BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Master Thesis

Human Motion Analysis Using 3-D Pose Estimation Input

Dennis Ritter
Matriculation Number: 874357

A thesis presented for the degree
Master of Science

in the course
Media Informatics
Department VI
Beuth University of Applied Sciences

Supervisor:	Prof. Dr. Kristian Hildebrand
Assistant Supervisor:	Prof. Dr. Joachim Schimkat

Berlin
Germany
02.12.2019

Contents

1 Introduction

1.1 Motivation and Objectives

Sport exercises are a common part of diverse medical therapies. The correct execution of those exercises is essential for the healing process and therefore the success of such therapies. Frequently supervising the execution of exercises is an important but also a time-consuming task for therapists. If patients have to perform exercises at home they are probably not supervised by a therapist anymore and must review the execution on their own. Therefore, automating the supervision process could benefit not only therapists but also treated patients.
In order to make assumptions about a patients or trainees performance during the execution of sport exercises a potential software should be capable of solving different problems. Each performed exercise including multiple repetitions must be split into subsequences of single iterations of that exercise in order to be able to rate each repetition individually. Furthermore, the type of the performed exercise should be automatically identified.
Performing the segmentation, identification and supervision tasks for sport exercise motion sequences requires high-quality motion tracking data. Motion tracking software improved in recent years [8] [16] due to artificial intelligence research, available Human Pose datasets [4] [25] [44] and increased computational power to perform motion tracking tasks in realtime or near-realtime.

A procedure of software performing the tasks stated above should be:
(1) Receive 3-D motion tracking data of a patient performing a set of exercises.
(2) Use the positional data to determine joint angles as a description of the tracked pose.
(3) Segment the calculated joint angle sequences into single iterations.
(4) Identify the type of exercise.
(5) Rate the execution performance for each iteration of the executed exercise.

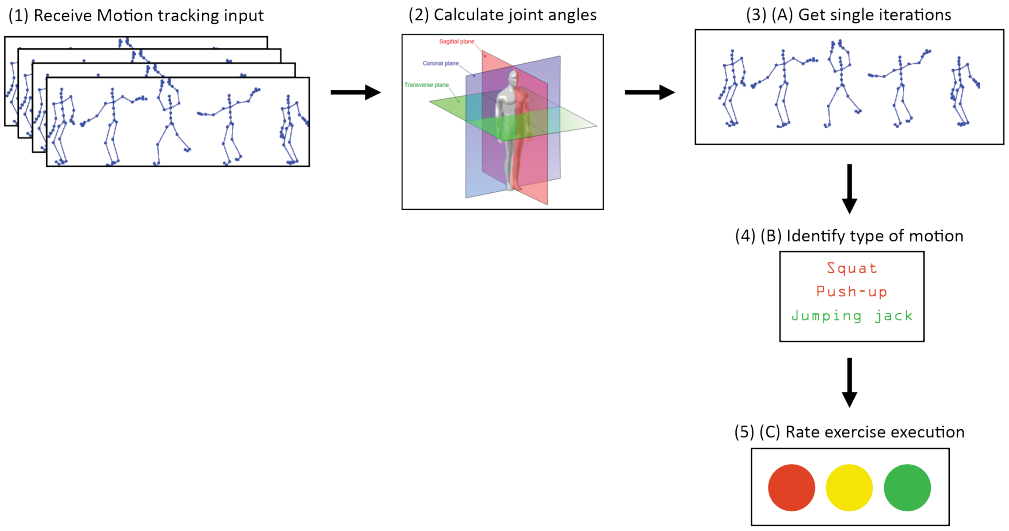

Figure 1.1: A basic software procedure illustrating the tasks of this work. 1-5 indicate the step in the procedure. A, B and C mark the corresponding main objectives of the thesis. *Skeleton figures image source:* [45, p. 2]. *Human planes of motion image source:* [51]

The described example procedure can also be seen in Figure 1.1.
To realize segmentation, identification and rating of exercises performed by patients, several problems must be solved. One problem in this context is the uniqueness of the human body. Humans differ in size and figure which complicates the process of comparing exercises of two different individual people. Additionally, the posture, range of motion and fitness level differs from person to person. This especially applies to very old people or people that are handicapped by injuries. Consequently, the definition of a correct exercise execution might vary for people with different body conditions.
Another problem is that even if a set of a type of exercise is already segmented into single iterations, the execution of each iteration might have been done at a different pace. But primarily the time of execution might vary considerably when comparing iterations done by different people.

After all, the objectives of this work can be summed up by naming four primary contributions:
(A) Generating single iteration subsequences from motion sequences that include multiple repetitions of a sport exercise.

(B) Matching motion sequences by determining their aligned distance to each other.
(C) Automatically rating the execution performance of sport exercises.
(D) Creating a 3-D motion sequence dataset of sport exercises.

1.2 Research Context

The research of this work is embedded in the *BewARe* [21] research project of the Beuth University of Applied Sciences. *BewARe* is an interdisciplinary research project including computer scientists, engineers and physiotherapists and aims to increase sport exercise-based therapies for hypertension patients. The goal is to develop a sensor-based Augmented Reality (AR) system that uses gamification elements to motivate elderly people to perform exercises as part of medical therapy. Based on acquired sensor data, the system adapts to the specific needs of users to optimize the individual training program. Additionally, the sensor data is used for instant user feedback while performing exercises in order to correct wrong executions and prevent possible negative effects and injuries.

The main focus of this thesis is to use depth sensor data to analyse performed exercises in order to gather information about the number of performed iterations and the correctness of each execution.

1.2.1 Project Conditions

In this section, the *BewARe* project conditions which influence this thesis are presented.

Depth Sensor Input

The depth sensor input is recorded by an *Intel RealSense D435* and *Intel RealSense D435i* [1] camera.

Human Pose Estimation Input

Human joint positions are estimated from depth camera input by a software that uses the Nuitrack SDK [2].

[1] https://www.intelrealsense.com/depth-camera-d435/
[2] https://nuitrack.com/

Exercises execution reference

The optimal execution of exercises is defined by medical human joint angles of the starting positions and end positions. A document stating such predefined exercises can be found on the attached data storage under *BewARe_exercise_concepts.pdf*. The document is written in german language.

1.3 Structure

The thesis is structured into six main chapters focussing on different aspects. This first chapter introduces the requirements, tasks and challenges of this work and outlines the motivation to solve them. In the second chapter, relevant and related work to the defined main objectives, *Subsequencing Motion Sequences*, *Matching Motion Sequences* and *Rating Execution of Exercises* is presented. After that, the approaches and challenges dealing with the main objectives are described in the main part of this work. Further, parts of the implemented prototype code are presented and explained. The fifth chapter then deals with the evaluation and presents the developed results. Lastly, the work will be concluded by discussing the contributions and results of chosen approaches and finally mention potential future work.

2 Related Work and Fundamentals

This chapter presents related work to the objectives of this thesis. First, methods and approaches to compute clinical meaningful human joint angles from motion tracking data will be introduced. Then, different approaches and domains to identify subsequences are briefly explained. Furthermore, motion matching, primarily using the Dynamic Time Warping (DTW) algorithm is introduced. Moreover, different approaches in former research of automatic exercise performance evaluations are explained.

2.1 Calculating Human Joint Angles

Calculating human joint angles from 3-D joint position data is a complicated task since "there are joints, such as the shoulder, for which there is no single definition of a joint angle that is anatomically meaningful for the full range of motion of the joint" [35, p. 50]. However, two approaches are frequently used to describe complex joint motions in an anatomical meaningful manner. One method uses *Euler-Cardan angles* which describe one rotation matrix interpreted as the matrix product of three sequential rotation matrices about three axes. [19] The other method is called *Joint Coordinate System (JCS)* and has been proposed by Grood and Suntay in 1983 [15]. The paper claims that the *JCS* method does not depend on a particular rotation sequence, which has been disproved by MacWilliams and Davis in [26]. Their work proves that the *JCS* method is always equivalent to the Euler-Cardan method for a particular sequence. Apart from the two mentioned methods, *Helical Angles* and *Quaternions* are also frequently used as representation for body part orientations as they are computationally robust. However, these methods are considered to not allow deriving clinically meaningful angles [47, p. 218]. As a consequence to their clinical meaninglessness, *Helical Angles* and *Quaternions* are not further investigated in this thesis.

Further, the medical joint angle definitions will be explained as well as the methods to describe relative human joint positions technically, aiming to express as close to the clinical definitions as possible.

2.1.1 Clinical Angle Definitions

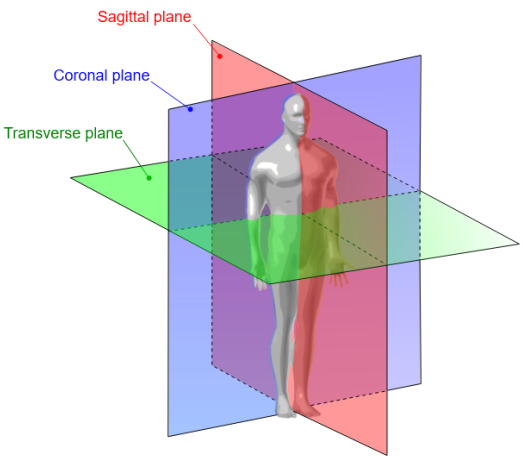

Figure 2.1: Anatomical planes of motion and anatomical position for humans. [51]

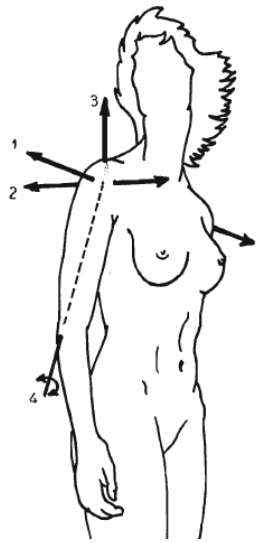

Figure 2.2: Rotation axes for clinical definitions of shoulder joint movements: (1) transverse axis, (2) sagittal axis, (3) vertical axis and (4) internal/external rotation axis. [17, p. 3]

Clinical joint positions are commonly described by four particular joint angles that represent a movement on a corresponding plane.

The *flexion* and *extension* angle represents movement on the *sagittal plane*. The movement is called *flexion* when the body part moves to the front or *extension* when it moves back. This rotation might also be described as a rotation about the *transverse axis* or *horizontal axis* which is a vector that points sideways with respect to the reviewed joint. The *transverse axis* is perpendicular to the *sagittal plane* whose origin lies in the center of the reviewed joint. The axis marked with number one (1) in Figure 2.2 displays the *transverse axis* for the shoulder. Figure 2.3 shows an Extension of 50 degrees on the left and a Flexion of 90 degrees on the right side. [17]

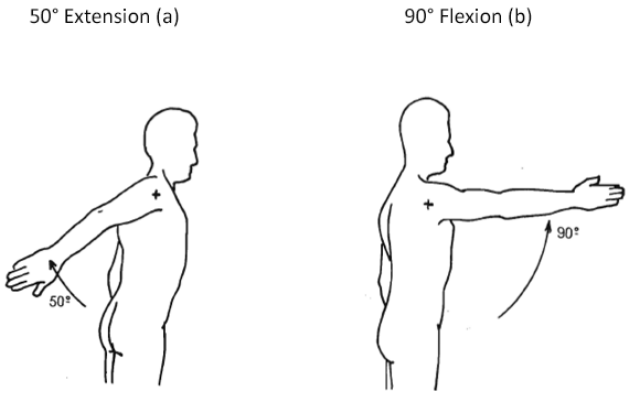

Figure 2.3: (a) 50° extension and (b) 90° flexion example for the right shoulder joint

[17, p. 5]

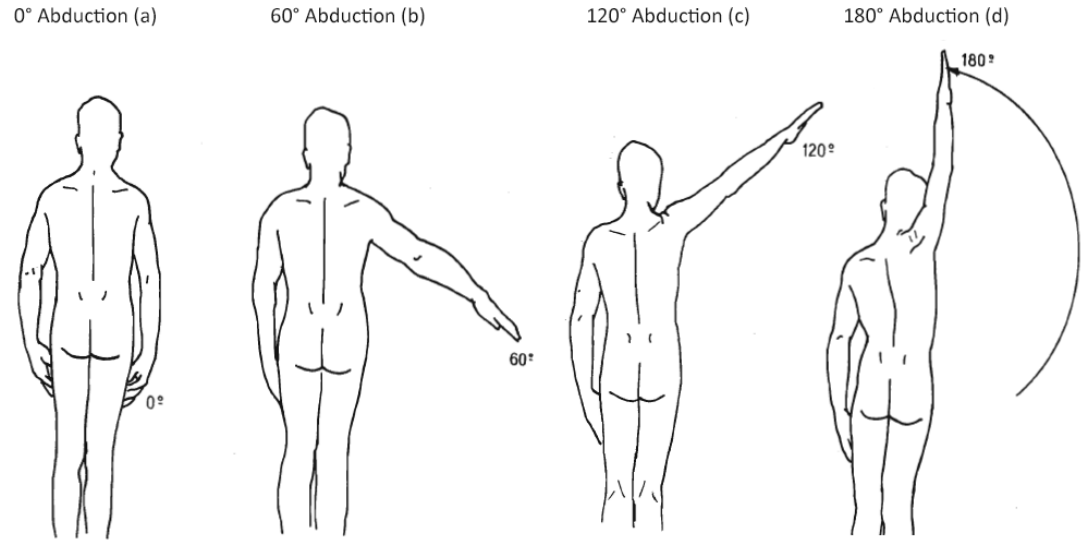

Figure 2.4: (a) 0° abduction, (b) 60° abduction, (c) 120° abduction and (d) 180° abduction for the right shoulder joint

[17, p. 7]

