equations 5.1 and 5.4.

Thus, in a MOB-IK optimization, the cost function to be minimized at each time step is defined as:

$$\begin{aligned} f(\mathbf{q}) &= w_M \cdot f_M(\mathbf{q}) + w_O \cdot f_O(\mathbf{q}) \\ &= w_M \left(\frac{\sum_{i\in\mathcal{M}} w_i \|\mathbf{x}_i^{exp} - \mathbf{x}_i(\mathbf{q})\|^2}{\sum_{i\in\mathcal{M}} w_i} + \frac{\sum_{j\in\mathcal{C}} w_j (q_j^{exp} - q_j)^2}{\sum_{j\in\mathcal{C}} w_j} \right) + \\ &\quad + w_O \left(\frac{\sum_{i\in\mathcal{I}} w_i \cdot \theta_i(\mathbf{q})^2}{\sum_{i\in\mathcal{I}} w_i} \right) \end{aligned} \tag{5.5}$$

where $f_M(\mathbf{q})$ is the cost function relative to MB-IK, $f_O(\mathbf{q})$ the cost function relative to OB-IK, and w_M and w_O two weights that allow to specify how strongly the marker positions should contribute to the total cost with respect to the IMU orientations.

Figure 5.5 shows the same full-body model presented in the previous sections, including the 32 markers used for the MB-IK and the 18 IMUs used for the OB-IK. The model represents the joint positions and link orientations that are estimated via markerless MoCap on RGB-D data by exploiting the Azure Kinect Body Tracking SDK.

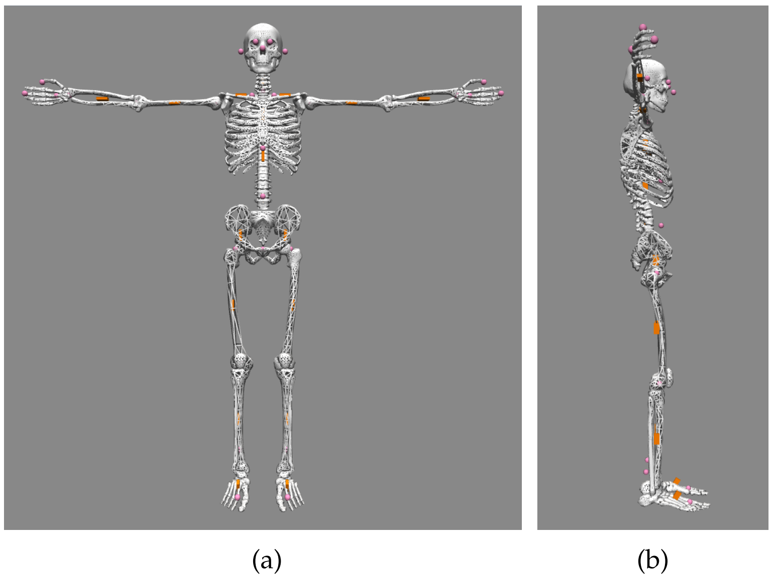


Figure 5.5: Full-body model including the marker positions and the link orientations estimated via markerless MoCap by the Azure Kinect Body Tracking SDK. Markers are represented as pink spheres, while orientations are used as fictional IMUs, here represented as orange cuboids. (a) Frontal view, (b) side view.

5.4.2 Real-Time Multi-Person Inverse Kinematics

Although the aforementioned adjustments enable multimodal fusion of heterogeneous sensors, additional modifications are necessary to allow online sensor-independent feed of data, multi-person analyses, and real-time performance of the IK optimization. As introduced in Section 5.3, OpenSim's IK tools solve a separate optimization problem for each recorded time frame. Thus, multiple frames can be analyzed concurrently by exploiting a multi-threaded architecture.

The developed system aims to expand the real-time implementation proposed in [197], where the authors interfaced a Vicon optoelectronic system with OpenSim to perform real-time IK and ID optimizations. To this end, this dissertation presents a revised multi-threaded architecture for concurrently solving multiple IK optimizations. Furthermore, the proposed algorithms support the simultaneous use of marker positions and IMU orientations to drive a common musculoskeletal model, allowing to perform multimodal sensor fusion of heterogeneous quantities independently of the specific hardware used for the motion assessment. Finally, the developed system expands the capabilities of classical IK approaches by enabling real-time assessments of multiple people's poses, not limiting the analysis to single-person activities.

The proposed multi-threaded architecture follows the programming design pattern of producers-consumers [198]. Producers are responsible for adding data to a shared buffer, while consumers access and modify such data. Only one entity can access the buffer's data at any given time.

Figure 5.6 shows a schematic representation of the developed software architecture. As we can see from the scheme, the input and output files containing the recorded measures are now replaced with *SkeletonGroup* messages[4] that can be exchanged at runtime among modules. To achieve this, the original IK tools provided by OpenSim were adapted to take their input from a memory buffer, instead of requiring pre-recorded data to be stored in a *.trc* file. In Figure 5.6, two separate queues are represented for the input and output data. This was chosen for clarity reasons, not to clutter the scheme excessively. However, the actual software implementation makes use of a unique queue, where each element allows to store both the measured quantities and the estimated body poses.

Each time a new message is received, it is added to the shared queue. One of the available threads takes ownership of the first unconsumed element in the queue and runs a single-frame IK optimization. Depending on the data stored in the message and on the musculoskeletal model being used, either a MB-IK optimization (Equation 5.1), an OB-IK optimization

[4]A detailed description of how a *SkeletonGroup* is defined within *Hi-ROS* is reported in Section 2.4

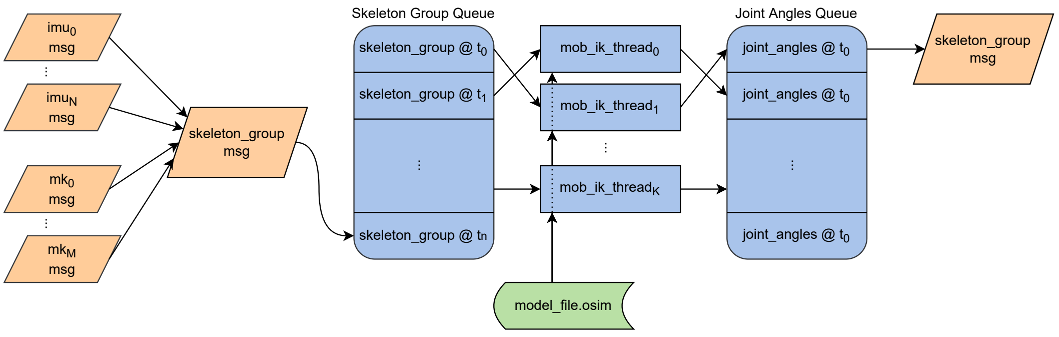


Figure 5.6: Real-time MOB-IK architecture. Input markers and IMUs are represented within a unique *SkeletonGroup* message and pushed into an ordered queue which can be accessed by multiple IK solver threads. The estimated joint angles are then re-converted into a *SkeletonGroup*. The orange boxes represent online messages, the blue boxes real-time operations, and the green box an offline file.

(Equation 5.4), or a MOB-IK optimization (Equation 5.5) can be progressed. As soon as the IK results are computed, an additional publisher thread converts the estimated joint angles describing the body pose back into a *SkeletonGroup* message and publishes the results for online consumption.

The *.xml* setup file used in the original OpenSim workflow to store the IK solver settings (Figure 5.2) is no longer necessary. In the proposed implementation, all the configuration parameters can be set and modified via the ROS parameter server[5]. The only required file is the description of the model to be used, which must be known prior to the analysis. Such a model can either already be scaled to match the specific analyzed subject, or online scaling can be progressed. In the last case, input data containing 3D marker positions are required.

The developed IK solver module is implemented as a ROS node. As a result, multiple solvers – one per tracked person – can be spawned and run concurrently on the same machine. Each solver performs an IK optimization of the specific subject assigned to it. The implementation choices also allow to split the computation among distributed PCs, if needed.

[5]In-depth analysis of the ROS capabilities, including a detailed description of what the parameter server represents in ROS, are presented in Section 2.2.1

5.4.2.1 Multi-Thread Architecture

As presented in Section 2.2.2, one of the reasons that guided the choice of OpenSim concerned its implementation. In fact, both OpenSim and Simbody, a well-known library for large-scale mechanical modeling on which OpenSim is based, are entirely written in C++, one of the most efficient high-level programming languages. This is extremely important, especially in the context of employing such libraries for the real-time assessment of human motion.

In this work, a multi-thread architecture was developed to maximize the achievable framerate of IK optimizations. Each thread works asynchronously from the others to solve a single-frame IK problem. To avoid multiple threads to access and modify the same element in the shared queue, this work relies on the concept of monitors [199]. Monitors are a natural generalization of the objects of object-oriented programming, that encapsulate data and operation declarations within a class. Unlike classes, however, in monitors only one process can execute an operation on a specific object at any one time. Monitors are used to provide a structured concurrent programming primitive that centralizes the responsibility for correctness into the module itself [198].

The multi-thread problem tackled in this work can be represented as a modified producers-consumers problem, with 1 producer and $n + 1$ consumers, where n is the number of IK solver threads to be spawned. The producer consists of the subscriber to the ROS topic where the measured markers and/or IMUs are published as a *SkeletonGroup* message. Each time a new message is received, the producer appends the data to the shared buffer and signals the presence of the new element to the consumers.

The buffer is implemented as an infinite singly-linked queue. Each node of the queue contains four elements: the measured quantities (i.e., marker positions and/or IMU orientations), the joint angles that will describe the model's pose after the IK optimization, a Boolean value indicating if the IK has already been solved, and a pointer to the next element of the queue. When the producer appends a new element, it updates the pointer of the previous last element of the queue, adds the new node, initializes its Boolean value to false, and signals the presence of a new element to the consumers.

Two different typologies of consumers have access to the queue: n IK solvers and 1 ROS publisher. A configurable number n of IK solvers consumes the first available non-processed elements of the queue. Each solver processes a single time frame and performs a single-frame IK optimization. This is possible since OpenSim's IK tools solve a separate optimization problem for each time sample. Thus, the results of each frame do not depend on the output of the previous frames.

When a frame is processed by one of the solvers, the corresponding joint angle values are updated in the node, the Boolean variable indicating if the element was correctly processed is set to true, and the presence of a new processed frame is signaled to the publisher. Then, the publisher checks whether the oldest element in the queue is already processed or not. If it is not processed, the publisher waits until a new element is processed. This is necessary since different frames might require varying computation times. For this reason, depending on the input data frequency and on the available computational resources, it is possible for a newer frame to be processed before one of the older frames.

The developed architecture allows to publish all the input frames in the correct order. The publisher waits until the oldest element is processed, then converts the joint angles describing the body pose into a *SkeletonGroup* message, publishes it in a dedicated topic, and removes the element from the queue.

5.4.2.2 Preliminary Results

Preliminary results were obtained by using a full-body musculoskeletal model to simulate motion from markerless motion capture data. To this end, the data recorded in the *UNIPD-BPE* dataset (Section 4.2) were used as input for the IK optimization. First, the separate outputs of the five Azure Kinect cameras used in the dataset were merged by exploiting the tracking pipeline presented in Section 4.3. This resulted in a set of measurements that contain, at each time frame, the 3D positions of 32 markers placed in specific landmarks on the body and the orientations of 18 links connected by pairs of markers. In fact, the Azure Kinect Body Tracking SDK permits to estimate both these quantities from RGB-D data. Details on the positions of the markers and the hierarchy defining the estimated link orientations are reported in Section 4.2.3.1.

In this work, the estimated orientations were considered as virtual IMUs placed in the corresponding body segments. This allowed to simulate the deployment of heterogeneous MoCap systems (e.g., an optoelectronic system to measure the marker positions and an inertial system to obtain the IMU orientations). In fact, the main objective of this analysis was to assess the real-time performance of the proposed system in complex scenarios (e.g., when a high number of markers and/or IMUs are used, or when multiple people are tracked). The data estimated exploiting markerless MoCap in the *UNIPD-BPE* dataset allowed to test the developed algorithms while simultaneously tracking the poses of four interacting people, each described by a large set of markers and IMUs. It is important to note that, by exploiting standard interfaces for the communication between modules, the developed IK solver algorithm is agnostic to the sensing systems used as input. As a result, marker positions estimated using optoelectronic MoCap or via markerless MoCap are represented within *Hi-ROS* using a unique structure (and the same holds for IMU orientations). Thus, by exploiting a unique system to feed both marker positions and IMU orientations, the computation was not simplified. On the contrary, this allowed to effectively simulate the usage of a high number of heterogeneous sensors, while at the same time providing data describing multiple interacting people.

The full-body musculoskeletal model developed by Rajagopal *et al.* [11] (32 DoFs) was used in the experiments. The virtual markers and IMUs in the model depicted in Figure 5.5 are specifically positioned to match the marker positions and link orientations estimated by the Azure Kinect Body Tracking SDK.

However, before motion estimation, the model needs to be scaled to match the individual anthropometric characteristics of each subject. To this end, the marker positions estimated during the first $2\,\mathrm{s}$ of the trials are used. Once the models are properly scaled, the estimated markers and IMUs are fed to the multi-thread MOB-IK solver module and processed online.

Performance was assessed on a desktop PC equipped with a 16-core Intel Core i9-9900K CPU @ $3.60\,\mathrm{GHz}$ and $64\,\mathrm{GB}$ of RAM. The same motion data were used as input, while the number of threads was varied between 1 and 32. The experiments were repeated in different test cases, in which one, two, three, or four subjects were analyzed. The solver accuracy was set to 1×10^{-4} across all the experiments. The results are visualized in Figure 5.7 and reported (up to 16 threads) in Table 5.1.

Focusing on single-person analyses, the single-thread implementation, with this specific configuration (i.e., musculoskeltal model, number of DoFs, number of markers, number of IMUs, solver accuracy) requires on average $17.23\,\mathrm{ms}$ per frame, allowing a maximum frequency of $58.06\,\mathrm{Hz}$.

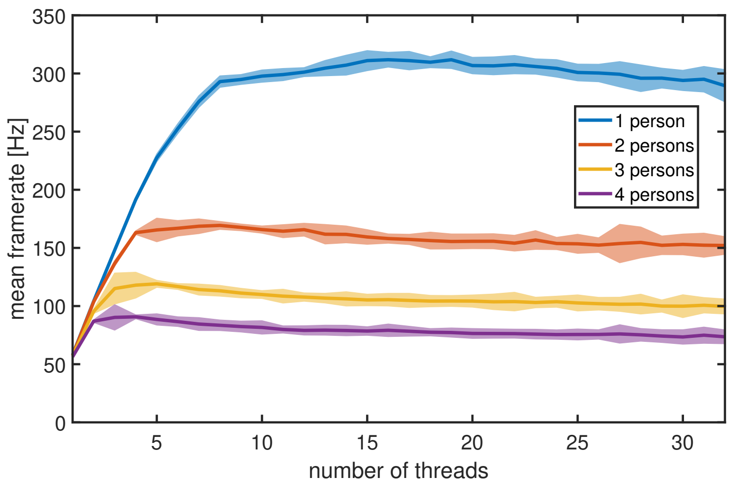


Figure 5.7: Maximum achievable framerate when varying the number of IK solver threads and the number of analyzed subjects using the full-body model from [11]. A set of 32 markers and 18 IMUs corresponding to the output of the Azure Kinect Body Tracking SDK was manually defined. The shaded areas indicate ±1 SD.

We can notice an almost linear increase in the achievable framerate up to 8 threads. Then, there is still a slight increase up to 16 threads, where a throughput of $311.82\,\mathrm{Hz}$ is obtained, followed by a slow decrease with a higher number of threads. This was expected since the number of processor cores of the PC used for the experiments is 16. Thus, a higher number of threads does not produce a substantial performance increase.

Nonetheless, the developed architecture allows for a performance ∼5.4 times faster than that of the original single-thread implementation. In addition, online input data can be obtained by any typology of MoCap system by simply representing the measured quantities as a *SkeletonGroup* message.

Proceeding with the analysis, we can notice a reduced performance gain in the range between 8 and 16 threads. This is explained by the fact that, despite each thread running in parallel, the access to the shared resource (i.e., the queue containing the input frames) is regulated by the monitor, preventing multiple threads from accessing the same element at once. Although the time required by each thread to process one frame does not depend on the total number of threads, the additional time to protect the access to the shared resources increases proportionally to the number of consumers.

Table 5.1: Maximum achievable framerate for different numbers of IK solver threads and analyzed subjects using the full-body model from [11], where 32 markers and 18 IMUs were defined. Bold values indicate the highest framerates. Data are presented as mean (SD).

# Threads	**Framerate** [Hz]			
	1 person	**2 persons**	**3 persons**	**4 persons**
1	58.06 (0.48)	57.60 (0.61)	57.31 (0.70)	56.34 (0.51)
2	104.11 (0.85)	103.59 (1.02)	95.41 (1.74)	86.99 (1.05)
3	148.07 (1.41)	136.64 (2.32)	115.07 (13.20)	90.25 (10.83)
4	191.59 (1.88)	163.12 (1.50)	117.98 (10.98)	**90.76 (1.58)**
5	227.49 (3.22)	165.44 (10.03)	**119.11 (2.90)**	88.50 (4.73)
6	252.59 (4.14)	166.88 (6.54)	116.78 (2.43)	86.59 (4.05)
7	275.82 (4.88)	168.63 (6.22)	114.18 (4.64)	84.52 (5.42)
8	293.00 (4.81)	**169.34 (3.32)**	113.13 (4.49)	83.46 (4.43)
9	294.81 (4.16)	167.65 (2.77)	111.16 (3.99)	82.37 (4.57)
10	297.73 (5.18)	165.75 (3.00)	109.90 (3.27)	81.62 (5.64)
11	299.04 (5.13)	164.37 (5.48)	108.52 (5.84)	79.86 (2.98)
12	301.15 (3.76)	165.67 (5.41)	107.73 (3.48)	79.07 (3.87)
13	304.58 (6.50)	161.77 (8.26)	106.82 (4.06)	79.27 (4.17)
14	307.15 (8.47)	161.63 (7.12)	106.17 (6.02)	78.97 (4.39)
15	311.04 (8.52)	159.40 (6.18)	105.24 (4.84)	78.52 (3.58)
16	**311.82 (6.25)**	158.03 (3.87)	105.49 (5.32)	79.16 (5.28)

When the number of threads exceeds the number of processor cores, no performance gain can be obtained. On the contrary, the performance degrades, due to the increased time required to control the access to the buffer. Thus, the overall throughput of the system gradually starts to decrease.

Finally, it is worth analyzing the delay introduced by the IK optimization. While multiple input frames can be processed in parallel, the time required by each IK solver in the developed multi-thread architecture is independent of the number of threads. Thus, each frame will always require on average $17.23\,\mathrm{ms}$ to be processed, introducing an approximately constant delay between the motion measurement and the IK result. These values, however, are well within the range to consider a system as real-time [200].

The performances obtained when analyzing multiple subjects confirm the previous analysis. Multi-person assessments were conducted by running multiple IK solvers on the same PC used in the single-person experiments. From Table 5.1, we can notice how the highest framerates are achieved with 8 threads when tracking 2 people, 5 threads when tracking 3 people, and 4 threads when tracking 4 people. This was expected, since the reported number of threads corresponds to the threads spawned for the analysis of each subject. Thus, peak performance, similarly to the single-person case, is always achieved with a total number of threads not greater than the number of processor cores (in this case, 16).

Finally, we can notice how the performance decrease is almost proportional to the number of analyzed subjects. However, taking as an example the two-person case, when using 8 threads it was possible to achieve $169.34\,\mathrm{Hz}$ per subject, resulting in an overall throughput of $338.68\,\mathrm{Hz}$, which is slightly higher than the single-person case with 16 threads. This can be explained by the fact that, with two persons, two independent queues are present. Thus, the time required to protect the access to the shared resources is lower than in the single-person case, where a double number of threads need to have access to the same queue.

5.5 CONCLUSIONS

This chapter analyzed the typical workflow used in state-of-the-art biomechanical analyzes, together with the novel methodologies and algorithms I developed to enable real-time multimodal IK optimizations of multiple people's poses.

The first part of the chapter was dedicated to presenting how the musculoskeletal models exploited in this work are defined. Despite being primarily used for human motion assessments, such models allow to represent any system of rigid bodies connected by joints, allowing maximum flexibility of the proposed algorithms.

Then, the mathematical principles behind MB-IK and OB-IK optimizations were analyzed. The procedure is formalized as a WLS optimization problem with the goal of minimizing the distance between all the experimental marker positions (or IMU orientations) and the corresponding virtual markers (or IMUs) defined in the model. However, the algorithms used to simulate the body movement are designed for offline analyses of single subjects, expecting input data to be already acquired and properly post-processed before reconstructing the motion.

Thus, the dissertation advanced by discussing the modifications required to enable simultaneous feed of heterogeneous quantities and real-time multi-person analyses. First, the modified WLS problem accepting both marker positions and IMU orientations was formalized. Subsequently, the architecture defined to enable real-time multimodal assessments was presented. The proposed multi-threaded system follows the programming design pattern of producers-consumers. A single producer (the ROS subscriber to the *SkeletonGroup* messages describing the persons' poses) is responsible for adding data to a shared buffer, while multiple consumers (the IK solver threads) access and modify such data. Multiple instances of IK solvers can run concurrently on the same machine, or in a distributed network of PCs, allowing to assess the motion of any number of subjects in real-time.

In this regard, the performances of the developed system were assessed by varying the number of spawned threads for the analyses of up to four people simultaneously. The results showed how the IK optimizations were able to achieve consistent real-time performance, with a maximum framerate of 312 Hz when analyzing one person's motion, and 91 Hz with four persons on the scene.

To conclude, my work within the *Modeling* level has the potential to push forward, via a multidisciplinary approach, the state-of-the-art on accurate unobtrusive assessment of human motion in unconstrained environments. Indeed, it enables to maximize the BPE accuracy with the simultaneous use of heterogeneous data and, at the same time, it allows to seamlessly select the optimal subset of sensors to be used. As a result, the proposed algorithms have the ability to enable the development of a variety of emerging applications, requiring precise knowledge of multiple people's poses, with a minimal number of on-body sensors, and in real-time.

6 Applications Enabled by Hi-ROS

Part of the work presented in this chapter has been published or submitted as scientific papers [168], [201], [202].
I have made a substantial and principal contribution in the conception and design of these studies, related software development, analyses and interpretations of the results, drafting, and critical revision of the final manuscripts.
Co-authors' permissions for the inclusion of the studies in this dissertation have been obtained.

6.1 Introduction

The effectiveness of the proposed framework is demonstrated through three case studies enabled by *Hi-ROS* in the contexts of Industry 5.0 and HRI. The first application (Section 6.2) aims to train a cobot to perform a task in a shared workspace, while efficiently avoiding a human operator, without the necessity of stopping its motion. In the performed experiments, *Hi-ROS* is used to estimate the real-time pose of the operator, feeding such information to the cobot. To ensure the safety of the human, the estimation must be accurate and, at the same time, robust, throughout the whole trial. The obtained results show how the robot is capable of correctly perceiving and avoiding the person.

The second application (Section 6.3) consists of the development of a platform for real-time ergonomic analyses and training. The platform allows to calculate multiple ergonomic scores in real-time based on the pose assumed by the worker. In addition, such analysis can be performed for multiple people simultaneously. The platform is designed to support any typology of MoCap system. This characteristic was achieved by exploiting the *Hi-ROS* framework for the assessment of the body poses.

Finally, the third application (Section 6.4), which is currently under development, aims to create a real-time control loop to give immediate feedback to a manual operator while performing a series of tasks. The proposed system is based on a set of control volumes that are built around specific positions on the workstation. In this regard, the location and pose of the operator are acquired in real-time by combining the feeds of multiple RGB-D cameras, required to cover the complete working area.

6.2 Feedback Motion Planning in Human-robot Shared Workspaces via Deep Reinforcement Learning

This work investigates the usage of a feedback motion planner based on DRL for human-robot coexistence (i.e., humans and robots share the same workspace, but execute separate tasks) in industrial contexts. In particular, the focus is on tasks in which humans and industrial manipulators share the workspace while operating parallel tasks without synchronization. The proposed approach models the feedback motion planning problem as a Markov Decision Process and applies DRL to learn a policy capable of well approximating the optimal feedback motion planner, defined by the reward function. The reward function takes into account three objectives: (1) reaching the target position, (2) avoiding collisions, and (3) minimizing the required actions.

Human-task independence and subject independence are obtained by training the DRL with a pseudo-random occupancy volume that models the human occupancy in the workspace. In this way, different body dimensions can be represented by varying the size of the occupancy volume describing the operator's location. This allows to address the problem of describing a large set of possible human tasks, including the high variance in their execution and of different persons' anthropometric characteristics.

Detailed analyses of the design of the DRL policy, as well as of the developed simulated environment to train the robotic agent, go beyond the scope of this dissertation. Therefore, the rest of this section will focus on the experimental validation of the proposed methodology and, specifically, on how the human operator is perceived by the robot for online re-planning of its motion.

The experimental setup consists of a $1.2\,\mathrm{x}0.6\,\mathrm{m}$ area that can be reached by the manipulator, defined within a $2.5\,\mathrm{x}1.0\,\mathrm{m}$ workspace where the human operator can freely move. An RGB-D camera (Microsoft Kinect v2) is integrated into the system and its pose with respect to the reference frame of the robot calibrated using the system proposed in [203]. For each trial, the target to be reached by the robot is defined by placing an AprilTag marker [204] at a random location within the goal space region. The pose of the operator is estimated via markerless MoCap by exploiting the modules presented in Section 4.3. This allowed to obtain robust accurate BPE and tracking of the human pose (including absolute position and velocity of the body centroid), in real-time, at a frequency of $30\,\mathrm{Hz}$ (maximum framerate of the Kinect).

Specifically, OpenPose [28] is used as detector. The 2D raw detections are acquired and reprojected in real-time by exploiting the depth map estimated by the Kinect camera to obtain a 3D representation of the operator's pose. The pose is then expressed with respect to the global reference frame defined in the initial calibration of the experimental setup. Finally, a safety bounding volume is defined, from the full-body pose, as the minimum volume containing all the 3D keypoints of the tracked person. The computation is based on an SVD approach, which computes the minimum-volume bounding box. The 3D absolute position and velocity of the volume's centroid, together with the radius of the cylindrical safety volume containing the human, are then fed to the robot in real-time.

During the experimental runs, the developed framework allowed to accurately track the operator's pose and feed such information to the robot in real-time, without missing frames. This was a key element in this work, since communication delays, or wrong estimations of the pose, would have led to an incorrect planning of the robot motion, with the risk of having a collision with the human operator.

The obtained results showed how the developed policy was able to correctly avoid the operator, even in the experiments where the human actively interfered with the robot. This is an additional demonstration of the *Hi-ROS* framework's capabilities. In fact, the agent (i.e., the robot) was trained in a simulated environment, with ideal knowledge of the operator's pose, without delays or measurement noise. When shifting from simulated to real environment, there was no degradation of the performance. Thus, we can conclude that the modules developed within my research were able to provide accurate markerless motion data, in real-time, and with negligible delays.