The *abduction* and *adduction* angle represents movement on the *frontal plane* or *coronal plane*. The movement is called an *abduction* when the body part moves away from the related fixed body or *adduction* when it moves to the body. This rotation might also be described as a rotation about the *sagittal axis* which is a frontal or backwards pointing vector with respect to the reviewed joint. The *sagittal axis*, which is also called the *antero-posterior axis*, is a perpendicular of the *frontal plane* whose origin lies in the center of the reviewed joint. The axis marked with number two (2) in Figure 2.2 displays the

Figure 2.5: 30° adduction for the right shoulder joint

[17, p. 5]

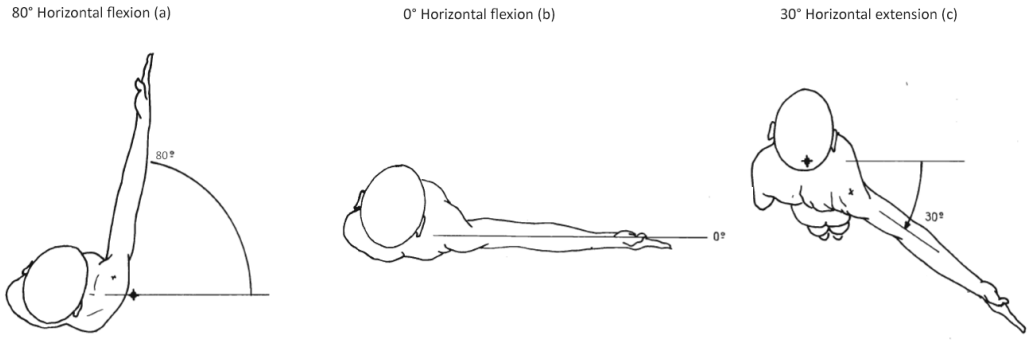

Figure 2.6: (a) 80° horizontal flexion, (b) 0° horizontal flexion and (c) 30° horizontal extension of the right shoulder

[17, p. 11]

sagittal axis for the shoulder. Figure 2.4 shows key frames of an abduction sequence example for the right shoulder. Figure 2.5 shows an example for a 30 degrees adduction of the right shoulder. [17]

The *horizontal flexion* and *horizontal extension* or *horizontal abduction* and *horizontal adduction* describe the same joint movement. It represents a movement on the *horizontal plane* that is running through the intersection of the frontal and sagittal plane. This rotation might also be described as rotation about the *vertical axis* which is an upwards or downwards pointing vector. The *vertical axis* is perpendicular to the *horizontal plane* whose origin lies in the center of the reviewed joint. The axis number three in Figure 2.2

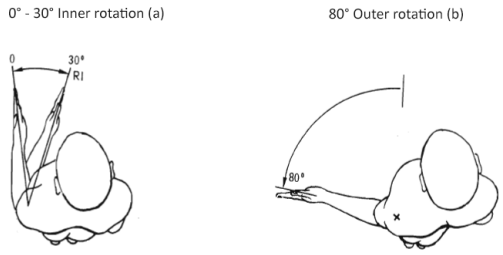

Figure 2.7: (a) 0°-30° internal rotation and (b) 30° external rotation. [17, p. 9]

displays the *vertical axis* for the shoulder. Figure 2.6 shows an example of *horizontal flexion* and *horizontal extension*. Person *a* in the picture displays zero degrees horizontal movement. Person *b* displays a 140 degrees horizontal flexion and person *c* shows a 30 degrees *horizontal extension*. [17]
The *internal rotation* and *external rotation* angle represent a rotation about an axis that connects the fixed joint and a corresponding free joint. In case of an *internal/external rotation* of the shoulder, these joints are the shoulder itself and the elbow. The movement is called an *internal rotation* when the body part rotates to the related body or *external rotation* when it rotates away from the body. The axis marked with number four (4) in Figure 2.2 displays the *internal/external rotation* axis for the shoulder. Figure 2.7 shows three different *internal/external rotations* of the left shoulder. *Person a* displays a 30 degrees *internal rotation* or the backward movement from 30 degrees *internal rotation* to zero degrees as indicated by the double-sided arrow. *Person b* shows an 80 degrees *external rotation*. [17]
Figure 2.1 shows the anatomical planes of motion described in the previous paragraphs as well as the *anatomical position* of human beings. The *anatomical position* is the one position where all clinical joint angles equal zero degrees. It is used as a reference when describing body parts in relation to each other. [42]

When computing joint angles from tracked motion data, it is required to transfer them to the medical definitions explained above. Clinical definitions of the presented motions are measured independently from each other in 2-D space, which does not allow to reliably break down clinical *flexion/extension*, *abduction/adduction* and *external/internal rotation* angles from 3-D human poses. Since the results for those clinical angles depend on the order of rotations, it is for example not possible to determine whether the movement was a *flexion* followed by an *abduction* or an *abduction* followed by a *flexion*. [5, chapter 8.1] [49, chapter 2.4]

2.1.2 Euler and Cardan Angles

Euler-Cardan angles are frequently utilised to describe joint movements close to their clinical definitions as explained in section 2.1.1. Generally, Euler-Cardan angles can be used to describe the orientation of an object in relation to a fixed coordinate system in three-dimensional space. In other words, they describe the orientation of a moving Local Coordinate System (LCS) relative to a fixed Global Coordinate System (GCS). [35, p. 50 ff.] Euler angles can describe a change of orientation as a sequence of three consecutive rotations. These rotations are defined as rotation matrices which can be seen in equation 2.2. [35, p. 51 ff]

In three dimensional space, twelve possible Euler-Cardan rotation sequences exist: *XYZ, XZY, YXZ, YZX, ZXY, ZYX, XYX, ZYZ, ZXZ, YXY, XZX, YZY*. [32] The first six sequences are also referred to as *Cardan angles* or *Tait-Bryan angles* and the last six sequences are also called *proper Euler angles*. [3] [11]
Each sequence of single rotations results in a different rotation matrix that is computed by the matrix product of those single rotations and represents a rotation about an unknown arbitrary axis. Equation 2.1 shows how to calculate the rotation matrix R for the Cardan sequence *XYZ*. Equation 2.3 shows the resulting rotation matrix R which is also called the *decomposition matrix*. Alpha (α) represents the angle for the first rotation about axis *X*. Beta (β) represents the angle for the second rotation about axis *Y*. Lastly, Gamma (γ) represents the angle for the third rotation about axis *Z*. The angles can be obtained directly from the rotation matrix R. Alpha (α) is computed from the elements (3,2) and (3,3) as seen in equation 2.4. Beta (β) is computed from the elements (1,1), (2,2) and (3,1) as seen in equation 2.5. Gamma (γ) is computed from the elements (2,1) and (1,1) as seen in equation 2.6. [35, p. 51 ff.]

$$R = RxRyRz \tag{2.1}$$

where

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} R_y = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} R_z = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

$$R = \begin{bmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha + \sin\gamma\cos\alpha & \sin\gamma\sin\alpha - \cos\gamma\sin\beta\cos\alpha \\ -\sin\gamma\cos\beta & \cos\alpha\cos\gamma - \sin\gamma\sin\beta\sin\alpha & \sin\gamma\sin\beta\cos\alpha + \cos\gamma\sin\alpha \\ \sin\beta & -\cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{2.3}$$

$$\alpha = \tan^{-1}\left(\frac{-R_{32}}{R_{33}}\right) \tag{2.4}$$

$$\beta = \tan^{-1}\left(\frac{R_{31}}{\sqrt{R_{11}^2 + R_{21}^2}}\right) \tag{2.5}$$

$$\gamma = \tan^{-1}\left(\frac{-R_{21}}{R_{11}}\right) \tag{2.6}$$

The most widely used Euler-Cardan sequence to determine clinically meaningful joint angles is *XYZ*, which is also called a Cardan sequence as described in the previous paragraph. If the *XYZ* sequence is used, it is important to define the Euclidean Coordinate System of the moving joint correctly. [35, p. 50 ff.]

Euler angles have a significant problem when using them to derive clinical meaningful angles from joint positions, which is the *Gimbal Lock*. A *Gimbal Lock* occurs when the second rotation of an Euler sequence is of 90 degrees because this leads to the first and third rotation axes being aligned to each other. Thus one of the three degrees of freedom (DOF) is eliminated as rotations about the first and third axes become equivalent to each other, which results in meaningless angles. [35, p. 53]
A general problem when computing ball joint angles for hip and shoulder is that the order of joint movements can not surely be determined. Clinical definitions of *flexion/extension* and *abduction/adduction* are only defined in 2-D and not consistent in 3-D space. This leads to differences because it is impossible to find out which action occurred first and which second. Thus leading to different results. [5, chapter 8.1] [49, chapter 2.4]

Euler angles are not used in this work, although their properties would theoretically allow it. Given the Gimbal Lock problem and sequence differences for joints with a high range of freedom as the shoulder, Euler-Cardan angles are not a perfect general solution to represent joint angles The given circumstances in this work allow a simpler method using *spherical coordinates* to determine *flexion/extension* and *abduction/adduction* angles that are close to their clinical definition.
One reason is that minor joint movements as *abduction/adduction* or *internal/external rotation* for joints like elbow or knee are ignored because the range of motion is very limited in most cases. It is assumed that they do not have a significant impact when evaluating sport exercises or analyzing human poses in a more general manner. Thus only the *flexion/extension* angle will be used for those joints which removes the necessity of complex rotation sequences. The only joints used in this work for which a single angle calculation is not viable are the hip and shoulder joints. These joints, especially the shoulder, still cause several challenges and problems that are not solved by using Euler-Cardan angles. Another reason to simplify the angle calculations is that the current input data does not allow to derive the orientation of body parts. Thus the *internal/external rotation* of hip and shoulder is not defined for several poses and will be ignored. So as Euler-Cardan angles do represent a generally safe and reliable method to compute all joint angles without significant problems, it has been decided to choose an easier method as long as the given circumstances remain.
Nevertheless, Euler angles played an important role to understand the challenges when calculating joint angles of complex motions in 3-D space. Most of all they helped to generally understand movements of ball joints and the challenges finding a standard method to compute clinical meaningful joint angles.

2.1.3 Joint Coordinate System

Grood and Suntay developed the *JCS Method* to describe three-dimensional joint motions as closely as possible to clinical definitions. In their work, a JCS is applied to the knee joint but it is even recommended to use the method for arbitrary joint motion descriptions. Generally, a JCS can be described as a three-dimensional Euclidean coordinate system whose origin lies in the center of rotation. Two of its three axes are called *fixed axes* because they are established by using other body part positions, thus they are fixed to the body. The third axis is called the *floating axis* or *free axis* because it is not directly attached

to some body part, but the calculated perpendicular of the fixed axes. In [15], a rotation about one of these axes represents a single medical joint movement. So a rotation about one axis represents a clinical *abduction* and *adduction*, another axis represents a *flexion* and *extension* and the last axis represents an *internal rotation* and *external rotation*. In their work [15], Grood and Suntay claim that their method is an improvement compared to the frequently used Euler-Cardan angles because it is independent to a rotation sequence. But in [26] it has been explained and proofed that the *JCS Method* described in [15] is always equivalent to at least one of the twelve Euler-Cardan sequences since the choice of axes in a JCS already imposes a rotation sequence.

In 2002 and 2005 the International Society of Biomechanics (ISB) released recommendations for the construction of JCS to measure human joint angles. The proposed set of different JCS are based on Grood and Suntay's work [15] and a proposal for reporting kinematic data by Wu and Cavanagh from 1995 [48]. The ISB's motivation was to improve communication and the exchange of research results among researchers and clinicians. The Standardization and Terminology Committee (STC) involved nearly 30 experts in joint biomechanics to develop proposals for several joints of the human body. [50][49]
The most interesting of the proposed JCSs for human joints for this work are those for shoulder and hip since they are ball joints and have three DOF which makes them hard to measure and translate into clinical definitions. As the orientation of extremities is controlled by them, they play an important role in a lot of sport exercises.
The ISB offers two different options to define axes of a JCS for shoulder joints. The first option places the coordinate systems origin to the estimated position of the *Glenohumeral* rotation center which is further simplified as the *shoulder joint*. The Z-axis represents the line connecting the shoulder joint and the center of the *most caudal point on the lateral epicondyle and medial epicondyle* which is further simplified as the *elbow joint*. Z-axis must point to the right for the right shoulder and left for the left shoulder. The X-axis is defined by the perpendicular to the plane formed by the shoulder joint and the *lateral and medial epicondyle* pointing forward. Lastly, the Z-axis is defined by the perpendicular line to the Y-axis and Z-axis. The second option differs only in the Z-axis definition. It utilizes the Y-axis of the corresponding elbow JCS which is defined by the line perpendicular to the plane constructed from the elbow joints position, the *most caudal-lateral point on the radial styloid* and the *most caudal-medial point on the ulnar styloid* pointing forward. [49]

Both recommendations of [49] can not be used in this work because of missing po-

sitional information. The first option is not usable as positions of the *lateral and medial epicondyle* are not available from motion tracking input data. As a result, the alignment of the Z-axis can not be determined. The second option can not be used because of missing positional information about the *radial and ulnar styloid* that is needed to construct a JCS for the elbow joint, whose Y-axis is utilized to construct the Z-axis for the shoulder joint JCS. Similarly, the hips joints JCS proposal from [50] can not be used in this work as positional information about the *anterior superior iliac spine*, *posterior superior iliac spine* and *femoral epicondyle* are missing. Therefore, a different shoulder and hip JCS from the recommondations of [49], which can be constructed from three-dimensional joint positions available from motion tracking data input will be defined later in this work.
Now that related work and accompanied challenges regarding joint angles have been introduced, the next section follows, dealing with the subsequencing of motion sequences.