6.3 A Real-Time Platform for Full-Body Ergonomic Assessment and Feedback in Manufacturing and Logistics Systems

While the first applications focused on providing information on human movement to a robotic agent in the context of HRI, this second case study aims to ensure the well-being and safety of a worker by monitoring and analyzing their ergonomics during assembly, manufacturing, and picking operations. Specifically, this section presents an innovative digital platform for ergonomics that provides real-time ergonomic assessments, posture feedback to workers, and productivity key performance indicators.

In its current state, the proposed system, namely *WEM-Platform*, evaluates eight NIOSH angles (National Institute for Occupational Safety and Health [205]) and four ergonomic risk indexes: RULA (Rapid Upper Limb Assessment [206]), REBA (Rapid Entire Body Assessment [207]), OWAS (Ovako Working posture Assessment System [208]), and PERA (Postural Ergonomic Risk Assessment [209]). They were selected for their versatility and extensive use in manual assembly environments. Moreover, their computation is mainly based on the postures assumed while performing the tasks, requiring only few additional information to be manually added in the software before each analysis. For these reasons, they represent an appropriate set of ergonomic indexes to be computed in real-time from the anatomical joint angles estimated by any typology of MoCap systems.

As in the previous section, the implementation details of the platform will be omitted from this dissertation, while focusing on the perception of the human required as input by the system. In-depth information on the choices behind the *WEM-Platform* design can be found in [168].

The *WEM-Platform* is designed to accept any typology of hardware that allows a real-time estimation of the operator's pose. The only requirement is an interpreter for its input. The possibility to use any MoCap system solution based on markerless MoCap, inertial MoCap, or optoelectronic

MoCap, allows the system to be extremely flexible and to comply with any environment, thus not limiting the monitoring and feedback capabilities to mere laboratory setups. This was achieved by exploiting *Hi-ROS* as the interface between the specific sensing device and the developed platform.

The validity of the proposed system was assessed with a laboratory case study focused on the assembly of medium-sized objects (Figure 6.1). To this end, a volunteer was involved in the assembly process of a bedside table. Only the assembly process for the bedside table frame was taken into account, as the drawers were considered as already sub-assembled components. His pose was estimated in real-time using an inertial suit (Xsens MVN Awinda), and communication with the *WEM-Platform* was achieved by exploiting the standard interfaces defined within *Hi-ROS* in Section 2.4. This allowed to compute and feed to the developed platform the set of anatomical joint angles required for the computation of the four aforementioned ergonomic indexes.

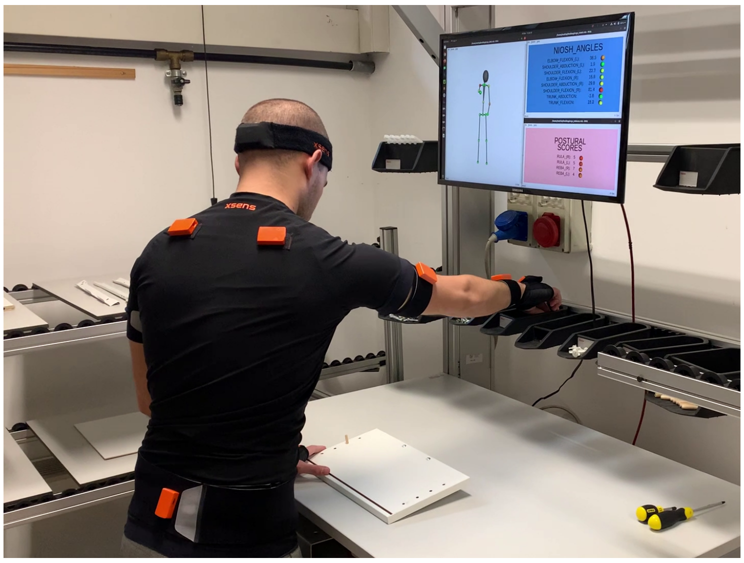


Figure 6.1: Experimental setup used for the assessment of the *WEM-Platform*. The volunteer is wearing a full-body Xsens MVN Awinda inertial suit while assembling a bedside table.

To assess the accuracy of the computation, 10 frames were extracted from the 7 min cycle time required for the assembly. The frames were selected to represent activities characterized by a high level of repeatability.

The ergonomic scores relative to such frames were then compared with those obtained both by computing ergonomic indexes with traditional post-processed video recording evaluations by an expert and by exploiting the Siemens Jack software [210], a well-known commercial simulation tool for ergonomic assessments.

The results for the proposed platform were particularly promising, showing an average difference of $1.6\,\%$ between the scores computed in real-time by the *WEM-Platform* and those calculated by the expert. Similarly, between the *WEM-Platform* scores and those computed by the Siemens Jack software, the average difference was equal to $3.2\,\%$. Thus, the scores computed with the proposed platform agreed with the analyses of both an expert and the Jack software. In addition, the *WEM-Platform* was able to provide multiple ergonomic indexes, and in real-time.

In few cases, the scores evaluated by the *WEM-Platform* were slightly higher than the manually calculated ones. This discrepancy occurred when the anatomical joint angles were close to an index threshold[1]. In fact, while the observer may wrongly classify an angle, with the high precision in the joint angle estimation obtained within *Hi-ROS*, the angle always falls into the correct range. Indeed, a few degrees (or even tens of degrees) can have a remarkable impact on the final score.

The obtained results show how *Hi-ROS* could successfully enable accurate ergonomic evaluations in unconstrained environments and in real-time. The proposed application was able to correctly compute four ergonomic risk indexes and eight NIOSH angles in real-time, as well as to provide visual feedback to the workers describing their postural risks independently of the technology used to assess human motion. Furthermore, the design choices behind the framework I developed within my Ph.D. recently allowed to extend the *WEM-Platform* capabilities to support real-time ergonomics evaluation and feedback in multi-manned assembly stations.

[1]The overall ergonomic scores combine the results of several subscores that depend on the specific range in which a certain joint angle falls. Interested readers are referred to [206]–[209].

6.4 A Closed-Loop Control for Manual Assembly Monitoring

Finally, the third application aims to create a real-time control loop to give immediate feedback to a manual operator while performing a series of tasks. The system is currently being implemented on a manual workstation at the Logistics Laboratory of the University of Padova.

In this case, the real-time location and pose of the operator are required to understand the typology and order of tasks being performed, based on a set of control volumes built around specific positions within the workstation.

The experimental setup will include a network of RGB-D cameras (Intel RealSense, Intel Corp., Santa Clara, CA, USA [121]) to estimate the pose of the operator using markerless MoCap. This will take advantage of all the *Hi-ROS* modules proposed within the *Tracking* level (Section 4.3). The real-time full-body poses will then be combined with a set of control volumes that are built around specific positions on the workstation (e.g., in the storage bins where the picking operations are performed). In this way, it will be possible to check whether a specific body part is located in a certain area (e.g., when the wrists approach the workbench dedicated to an assembly task, to indicate the onset of the task).

The approach relying on control volumes can also be applied to check the correctness of the task sequence of particular assembly operations. As an example, it will be possible to detect if the operator is picking an item from the wrong box, or if a required task was omitted.

Ideally, as soon as an error is detected, some sort of feedback will be immediately provided to the operator. The best solution is currently being discussed, as different approaches are possible. As an example, feedback could be achieved by prompting a message on a screen mounted on the workstation, through augmented reality, or by means of the vibration of a bracelet.

The final target of the complete multi-camera tracking and feedback system is to allow accurate monitoring of the tasks performed in the entire workstation. Thus, the assessment of human motion needs to be performed within several square meters of walking area in a cluttered environment affected by strong occlusions (e.g., workstations, racks, semi-assembled products, etc.). The final setup could also be used to study the learning curves of different operators aided by the proposed control loop. At the same time, it will be possible to compare the obtained results with the curves of traditional learning approaches.

6.5 CONCLUSIONS

This chapter presented three applications enabled by the algorithms developed within my Ph.D. While focusing on different topics, all the applications belong to the HRI and Industry 5.0 contexts and require accurate real-time data describing one or multiple people's body poses. The obtained results demonstrated the effectiveness of the *Hi-ROS* framework when employing different sensing systems in heterogeneous scenarios.

The first application investigated the usage of a feedback motion planner based on DRL in human-robot shared workspaces. The system was designed for applications where humans and robots share the same workspace, but execute parallel asynchronous tasks. The goal consisted in training a collaborative robot to re-plan its motion, in real-time, to avoid possible collisions with a human operator while reaching a predefined target. In this work, *Hi-ROS* was used to obtain robust accurate real-time BPE and tracking of the operator's pose, by exploiting a Kinect v2 RGB-D camera. During the experimental runs, the developed framework was able to correctly track the operator's pose and feed this information to the robot, without missing frames. This was a key element in this work, since delays or wrong pose estimations would have led to incorrect planning of the robot motion, with possible collision with the human.

The focus then shifted to ensuring the operator's well-being, with the development and validation of a platform for the real-time assessment of workers' ergonomics. The proposed platform allows to evaluate several ergonomic indexes (i.e., RULA, REBA, OWAS, and PERA) in real-time, while also providing visual feedback on the correctness of the assumed poses to one or multiple workers. The system supports different typologies of sensing systems for the real-time estimation of the body poses. This was easily achieved by exploiting *Hi-ROS* as the interface between the specific sensing device and the developed ergonomic tools. The results obtained in the proposed case study, consisting of the assembly process of a bedside table, showed how the developed platform was able to accurately compute the aforementioned indexes throughout the whole task.

Finally, the third work presented a system that is currently being implemented in the Logistics Laboratory of the University of Padova. The goal, in this case, is to give a worker immediate feedback describing how the set of assigned tasks is being performed. To achieve that, a real-time control loop is being designed based on a set of control volumes defined at specific locations within the workstation. This will require the operator's full-body pose to be estimated within several square meters of walking area in a cluttered environment affected by strong occlusions. To this end, all the modules developed within the *Tracking* level of this dissertation will be used. This will allow to provide to the operator a feedback (i.e., by prompting a message on a screen mounted on the workstation, through augmented reality, or by means of the vibration of a bracelet) to signal possible inaccuracies in the task sequence as soon as an error is detected.

7 Final Discussion

The core objective of the research activities that I conducted during my Ph.D. aimed at taking a step forward on enabling accurate assessment of human motion in unconstrained environments and in real-time. To address this complex challenge, my work focused on the definition and implementation of novel algorithms and methodologies to seamlessly interface heterogeneous sensors, enhance the body pose estimation precision when exploiting multi-sensor networks, and perform real-time IK optimizations based on highly accurate musculoskeletal models of the analyzed persons. The extensive work on these directions converged on the development of an open-source efficient, flexible, modular framework, namely *Hi-ROS*, with the final goal of enabling accurate and real-time assessment of human motion independently of the employed sensing devices and of the environments where the acquisitions are progressed.

This required an in-depth investigation of the characteristics of different state-of-the-art technologies for the assessment of human motion, which led to the definition of common interfaces to enable the usage of the developed tools on heterogeneous quantities measured/estimated by different sensing systems. The choices driving the definition of how a person is represented within *Hi-ROS* aimed at maximizing the proposed framework's flexibility. In fact, the number and typology of sensors can easily be modified during the execution of the final application itself, without requiring any modification of the software enabling the estimation of the body pose. Thus, the maintenance and upgrade times of a system can be markedly reduced. This key feature enables the usage of the proposed tools in different scenarios, since both the hardware and the software components can be easily tuned to the specific needs.

My work proceeded with the development of novel algorithms for robust multi-person tracking in distributed camera networks. Several challenges arising from the aforementioned setups were identified and solved to maximize the achievable accuracy while maintaining real-time performance.

Finally, the last contribution of my research allowed to adapt state-of-the-art methodologies borrowed from the biomechanics community to enable their usage in any application benefiting from real-time knowledge of how one (or multiple) persons are behaving. In this regard, highly accurate musculoskeletal models describing the human body are included in the framework and used to simulate motion based on different physical quantities measured (or estimated) by any typology of MoCap system. This is the last key feature required to maximize the flexibility of the tools developed within my Ph.D. It allows to seamlessly select the optimal subset of sensors to be used, as well as to perform robust multimodal sensor fusion among heterogeneous quantities since all the data refer to the same underlying model.

The correctness and effectiveness of the developed tools were proven in three distinct applications. The first focused on online safe re-planning of a manipulator in human-robot shared workspaces to ensure the operator's safety. The second addressed multi-person real-time ergonomic assessment and feedback for the prevention of work-related musculoskeletal disorders. Finally, the third application, which is currently under development, aims to create a real-time control loop to provide immediate feedback to a manual operator while performing a generic set of tasks.

These initial test cases clearly demonstrate the flexibility of the proposed tools. Thus, the developed framework has the potential to enable a variety of new emerging applications requiring precise knowledge of the human pose in unpredictable environments, with a minimal number of on-body sensors, and in real-time.

7.1 SUMMARY

My Ph.D. research aimed to push forward, via a multidisciplinary approach, the state-of-the-art in accurate, real-time, and unobtrusive assessment of human motion in unconstrained environments. To achieve this complex objective, I divided my work into three macro levels: the *Sensing* level, to seamlessly integrate heterogeneous sensors within the proposed framework; the *Tracking* level, to maximize the accuracy in distributed multi-sensor networks; and the *Modeling* level, to enable multimodal sensor fusion via musculoskeletal modeling.

Chapter 1 presented the current challenges in the assessment of human motion, as well as the solutions proposed within my Ph.D. After an initial overview of the research scenario, the available technologies are analyzed, with a focus on their characteristics and limitations. Human MoCap systems can be divided into three categories: optoelectronic systems, requiring a set of retroreflective markers to be applied on the body; inertial systems, based on a chain of IMUs worn by the subject; and markerless systems, which do not require any sensor or marker on the body and rely on deep learning algorithms for the estimation of motion.