2.2 Subsequencing Motion Sequences

Subsequencing of motion sequences have been researched in order to solve problems in various contexts. But mostly to either retrieve motion sequences from a database by a given query sequence or to improve synthesising motion sequences based on other sequences or parts of them. Generally, the subsequencing process is referred to as indexing, as the classification of such subsequences for easier retrieval and therefore time-saving in various practical use-cases is the main motivation. [18]
Keogh et al. used a lower bounding distance and uniform scaling approach in [18] to retrieve motion sequences, whose first frames are similar to the end frames of other sequences. Targeting to improve the process of concatenating small sequences to produce a longer, complex sequence. Like for example a martial arts sequence of a punch, followed by a kick, followed by a jumping kick. Where the punch, kick and jumping kick would be singular sequences that allow easy concatenation because of their fitting start and end subsequences.
A more general approach is the decomposition of motions into elemental building blocks, from which more complex motions can be constructed. In [7], the term *Moveme* has been introduced. A *Moveme* describes a decomposed primitive motion of some elementary action, similar to phonemes in spoken language. Del Vecchio et al. further researched the concept of *Movemes* in order to create a motion alphabet of them [14].
LaViers et al. specified a grammar for ballet leg positions in [23] in order to improve human-like robot motions of other formalised motions like human gestures or other types

of motions, which allow human robots to better react to dynamic environments.
In [9], Chang et al. are counting repetitions of sport exercises analysing tracked accelerometer data sequence graphs for repeating patterns. Das et al. also used accelerometer data of a smartwatch in [13]. In their paper, they analyse the accelerometer sequences for peaks and valleys which represent single iterations of the performed exercise. In [24], Li et al. analysed an improved accelerometer signal for peaks to successfully identify repetitions in free weight exercises.

Some of the presented papers use a general approach in order to classify and synthesise motions in general, meaning motions of different domains. A goal of this thesis is to successfully identify single iterations of performed exercises. Papers that analyse repetitions of sport exercises mostly use a pattern or peak analysis approach. As the context of the latter is very similar to the context of this thesis, the challenge of identifying subsequences is approached by analysing joint angle sequences within performed exercise sets.

Smoothing Joint Angle Sequence Data

It is common practice to filter raw input data or directly derived input data in order to remove noise and outliers reasoned by technical or environmental inaccuracy. In 1964 Savitzky and Golay proposed a data smoothing method that is logically based on local least-squares and polynomial approximation for single data points within longer sequences. [38] Their method is used in so-called Savitzky-Golay-Filters in order to not only smooth noisy data in signal processing, but also in other domains. Such filters differ in a window size and polynomial order. Meaning the use of a polynomial order to approximate several data points. [39]
A Savitzky-Golay filter will be used to smooth the input data for the subsequencing process, as it is an adequate method to remove noise and outliers that appear due to inaccuracies of the former motion tracking.

2.3 Matching Motion Sequences

Research in determining the similarity of motion sequences has increased with the usage of data-driven animation techniques in computer games and animated movies. [18] Using existing motion sequences from a database as a starting point for new varieties of such animations or in order to synthesise derived animations may reduce the production workload extensively.

Different approaches have been developed to deal with the challenges of matching motion sequences and the often accompanied segmentation of complex human motion sequences that may contain various actions. Segmenting or indexing techniques, for example used in [18], [6] and [30], are aiming to recognise small and simple movements that can be decomposed from more complex motions. However, in the context of this thesis, complex sequences are sequences of multiple iterations performing an exercise and the variety in those complex sequences is rather small as they do not contain different exercises. Thus the decomposition of such sequences is not accounted in this chapter but is addressed in chapter 2.2.
Focussing on the pure matching of two motion sequences, the most common approaches are different approaches of Template Matching [10] [33] [31], Hidden Markov Models [7] [34] and Dynamic Time Warping [1] [29].
In [29, chapter 10], Mueller describes the usage of DTW for motion comparison and retrieval. DTW is an algorithm that is widely used across different fields of research to measure a distance of two arbitrary time series (find a detailed explanation in chapter 2.3.1). In [29, chapter 10], two different approaches to determine pose distances are pointed out. One DTW approach utilises sequences of tracked 3-D point cloud data and the other approach uses sequences of human joint angles in order to represent motion sequences. One feature of the joint angle representation is that they are invariant to translations, rotations and scalings of the whole skeleton. This can be an advantage and disadvantage at the same time, as a motion performed in different body orientations is still considered similar, which is desired in most cases. But in some cases, this is a disadvantage, for example, a push up exercise is very different from motions where a person is standing and pushes the arms to the front. But justified by the rotation invariance, both motions may be considered very similar. [29, chapter 10] Moreover, some joint rotations might have a greater effect on the pose of a person. As a rotation of the shoulder joint arguably has a greater impact than the wrist angles to the pose as a whole as the complete arm is moved when the shoulder rotates

and not just the hand. This problem is addressed by assigning weights to different joints to control the influence of each joint angle. But the importance of particular joint angles may also change within a sequence, depending on the current overall pose or context of the motion. However, overall joint angles, as well as point cloud data, are viable inputs for the DTW algorithm in order to determine a meaningful distance to distinguish motion sequencen from each other. [29]
The DTW approach is well suited to perform a comparison between two motion sequences of sport exercises as the algorithm handles the possible time shift between those and calculates a distance at the same time. Therefore, DTW will be used for the matching task in this thesis. Moreover, both types of input data that are recommended in [29, chapter 10] are very similar to the available calculated pose describing data of this work. Chapter 2.3.1 describes the functionality of the DTW algorithm in more detail.

2.3.1 Dynamic Time Warping (DTW)

DTW is an algorithm that is used to measure the similarity of two time series, which may vary in length. The algorithm has been originally developed and used for speech recognition [36] but has been applied to several other applications like handwriting recognition [43], gesture recognition [22] [12] and computer vision [28].
To measure the similarity of two time series, the DTW algorithm elastically transforms the two time series in order to align them as optimal as possible. The alignment results in more accurate similarity measurements by ignoring time shifts within the two time series. The optimal alignment is represented by a warp path that has two general constraints. The path must start at the beginning of both time series and also finish at the end of both time series, to ensure that every index of both have been referenced in it. And secondly, the time indices must increase monotonically to prevent the path from overlapping. The warp path runs through a cost matrix that has an x-axis size that equals the length of time series one and a y-axis size that equals the length of time series two. This cost matrix stores the distances of the corresponding indices in the time series to each other. The DTW algorithm calculates that warp path through this cost matrix, which results in the minimum total distance, thus representing the optimal alignment of both time series following the mentioned constraints. [37]
Figure 2.8 shows two arbitrary time series and their matched time indices that have been determined by the DTW algorithm. Figure 2.9 shows the same time series as Figure 2.8 and the resulting warp path running through the cost matrix, representing the optimal alignment

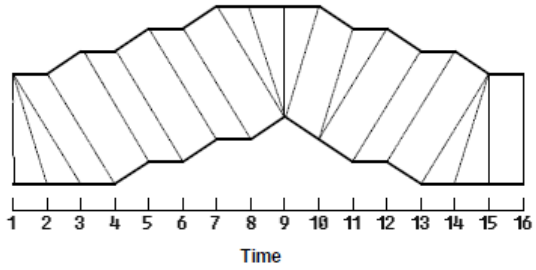

Figure 2.8: Two arbitrary time series and their optimal alignment visualised by lines that connect the matched data indices. [37, p. 1]

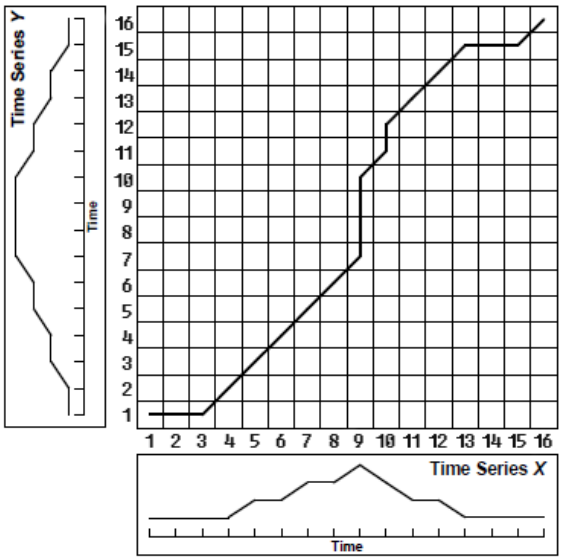

Figure 2.9: A cost matrix with the minimum-distance warp path of two time series. [37, p. 2]

of these time series. In this case, the optimal matching index pairs are: (1,1), (2,1), (3,1), (4,2), (5,3), (6,4), (7,5), (8,6), (9,7), (9,8), (9,9), (9,10), (10,11), (10,12), (11,13), (12,14), (13,15), (14,15), (15,15), (16,16). [37]

One disadvantage of the original DTW algorithm is its exponential complexity, which limits the usage to rather small sequences. As a consequence, Salvador and Chan presented the *fastDTW* algorithm in their work *FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space* [37]. The paper names three main approaches to optimise the complexity of the original DTW algorithm. (1) Shrinking the sequences into smaller sequences being as similar as possible to their original. (2) Finding a minimum-distance warp path at lower resolutions in order to use that as initial input at higher resolutions. (3) Refining the initial warp path projected from a lower resolution by applying local

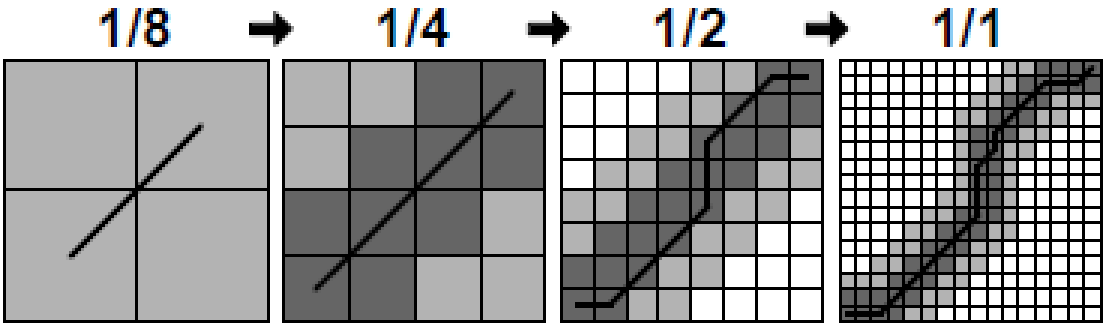

Figure 2.10: Four different resolutions of a warp path calculated by the fastDTW algorithm. [37, p. 4]

adjustments. The described procedure is visualised in Figure 2.10, where four different resolutions of a warping path are displayed. [37]

With the DTW and *fastDTW* algorithm, the matching of two similar motion sequences could be realised, as the calculated distance of the optimal time series alignment represents the similarity of two time series. Based on that fact, it might be possible to identify the type of a particular exercise by comparing it to different ground-truth sequences of different types of exercises.

2.4 Rating Execution of Exercises

Automatically rating the execution of exercises has gained interest with increasing quality of motion tracking possibilities. Some approaches use so-called gold-standards, which are ground-truth motion sequences. Those are used to evaluate the trainees performance by comparing them to the trainees motion. [2] [41] Others use predefined rulesets for a direct comparison and performance evaluation [52] or determine exhaustion levels of trainees in order to guide them during the exercise [13].
In [13], Das et al. calculate an *Ease Factor*, which represents how comfortable a trainee is performing an exercise variation by determining the variance in execution speed. Additionally, a trainee's *Comfort Factor* is calculated utilising a relation between heart rate variance and hand tremor.
In [2], an evaluation of dancing performances is done comparing tracked sequences to a defined gold-standard. In order to determine the choreography score, joint velocity vectors

and spatial joint positions are utilised.
In 2013, Su developed a Kinect-based system in [41], that determines the quality of rehabilitation exercise performances. In the paper, a gold-standard approach that uses personalised recorded exercise executions of a guiding physician is used. Patients' exercise performances are evaluated against the physicians recorded gold-standard in order to determine the quality of execution. To achieve that, a trajectory path and speed variation distance are determined using the DTW algorithm.
In [52], Zhao et al. also designed and implemented a Kinect-based system to guide users while performing rehabilitation exercises. The approach is based on a rule set for each exercise that defines certain key angles and bone orientations that represent a correct execution of the respective exercise. The software will then give guidance to the trainee, depending on the comparison results of the tracked execution to the defined ruleset.

Two general approaches to evaluate exercise performances are promising and suit the context and requirements of this thesis well. On the one hand, predefined rulesets for each particular exercise may be able to give high-quality exercise performance feedback back to the user. Based on such predefined rules, a system could guide a user very accurately, respecting each rule separately. On the other hand, matching tracked trainee sequences to a ground-truth standard may also be suited to evaluate the trainees performance. Personalized guidance with this approach could be better, as a trainer or physician is able to personalize the ground-truth definition of exercises for each user by tracking the gold-standard together with them as done in [41].
The project context of this thesis allows defining rulesets based on predefined joint angles for start and end positions of exercises (see chapter 1.2). Apart from that, the comparison of a motion sequence to a ground-truth standard is already being researched by approaching the matching challenge of this thesis (contribution (B) in chapter 1.1). Given these conditions, both approaches will be examined in this thesis in order to determine the quality of exercise executions.

2.5 Summary

This chapter introduces research and standardisation approaches to calculate medically meaningful joint angles as they are utilised for pose and motion representations and used to solve all the main challenges of this thesis defined in chapter 1.1. It also gives a brief

introduction to former research approaches related to these main contributions. Starting with different contributions regarding subsequencing and indexing for motion sequences, followed by motion matching approaches with a focus on Dynamic Time Warping and lastly presenting recent research that aims to automatically rate the execution of exercises. The research showed, that the tasks of this thesis are not only subject to computer vision challenges but moreover challenges across different fields of research like for example medicine, biomechanics, robotics and signal processing.