The first gained the role of the *de-facto* standard in biomechanical analyses, due to their submillimeter precision. However, optoelectronic systems require delicate hardware and specialized personnel for the correct positioning of the markers on the body. Therefore, they are typically confined to dedicated laboratories, preventing extensive use in unconstrained environments.

Inertial systems, on the other hand, exploit multiple IMUs worn on the body and combine the estimated orientations with a personalized model of the subject to reconstruct their motion. Despite their accuracy approaching that of optoelectronic systems without the necessity of dedicated laboratories or specialized personnel, inertial MoCap still requires several sensors to be applied to the body. This, especially during prolonged sessions, can cause discomfort and limit the user's dexterity.

Finally, markerless MoCap does not require any sensor or marker on the body. Such systems rely on deep learning algorithms for the estimation of the body poses based on RGB or depth information of the scene. However, the achievable accuracy is at best one order of magnitude lower than that of the other systems.

None of the aforementioned systems is able to provide highly accurate measurements, without requiring complex hardware setups nor hindering the freedom of movement in any way, and in real-time. Thus, my research followed two parallel but interlaced directions: maximize the sensorless accuracy, and minimize the number of required sensors. The two approaches can be combined to identify the best trade-off between the required precision and the maximum number of sensors allowed by the specific application.

To this end, the rest of the chapter analyzed how my research was organized, introducing the three macro levels that I defined within my research: the *Sensing* level, the *Tracking* level, and the *Modeling* level.

Before investigating the design choices made within the three aforementioned levels, an additional preliminary analysis is required. In fact, all the work developed in my Ph.D. required the selection and use of state-of-the-art tools borrowed from the robotics and biomechanics communities. The definition of such tools, together with a detailed analysis of their characteristics, were reported in Chapter 2.

The first tool consists of ROS, a middleware that rapidly gained the role of the *de-facto* standard for advanced distributed robotics applications. ROS provides high-level tools to simplify the development of complex robotics applications requiring efficient online communication among distributed sensors. In this work, ROS was selected to enable efficient, reliable, and robust communication between all the different hardware and software components defined in the proposed framework. Furthermore, due to the widespread usage of ROS, a large number of hardware manufacturers release official drivers to interface their products with the middleware. As a result, several different brands of sensors (as well as combinations of sensors) can be seamlessly used within the proposed framework, resulting in great flexibility of the developed tools.

The algorithms developed for real-time model-based IK optimizations, on the other hand, rely on OpenSim, a well-known library for biomechanical analyses. OpenSim was chosen for multiple reasons. First, it includes several already validated musculoskeletal models. This is a key feature of OpenSim, as the creation of new accurate human models *ex nihilo* is a complex task that requires a deep understanding of human anatomy. The second reason is the open-source nature of the project, which allows modifying any model, as well as the source code of its libraries, to alter the developed algorithms to comply with different application requirements. Finally, the last motivation is the reliability, precision, and efficiency of the tools provided for IK optimizations. In fact, this was a necessary requirement to achieve real-time performance of the developed algorithms.

The chapter continued by analyzing the structure of the proposed framework, which mimics the three levels on which I divided my research. First, the typologies of sensors supported in the *Sensing* level and the common interfaces defined for the representation of a person's pose were presented. Then, an analysis of the main challenges faced in multi-sensor networks was reported, together with the characteristics of the modules developed within the *Tracking* level to tackle such problems. Finally, a description of the workflow typically used in biomechanical analyses, with the modified methodology proposed within the *Modeling* level to enable real-time IK optimizations was presented.

After describing the structure and characteristics of the developed framework, Chapter 3 dived into the details of the *Sensing* level. This chapter focused on the different typologies of sensors that can be used to measure physical quantities describing a person's pose. Specifically, marker positions placed on specific body landmarks and IMU orientations attached to the body segments. Such quantities can be measured by optoelectronic systems (markers) and inertial MoCap (IMUs), or estimated via markerless BPE.

Due to the limitations of optoelectronic systems, which typically confine their usage to specialized laboratories, the dissertation primarily focused on markerless and inertial MoCap. To this end, the different typologies of cameras supported within *Hi-ROS* were analyzed. They are divided into RGB and RGB-D cameras, depending on whether depth information is produced or not. Special attention was given to the different technologies used to estimate the depth of the scene, divided into stereo vision, structured light, and ToF sensors.

The focus then shifted to the usage of IMUs for the assessment of human motion. While in markerless MoCap a single camera can be sufficient to estimate the full-body motion of a person, inertial MoCap requires several sensors. However, the usage of multiple (and, possibly, wireless) sensors introduces new challenges. In fact, the measurements might refer to different time frames, and communication (and/or computation) delays can indeed alter the order in which distributed data are received and combined. To this end, an efficient implementation of a driver was proposed to interface multiple Xsens MTw Awinda IMUs within *Hi-ROS*. The developed software also includes an accurate synchronization algorithm to ensure that each time frame contains consistent data from all the sensors required by the MoCap system.

Finally, an in-depth investigation of the performance of three different manufacturers of IMUs (i.e., Xsens, MbientLab, and InvenSense) for human motion tracking was proposed. A direct driver servomotor was used to move the IMUs following sinusoidal references at specific amplitudes and frequencies, carefully defined to mimic the characteristics and RoM of human movements. The results showed that Xsens outperforms the other IMUs at the highest frequencies, while at the lowest frequencies all sensors perform equally. This means that for slow movements (e.g., in a rehabilitation context), cheaper sensors can be adequate.

The work within the *Tracking* level was presented in Chapter 4. The contribution of this chapter was twofold. In fact, the main focus was on presenting my research on the enhancement of multiple people's poses obtained from a distributed multi-sensor network in real-time. However, to properly assess the performance of the proposed system, a novel extensive dataset of movements was acquired.

The dataset, namely *UNIPD-BPE*, contains synchronized RGB, depth, and inertial data of several sequences with up to four interacting people, recorded from five Microsoft Azure Kinect cameras and two Xsens MVN Awinda full-body suits. Single-person sequences include 15 participants performing a set of 12 ADLs, while multi-person sequences include seven actions with two to four persons interacting in a confined area. The multi-person sequences offer challenging scenarios where multiple (partially or fully) occluded persons move and interact in a restricted space. The dataset aims to push forward the development of multi-people and multi-sensor markerless BPE and tracking algorithms, as well as multimodal sensor fusion, without the necessity of expensive hardware and bulky acquisition setups.

The chapter continued with a detailed description of the algorithms I developed to allow real-time temporal tracking of multiple people using a network of homogeneous sensors. The proposed system consists of four modules: (1) a *Skeleton Tracker* to perform pure frame-by-frame tracking of the detected body poses; (2) a *Skeleton Merger* to fuse detections from multiple cameras and detect possible outliers; (3) a *Skeleton Optimizer* to perform a global optimization to ensure consistency of each person's body dimensions; and (4) a *Skeleton Filter* to perform real-time filtering of the estimated marker positions and link orientations. No assumptions are made about the typology or number of sensors being used, nor about the detection algorithm that extracts the 3D poses of the persons.

Robustness, accuracy, and real-time performance were evaluated in the *UNIPD-BPE* dataset. The results showed that, by exploiting the proposed methodology, it was possible to reduce BPE errors by up to $35\,\%$ when compared to a pure tracking-by-detection approach. At the same time, the system was able to assign correct IDs to each tracked person, even during periods of close proximity of two or more people. Finally, the system was proven to be efficient and suited for real-time applications, requiring 8.3 ms per frame to track four people.

Chapter 5 concluded the analysis of the three macro levels on which I divided my research by analyzing the *Modeling* level. The work within this level is fundamental to enable the simultaneous use of heterogeneous sensors for the assessment of human motion. In fact, my Ph.D. was driven by the strong idea that proper multimodal sensor fusion must be performed at the *Modeling* level. Different sensors, MoCap systems, and BPE algorithms typically rely on unrelated models (if any). Therefore, a direct fusion of their measured quantities is prone to lead to erroneous results. However, by defining a unique underlying model, it is possible to actively use all the available data, independently of their source, with a common final goal.

To this end, the chapter included a detailed analysis of how a generic musculoskeletal model can be represented in *Hi-ROS* using OpenSim. Although being primarily used for musculoskeletal simulations, OpenSim models can represent any system of rigid bodies connected by frictionless joints that are acted upon by forces to produce motion.

The discussion advanced by analyzing the mathematical principles behind IK simulations. Depending on the measured (or estimated) quantities (i.e., marker positions or IMU orientations), the procedure is formalized as a WLS optimization problem with the goal of minimizing the distance between an experimental marker (or IMU) and the corresponding virtual marker (or IMU) in the model.

To enable an online sensor-independent feed of data and real-time performance of the IK optimization, the original methodology used in biomechanics required several modifications. First, the input data, typically prerecorded and stored in a dedicated file, are now fed to the solver at runtime by means of custom-defined ROS messages. The definition of the structures used for the communication among all the modules developed within my research, described in Section 2.4, allows to use a unique representation independently of the typology or number of sensors employed. Moreover, any combination of sensing systems can be used or changed without requiring any modification of the developed software.

Real-time performance, on the other hand, was achieved by exploiting the concurrent programming design pattern of producers-consumers. Multiple threads have access to a shared queue, allowing them to run multiple IK optimizations simultaneously. Furthermore, the developed architecture maintains real-time capabilities while also enabling multi-person IK optimizations.

In fact, preliminary tests showed how the proposed system was able to achieve 312 Hz when using a full-body model with 32 DoFs, driven by a set of 32 markers and 18 IMUs, 169 Hz when tracking two persons, 119 Hz when tracking three persons, and 91 Hz when tracking four persons on the same machine. Furthermore, the developed system can run on a distributed network of PCs, attaining even faster framerates, if needed.

This work has the potential to enable all the characteristics of the ideal MoCap system: providing highly accurate measurements of multiple people's poses, without requiring complex hardware setups nor hindering the freedom of movement in any way, and in real-time. In this regard, the optimal solution might involve a limited number of cameras to have an initial (possibly partial) estimate of the pose, coupled with a reduced set of IMUs on the most crucial body segments to overcome occlusions and increase the BPE accuracy. In fact, by properly fusing information from heterogeneous sensors (e.g., from markerless and inertial MoCap), it is possible to maximize the advantages of the employed systems while minimizing their weaknesses.

Finally, the algorithms, methodologies, and tools developed within my Ph.D. were effectively applied in three collaborations in the contexts of HRI and Industry 5.0, presented in Chapter 6.

The first application focuses on feedback motion planning in human-robot shared workspaces via DRL. In this work, a collaborative robot is trained to re-plan its motion online with the goal of avoiding possible collisions with a human operator. In particular, the focus is on tasks in which humans and manipulators share the same workspace while operating in parallel, without synchronization. The proposed experiments, which required robust and accurate real-time knowledge of the operator's pose to guarantee their safety, showed a success rate close to 90 %. In this regard, *Hi-ROS* was used to ensure real-time estimation and tracking of the operator's motion throughout the whole duration of the experiments.

The second work consists of the development and validation of a platform for real-time assessment of workers' ergonomics. This innovative platform allows to evaluate a set of ergonomic indexes (i.e., RULA, REBA, OWAS, and PERA) and to provide visual feedback to workers regarding posture and physical fatigue metrics in real-time. While the performed validation exploited inertial MoCap for the assessment of the operator's pose, the system is designed to support the use of any MoCap system as input.

This was achieved by exploiting *Hi-ROS* for the assessment of the body poses.

Finally, the third and last work, currently under development, aims to give real-time feedback to a worker while performing a generic set of assigned tasks. The goal is to create a real-time control loop to give immediate feedback to a manual operator based on a set of control volumes that are built around specific positions on the workstation. In this regard, the location and pose of the operator are acquired in real-time by exploiting the modules developed within the *Tracking* level to combine the feeds of multiple RGB-D cameras, required to cover the entire working area.

7.2 Future Works

The work presented in this dissertation analyzed the current state of the art and the major challenges in the assessment of human motion. Particular importance was given to maximizing the pose estimation accuracy while minimizing the intrusiveness of the required hardware and providing real-time results. The proposed methodologies and algorithms converged on the development of an open-source efficient modular framework, with the aim of laying the basis for new challenging applications where real-time knowledge of the human pose in everyday living and working environments is fundamental. Based on the results obtained during my work, together with the insight gained with several collaborations within the HRI and Industry 5.0 contexts, new challenges and possible directions arise for future research.

One of the primary goals of my work consisted in combining the output of heterogeneous MoCap systems to allow the selection and minimization of the sensors to be worn on the body, depending on the specific application and environment. The motivation behind this stands on the maximization of the user's comfort, without limiting their dexterity, and enabling the motion assessment to be conducted in unconstrained environments.

However, similar results might be achieved by following a different direction. The idea, in this sense, is to minimize the intrusiveness of the sensing system not by decreasing the number of intrusive sensors, but by reducing the bulkiness of each sensor. To this end, smart textiles systems (STS) are a promising technology. Smart textiles are "intelligent" fabrics that embed different typologies of sensors to perceive environmental stimuli, which can be mechanical, chemical, biological, thermal, among others [211].

The interest in STS has grown rapidly in recent years. Proesmans *et al.* [212] proposed an open-source, wireless, and modular piezoresistive smart textile to measure the mechanical stress of the cloth in which it is embedded. Seoane *et al.* [213], on the other hand, developed a prototype sensorized textile with integrated textile electrodes to obtain electrocardiogram (ECG) recordings. Finally, Kang *et al.* [214] developed a conductive yarn to embed multiple IMUs on a smart shirt used to monitor the sitting posture of the user. Although the aforementioned examples show the flexibility of STS, the number of applications including smart textiles is bound to grow, with sensorized garments embedding also electroencephalogram (EEG) sensors, respiratory sensors, pulse oximeters, temperature sensors, etc. [215].

Current applications embedding IMUs are limited to the estimation of reduced sets of joint angles. Future applications should try to maximize the number of embedded IMUs in shirts and trousers, and use the estimated orientations as input for an OB-IK optimization to estimate the full-body pose. In fact, one of the main limitations of inertial MoCap systems is the need to apply several sensors on the clothes, which can cause discomfort during prolonged MoCap sessions. In a recent study [216], 18 participants were involved in evaluating the usability of an STS compared to a commercial inertial system. The results showed how the majority of the participants indicated that they would prefer to wear a smart textile system to an inertial system, principally due to its usability. Thus, a full-body smart textile inertial suit would constitute a breakthrough for non-intrusive MoCap systems.

To this end, the work proposed in this dissertation has the potential to enable the use of such technologies for accurate real-time full-body assessments. In fact, despite the different characteristics of the hardware, STS produce similar outputs with respect to currently available MoCap systems (e.g., a smart textile shirt embedding multiple IMUs will estimate the same quantities of an inertial suit). Since the algorithms developed in my research support different sensors' outputs combined in a common musculoskeletal model of the human, STS can be seamlessly integrated in the proposed framework.

At the current stage of development, *Hi-ROS* does not yet support ID optimizations. Such analyses, in fact, require other typologies of sensors (e.g., force platforms to measure GRFs, or EMG sensors to measure the muscles' activation) to estimate the internal joint torques. These technologies are extremely delicate, thus limiting their use to confined dedicated laboratories. For this reason, it was chosen not to include (yet) such sensors in the proposed framework.

However, new STS embedding EMG sensors have the potential of enabling in-depth analyses in unconstrained environments. Furthermore, recent works focusing on the integration of force sensors in shoe soles are showing promising results [217], [218]. In this regard, the integration of such sensors and, as a result, of real-time ID optimizations should be one of the next steps in extending the proposed framework.

This would be particularly interesting in the industrial scenario to monitor the fatigue of an operator during their shifts. In fact, while extensive work has been (and is still being) conducted on ergonomic assessments [219], the mere pose of the worker in this context is not sufficient to provide a full view of their physical condition. Future applications should investigate the usage of heterogeneous sensors (e.g., vision/inertial MoCap, EMG sensors, force sensors) to monitor and/or estimate additional quantities describing the operator's fatigue.

Within the *Modeling* level, the workflow on pure OB-IK currently requires an initial calibration to define the relative orientations between each IMU and the corresponding virtual IMU in the body. In this sense, the user needs to assume a known static pose during a calibration trial (typically, a neutral pose in which all the anatomical joint angles are equal to $0°$). However, the pose assumed by the person will never perfectly match the one defined in the model, resulting in subsequent pose estimation errors whose magnitude depends on how close the initial pose could match the ideal one.

Multimodal sensor fusion, in this context, could allow to minimize such errors. Future research should strive to simplify the initial calibration process by exploiting the pose estimated, as an example, by a single RGB-D camera as the initial pose on which the IMU orientations are matched.

Furthermore, it would be interesting to evaluate how different configurations of cameras and IMUs affect the achievable accuracy of MOB-IK optimizations. Within the *Modeling* level, the main research question to answer was whether it was possible to perform multi-person IK optimizations in real-time, while at the same time enabling multimodal sensor fusion of heterogeneous sensing systems. Thus, the focus was not on assessing the achievable accuracy, but on quantifying the real-time capabilities of the proposed system. In fact, the accuracy primarily depends on the quality of the specific sensors being used, rather than on the IK algorithm, which has already been validated by means of several biomechanical analyses (e.g., [197], [220], [221]). However, it would be interesting to compare how different sensing setups (and, specifically, different numbers of sensors) impact the overall precision of the pose estimation. An thorough assessment will be required to define the minimum number of cameras and IMUs that allow to achieve adequate accuracy without hindering the user's dexterity.

Proper scaling of a generic musculoskeletal model and, specifically, the correct positioning of the virtual markers and IMUs on the model is still an open challenge. The common approach in biomechanics, in fact, consists of performing a linear scaling of generic models retrieved from human specimens to match the individual anthropometric characteristics of each subject. Therefore, the estimated quantities (e.g., joint angles and moments) are not based on a suitable subject-specific model, which limits the accuracy of the simulated motion kinematics and dynamics [222].

Although novel approaches that combine statistical shape methods with medical imaging databases have the potential to create fully subject-specific anatomical models [223], the correct placement of markers (both in the model and in the subject's body) remains fundamental. With IMUs, this problem is not as strict, since the main source of errors is not the relative position between an IMU and its body segment, but the relative orientation. Thus, the main research problem in OB-IK optimizations consists of maximizing the accuracy of the initial calibration.

The markers used in MB-IK, on the other hand, must perfectly match their virtual and real positions. This is one of the main reasons why specialized personnel are needed for optoelectronic analyses.

In markerless MoCap, the estimated markers are automatically placed on the subject's body by the BPE algorithm. However, different people have different weights, heights, and physical structures. As a result, some markers (e.g., the chest) may be placed in slightly different locations, depending on the person being analyzed. In this regard, it would be interesting to investigate an automatic placement procedure of the virtual markers in the model, after the initial scaling of the model dimensions.

A possible solution might divide the scaling procedure into two separate steps. First, the mesh describing the body of the analyzed subject is extracted from the depth, and the different parts of the body are segmented.

This would allow to perform the actual scaling of the model to match the user's anthropometric characteristics. Once the model is scaled, the estimated marker positions could be compared to the virtual markers' ones, and the latter altered, within an acceptable range, to minimize the residual error obtained within an IK optimization performed on the calibration trial. This would allow to optimize the placement of the virtual markers on the model, enabling a subject-specific marker set.

7.3 Conclusions

This thesis focused on providing accurate assessment of multi-person motion in complex environments, with minimal intrusiveness of the required hardware, and in real-time. To achieve this complex goal, an in-depth analysis of the currently available MoCap systems was progressed. The requirements of an ideal system, along with the limitations of the current technologies, allowed to define three macro levels in which the overall problem was divided. The extensive work on sensing, tracking, and modeling resulted in the development of an open-source efficient modular framework where different typologies of sensors can be integrated in a plug-and-play fashion. Particular attention was given to distributed sensors networks (e.g., camera networks) to maximize the achievable accuracy while overcoming the several challenges typical of such setups. Finally, new methodologies to enable the real-time performance of state-of-the-art biomechanical approaches were successfully adapted in the context of Industry 5.0 and HRI. By exploiting a common musculoskeletal model of the human, multimodal sensor fusion of heterogeneous quantities was achieved.

To conclude, the work developed within my Ph.D. research aimed to push forward, via a multidisciplinary approach, the state-of-the-art on accurate unobtrusive assessment of human motion in everyday living and working environments. The developed algorithms and methodologies permit to compare different sensor configurations on the fly, allowing to fine-tune the performance (both with regard to the required accuracy and to the maximum setup complexity) to respect the requirements of any typology of application. Moreover, modularity permeated the design of all the modules defined within the three levels of my research. This allows to modify the employed MoCap setups without requiring any modification on the software used for real-time assessment of motion. To this end, the proposed work has the potential to enable the development of a variety of emerging applications, where precise knowledge of multiple people's poses in unconstrained environments, with a minimal number of on-body sensors, and in real-time, is a fundamental requirement.

Bibliography

[1] A. Ancillao, "Analysis and Measurement of Human Motion: Modern Protocols and Clinical Considerations", *Journal of Robotics and Mechanical Engineering Research*, vol. 1, no. 4, pp. 30–37, 2016.

[2] R. Klette and G. Tee, "Understanding Human Motion: A Historic Review", in *Human Motion*, Springer, 2008, pp. 1–22.

[3] S. L. Colyer, M. Evans, D. P. Cosker, and A. I. Salo, "A Review of the Evolution of Vision-Based Motion Analysis and the Integration of Advanced Computer Vision Methods Towards Developing a Markerless System", *Sports Medicine - Open*, vol. 4, no. 1, pp. 1–15, 2018.

[4] X. Xu, Y. Lu, B. Vogel-Heuser, and L. Wang, "Industry 4.0 and Industry 5.0 — Inception, Conception and Perception", *Journal of Manufacturing Systems*, vol. 61, pp. 530–535, 2021.

[5] A. Hentout, M. Aouache, A. Maoudj, and I. Akli, "Human-Robot Interaction in Industrial Collaborative Robotics: A Literature Review of the Decade 2008-2017", *Advanced Robotics*, vol. 33, no. 15-16, pp. 764–799, 2019.

[6] D. Rossell, "Chronophotography in the Context of Moving Pictures", *Early Popular Visual Culture*, vol. 11, no. 1, pp. 10–27, 2013.

[7] Aristotele, Περὶ πορείας ζῴω. Eng. Transactions by A.S.L. Farquharson, eBooks@Adelaide, 2004.

[8] R. B. Martin, "A Genealogy of Biomechanics", in *23rd Annual Conference of the American Society of Biomechanics*, vol. 23, 1999.

[9] S. L. Delp, J. P. Loan, M. G. Hoy, F. E. Zajac, E. L. Topp, and J. M. Rosen, "An Interactive Graphics-Based Model of the Lower Extremity to Study Orthopaedic Surgical Procedures", *IEEE Transactions on Biomedical Engineering*, vol. 37, no. 8, pp. 757–767, 1990.

[10] D. Blana, J. G. Hincapie, E. K. Chadwick, and R. F. Kirsch, "A Musculoskeletal Model of the Upper Extremity for Use in the Development of Neuroprosthetic Systems", *Journal of Biomechanics*, vol. 41, no. 8, pp. 1714–1721, 2008.

[11] A. Rajagopal, C. L. Dembia, M. S. DeMers, D. D. Delp, J. L. Hicks, and S. L. Delp, "Full-Body Musculoskeletal Model for Muscle-Driven Simulation of Human Gait", *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 10, pp. 2068–2079, 2016.

[12] D. A. Winter, *Biomechanics and Motor Control of Human Movement*. John Wiley & Sons, 2009.

[13] V. Camomilla, A. Cereatti, A. G. Cutti, S. Fantozzi, R. Stagni, and G. Vannozzi, "Methodological Factors Affecting Joint Moments Estimation in Clinical Gait Analysis: A Systematic Review", *Biomedical Engineering Online*, vol. 16, no. 1, pp. 1–27, 2017.

[14] V. N. Gandbhir and B. Cunha, "Goniometer", 2020.

[15] G. E. Hancock, T. Hepworth, and K. Wembridge, "Accuracy and Reliability of Knee Goniometry Methods", *Journal of Experimental Orthopaedics*, vol. 5, no. 1, pp. 1–6, 2018.

[16] J. Atha, "Current Techniques for Measuring Motion", *Applied Ergonomics*, vol. 15, no. 4, pp. 245–257, 1984.

[17] M. Sandau, H. Koblauch, T. B. Moeslund, H. Aanæs, T. Alkjær, and E. B. Simonsen, "Markerless Motion Capture Can Provide Reliable 3D Gait Kinematics in the Sagittal and Frontal Plane", *Medical Engineering & Physics*, vol. 36, no. 9, pp. 1168–1175, 2014.

[18] A. Cappozzo, U. Della Croce, A. Leardini, and L. Chiari, "Human Movement Analysis Using Stereophotogrammetry: Part 1: Theoretical Background", *Gait & Posture*, vol. 21, no. 2, pp. 186–196, 2005.

[19] J. O'Rourke and N. I. Badler, "Model-Based Image Analysis of Human Motion Using Constraint Propagation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 6, pp. 522–536, 1980.

[20] M. K. Leung and Y.-H. Yang, "First Sight: A Human Body Outline Labeling System", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 4, pp. 359–377, 1995.

[21] L. Mündermann, S. Corazza, and T. P. Andriacchi, "The Evolution of Methods for the Capture of Human Movement Leading to Markerless Motion Capture for Biomechanical Applications", *Journal of Neuroengineering and Rehabilitation*, vol. 3, no. 1, pp. 1–11, 2006.

[22] V. Joukov, J. Ćesić, K. Westermann, I. Marković, D. Kulić, and I. Petrović, "Human Motion Estimation on Lie Groups Using IMU Measurements", in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 1965–1972.

[23] D. Jurman, M. Jankovec, R. Kamnik, and M. Topič, "Calibration and Data Fusion Solution for the Miniature Attitude and Heading Reference System", *Sensors and Actuators A: Physical*, vol. 138, no. 2, pp. 411–420, 2007.

[24] L. Lou, X. Xu, J. Cao, Z. Chen, and Y. Xu, "Sensor Fusion-Based Attitude Estimation Using Low-Cost MEMS-IMU for Mobile Robot Navigation", in *2011 6th IEEE International Conference on Information Technology and Artificial Intelligence*, IEEE, vol. 2, 2011, pp. 465–468.

[25] A. Toshev and C. Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1653–1660.

[26] R. A. Güler, N. Neverova, and I. Kokkinos, "Densepose: Dense Human Pose Estimation in the Wild", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7297–7306.

[27] A. Mathis, P. Mamidanna, K. M. Cury, *et al.*, "DeepLabCut: Markerless Pose Estimation of User-Defined Body Parts With Deep Learning", *Nature Neuroscience*, vol. 21, no. 9, pp. 1281–1289, 2018.

[28] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[29] T. von Marcard, B. Rosenhahn, M. J. Black, and G. Pons-Moll, "Sparse Inertial Poser: Automatic 3D Human Pose Estimation From Sparse IMUs", in *Computer Graphics Forum*, Wiley Online Library, vol. 36, 2017, pp. 349–360.