3 Methodology

In this chapter, the main part of this thesis is presented. Therefore, chosen approaches in order to solve the main challenges of this work are explained in detail.
First, the acquisition and representation of used input data is introduced. Explaining the creation of a labled dataset of exercise motion sequences as well as defining the representation of motion sequences and exercises in the context of this thesis.
Subsequently, the approach of human joint angle calculations is presented. Including the transformation from a GCS to a JCS and the angle calculations for ball joints and non-ball joints utilising spherical coordinates. The calculated joint angles lie the foundation for the approaches that handle the challenges of the main objectives, as they are used as input data, directly derived from 3-D position data of sequence representations.
Afterward, the subsequencing approach is explained along with examples for each part of the process. This includes the identification of local extrema in prioritised joint angle sequences and subsequent filters that lead to the identification of single iteration subsequences in motion sequences of exercise sets.
The next section deals with the subject of matching motion sequences based on the DTW algorithm and Euclidean distances. Highlighting the importance of aligning the examined sequences before determining their similarity.
Further, the two approaches to rate performed exercises will be presented. Comparing predefined clinical target angles to the joint angles of the performing trainee on the one hand. And deriving quality ratings from the similarity between a query sequence and a ground-truth exercise sequence on the other hand.
Lastly, the presented approaches will be summed up and briefly put in context with each other.

3.1 Data Acquisition

In order to test and evaluate the methods, which are developed in this thesis, a labeled dataset of exercise motion sequences has been created. In this chapter, the creation process of this dataset is presented followed by definitions of motion sequences and exercises in context of this thesis.

3.1.1 Test Dataset

The recorded test dataset to test and evaluate approaches regarding the main objectives of this thesis contains 1546 sequence files. 1407 of those files are single iteration exercise sequences and 139 files are sequences including ten iterations of an exercise. Resulting in a maximum of 2797 exercise iterations. The complete dataset can be found on the attached data storage under *motion_tracking_dataset.zip*.

Figure 3.1: A montage of five images displaying trainees performing exercises.

The tracking software is a Unity program that uses the *Nuitrack SDK* for the identification of human body parts. The sequences have been recorded with an *Intel Realsense D435* and an *Intel Realsense D435i* camera simultaneously. Both cameras were placed on tripods right next to each other and positioned in front of the test person. The positions of the cameras have been set to a height of 85cm from the ground and a distance of 200cm to the

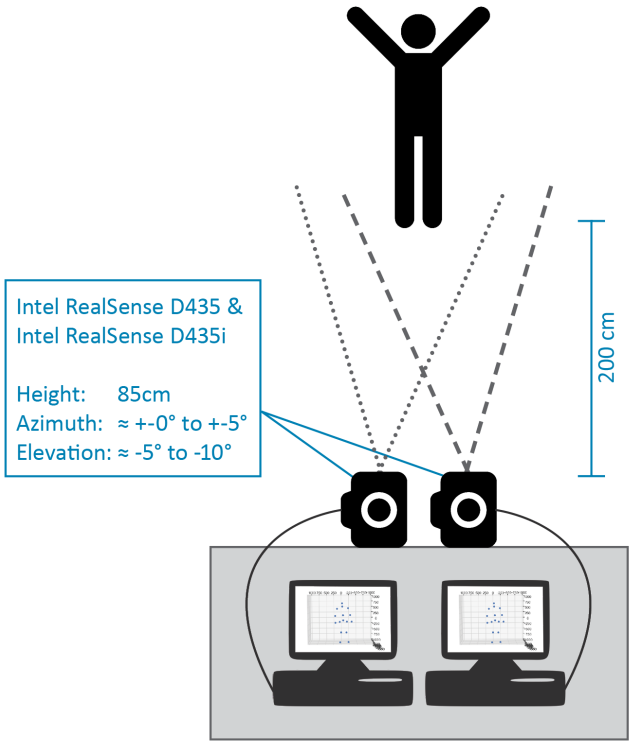

Figure 3.2: A schematic illustration of the motion tracking setup described in this thesis.

test person marker. The cameras were turned by 90 degrees to record in portrait mode in order to increase the vertical range of the viewport. The orientation of the cameras has been set to approximately minus five respectively five degrees azimuth angle and minus five to minus ten degrees elevation angle. The settings of the orientation were done by hand, with the goal to cover the entire range of the trainees motion. Figure 3.2 shows a schematic illustration of the tracking setup.

The described setup has been used to track eleven different exercises performed by eight test person. Table 3.1 shows the distribution of sequence files and resulting iterations across the recorded exercises. It shows how many sequence files and iterations have been recorded for each exercise. Differences in the number of files and iterations as between squat and triceps extensions occur due to removed sequences were tracking issues made the sequences unusable or users that did not want or could not perform specific exercises. Each recorded motion sequence is represented by a resulting JavaScript Object Notation (JSON) file, which most importantly includes a list of 3-D positions of all tracked body parts. The next section describes what is included in such output file and explains how it is further utilised.

Exercise	No. of Sequence Files	No. of Single i Seqs	No. of Ten i Seqs	No. of Total Iterations
biceps curl left	130	118	12	238
biceps curl right	132	120	12	240
knee lift left	134	122	12	242
knee lift right	136	124	12	244
lunge left	132	120	12	240
lunge right	132	120	12	240
overhead press	155	141	14	281
side step	173	158	15	308
squat	171	155	16	315
triceps extension left	131	119	12	239
triceps extension right	120	110	10	210
Overall	1546	1407	139	2797

Table 3.1: The number of sequence files, single-iteration-sequences, ten-iteration-sequences and total iterations per exercise listed by exercises.

3.1.2 Motion Sequences

A motion sequence is defined by a *JSON* file, which consists of information about the sequences' name, the date of recording and most relevantly the positions of each tracked body part for all frames of the sequence. Moreover, such sequence file contains the timestamp for each frame of the sequence and the tracking format, which maps body parts to indices in the *3D positions list*. Listing 3.1 shows an example of a sequence file containing only one frame. The positional information is available under the *frames* key. Each list element of the *frames* list is a list itself and includes the 3-D positions of one frame for each body part. It is notable, that every three values in that list represent the x, y and z position of one body part. To assign the correct body part, the *format* dictionary (see Listing 3.1 line 4-12) must be utilised. Every number value in that dictionary multiplied by three represents the x position of the respective body part, the following two elements then represent the y and z position.

```
{
  "name": "Sequence Data",
  "date": "2019-10-24T14:33:45.4612904+02:00",
  "format": {
    "LeftWrist": 0,
    "LeftElbow": 1,
    "LeftShoulder": 2,
  ...
    "RightElbow": 13,
    "RightShoulder": 14,
    "Head": 15
  },
  "timestamps": [
    1571920425462.283
  ],
  "frames": [
    [
      268.1098,46.3126831,1913.36121,
      252.486618,335.118378,1984.90454,
      175.214462,644.7952,1951.10229,
      ...
      -247.3602,360.497742,2104.1543,
      -179.442459,661.147,2014.65271,
      -5.89409542,867.4157,1939.42981
    ]
  ]
}
```

Listing 3.1: Examplary JSON file representation of a motion sequence.

In order to analyse a sequence, the body part positions for each frame are retrieved and generalised in a preprocessing step. To utilise specific motion tracking input, a processing function for that input must be implemented. Such processing function initialises and returns a *Sequence* object instance that contains all available information of a motion sequence in a generalised format. Furthermore, the returned instance is enhanced by

joint angles, which are calculated from the positional information as described in 3.2. Additionally, the object provides merge and slicing functions. The *Sequence* object will be used for further steps in which the data is going to be analysed and interpreted.

3.1.3 Exercises

Exercises are defined by JSON files that contain all necessary information to analyse motion sequences with respect to that exercise. Besides a name and a description of an exercise, such files hold information about the target angles for all body parts and each type of angle for these body parts. A target angle specifies, what value range of a particular angle should be reached at the start pose and the end pose when performing the exercise. In addition to that, each particular angle also has a priority value from 0.0 to 1.0, which specifies the importance of that angle for an exercise. A priority of 0.0 would mean the respective angle is of no importance at all. A priority of 1.0 means, that it is very important and characterises that exercise. Such angles with a priority of 1.0 are also called *prioritised angles* within this thesis.
As an example, the prioritised angles of a squat exercise might be the left and right hips *flexion/extension* angles and the left and right knee *flexion/extension angles*. The target angle ranges for these prioritised angles then could be 0 to 10 for the starting positions and 45 to 90 degrees for the end positions.
As trainees that perform an exercise might have injuries or other limitations that require personalised target angles, the exercise JSON file also has a *userId* field. Consequently, every exercise can be personalised to specific circumstances that affect the defined target angles. As an example, listing 3.2 shows an excerpt of an exercise file describing a squat exercise.

```
{
  "name": "Squat",
  "userId": 1,
  "description": "Lower the hips and stand back up.",
  "angles": {
    "start": {
    "hip_left": {
        "flexion_extension": {
          "angle": [0, 10],
          "priority": 1.0
        },
        "abduction_adduction": {
          "angle": [0, 0],
          "priority": 0.5
        },
        "innerrotation_outerrotation": {...},
    },
    "hip_right": {...},
    ...
    },
    "end": {
      "hip_left": {
        "flexion_extension": {WW
          "angle": [45, 90],
          "priority": 1.0
        },
        "innerrotation_outerrotation": {...},
        "abduction_adduction": {...}
    },
    "hip_right": {...},
    ...
  }
}
```

Listing 3.2: Extraction from exercise file Squat.json

3.2 Calculating Human Joint Angles

Computing joint angles from the preprocessed input data (see section 3.1.2) involves up to three main steps that are performed consecutively for each frame of a sequence.
For ball joints, a local Joint Coordinate System (JCS) is constructed first. After that, two angles are computed by utilising spherical positions. Lastly, the computed angles are interpreted and further processed for their specific usage.
The angles for non-ball joints are computed without creating a local JCS. Since only one angle is considered relevant, the spherical positions do not have to be used. Instead, the angle can be computed by utilizing the *dot product* of two vectors related to the joint.

3.2.1 Joint Coordinate System Creation

A JCS is created to perform angle calculations for ball joints which currently involves the left and right hip and shoulder joints. The JCS is needed to get a spatially moving joint position relative to the center of rotation. The center of rotation and origin of the JCS is the position of the joint whose angles are going to be computed. The spatially moving joint is a specific joint that changes its position as a result of such a rotation because it is connected to it through a limb. In case of the right shoulder, the rotating joint would be the right shoulder joint, the spatially moving joint would be the right elbow joint and the connecting limb would be the upper arm.
As a consequence, the spatially moving joints' 3-D position can be converted into spherical coordinates with the shoulder joint position as center of the sphere. The resulting two angles then can be adjusted to correspond to clinical *flexion/extension* and *abduction/adduction* angles.

The general idea is to find a transformation that transforms all joint positions in such a way that their new position is relative to the rotating joint.
Three positional 3-D positions in the Global Coordinate System (GCS) are needed to perform this task. (1) The intended origin of the JCS, which is the position of the rotating joint. (2) A point in the GCS that describes the orientation of the x-axis of the JCS and is further referenced as the *x-orientation-point*. (3) And a point in the GCS that describes the orientation of the y-axis of the JCS and is further referenced as the

y-orientation-point. The final resulting JCS will be a right-handed Euclidean Coordinate System with a x-axis pointing left, a y-axis pointing up and a z-axis pointing to the rear from camera perspective.

Finding JCS Axes

As a first step, the x-axis direction is computed by subtracting the origin from the *x-orientation-point* which results in a vector vx that starts at the origin point and ends at the *x-orientation-point*.
The x-position of the 3-D vector vx points to will then be checked against zero, which is the x-position of the origin point o, to ensure the intended direction of the x-axis. If the GCS x-axis points to the left and vx_x is positive, vx points to the left already. If vx_x is negative, vx currently points to the right and must be flipped by negating the coordinates of vx. The same method is used to ensure the JCS y-axis to point up and the z-axis to point away from the camera.
The second step is the computation of the z-axis direction vz. vz is a perpendicular vector to vx and the vector from origin o to the *y-orientation-point*, which is only a point on the xy-plane but not representing vy. After that, it is ensured that vz points away from the camera.
The third step is the computation of the y-axis vy, which is a perpendicular vector to vx and vz that is assured to point up.
The results are three vectors vx, vy and vz, which can now be used to find a transformation to project any point from the GCS to the JCS.

Transforming Joint Positions from GCS to JCS

To transform any point from the GCS to the JCS, a transformation must be found that results in the GCS covering the JCS.
The positional and orientational alignment is performed by three consecutive transformations. The first transformation is a translation T from the zero position $vzero$ to the new *JCS* origin o, which is computed by subtracting o from $vzero$.
The second transformation Rx must represent a rotation that aligns the JCS x-axis direction vector vx to the GCS x-axis direction vector, which is being defined as $[1,0,0]$. To construct the rotation matrix Rx, a rotation axis and an angle α is needed. The rotation axis is calculated by the cross product between the normalised vx and $[1,0,0]$ vectors, which

represents a perpendicular vector to these two. Then the angle between vx and $[1,0,0]$ is calculated by the dot product of these. As a consequence, we are able to define the rotation matrix Rx by using the *Euler-Rodrigues formula* [27].
The third transformation Ry must represent a rotation that aligns the JCS y-axis direction vector vy to the GCS y-axis direction vector $[0,1,0]$ after the x-axes have been aligned already. Again a rotation axis and angle is needed to construct the Rotation matrix Ry. To find the rotation angle, a matrix multiplication of vy and Rx is performed to rotate vy. After that, the angle between $Rx \cdot vy$ and $[0,1,0]$ can be determined by calculating the dot product of them. The rotation axis will be vx as we don't want to change any x-positions after performing the rotation Rx.
As a result, three 4x4 transformation matrices have been determined: T, Rx and Ry. To summarise these transformations in one 4x4 transformation matrix M, the matrix product $M = T \cdot Rx \cdot Ry$ is determined. Further, it is possible to project any point from the GCS into the JCS by multiplying the point with the transformation matrix M.