[30] L. Chiari, U. Della Croce, A. Leardini, and A. Cappozzo, "Human Movement Analysis Using Stereophotogrammetry: Part 2: Instrumental Errors", *Gait & Posture*, vol. 21, no. 2, pp. 197–211, 2005.

[31] A. Leardini, L. Chiari, U. Della Croce, and A. Cappozzo, "Human Movement Analysis Using Stereophotogrammetry: Part 3. Soft Tissue Artifact Assessment and Compensation", *Gait & Posture*, vol. 21, no. 2, pp. 212–225, 2005.

[32] U. Della Croce, A. Leardini, L. Chiari, and A. Cappozzo, "Human Movement Analysis Using Stereophotogrammetry: Part 4: Assessment of Anatomical Landmark Misplacement and its Effects on Joint Kinematics", *Gait & Posture*, vol. 21, no. 2, pp. 226–237, 2005.

[33] J. G. Richards, "The Measurement of Human Motion: A Comparison of Commercially Available Systems", *Human Movement Science*, vol. 18, no. 5, pp. 589–602, 1999.

[34] *Vicon*, `https://www.vicon.com`, Accessed: May 2022.

[35] *Qualisys*, `https://www.qualisys.com`, Accessed: May 2022.

[36] *BTS Bioengineering*, `https://www.btsbioengineering.com`, Accessed: May 2022.

[37] M. Sati, J. de Guise, S. Larouche, and G. Drouin, "Quantitative Assessment of Skin-Bone Movement at the Knee", *The Knee*, vol. 3, no. 3, pp. 121–138, 1996.

[38] C. Reinschmidt, A. van den Bogert, B. Nigg, A. Lundberg, and N. Murphy, "Effect of Skin Movement on the Analysis of Skeletal Knee Joint Motion During Running", *Journal of Biomechanics*, vol. 30, no. 7, pp. 729–732, 1997.

[39] A. Peters, B. Galna, M. Sangeux, M. Morris, and R. Baker, "Quantification of Soft Tissue Artifact in Lower Limb Human Motion Analysis: A Systematic Review", *Gait & Posture*, vol. 31, no. 1, pp. 1–8, 2010.

[40] E. Szczerbik and M. Kalinowska, "The Influence of Knee Marker Placement Error on Evaluation of Gait Kinematic Parameters", *Acta of Bioengineering and Biomechanics*, vol. 13, no. 3, pp. 43–46, 2011.

[41] B. E. Groen, M. Geurts, B. Nienhuis, and J. Duysens, "Sensitivity of the OLGA and VCM Models to Erroneous Marker Placement: Effects on 3D-Gait Kinematics", *Gait & Posture*, vol. 35, no. 3, pp. 517–521, 2012.

[42] E. Growney, D. Meglan, M. Johnson, T. Cahalan, and K.-N. An, "Repeated Measures of Adult Normal Walking Using a Video Tracking System", *Gait & Posture*, vol. 6, no. 2, pp. 147–162, 1997.

[43] H. Tsushima, M. E. Morris, and J. McGinley, "Test-Retest Reliability and Inter-Tester Reliability of Kinematic Data From a Three-Dimensional Gait Analysis System", *Journal of the Japanese Physical Therapy Association*, vol. 6, no. 1, pp. 9–17, 2003.

[44] A. Filippeschi, N. Schmitz, M. Miezal, G. Bleser, E. Ruffaldi, and D. Stricker, "Survey of Motion Tracking Methods Based on Inertial Sensors: A Focus on Upper Limb Human Motion", *Sensors*, vol. 17, no. 6, p. 1257, 2017.

[45] S. Sukkarieh, E. M. Nebot, and H. F. Durrant-Whyte, "A High Integrity IMU/GPS Navigation Loop for Autonomous Land Vehicle Applications", *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 572–578, 1999.

[46] L. Sahawneh and M. Jarrah, "Development and Calibration of Low Cost MEMS IMU for UAV Applications", in *2008 5th International Symposium on Mechatronics and Its Applications*, IEEE, 2008, pp. 1–9.

[47] J. Yi, J. Zhang, D. Song, and S. Jayasuriya, "IMU-Based Localization and Slip Estimation for Skid-Steered Mobile Robots", in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2007, pp. 2845–2850.

[48] H. Hyyti and A. Visala, "A DCM Based Attitude Estimation Algorithm for Low-Cost MEMS IMUs", *International Journal of Navigation and Observation*, vol. 2015, 2015.

[49] S. Sabatelli, M. Galgani, L. Fanucci, and A. Rocchi, "A Double-Stage Kalman Filter for Orientation Tracking With an Integrated Processor in 9-D IMU", *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 3, pp. 590–598, 2012.

[50] W. Li and J. Wang, “Effective Adaptive Kalman Filter for MEMS-IMU/Magnetometers Integrated Attitude and Heading Reference Systems”, *The Journal of Navigation*, vol. 66, no. 1, pp. 99–113, 2013.

[51] R. V. Vitali, R. S. McGinnis, and N. C. Perkins, “Robust Error-State Kalman Filter for Estimating IMU Orientation”, *IEEE Sensors Journal*, vol. 21, no. 3, pp. 3561–3569, 2020.

[52] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel, “A Complementary Filter for Attitude Estimation of a Fixed-Wing UAV”, in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2008, pp. 340–345.

[53] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan, “Estimation of IMU and MARG Orientation Using a Gradient Descent Algorithm”, in *2011 IEEE International Conference on Rehabilitation Robotics*, IEEE, 2011, pp. 1–7.

[54] S. P. Tseng, W.-L. Li, C.-Y. Sheng, J.-W. Hsu, and C.-S. Chen, “Motion and Attitude Estimation Using Inertial Measurements With Complementary Filter”, in *2011 8th Asian Control Conference (ASCC)*, IEEE, 2011, pp. 863–868.

[55] J. Zhao, “A Review of Wearable IMU (Inertial-Measurement-Unit)-Based Pose Estimation and Drift Reduction Technologies”, in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1087, 2018, p. 042 003.

[56] F. Dadashi, F. Crettenand, G. P. Millet, and K. Aminian, “Front-Crawl Instantaneous Velocity Estimation Using a Wearable Inertial Measurement Unit”, *Sensors*, vol. 12, no. 10, pp. 12 927–12 939, 2012.

[57] M. Schepers, M. Giuberti, G. Bellusci, *et al.*, “Xsens MVN: Consistent Tracking of Human Motion Using Inertial Sensing”, *Xsens Technol*, pp. 1–8, 2018.

[58] L. Zhang, J. Sturm, D. Cremers, and D. Lee, “Real-Time Human Motion Tracking Using Multiple Depth Cameras”, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 2389–2395.

[59] S. Asteriadis, A. Chatzitofis, D. Zarpalas, D. S. Alexiadis, and P. Daras, “Estimating Human Motion From Multiple Kinect Sensors”, in *6th International Conference on Computer Vision/Computer Graphics Collaboration Techniques and Applications*, 2013, pp. 1–6.

[60] M. Munaro, F. Basso, and E. Menegatti, "OpenPTrack: Open Source Multi-Camera Calibration and People Tracking for RGB-D Camera Networks", *Robotics and Autonomous Systems*, vol. 75, pp. 525–538, 2016.

[61] *Azure Kinect DK*, `https://azure.microsoft.com/services/kinect-dk`, Accessed: May 2022.

[62] P. Karashchuk, K. L. Rupp, E. S. Dickinson, *et al.*, "Anipose: A Toolkit for Robust Markerless 3D Pose Estimation", *Cell Reports*, vol. 36, no. 13, p. 109 730, 2021.

[63] A. Malaguti, M. Carraro, **M. Guidolin**, L. Tagliapietra, E. Menegatti, and S. Ghidoni, "Real-Time Tracking-by-Detection of Human Motion in RGB-D Camera Networks", in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, IEEE, 2019, pp. 3198–3204.

[64] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep It SMPL: Automatic Estimation of 3D Human Pose and Shape From a Single Image", in *European Conference on Computer Vision*, Springer, 2016, pp. 561–578.

[65] M. Windolf, N. Götzen, and M. Morlock, "Systematic Accuracy and Precision Analysis of Video Motion Capturing Systems — Exemplified on the Vicon-460 System", *Journal of Biomechanics*, vol. 41, no. 12, pp. 2776–2780, 2008.

[66] Y. Desmarais, D. Mottet, P. Slangen, and P. Montesinos, "A Review of 3D Human Pose Estimation Algorithms for Markerless Motion Capture", *Computer Vision and Image Understanding*, vol. 212, p. 103 275, 2021.

[67] V. Belagiannis, S. Amin, M. Andriluka, B. Schiele, N. Navab, and S. Ilic, "3D Pictorial Structures for Multiple Human Pose Estimation", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1669–1676.

[68] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "SCAPE: Shape Completion and Animation of People", in *ACM SIGGRAPH 2005 Papers*, 2005, pp. 408–416.

[69] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A Skinned Multi-Person Linear Model", *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, pp. 1–16, 2015.

[70] J. Maycock, T. Rohlig, M. Schroder, M. Botsch, and H. Ritter, "Fully Automatic Optical Motion Tracking Using an Inverse Kinematics Approach", in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, IEEE, 2015, pp. 461–466.

[71] M. Quigley, K. Conley, B. Gerkey, *et al.*, "ROS: An Open-Source Robot Operating System", in *ICRA Workshop on Open Source Software*, Kobe, Japan, vol. 3, 2009, p. 5.

[72] **M. Guidolin**, E. Menegatti, and M. Reggiani, "UNIPD-BPE: Synchronized RGB-D and Inertial Dataset for Multimodal Body Pose Estimation and Tracking", *Data*, [submitted - under review]).

[73] S. L. Delp, F. C. Anderson, A. S. Arnold, *et al.*, "OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement", *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 11, pp. 1940–1950, 2007.

[74] *OpenSim Statistics*, `https://simtk.org/plugins/reports/index.php?type=group&group_id=91`, Accessed: May 2022.

[75] *Xsens MVN*, `https://www.xsens.com/products/mvn-analyze`, Accessed: May 2022.

[76] Y. Li, "Deep Reinforcement Learning: An Overview", *arXiv Preprint arXiv:1701.07274*, 2017.

[77] **M. Guidolin**, L. Tagliapietra, E. Menegatti, and M. Reggiani, "Hi-ROS: Open-Source Multi-Camera Sensor Fusion for Real-Time People Tracking", *Computer Vision and Image Understanding*, [submitted - under review].

[78] H. Modares, I. Ranatunga, F. L. Lewis, and D. O. Popa, "Optimized Assistive Human-Robot Interaction Using Reinforcement Learning", *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 655–667, 2015.

[79] A. Q. Keemink, H. van der Kooij, and A. H. Stienen, "Admittance Control for Physical Human-Robot Interaction", *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1421–1444, 2018.

[80] D. Andronas, E. Kampourakis, K. Bakopoulou, C. Gkournelos, P. Angelakis, and S. Makris, "Model-Based Robot Control for Human-Robot Flexible Material Co-Manipulation", in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2021, pp. 1–8.

[81] T.-W. Lu and J. O'Connor, "Bone Position Estimation From Skin Marker Co-ordinates Using Global Optimisation With Joint Constraints", *Journal of Biomechanics*, vol. 32, no. 2, pp. 129–134, 1999.

[82] A. Shafti, A. Ataka, B. U. Lazpita, A. Shiva, H. A. Wurdemann, and K. Althoefer, "Real-Time Robot-Assisted Ergonomics", in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 1975–1981.

[83] L. Fortini, M. Lorenzini, W. Kim, E. De Momi, and A. Ajoudani, "A Real-Time Tool for Human Ergonomics Assessment Based on Joint Compressive Forces", in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, IEEE, 2020, pp. 1164–1170.

[84] P. Petz, F. Eibensteiner, and J. Langer, "Sensor Shirt as Universal Platform for Real-Time Monitoring of Posture and Movements for Occupational Health and Ergonomics", *Procedia Computer Science*, vol. 180, pp. 200–207, 2021.

[85] D. Bakken, "Middleware", *Encyclopedia of Distributed Computing*, vol. 11, 2001.

[86] A. Hentout, A. Maoudj, and B. Bouzouia, "A Survey of Development Frameworks for Robotics", in *2016 8th International Conference on Modelling, Identification and Control (ICMIC)*, IEEE, 2016, pp. 67–72.

[87] S. Vinoski, "CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments", *IEEE Communications Magazine*, vol. 35, no. 2, pp. 46–55, 1997.

[88] H. Bruyninckx, "Open Robot Control Software: the OROCOS Project", in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, IEEE, vol. 3, 2001, pp. 2523–2528.

[89] H. Utz, S. Sablatnog, S. Enderle, and G. Kraetzschmar, "Miro-Middleware for Mobile Robot Applications", *IEEE Transactions on Robotics and Automation*, vol. 18, no. 4, pp. 493–497, 2002.

[90] B. Gerkey, R. T. Vaughan, A. Howard, *et al.*, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems", in *Proceedings of the 11th International Conference on Advanced Robotics*, Citeseer, vol. 1, 2003, pp. 317–323.

[91] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, and W.-K. Yoon, "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)", in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2005, pp. 3933–3938.

[92] G. Metta, P. Fitzpatrick, and L. Natale, "YARP: Yet Another Robot Platform", *International Journal of Advanced Robotic Systems*, vol. 3, no. 1, p. 8, 2006.

[93] R. Diankov and J. Kuffner, "OpenRAVE: A Planning Architecture for Autonomous Robotics", *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, vol. 79, 2008.

[94] A. Elkady and T. Sobh, "Robotics Middleware: A Comprehensive Literature Survey and Attribute-Based Bibliography", *Journal of Robotics*, vol. 2012, 2012.