3.2.2 Angle Calculation

This section explains how the actual joint angles are computed. Priorly it is important to note that *internal/external rotations* will be ignored for all joints since the current input data does not allow to identify the orientation of body parts without exceeding the effort of this work.
Additionally, the range of motion for non-ball joints is simplified to one angle. *Flexion/Extension* is considered the only relevant motion for such joints. Since *abduction/adduction* and *internal/external rotation* movements are uncharacteristic for non-ball joints, no significant importance is assumed of such to solve the objectives of this thesis.

Non-Ball Joints

As explained in section 3.2, the non-ball joint *flexion* angles are computed by utilising the dot product of two vectors that represent the limbs which are connected to the examined joint. The two vectors both have their origin in the examined joint and point to another joint that represents the end of a limb. For example, to compute the right elbow *flexion*

angle, the right elbow, right wrist and right shoulder joint positions are required. The three joint positions allow to determine a vector from elbow to shoulder and another vector from elbow to wrist. Then the dot product of the two vectors is calculated after normalising them. The result is the lowest angle between the vectors. To derivate a clinically meaningful angle from that, the result in degrees is subtracted from 180. This results in degrees near zero if the arm is straight and higher degrees if the arm is flexed, which accompanies the clinical flexion angle definitions.

Ball Joints

As mentioned in section 2.1.2, Euler-Cardan angles are not used to compute the joint angles for ball angles in this work. Instead, the *flexion/extension* and *abduction/adduction* angles are determined by utilising the concept spherical coordinates.

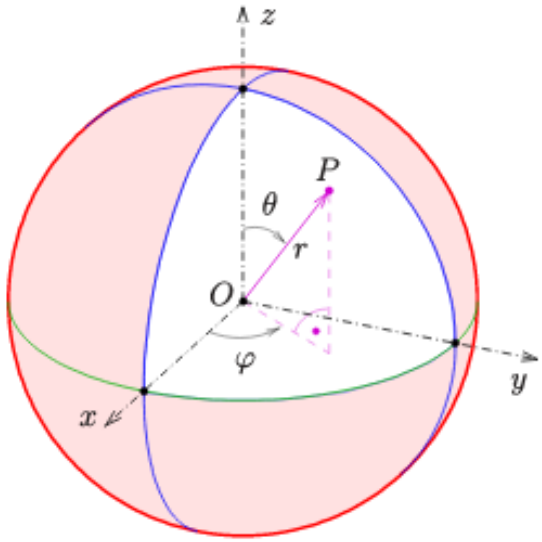

Figure 3.3: Spherical angles representation of euclidean coordinates

[20]

The method requires the position of a spatially moving joint that is connected to the rotating joint in the priorly constructed JCS, as explained in the first paragraph of section 3.2.1. This allows to compute the spherical coordinates denoted as radial distance (r), elevation angle theta (θ) and azimuth angle phi (ϕ) for a right-handed coordinate system as displayed in Figure 3.3 (see eq. 3.1, 3.2, 3.3). To transfer this principle to meaningful joint angles, the computations must be applied using other axes than displayed in the example in figure

Figure 3.3.
Firstly, the axes of the Euclidean coordinate system must be turned by 90 degrees anti-clockwise to represent the constructed JCS as the z-axis must point to the rear and the y-axis must point up. After that, the computation of ϕ is changed to represent a rotation about the x-axis, instead of the z-axis. The angle ϕ can now be interpreted as a *flexion/extension* angle, which is a rotation about the *transverse axis*. Similarly, the angle θ is not computed as being the elevation angle of the z-axis, but the JCS negative x-axis instead. If θ is subtracted from 90 degrees now, the resulting angle describes the angle between a vector pointing from origin to another point and the YZ-plane. This is suited to represent the summarised *abduction/adduction* and *horizontal abduction/adduction*.
Without a postprocessing step, the order of rotations will always be a *flexion/extension* followed by a *horizontal abduction/adduction*. But the resulting angles of an *abduction/adduction* followed by a flexion can be derived from ϕ and θ as explained in section 3.2.3.

A minor problem of this method, similar to the *Gimbal lock* of Euler-Cardan angles, is an *abduction/adduction* angle θ of 90 degrees. In that case, the examined point lies exactly on the x-axis of the JCS and the rotation angle about the x-axis can not be computed anymore. This problem is addressed by checking whether θ is of 90 degrees and if that is the case, to set ϕ to 0 before the postprocessing step.

$$r = \sqrt{x^2 + y^2 + z^2} \tag{3.1}$$

$$\theta = \arccos \frac{z}{r} \tag{3.2}$$

$$\phi = \begin{cases} \arctan\left(\frac{y}{x}\right), \text{ if } x > 0 \\ \operatorname{sgn}(y)\,\frac{\pi}{2}, \text{ if } x = 0 \\ \arctan\left(\frac{y}{x}\right) + \pi, \text{ if } x < 0 \wedge y \geq 0 \\ \arctan\left(\frac{y}{x}\right) - \pi, \text{ if } x < 0 \wedge y < 0 \end{cases} \tag{3.3}$$

3.2.3 Angle Results Postprocessing

After computing azimuth angle ϕ and elevation angle θ and reinterpreting them as a *flexion/extension* followed by an *abduction/adduction* as explained in section 3.2.2, postprocessing is needed to adjust the results for a specific use-case.

Evaluation of Exercises

One use-case is the evaluation of performed exercises, based on specified joint angles for start and end position. Assuming a target joint angle of the right shoulder should be of 180 degrees *abduction* for an arbitrary exercise. And if a performing person executes the exercise perfectly, the computed *flexion* angle ϕ will be 180 degrees and the *abduction/adduction* angle will be of 0 degrees. As it is impossible to determine the order of rotations the person really did, the most likely order will be expected. Since a 180 degrees *abduction* end position is specified in this example exercise, the results are also interpreted as if an *abduction* has been performed first.
The postprocessing step is suited to ensure meaningful angles for both possible orders of motion. It is adjustable to possible future requirements of clinicians without changing the angle computations it is based on, which are clinically meaningful angles by themselves, representing a *flexion/extension* followed by an *abduction/adduction*.

3.3 Subsequencing Motion Sequences

Identifying single iterations in motion sequences including multiple iterations of sport exercises is one of the main goals of this thesis. In order to perform this task, calculated joint angles of prioritised body parts for a specific exercise are analysed. The result is an assertion about the number of single iteration subsequences and their respective frame range within the analysed sequence. The general steps of this process are presented in the list at the bottom of this paragraph. Then the single parts of this process are explained in detail within the following chapters.

General steps to identify iterations in a sequence:

1. Retrieve prioritised body part joint angles for each frame of the sequence.
2. Identify local extrema in smoothed sequences of prioritised body part angles.

3. Find iterations by comparing the identified extrema of all prioritised body parts to each other.

3.3.1 Prioritised Body Part Angles

To identify single iterations in a sport exercise motion sequence, a list of prioritised body part angles for a particular exercise is required. It is assumed that the prioritised body parts provide the most meaningful information about an exercise sequence. As they are important indicators of whether the exercise has been performed correctly or not. It is also assumed that the prioritised body parts will be moved when performing the exercise. Meaning that corresponding joint angles will change when moving from the specified start position to the end position of that exercise. Other body parts might not be moved significantly and therefore are less suited to identify start and end keyframes of single iteration subsequences. As a consequence, only the prioritised body part angle data is used in the process of identifying single iterations.
The information which body part angles are prioritised for a particular exercise is stored within an exercise file as explained in chapter 3.1.3.

3.3.2 Identifying Local Extrema of Prioritised Joint Angles

After retrieving the raw data of prioritised body part angles for each frame of the examined sequence, the joint angle values must be altered to fit the type of movement for the corresponding exercise as explained in chapter 3.2.3.
Then, the joint angle sequences are smoothed in order to simplify the identification of meaningful local extrema, which might later represent the start, turning or end key frames for single iterations of the analysed exercise set.

Figure 3.4 serves for a better understanding of the following chapters. On top, it shows graphs of the hips and knees joint angles of a trainee that performed ten iterations of a squat exercise. The middle part of the figure shows a zoom into the first iteration of those ten iterations. The images of the trainee at the bottom correspond to the frame numbers their respective lines point to. The type of plot that can be seen at the top of Figure 3.4 will be used to help further explain the process of subsequencing motion sequences.

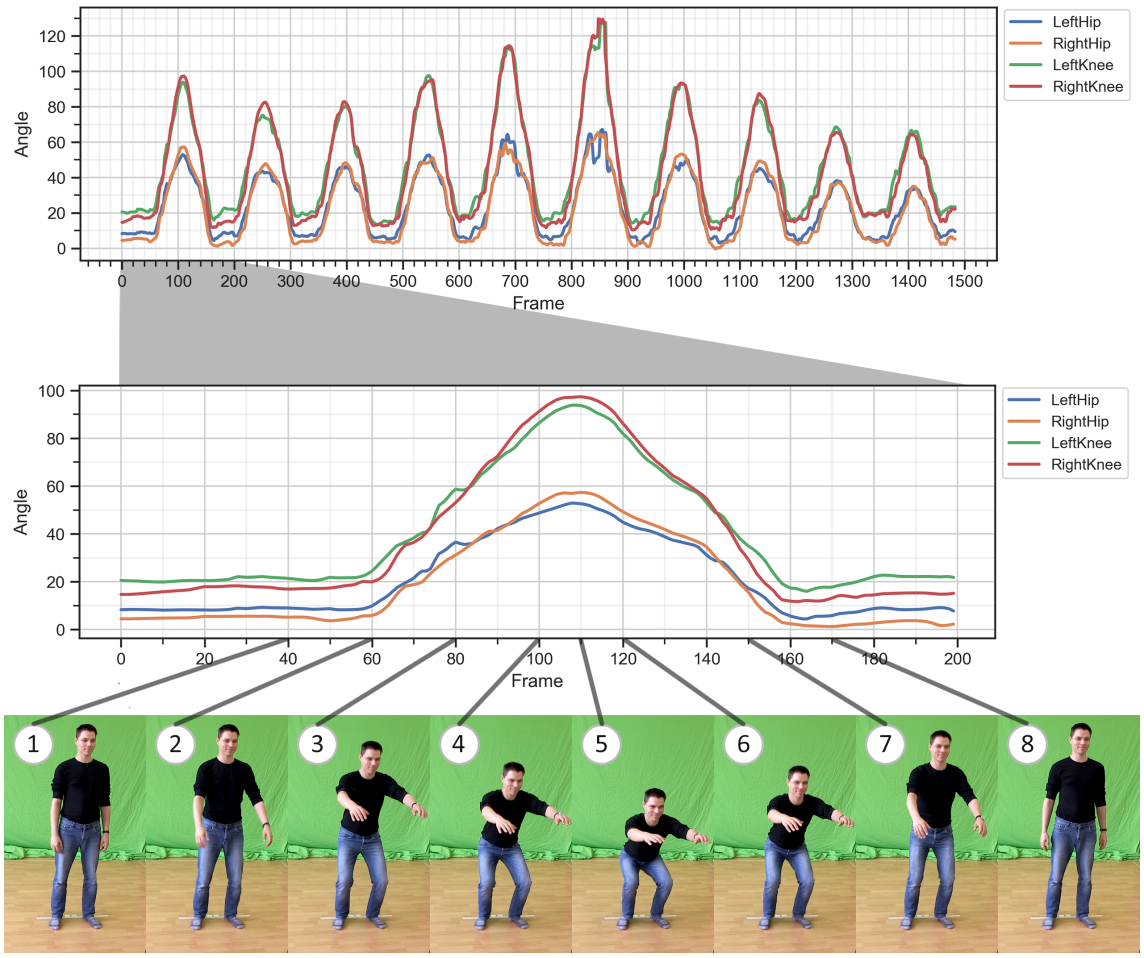

Figure 3.4: Graphs of prioritised body part angles of a squat exercise motion sequence for ten iterations (top) and another plot that zooms into the first iteration (mid). Corresponding RGB Images of the trainee while performing the exercise (bottom) including connections to the respective frames.

Smoothing Joint Angle Data

The altered joint angle data might still be subject to strong fluctuations as they are derived from the unfiltered motion tracking input data. As outliers and irregularities complicate the identification of meaningful local extrema that mark a start and end frame of an iteration, the joint angle data is smoothed by applying a Savitzky–Golay Filter (see chapter 2.2) to each sequence of body part angles. Figure 3.5 shows four graphs that map the raw joint angles on frames of the examined sequence for an exemplary execution of ten iterations of

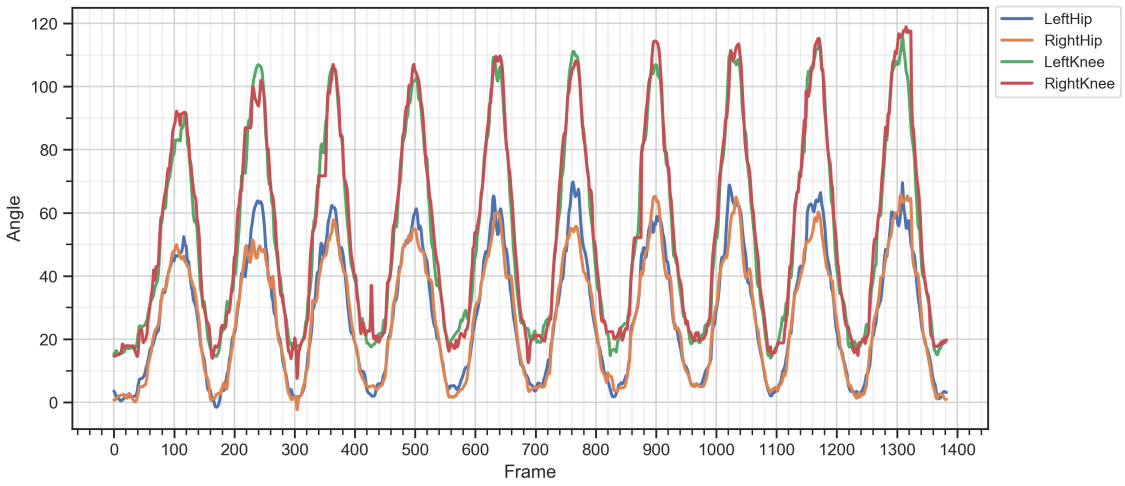

Figure 3.5: Raw prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise.