[95] E. Tsardoulias and P. Mitkas, "Robotic Frameworks, Architectures and Middleware Comparison", *arXiv Preprint arXiv:1711.06842*, 2017.

[96] *ROS Wiki*, `https://wiki.ros.org/ROS/Introduction`, Accessed: May 2022.

[97] M. Damsgaard, J. Rasmussen, S. T. Christensen, E. Surma, and M. De Zee, "Analysis of Musculoskeletal Systems in the AnyBody Modeling System", *Simulation Modelling Practice and Theory*, vol. 14, no. 8, pp. 1100–1111, 2006.

[98] *BoB (Biomechanics of Bodies)*, `https://www.bob-biomechanics.com`, Accessed: May 2022.

[99] *SIMM (Software for Interactive Musculoskeletal Modeling)*, `https://motionanalysis.com/simm`, Accessed: May 2022.

[100] D. W. Wagner, V. Stepanyan, J. M. Shippen, *et al.*, "Consistency Among Musculoskeletal Models: Caveat Utilitor", *Annals of biomedical engineering*, vol. 41, no. 8, pp. 1787–1799, 2013.

[101] J. B. Langholz, G. Westman, and M. Karlsteen, "Musculoskeletal Modelling in Sports - Evaluation of Different Software Tools With Focus on Swimming", *Procedia Engineering*, vol. 147, pp. 281–287, 2016.

[102] Y. Kim, Y. Jung, W. Choi, K. Lee, and S. Koo, "Similarities and Differences Between Musculoskeletal Simulations of OpenSim and AnyBody Modeling System", *Journal of Mechanical Science and Technology*, vol. 32, no. 12, pp. 6037–6044, 2018.

[103] U. Trinler, H. Schwameder, R. Baker, and N. Alexander, "Muscle Force Estimation in Clinical Gait Analysis Using AnyBody and OpenSim", *Journal of Biomechanics*, vol. 86, pp. 55–63, 2019.

[104] M. A. Sherman, A. Seth, and S. L. Delp, "Simbody: Multibody Dynamics for Biomedical Research", *Procedia Iutam*, vol. 2, pp. 241–261, 2011.

[105] P. J. Bishop, "Testing the Function of Dromaeosaurid (Dinosauria, Theropoda) 'Sickle Claws' Through Musculoskeletal Modelling and Optimization", *PeerJ*, vol. 7, e7577, 2019.

[106] **M. Guidolin**, E. Menegatti, M. Reggiani, and L. Tagliapietra, "A ROS Driver for Xsens Wireless Inertial Measurement Unit Systems", in *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, IEEE, vol. 1, 2021, pp. 677–683.

[107] **M. Guidolin**, R. A. B. Petrea, R. Oboe, M. Reggiani, E. Menegatti, and L. Tagliapietra, "On the Accuracy of IMUs for Human Motion Tracking: A Comparative Evaluation", in *2021 IEEE International Conference on Mechatronics (ICM)*, IEEE, 2021, pp. 1–6.

[108] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms", *International Journal of Computer Vision*, vol. 47, no. 1, pp. 7–42, 2002.

[109] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in Computational Stereo", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993–1008, 2003.

[110] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.

[111] S. Van der Jeught and J. J. Dirckx, "Real-Time Structured Light Profilometry: A Review", *Optics and Lasers in Engineering*, vol. 87, pp. 18–31, 2016.

[112] S. Mattoccia, "Stereo Vision Algorithms for FPGAs", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, pp. 636–641.

[113] J. Geng, "Structured-Light 3D Surface Imaging: A Tutorial", *Advances in Optics and Photonics*, vol. 3, no. 2, pp. 128–160, 2011.

[114] S. Zhang, "High-Speed 3D Shape Measurement With Structured Light Methods: A Review", *Optics and Lasers in Engineering*, vol. 106, pp. 119–131, 2018.

[115] R. Lange and P. Seitz, "Solid-State Time-of-Flight Range Camera", *IEEE Journal of Quantum Electronics*, vol. 37, no. 3, pp. 390–397, 2001.

[116] S. Hsu, S. Acharya, A. Rafii, and R. New, "Performance of a Time-of-Flight Range Camera for Intelligent Vehicle Safety Applications", in *Advanced Microsystems for Automotive Applications 2006*, Springer, 2006, pp. 205–219.

[117] T. Ringbeck and B. Hagebeuker, "A 3D Time of Flight Camera for Object Detection", *PMD Technologies GmbH, Siegen*, vol. 2, 2007.

[118] R. Horaud, M. Hansard, G. Evangelidis, and C. Ménier, "An Overview of Depth Cameras and Range Scanners Based on Time-of-Flight Technologies", *Machine Vision and Applications*, vol. 27, no. 7, pp. 1005–1020, 2016.

[119] Y. He and S. Chen, "Recent Advances in 3D Data Acquisition and Processing by Time-of-Flight Camera", *IEEE Access*, vol. 7, pp. 12 495–12 510, 2019.

[120] *Sensors supported by ROS*, `https://wiki.ros.org/Sensors`, Accessed: May 2022.

[121] *Intel RealSense*, `https://www.intelrealsense.com`, Accessed: May 2022.

[122] *Orbbec 3D*, `https://www.orbbec3d.com`, Accessed: May 2022.

[123] R. G. Valenti, I. Dryanovski, and J. Xiao, "Keeping a Good Attitude: A Quaternion-Based Orientation Filter for IMUs and MARGs", *Sensors*, vol. 15, no. 8, pp. 19 302–19 330, 2015.

[124] M. Perlmutter and L. Robin, "High-Performance, Low Cost Inertial MEMS: A Market in Motion!", in *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*, IEEE, 2012, pp. 225–229.

[125] M. Narasimhappa, A. D. Mahindrakar, V. C. Guizilini, M. H. Terra, and S. L. Sabat, "MEMS-Based IMU Drift Minimization: Sage Husa Adaptive Robust Kalman Filtering", *IEEE Sensors Journal*, vol. 20, no. 1, pp. 250–260, 2019.

[126] F. Wittmann, O. Lambercy, and R. Gassert, "Magnetometer-Based Drift Correction During Rest in IMU Arm Motion Tracking", *Sensors*, vol. 19, no. 6, p. 1312, 2019.

[127] M. Falbriard, F. Meyer, B. Mariani, G. P. Millet, and K. Aminian, "Drift-Free Foot Orientation Estimation in Running Using Wearable IMU", *Frontiers in Bioengineering and Biotechnology*, vol. 8, p. 65, 2020.

[128] L. Ricci, F. Taffoni, and D. Formica, "On the Orientation Error of IMU: Investigating Static and Dynamic Accuracy Targeting Human Motion", *PLoS One*, vol. 11, no. 9, 2016.

[129] L. Tagliapietra, L. Modenese, E. Ceseracciu, C. Mazzà, and M. Reggiani, "Validation of a Model-Based Inverse Kinematics Approach Based on Wearable Inertial Sensors", *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 21, no. 16, pp. 834–844, 2018.

[130] R. M. Chapman, W. E. Moschetti, and D. W. Van Citters, "Stance and Swing Phase Knee Flexion Recover at Different Rates Following Total Knee Arthroplasty: An Inertial Measurement Unit Study", *Journal of Biomechanics*, vol. 84, pp. 129–137, 2019.

[131] Y.-J. Lu, C.-J. Chang, C.-W. Chang, and S.-W. Yang, "Accuracy Comparisons in IMU Sensor and Motion Analysis Software", in *Proceedings of the 2018 2nd International Conference on Mechatronics Systems and Control Engineering*, 2018, pp. 13–16.

[132] *MTw Awinda User Manual*, `https://www.xsens.com/hubfs/Downloads/Manuals/MTw_Awinda_User_Manual.pdf`, Accessed: May 2022.

[133] *GitHub Repository*, `https://github.com/qleonardolp/xsens_mtw_driver-release`, Accessed: May 2022.

[134] *GitHub Repository*, `https://github.com/Raffa87/xsense-awinda`, Accessed: May 2022.

[135] *ROS Best Practices*, `https://wiki.ros.org/BestPractices`, Accessed: May 2022.

[136] *Xsens MTi Products*, `https://www.xsens.com/mti-products`, Accessed: May 2022.

[137] Q. Yuan, E. Asadi, Q. Lu, G. Yang, and I. Chen, "Uncertainty-Based IMU Orientation Tracking Algorithm for Dynamic Motions", *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 2, pp. 872–882, 2019.

[138] K. H. E. Beange, A. D. C. Chan, and R. B. Graham, "Evaluation of Wearable IMU Performance for Orientation Estimation and Motion Tracking", in *2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, 2018, pp. 1–6.

[139] *MPU-9250 Product Specification*, `https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf`, Accessed: May 2022.

[140] *MetaMotionR Product Specification*, `https://mbientlab.com/documents/MetaMotionR-PS3.pdf`, Accessed: May 2022.

[141] A. Newell, K. Yang, and J. Deng, "Stacked Hourglass Networks for Human Pose Estimation", in *European Conference on Computer Vision*, Springer, 2016, pp. 483–499.

[142] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, "Cascaded Pyramid Network for Multi-Person Pose Estimation", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7103–7112.

[143] J. Shotton, A. Fitzgibbon, M. Cook, *et al.*, "Real-Time Human Pose Recognition in Parts From Single Depth Images", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Ieee, 2011, pp. 1297–1304.

[144] H. Alabbasi, A. Gradinaru, F. Moldoveanu, and A. Moldoveanu, "Human Motion Tracking & Evaluation Using Kinect v2 Sensor", in *2015 E-Health and Bioengineering Conference (EHB)*, IEEE, 2015, pp. 1–4.

[145] J. Kim, I. Lee, J. Kim, and S. Lee, "Implementation of an Omnidirectional Human Motion Capture System Using Multiple Kinect Sensors", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 98, no. 9, pp. 2004–2008, 2015.

[146] A. Bilesan, S. Behzadipour, T. Tsujita, S. Komizunai, and A. Konno, "Markerless Human Motion Tracking Using Microsoft Kinect SDK and Inverse Kinematics", in *2019 12th Asian Control Conference (ASCC)*, IEEE, 2019, pp. 504–509.

[147] L. Sigal, A. O. Balan, and M. J. Black, "HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion", *International Journal of Computer Vision*, vol. 87, no. 4, pp. 4–27, 2010.

[148] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, 2013.

[149] M. Munaro, G. Ballin, S. Michieletto, and E. Menegatti, "3D Flow Estimation for Human Action Recognition From Colored Point Clouds", *Biologically Inspired Cognitive Architectures*, vol. 5, pp. 42–51, 2013.

[150] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Berkeley MHAD: A Comprehensive Multimodal Human Action Database", in *2013 IEEE Workshop on Applications of Computer Vision (WACV)*, IEEE, 2013, pp. 53–60.

[151] H. Joo, H. Liu, L. Tan, *et al.*, "Panoptic Studio: A Massively Multiview System for Social Motion Capture", in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3334–3342.

[152] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1010–1019.

[153] C. Chen, R. Jafari, and N. Kehtarnavaz, "UTD-MHAD: A Multimodal Dataset for Human Action Recognition Utilizing a Depth Camera and a Wearable Inertial Sensor", in *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2015, pp. 168–172.

[154] M. Trumble, A. Gilbert, C. Malleson, A. Hilton, and J. P. Collomosse, "Total Capture: 3D Human Pose Estimation Fusing Video and Inertial Sensors", in *British Machine Vision Conference (BMVC)*, vol. 2, 2017, pp. 1–13.

[155] P. Maurice, A. Malaisé, C. Amiot, *et al.*, "Human Movement and Ergonomics: An Industry-Oriented Dataset for Collaborative Robotics", *The International Journal of Robotics Research*, vol. 38, no. 14, pp. 1529–1537, 2019.

[156] A. Chatzitofis, L. Saroglou, P. Boutis, *et al.*, "HUMAN4D: A Human-Centric Multimodal Dataset for Motions and Immersive Media", *IEEE Access*, vol. 8, pp. 176 241–176 262, 2020.

[157] M. Tölgyessy, M. Dekan, L. Chovanec, and P. Hubinskỳ, "Evaluation of the Azure Kinect and Its Comparison to Kinect v1 and Kinect v2", *Sensors*, vol. 21, no. 2, p. 413, 2021.

[158] *Azure Kinect Body Tracking SDK Documentation*, `https://microsoft.github.io/Azure-Kinect-Body-Tracking/release/1.1.x/index.html`, Accessed: May 2022.

[159] M. Paulich, M. Schepers, N. Rudigkeit, and G. Bellusci, "Xsens MTw Awinda: Miniature Wireless Inertial-Magnetic Motion Tracker for Highly Accurate 3D Kinematic Applications", *Xsens: Enschede, The Netherlands*, 2018.

[160] *Rosbag Tutorials*, `http://wiki.ros.org/rosbag/Tutorials`, Accessed: May 2022.

[161] D. L. Mills, "Internet Time Synchronization: The Network Time Protocol", *IEEE Transactions on Communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[162] L. Romeo, R. Marani, M. Malosio, A. G. Perri, and T. D'Orazio, "Performance Analysis of Body Tracking with the Microsoft Azure Kinect", in *2021 29th Mediterranean Conference on Control and Automation (MED)*, IEEE, 2021, pp. 572–577.

[163] *Azure Kinect Body Tracking Joints*, `https://docs.microsoft.com/azure/kinect-dk/body-joints`, Accessed: May 2022.

[164] *Xsens MVN User Manual*, `https://www.xsens.com/hubfs/Downloads/usermanual/MVN_User_Manual.pdf`, Accessed: May 2022.

[165] A. Vysocky and P. Novak, "Human-Robot Collaboration in Industry", *MM Science Journal*, vol. 9, no. 2, pp. 903–906, 2016.