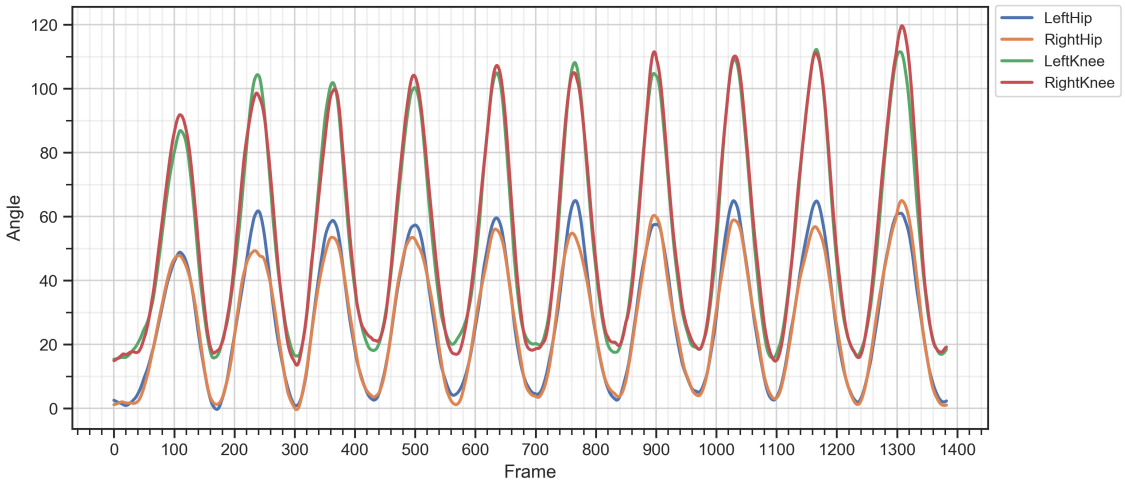

Figure 3.6: Smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise after applying a Savitzky-Golay filter.

a squat exercise. In contrast, Figure 3.6 shows the joint angles for the same sequence as in Figure 3.5 after applying a Savitzky-Golay filter, where irregularities have been removed successfully and facilitate the identification of meaningful extrema.

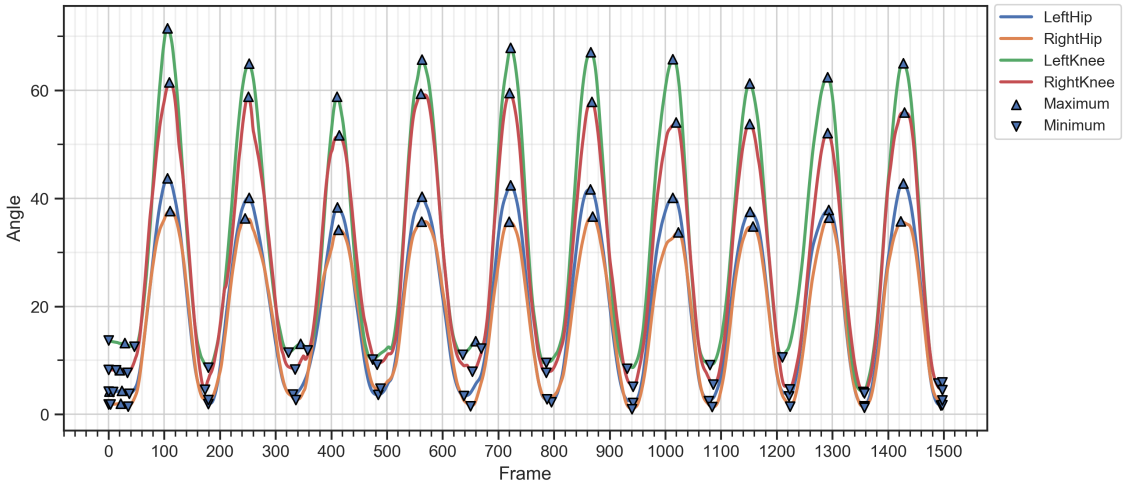

Figure 3.7: Marked minima and maxima from smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise.

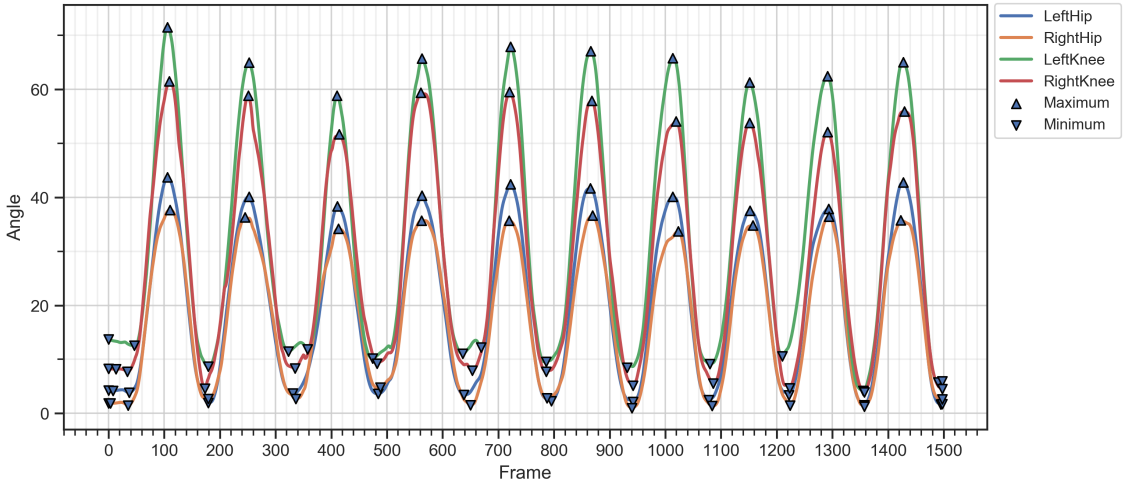

Figure 3.8: Marked extrema of smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise after removing extrema markings that are not close enough to their respective target angle.

Identifying Extrema

After the joint angles have been smoothed, minima and maxima of each joint angle sequence are identified by comparing the angle value of each frame to a group of adjacent frames. If the joint angle values of all adjacent frames are greater than the current examined value, the frame is considered to be a local minimum. If on the other hand, all values are less, it is considered to be a local maximum. Figure 3.7 shows the smoothed joint angle data of another squat exercise sequence including identified minima (arrowhead pointing down)

and maxima (arrowhead pointing up). It may be observed, that significant local extrema, which might represent start and end frames of an iteration, have been identified. But in addition to that, some unwanted extrema were found around the frames 20 to 40, frames 340 to 360 and frames 660 to 680. The identified maxima in these frame ranges are unwanted, as they only represent maximum values that are very close to their adjacent minima and not helpful in identifying single iterations.
In order to remove this kind of unwanted extrema, the joint angle values of identified local minimum and maximum frames are compared to the exercises defined target angles for start and end position. Before that, it is necessary to determine what kind of joint motion must be performed for the current exercise. If the joint motion is a *flexion* or *abduction*, the exercises' end-state target angle is greater than the start state target angle. As a consequence, the maxima of such body part joint angles shall represent frames where the angle value is as close as possible to the exercises' defined end-state angle value (see 3.2.2). If on the other hand, the joint motion is an *extension* or *adduction*, the exercises end-state target angle is less than the exercises start state target angle (see 3.2.2) and identified maxima shall represent frames where the angle value is as close as possible to the exercises defined start state angle value.
Given the information of what minima and maxima should represent, extrema are filtered by determining the distance of the angle values for extrema frames to the desired target angle value and to the unwanted target angle value. If the distance to the unwanted target is less than to the desired target, the extremum is removed. Figure 3.8 shows the same data as Figure 3.7 after applying the filter stated above. It can be observed, that the unwanted maxima in frame ranges 20 to 40, 340 to 360 and 660 to 680 observed in Figure 3.7 have been successfully removed.

3.3.3 Identifying Iterations

Based on the identified and filtered extrema of prioritised body parts, as explained in chapter 3.3.2, the following steps aim to identify start, turning and end frames of single iterations of the performed exercise set. Identified start and end frames should ideally be located at the frame, where the exercises starting position has been reached or was intended to be reached. Identified turning frames should ideally mark the location where the target position of the exercise has been reached and the reversing motion is started to reach the starting position again. One iteration is then defined as a start/end frame, followed by a

turning frame, followed by a start/end frame.
The process of deriving start, turning and end frames from extrema of smoothed body part angles is explained in the following sections.

Confirming Key Frame Candidates

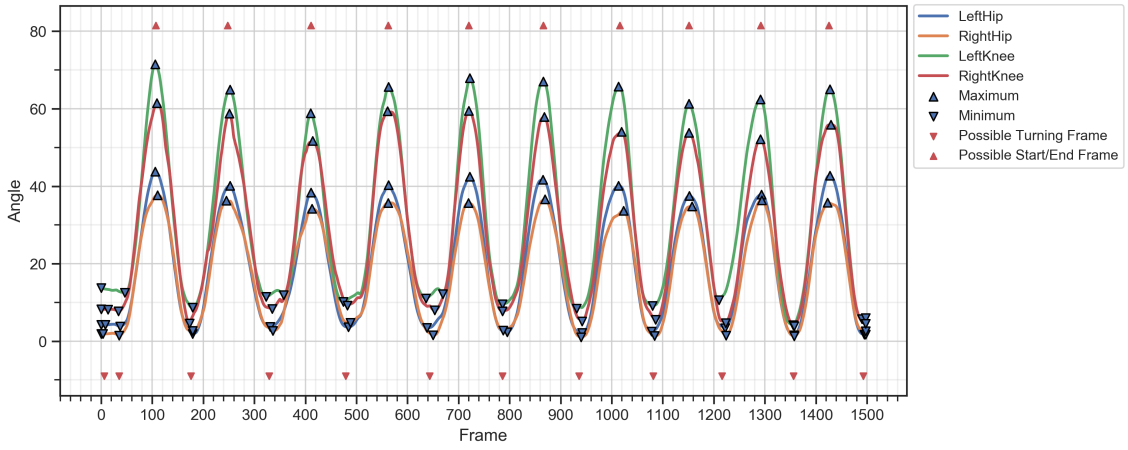

Figure 3.9: Marked extrema of smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise after removing extrema markings that are not close enough to their respective target angle. Enhanced by marked frames that represent confirmed keypoint candidates of single iterations.

Depending on the types of intended exercise motions, minima and maxima of each body part angle sequence should represent key frame candidates. Meaning either start/end frames or turning frames of an iteration, as explained in the *identifying extrema* section of chapter 3.3.2. Consequently, all of the identified extrema are either start/end frame candidates or turning frame candidates. The goal is now, to find key frame candidates, which recur in all or nearly all of the prioritised body part angle sequences within a specified frame range. Moreover, if more than one key frame candidates are present within that range, that belong to one body part angle sequence, all but the first candidate will be removed. In case a group of start/end frame candidates or turning frame candidates along all prioritised body parts is found, the arithmetic mean of their frame numbers is calculated. The result then represents a confirmed key frame candidate.
Figure 3.9 shows the same data as Figure 3.8, enriched by markings for confirmed key frames. It can be observed, that groups of single body part angle key frames have been

summed up to one confirmed key frame candidate. It can also be seen, that the redundant minima at frame 360 and 670 did not result in a second confirmed key frame candidate, but only the first candidate in that range has been accounted for.

Determine Iteration Key Frames

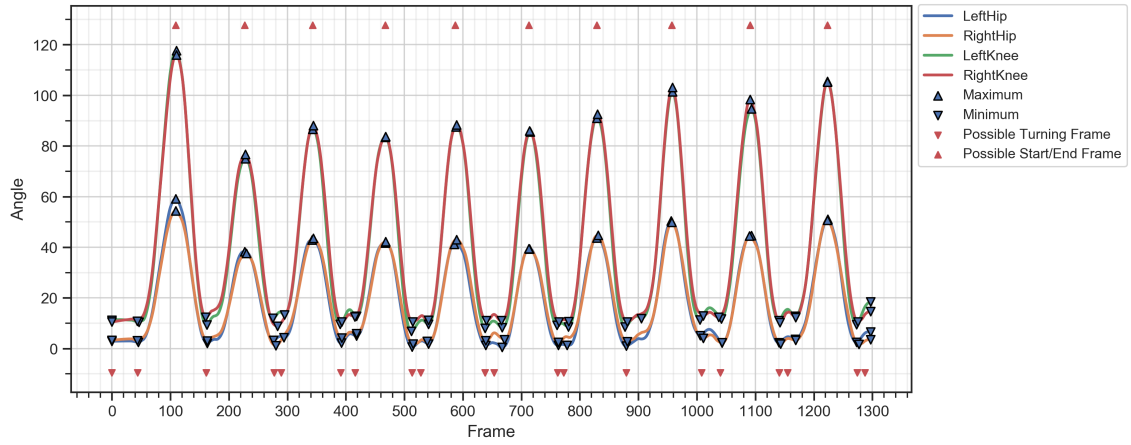

Figure 3.10: Marked extrema of smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise after removing extrema markings that are not close enough to their respective target angle. Enhanced by marked frames that represent confirmed keypoint candidates of single iterations. Serves as an example for many multiple confirmed key frame candidates in a sequence.

The last step to identify single iterations of an exercise sequence is to form groups of three key frames. The start frame, turning frame and end frame number of an iteration. To achieve that, it must be assured that only those confirmed key frame candidates are grouped, where a start frame lies before a turning frame and that turning frame lies before an end frame. It must also be respected, that iterations do not overlap each other.
In Figure 3.9 it can be observed, that two confirmed start/end frame candidates have been identified at the start of the sequence. But as there is no confirmed turning frame between the two, one of them must be removed in order to meet the order condition stated in the paragraph above. The problem might become clearer when observing Figure 3.10, which is another sequence of ten squat iterations, where the problem of multiple adjacent confirmed start/end frame candidates is very prominent.
The result of which key frames are finally being grouped up to iterations can be observed in Figure 3.11. The plot displays the same sequence as Figure 3.10 but is enriched by markings

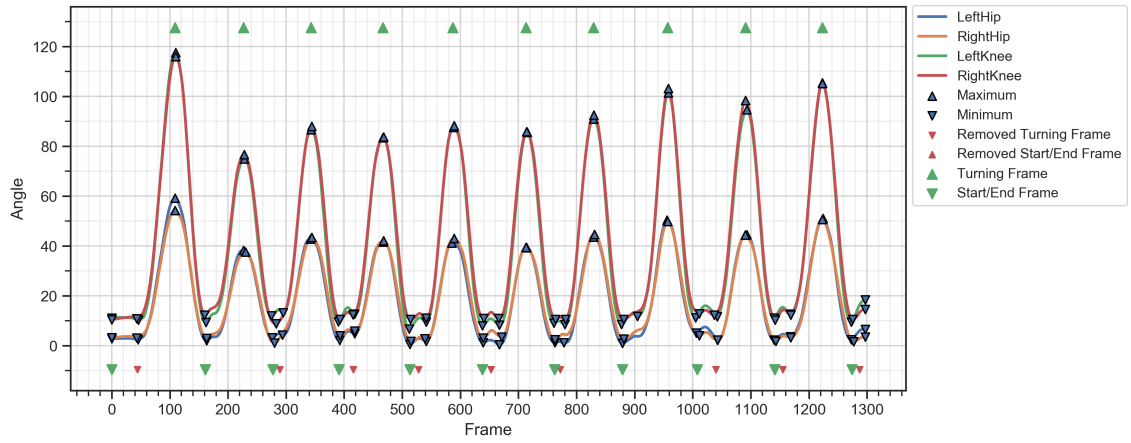

Figure 3.11: Marked extrema of smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise after removing extrema markings that are not close enough to their respective target angle. Enhanced by marked frames that represent confirmed keypoint candidates that have been removed and other marked frames that represent the final key frames of single iterations.

that represent the frames which are part of an identified iteration and the confirmed key frame candidates that have been removed, in order to meet the order and overlapping rules stated in the first paragraph of this chapter. In the displayed sequence, all iterations of the performed squat exercise have successfully been identified.

3.4 Matching Motion Sequences

In this chapter, the approach to identify an exercise by comparing it to ground-truth exercise sequences is presented. The result is an Euclidean distance for each ground-truth sequence the query sequence has been compared to. The exercise that results in the lowest distance should be the same as the performed exercise of the query sequence.

3.4.1 Determining DTW Distances of Motion Sequences

The identification of the exercise that is performed in a motion sequence is based on sequences of joint angles, which serve as transformational invariant representations of body poses. So each pose in a sequence is represented by several joint angles. The similarity between two of such poses can be determined by calculating the Euclidean distance between

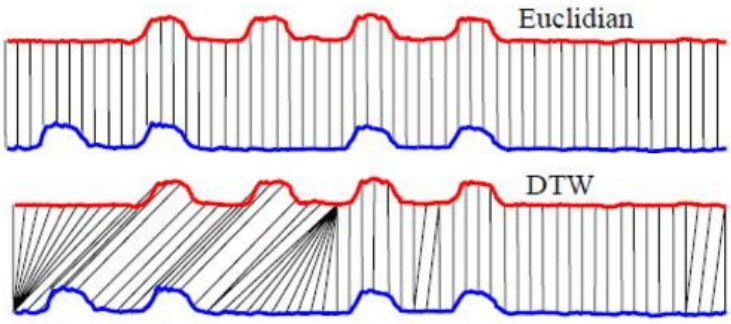

Figure 3.12: Visualised difference of Euclidean distance and dynamic time warped Euclidean distance determination between two sequences.

[46]

these pose representations.
Consequently, an Euclidean distance between all frames of a motion sequence represented by joint angles could describe the similarity of those sequences. However, this is only true if the compared sequences are of the same length. But the length of motion sequences of performed exercises depends on the trainees' speed of action. Thus, two of these motion sequences will rarely be of the same length. As a consequence, two very similar motions performed at different pace may result in compressed, stretched or time-shifted versions of each other. The Euclidean distance between such sequences would not represent a meaningful similarity measure to match the type of motion. Because the Euclidean distance is measured frame by frame and without respect to the mentioned differences in time. The upper part of Figure 3.12 shows a visualisation of an Euclidean distance measurement for two arbitrary sequences. It can be observed, that each frame of one sequence is compared to the respective frame of the other. For instance, the first peak of the blue graph is compared to the flat values of the red graph. Resulting in rather high distances caused by a time shift, which is not desired.
However, the Euclidean distance represents a meaningful similarity in order to compare two motion sequences that are ideally aligned with each other. For this purpose, the DTW algorithm is perfectly suited. As explained in chapter 2.3.1, the DTW algorithm aligns two motion sequences to each other by matching those indices that result in the lowest possible distance. Resulting in a lowest possible total Euclidean distance of the compared time series, which is the optimal representation of their similarity to each other. The bottom part of Figure 3.12 shows a visualisation of the distance measurement and alignment performed by the DTW algorithm. The black lines show which values of both sequences are compared

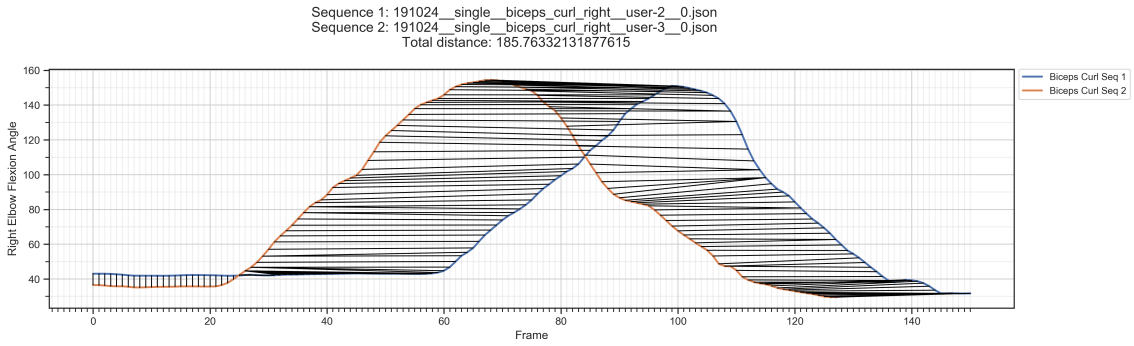

Figure 3.13: Two left elbow angle sequences of two biceps curl exercise motion sequences and the calculated DTW path for them.

to each other as they result in the lowest distance to each other.

Transferred to the exercise matching task of this thesis, the DTW algorithm is perfectly suited to compare one single iteration exercise motion sequence to other ground-truth exercise sequences. The lowest resulting total DTW distance of all comparisons including all body part angles is then considered a match. Figure 3.13 visualises the DTW alignment for the left elbow angle of two tracked motion sequences of a biceps curl exercise. It can be observed, that the sequences primarily differ in a time shift caused by different inactivity times before the arm has been flexed. As the DTW algorithm aligned the most similar values of each sequence, indicated by the black lines, that timeshift does not affect the resulting distance greatly. For illustration purposes, Figure 3.13 only shows the right elbow angles for both sequences. But it is important to note, that the actual matching involves all body part angles.

3.5 Rating Execution of Exercises

In order to rate executed exercises, two different approaches will be presented. One approach is based on a comparison between predefined target joint angles for exercises and determined joint angles of a trainee while performing these exercises. The other approach is based on the matching method used in chapter 3.4 and derives execution quality ratings from the similarities between joint angle sequences of a query sequence and ground-truth sequences.

3.5.1 Comparing Joint Angles

This approach to determine the quality of executed exercises directly utilises the calculated joint angles as explained in chapter 3.2. Aiming to deliver single frame feedback for all joint angles, whether the respective target angles are currently undercut, exceeded or reached.

Intended exercise executions are defined by clinical human joint angles of a start and end position for a particular exercise. This circumstance allows to use the clinical meaningful angles that are calculated from the 3-D tracking positions of a trainee in order to compare them to those predefined target angles. Unlike in the subsequencing and matching approaches, the calculated joint angles are not only used as input data for the procedure but are directly compared to the exercises target angles. To be able to do that, it is important that the calculated joint angles are clinically meaningful and not just of mathematical nature.
To be able to compare the joint angles of the performing trainee to the target angles of the exercise, the desired target state for each frame of the examined sequence must be known. At the beginning of the process, the target state is always the *end state*, as it is assumed that the exercise motion sequence that is about to be examined starts in the desired starting position. The identification of the turning point, the frame at which the target state changes to the *start state*, is determined by the subsequencing process explained in chapter 3.3.3. Consequently, the calculated joint angles of all following frames after that turning point must be compared to the desired target joint angles of the predefined angles for the *start state*.
The resulting feedback of the comparison consists of a current result state for each of the joint angles. The three result states are *undercut*, *in range* and *exceeded*. The *undercut* state means, that the predefined target angle for this particular joint has not been reached in this particular frame. The *in range* state means, that the target angle is reached right now. And the *exceeded* state means, that the target joint angle has already been exceeded.
It is intended, that the returned feedback states for each joint angle allow other software to generate high-quality notifications for their potential users. Being able to address specific body parts and timeframes to give users feedback about their performance right after every execution of an exercise.

3.5.2 Deriving Quality Ratings from Similarities

This second approach to rate a trainees exercise performance differs greatly from the method described in chapter 3.5.1. Instead of comparing calculated angles of a trainee to predefined exercise target angles, this method utilises the distance-based matching process described in chapter 3.4. In this approach, the similarity between the joint angle sequences of a query sequence and a ground-truth sequence of an exercise is used to derive assumptions about the quality of a trainees' exercise performance.

In chapter 3.4 the DTW Euclidean distance is used to identify types of exercises by comparing them to labeled sequences of different exercises in form of joint angle sequences. In that use-case, the labeled sequence that has the lowest resulting distance to the query sequence is considered to be the performed exercise in the query sequence.
In this performance quality determination use-case, the DTW distance can be reinterpreted. For that, a ground-truth or gold standard sequence is defined, which reflects the perfect execution of a particular exercise. The DTW distance of a query sequence to this ground-truth sequence then describes the difference of those to each other. Respectively, the distance describes the deviation from the ground-truth, the perfect execution. So the smaller the distance between query sequence and ground-truth, the better the execution of the exercise. In order to not only being able to give feedback about the overall exercise performance quality but also mentioning specific body parts, the distance is calculated for each body part angle sequence respectively. This allows to give more specific feedback with regard to particular body parts that might differ more from the ground-truth than others.
The resulting quality rating of this approach probably is not suited to absolutely determine whether the execution was *still okay* or *wrong*. As slight differences in motion might sometimes make the difference of a good and an injurious performance of an exercise. But based on the similarity determination, it is still possible to notify users about specific body parts they should give attention to. In a use-case as described in [41], where physicians record personalised ground-truth sequences with rehabilitation patients, this kind of feedback still can be of high value. Especially if the users are able to watch a video of the ground-truth sequence in order to compare their performance to that.

3.5.3 Discussion of Exercise Rating Approaches

The presented approaches to rate exercise performances are very different from each other. (1) The first approach directly compares predefined exercise target joint angles and joint angles of a performing trainee to each other in order to derive concrete feedback about each body part angle on a one frame basis. The outcome is a set of result states for each body part angle, whether it is *undercut*, *in target range* or *exceeded*. (2) The other approach derives the quality of execution from the distance between joint angle sequences of each body part angle and respective sequences of a gold-standard.
On the one hand, the first approach (1) is suited to give more accurate feedback about whether a motion is *okay* or *wrong* even if the result depends on minor differences in motion. On the other hand, that method can only be applied to predefined exercises as described in chapter 3.1.3. Therefore, slight changes in the desired motions may require customisation of various target angles in the exercise definition files. Further, this angle comparison based approach requires clinical meaningful angles to be able to compare medical target angles defined by physicians, to the resulting angles of a trainees' or patients' motions.
The second approach (2), which is based on the similarity of two motion sequences, is not as suited as (1) to give detailed feedback about whether a motion is injurious or not. The result is more of a hint, that mentions body parts that should receive special attention because their motion was different from the ground-truth sequence. Although the results of this method are less accurate, it has some other advantages over (1). It does not require predefined exercise files and target angles, but only a ground-truth sequence to evaluate against. Therefore, the number of exercises and their individual complexity is not limited, which makes the approach very generic and possibly transferable to other motions than sport exercises. Additionally, this approach is not only comparing against target angles of a *start pose* or *end pose* but compares the whole motion instead. Moreover, this approach does not necessarily require *clinically meaningful* joint angles, but could also use other angle representations or possibly even completely different input sequences.

In conclusion, the first approach (1) gives more accurate feedback but is limited to predefined exercises and specific key points. It is arguably harder to personalise exercises or create complex exercises that might consist of concatenated exercises. The second approach

(2), however, produces less accurate feedback. Nevertheless, it does not necessarily require *clinically meaningful* joint angles and predefined exercises. Furthermore, it allows to evaluate complex motions and considers the whole motion instead of specific keypoints.