[166] S. Bragança, E. Costa, I. Castellucci, and P. M. Arezes, "A Brief Overview of the Use of Collaborative Robots in Industry 4.0: Human Role and Safety", *Occupational and Environmental Safety and Health*, pp. 641–650, 2019.

[167] S. Lim and C. D'Souza, "A Narrative Review on Contemporary and Emerging Uses of Inertial Sensing in Occupational Ergonomics", *International Journal of Industrial Ergonomics*, vol. 76, p. 102 937, 2020.

[168] D. Battini, N. Berti, S. Finco, **M. Guidolin**, M. Reggiani, and L. Tagliapietra, "WEM-Platform: A Real-Time Platform for Full-Body Ergonomic Assessment and Feedback in Manufacturing and Logistics Systems", *Computers & Industrial Engineering*, vol. 164, p. 107 881, 2022.

[169] E. Bochinski, V. Eiselein, and T. Sikora, "High-Speed Tracking-by-Detection Without Using Image Information", in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, IEEE, 2017, pp. 1–6.

[170] D. Mehta, S. Sridhar, O. Sotnychenko, *et al.*, "VNect: Real-Time 3D Human Pose Estimation With a Single RGB Camera", *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–14, 2017.

[171] D. Mehta, O. Sotnychenko, F. Mueller, *et al.*, "XNect: Real-Time Multi-Person 3D Human Pose Estimation With a Single RGB Camera", *arXiv Preprint arXiv:1907.00837*, 2019.

[172] A. Martínez-González, M. Villamizar, O. Canévet, and J.-M. Odobez, "Real-Time Convolutional Networks for Depth-Based Human Pose Estimation", in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 41–47.

[173] B. Rim, N.-J. Sung, J. Ma, Y.-J. Choi, and M. Hong, "Real-Time Human Pose Estimation Using RGB-D Images and Deep Learning", *Journal of Internet Computing and Services*, vol. 21, no. 3, pp. 113–121, 2020.

[174] L. Chen, H. Ai, R. Chen, Z. Zhuang, and S. Liu, "Cross-View Tracking for Multi-Human 3D Pose Estimation at Over 100 FPS", in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2020, pp. 3279–3288.

[175] N. D. Reddy, L. Guigues, L. Pishchulin, J. Eledath, and S. G. Narasimhan, "TesseTrack: End-to-End Learnable Multi-Person Articulated 3D Pose Tracking", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 190–15 200.

[176] H. Chu, J.-H. Lee, Y.-C. Lee, C.-H. Hsu, J.-D. Li, and C.-S. Chen, "Part-Aware Measurement for Robust Multi-View Multi-Human 3D Pose Estimation and Tracking", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1472–1481.

[177] S. Bultmann and S. Behnke, "Real-Time Multi-View 3D Human Pose Estimation using Semantic Feedback to Smart Edge Sensors", *arXiv Preprint arXiv:2106.14729*, 2021.

[178] S. Moon, Y. Park, D. W. Ko, and I. H. Suh, "Multiple Kinect Sensor Fusion for Human Skeleton Tracking Using Kalman Filtering", *International Journal of Advanced Robotic Systems*, vol. 13, no. 2, p. 65, 2016.

[179] A. Kadkhodamohammadi, A. Gangi, M. de Mathelin, and N. Padoy, "A Multi-View RGB-D Approach for Human Pose Estimation in Operating Rooms", in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2017, pp. 363–372.

[180] G. Liu, G. Tian, J. Li, X. Zhu, and Z. Wang, "Human Action Recognition Using a Distributed RGB-Depth Camera Network", *IEEE Sensors Journal*, vol. 18, no. 18, pp. 7570–7576, 2018.

[181] K. Ryselis, T. Petkus, T. Blažauskas, R. Maskeliūnas, and R. Damaševičius, "Multiple Kinect Based System to Monitor and Analyze Key Performance Indicators of Physical Training", *Human-Centric Computing and Information Sciences*, vol. 10, no. 1, pp. 1–22, 2020.

[182] L. Zhou, N. Lannan, G. Fan, and J. Hausselle, "Human Motion Enhancement via Joint Optimization of Kinematic and Anthropometric Constraints", *EAI Endorsed Transactions on Bioengineering and Bioinformatics*, vol. 1, e1, 2021.

[183] A. Fender and J. Müller, "Velt: A Framework for Multi RGB-D Camera Systems", in *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces*, 2018, pp. 73–83.

[184] J. Munkres, "Algorithms for the Assignment and Transportation Problems", *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[185] S. Agarwal, K. Mierle, and T. C. S. Team, *Ceres Solver*, version 2.1, Mar. 2022. [Online]. Available: `https://github.com/ceres-solver/ceres-solver`.

[186] C. Pizzolato, D. G. Lloyd, M. Sartori, *et al.*, "CEINMS: A Toolbox to Investigate the Influence of Different Neural Control Solutions on the Prediction of Muscle Excitation and Joint Moments During Dynamic Motor Tasks", *Journal of Biomechanics*, vol. 48, no. 14, pp. 3929–3936, 2015.

[187] J. Diebel, "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors", *Matrix*, vol. 58, no. 15-16, pp. 1–35, 2006.

[188] G. Wu, S. Siegler, P. Allard, *et al.*, "ISB Recommendation on Definitions of Joint Coordinate System of Various Joints for the Reporting of Human Joint Motion — Part I: Ankle, Hip, and Spine", *Journal of Biomechanics*, vol. 35, no. 4, pp. 543–548, 2002.

[189] G. Wu, F. C. van der Helm, H. (DirkJan) Veeger, *et al.*, "ISB Recommendation on Definitions of Joint Coordinate Systems of Various Joints for the Reporting of Human Joint Motion — Part II: Shoulder, Elbow, Wrist and Hand", *Journal of Biomechanics*, vol. 38, no. 5, pp. 981–992, 2005.

[190] M. Jackson, B. Michaud, P. Tétreault, and M. Begon, "Improvements in Measuring Shoulder Joint Kinematics", *Journal of Biomechanics*, vol. 45, no. 12, pp. 2180–2183, 2012.

[191] M. Al Borno, J. O'Day, V. Ibarra, *et al.*, "OpenSense: An Open-Source Toolbox for Inertial-Measurement-Unit-Based Measurement of Lower Extremity Kinematics Over Long Durations", *Journal of Neuroengineering and Rehabilitation*, vol. 19, no. 1, pp. 1–11, 2022.

[192] N. Sarafianos, B. Boteanu, B. Ionescu, and I. A. Kakadiaris, "3D Human Pose Estimation: A Review of the Literature and Analysis of Covariates", *Computer Vision and Image Understanding*, vol. 152, pp. 1–20, 2016.

[193] *OpenSim Documentation*, `https://simtk-confluence.stanford.edu/display/OpenSim/Documentation`, Accessed: May 2022.

[194] M. G. Pandy, K. Sasaki, and S. Kim, "A Three-Dimensional Musculoskeletal Model of the Human Knee JoInternational Part 1: Theoretical Construction", *Computer Methods in Biomechanics and Bio Medical Engineering*, vol. 1, no. 2, pp. 87–108, 1997.

[195] M. Mirakhorlo, J. M. Visser, B. Goislard de Monsabert, F. Van der Helm, H. Maas, and H. Veeger, "Anatomical Parameters for Musculoskeletal Modeling of the Hand and Wrist", *International Biomechanics*, vol. 3, no. 1, pp. 40–49, 2016.

[196] B. Siciliano, O. Khatib, and T. Kröger, *Springer Handbook of Robotics*. Springer, 2008, vol. 200.

[197] C. Pizzolato, M. Reggiani, L. Modenese, and D. G. Lloyd, "Real-Time Inverse Kinematics and Inverse Dynamics for Lower Limb Applications Using OpenSim", *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 20, no. 4, pp. 436–445, 2017.

[198] M. Ben-Ari, *Principles of Concurrent and Distributed Programming, 2nd Edition*. Addison-Wesley, 2006.

[199] P. A. Buhr, M. Fortier, and M. H. Coffin, "Monitor Classification", *ACM Computing Surveys (CSUR)*, vol. 27, no. 1, pp. 63–107, 1995.

[200] O. A. Kannape and O. Blanke, "Self in Motion: Sensorimotor and Cognitive Mechanisms in Gait Agency", *Journal of Neurophysiology*, vol. 110, no. 8, pp. 1837–1847, 2013.

[201] G. Nicola, L. Tagliapietra, **M. Guidolin**, S. Ghidoni, and N. Pedrocchi, "Feedback Motion Planning in Human-Robot Shared Workspace via Deep Reinforcement Learning", *IEEE Transactions on Automation Science and Engineering*, [submitted - under review].

[202] N. Berti, S. Finco, **M. Guidolin**, M. Reggiani, and D. Battini, "Real-Time Postural Training Effects on Single and Multi-Person Ergonomic Risk Scores", *IFAC-PapersOnLine*, [accepted].

[203] R. Y. Tsai, R. K. Lenz, *et al.*, "A New Technique for Fully Autonomous and Efficient 3D Robotics Hand/Eye Calibration", *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, 1989.

[204] E. Olson, "AprilTag: A Robust and Flexible Visual Fiducial System", in *2011 IEEE International Conference on Robotics and Automation*, IEEE, 2011, pp. 3400–3407.

[205] T. R. Waters, V. Putz-Anderson, A. Garg, and L. J. Fine, "Revised NIOSH Equation for the Design and Evaluation of Manual Lifting Tasks", *Ergonomics*, vol. 36, no. 7, pp. 749–776, 1993.

[206] L. McAtamney and E. N. Corlett, "RULA: A Survey Method for the Investigation of Work-Related Upper Limb Disorders", *Applied Ergonomics*, vol. 24, no. 2, pp. 91–99, 1993.

[207] S. Hignett and L. McAtamney, "Rapid Entire Body Assessment (REBA)", *Applied Ergonomics*, vol. 31, no. 2, pp. 201–205, 2000.

[208] O. Karhu, P. Kansi, and I. Kuorinka, "Correcting Working Postures in Industry: A Practical Method for Analysis", *Applied Ergonomics*, vol. 8, no. 4, pp. 199–201, 1977.

[209] D. S. Chander and M. P. Cavatorta, "An Observational Method for Postural Ergonomic Risk Assessment (PERA)", *International Journal of Industrial Ergonomics*, vol. 57, pp. 32–41, 2017.

[210] M. Hovanec, P. Korba, and M. Šolc, "Tecnomatix for Successful Application in the Area of Simulation Manufacturing and Ergonomics", *15th International SGEM Geoconference on Informatics, Albena*, 2015.

[211] X. Tao, *Smart Fibres, Fabrics and Clothing: Fundamentals and Applications*. Woodhead Publishing Limited, 2001.

[212] R. Proesmans, A. Verleysen, R. Vleugels, P. Veske, V.-L. De Gusseme, and F. Wyffels, "Modular Piezoresistive Smart Textile for State Estimation of Cloths", *Sensors*, vol. 22, no. 1, p. 222, 2021.

[213] F. Seoane, A. Soroudi, K. Lu, *et al.*, "Textile-Friendly Interconnection Between Wearable Measurement Instrumentation and Sensorized Garments — Initial Performance Evaluation for Electrocardiogram Recordings", *Sensors*, vol. 19, no. 20, p. 4426, 2019.

[214] S.-W. Kang, H. Choi, H.-I. Park, *et al.*, "The Development of an IMU Integrated Clothes for Postural Monitoring Using Conductive Yarn and Interconnecting Technology", *Sensors*, vol. 17, no. 11, p. 2560, 2017.

[215] A. Angelucci, M. Cavicchioli, I. A. Cintorrino, *et al.*, "Smart Textiles and Sensorized Garments for Physiological Monitoring: A Review of Available Solutions and Techniques", *Sensors*, vol. 21, no. 3, p. 814, 2021.

[216] M. I. Mokhlespour Esfahani and M. A. Nussbaum, "Preferred Placement and Usability of a Smart Textile System vs. Inertial Measurement Units for Activity Monitoring", *Sensors*, vol. 18, no. 8, p. 2501, 2018.

[217] S. Muzaffar and I. A. M. Elfadel, "Shoe-Integrated, Force Sensor Design for Continuous Body Weight Monitoring", *Sensors*, vol. 20, no. 12, p. 3339, 2020.

[218] M. A. Wahid, S. Saragih, D. B. Wibowo, and I. Haryanto, "Simple Insole for Dynamic Foot Plantar Measurement in Walking Gait Analysis Using Force Sensitive Resistor", *International Research Journal of Innovations in Engineering and Technology*, vol. 5, no. 11, p. 57, 2021.

[219] M. Joshi and V. Deshpande, "A Systematic Review of Comparative Studies on Ergonomic Assessment Techniques", *International Journal of Industrial Ergonomics*, vol. 74, p. 102 865, 2019.

[220] U. Trinler and R. Baker, "Estimated Landmark Calibration of Biomechanical Models for Inverse Kinematics", *Medical Engineering & Physics*, vol. 51, pp. 79–83, 2018.

[221] P. Puchaud, C. Sauret, A. Muller, *et al.*, "Accuracy and Kinematics Consistency of Marker-Based Scaling Approaches on a Lower Limb Model: A Comparative Study With Imagery Data", *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 23, no. 3, pp. 114–125, 2020.

[222] P. Gerus, M. Sartori, T. F. Besier, *et al.*, "Subject-Specific Knee Joint Geometry Improves Predictions of Medial Tibiofemoral Contact Forces", *Journal of Biomechanics*, vol. 46, no. 16, pp. 2778–2786, 2013.

[223] J. Fernandez, J. Zhang, T. Heidlauf, *et al.*, "Multiscale Musculoskeletal Modelling, Data - Model Fusion and Electromyography-Informed Modelling", *Interface Focus*, vol. 6, no. 2, p. 20 150 084, 2016.