3.6 Summary

This chapter described the approaches and methods to handle the main objectives of this thesis. The calculation of clinically meaningful joint angles, and their usage across the following chapters have been presented. Further, approaches to retrieve single iteration subsequences from exercise sequences and to match motion sequences based on the DTW algorithm have been presented. Followed by two different approaches in order to rate executions of sport exercises, including a conclusive discussion of both approaches.
The presented approaches will be evaluated in chapter 5 using a self-created dataset, which is also a contribution of this thesis. The next chapter briefly presents selected implementations of the joint angle calculations and the subsequencing process.

4 Implementation

The Python 3.7 prototype has been implemented in order to demonstrate the feasibility of the developed methods, which tackle the main objectives of this thesis. In this chapter, selected parts from the implemented human motion analysis prototype will be presented. The first part deals with the implementation of the joint angle calculations explained in chapter 3.2. The second part presents the subsequencing procedure introduced in 3.3. The prototypes' source code can be found on the attached data storage under *source_code.zip*.

4.1 Joint Angle Calculation

Whenever a sequence object (see 3.1.2) is initialised, joint angles for 3-D positional body joint data is calculated. For non-ball joints, the joint *flexion/extension* angles can directly be calculated. For ball joints, a JCS must be constructed before *flexion/extension* and *abduction/adduction* angles can be determined.

4.1.1 JCS Construction

For every frame and ball joint of a sequence, the *align_coordinates_to* function (see Listing 4.1) is called in order to construct a Joint Coordinate System (JCS), which is needed to determine the respective joint angles. The function expects four arguments to perform this task. Three specific body part indices and the 3-D joint positions for one specific frame. The body part indices *origin_bp_idx*, *x_direction_bp_idx* and *y_direction_bp_idx* determine which body part positions are used to construct the axes of the JCS.
Then, the JCS axes *vx*, *vy* and *vz* are calculated as described in chapter 3.2.1 in the *Finding JCS Axes* section. The corresponding code part for the procedure can be observed in Listing 4.1 from line 8 to line 18.
After the direction of the axes have been calculated, the translation matrix T and rotation

matrices *Rx* and *Ry* are determined as explained in chapter Section 3.2.1 in the *Transforming Joint Positions from GCS to JCS* section. The corresponding code part in Listing 4.1 can be seen from line 21 to 30. At line 21 the translation matrix is obtained to move the JCS origin into the origin joints position. At line 23 to 25, the x-axis rotation is determined by constructing a rotation axis *x_rot* that is perpendicular to the calculated JCS axis *vx* and the target direction vector *[1,0,0]*. Then the angle between *vx* and *[1,0,0]* is calculated in order to construct the rotation matrix *Rx* at line 25, which represents the rotation from *vx* to *[1,0,0]* about the *x_rot* axis. After that, the third and last transformation is determined from line 27 to 30. At line 29 the angle is calculated as for *Rx* but this time, after the rotation *Rx* has been applied to *vy* at line 27.
Further, the complete transformation matrix *M* is determined at line 33, representing the dot product of *T*, *Rx* and *Ry*. Lastly, the transformation *M* is applied to all positions of the *positions* argument at line 35. As a result, the transformed positions are returned.

```python
def align_coordinates_to(origin_bp_idx: int, x_direction_bp_idx: int, y_direction_bp_idx: int,
 positions: np.ndarray):
    # Positions of given orientation joints in GCS
    origin = positions[origin_bp_idx]
    x_direction_bp_pos = positions[x_direction_bp_idx]
    y_direction_bp_pos = positions[y_direction_bp_idx]

    # New X-Axis from origin to x_direction
    vx = x_direction_bp_pos - origin
    if vx[0] < 0:
        vx = -vx
    # New Z-Axis is perpendicular to the origin-y_direction vector and vx
    vz = get_perpendicular_vector((y_direction_bp_pos - origin), vx)
    if vz[2] < 0:
        vz = -vz
    # New Y-Axis is perpendicular to new X-Axis and Z-Axis
    vy = get_perpendicular_vector(vx, vz)
    if vy[1] < 0:
        vy = -vy

    # Construct translation Matrix to move given origin to zero-position
    T = translation_matrix_4x4(np.array([0, 0, 0])-origin)
    # Construct rotation matrix for X-Alignment to rotate about x_rot_axis for the angle theta
    x_rot_axis = get_perpendicular_vector(vx, np.array([1, 0, 0]))
    theta_x = get_angle(vx, np.array([1, 0, 0]))
    Rx = rotation_matrix_4x4(x_rot_axis, theta_x)
    # Use new X-Axis axis for y rotation and Rotate Y-direction vector to get rotation angle
for Y-Alignment
    y_rot_axis = vx
    vy_rx = np.matmul(Rx, np.append(vy, 1))[:3]
    theta_y = get_angle(vy_rx, np.array([0, 1, 0]))
    Ry = rotation_matrix_4x4(norm(y_rot_axis), theta_y)
    # Transform all positions
    transformed_positions = []
    M = np.matmul(T, Rx, Ry)
    for pos in positions:
        pos = np.matmul(M, np.append(pos, 1))[:3]
        transformed_positions.append(pos)

    return transformed_positions
```

Listing 4.1: JCS axes determination and transformation from hma.movement_analysis.transformations.py

4.1.2 Angle Calculation

The *calc_angles_shoulder_left* function calculates the *flexion/extension* and *abduction/adduction* angles of the left shoulder joint for every frame of a sequence. The function can be observed in Listing 4.2. After the original joint positions have been transformed by the *align_coordinates_to* function at line 7, the *x*, *y* and *z* coordinates of the spatially influenced joint, in this case, the left elbow, are obtained. Further, the spherical coordinates *r* (line 13), *theta* (line 17) and *phi* (line 28) are calculated to determine clinical meaningful angles as explained in the *Ball Joints* section of chapter 3.2.2. The elevation angle *theta* represents the *abduction/adduction* angle and must be subtracted from 90.0 degrees (line 18) to equal zero in the anatomical position. Similarly, the azimuth angle *phi*, which represents the flexion as a rotation about the x-axis, must be increased by 90.0 (line 29). Moreover, 360.0 degrees must be subtracted from *phi* if *phi* is greater than 180.0 degrees (line 31-32) to move its range from 0 to 360.0 degrees to -180 to 180 degrees as *extension* angles are represented by negative *phi* values. As *phi* is not defined if the elbow lies directly on the x-axis, *phi* is set to zero whenever the angle of the x-axis and the vector from the JCS origin to the left elbow vector is zero (line 22-24). The result is then stored in a predefined numpy array *angles* (line 4) at the positions for the current frame and the respective angle type (line 35-36). When all angles of the respective body joint have been calculated for all frames, the *angles* numpy array is finally returned (line 38).

```python
def calc_angles_shoulder_left(positions: list, shoulder_left_idx: int, shoulder_right_idx: int, torso_idx: int, elbow_left_idx: int) -> np.ndarray:
    n_frames = len(positions)
    n_angle_types = len(AngleTypes)
    angles = np.zeros((n_frames, n_angle_types))
    for frame in range(0, n_frames):
        # Move coordinate system to left Shoulder
        left_shoulder_aligned_positions = transformations.align_coordinates_to(shoulder_left_idx, shoulder_right_idx, torso_idx, positions[frame])

        ex = left_shoulder_aligned_positions[elbow_left_idx][0]
        ey = left_shoulder_aligned_positions[elbow_left_idx][1]
        ez = left_shoulder_aligned_positions[elbow_left_idx][2]
        # Convert to spherical coordinates
        er = math.sqrt(ex**2 + ey**2 + ez**2)

        # Theta is the angle of the Shoulder-Elbow Vector to the YZ-Plane
        # Represents an abduction/adduction
        theta = math.degrees(math.acos(ex/er))
        theta = 90.0 - theta
        abduction_adduction = theta

        # Phi is arbitrary when point is on rotation axis, so we set it to zero in that case
        elbow_xaxis_angle = transformations.get_angle(np.array([1, 0, 0]), left_shoulder_aligned_positions[elbow_left_idx])
        if elbow_xaxis_angle == 0.0 or elbow_xaxis_angle == math.pi:
            phi = 0
        else:
            # Phi is the angle of the Elbow around the X-Axis (Down = 0)
            # Represents flexion/extension angle
            phi = math.degrees(math.atan2(ey, -ez))
            phi += 90.0
            # An Extension should be represented in a negative angle
            if phi > 180.0:
                phi -= 360.0
        flexion_extension = phi

        angles[frame][AngleTypes.FLEX_EX.value] = flexion_extension
        angles[frame][AngleTypes.AB_AD.value] = abduction_adduction

    return angles
```

Listing 4.2: Left shoulder angle calculation from hma.movement_analysis.angle_calculations.py

4.2 Subsequencing Motion Sequences

The subsequencing algorithm is explained in detail in chapter 3.3. The subsequencing procedure is part of the *ExerciseEvaluator* class. The class receives an *Exercise Object* and a *Sequence Object* as initial parameters as introduced in chapter 3.1. On initialisation of the *ExerciseEvaluator* class the *target angles* and *prioritised angles* are parsed from the given *Exercise Object* instance and also stored as an attribute. As the last step, the calculated joint angles of the sequence are processed further with respect to the given *Exercise Object* as described in 3.2.3.
The *ExerciseEvaluator* class also offers setter functions to change the *Exercise* and *Sequence* on runtime, which again triggers the parsing of *target angles* and *prioritised angles* as well as the postprocessing of the angles.
The following chapters present selected code from the subsequencing implementation.

Identifying Extrema

As a first step of the algorithm, the *Sequence Objects* joint angle data of the *Exercise Objects* prioritised angles is smoothed using a *Savitzky Golay Filter* as explained in the *Smoothing Joint Angle Data* section of chapter 3.3.2. In the displayed example of Listing 4.3 the *Savitzky Golay Filter* function from the *scipy.signal* package is applied at line 9 using a window size of 51 and the order 3. These window size and order parameters produced good results, but are further examined when evaluating the procedure.
After the joint angle data has been smoothed, the *argrelextrema* function of the *scipy.signal* package is used to identify a list of minima and maxima in the smoothed data (Listing 4.3 line 15-16). The order parameter of the *argrelextrema* function determines the number of values on both sides of a currently examined value that must fulfil the given condition. In this case, the conditions are *np.greater* and *np.less* and the order is set to 10. So a maximum is identified whenever the examined data point is greater than the prior ten values and next ten values. Similarly, a minimum is identified if the examined value is less than its neighbours. Same as the *Savitzky Golay Filter* parameters, the best value for the *argrelextrema* functions order parameter must be further examined during the evaluation of the subsequencing procedure.
Additionally, a minimum or maximum, depending on whether the exercises' target end value is greater than the target start value, is added to the first and last frame of the sequence.

But only if the angle values are closer to the respective exercises target angles than the defined tolerance. (Listing 4.3 line 20-29). Otherwise, if these two extrema are not added, the first and last frames of a sequence will only rarely be identified as extrema, because the *argrelextrema* functions' order parameter requires a constant fulfilment of its comparator, which is rarely the case before the first and after the last maximum of a sequence. Therefore, it is assumed that the first frame of the sequence is a starting frame and the last frame of the sequence is an end frame.
Figure 3.7 represents the current stage of the algorithm. It shows smoothed prioritised angle values, identified extrema from the *scipy.signal.argrelextrema* function and added extrema for the first and last frame.

```
    ...
        for prio_idx, (body_part_idx, angle_type) in enumerate(self.prio_angles):

            # Get calculated angles of a specific type for a specific body part for all frames
            angles = seq.joint_angles[:, body_part_idx, angle_type.value]

            # Apply a Savitzky-Golay Filter to get a list of smoothed angles.
            savgol_window = 51
            angles_savgol = savgol_filter(angles, savgol_window, 3, mode="nearest")
            angles_savgol_all_bps[prio_idx] = angles_savgol
            angles_all_bps[prio_idx] = angles
            angles_legend[prio_idx] = body_part_idx

            # Find Minima and Maxima of angles after applying a Savitzky-Golay Filter
            maxima = argrelextrema(angles_savgol, np.greater, order=10)[0]
            minima = argrelextrema(angles_savgol, np.less, order=10)[0]

            # Get Exercise targets for the current angle type
            ex_targets = self.target_angles[body_part_idx][angle_type.value]
            # Check if Exercise targets of target state END are greater than targets of START
            # We need this information to identify whether local MAXIMA or MINIMA represent
start/end of a subsequence
            target_end_greater_start = min(ex_targets[AngleTargetStates.END.value]) > min(
ex_targets[AngleTargetStates.START.value])
            # Add minimum to first and last frame if start_frame_min_dist/end_frame_min_dist
            # param value is not less than the actual distance to the target angle
            target_distance_tolerance = abs(int(mean(ex_targets[AngleTargetStates.END.value])
- mean(ex_targets[AngleTargetStates.START.value])))
            angles_savgol = np.array(angles_savgol)
```

```
27            target_start_range = self.target_angles[body_part_idx][angle_type.value][
AngleTargetStates.START.value]
28            if (min(target_start_range) - target_distance_tolerance < angles_savgol[0] < max(
target_start_range) + target_distance_tolerance):
29                minima = np.insert(minima, 0, 0)
30            if (min(target_start_range) - target_distance_tolerance < angles_savgol[-1] < max(
target_start_range) + target_distance_tolerance):
31                minima = np.append(minima, len(angles_savgol)-1)
```

Listing 4.3: Smoothing and identification of initial minima and maxima of the find_iteration_keypoints function from hma.movement_analysis.exercise_evaluator.py

Filtering Extrema

Some of the extrema identified in the stage described in the last section must be filtered because they do not represent key points of an iteration. Listing 4.4 shows the first filter, which removes extrema whose respective angle values are closer to the opposite target value than the desired target value. The defined lambda function filter in Listing 4.4 line 4 returns *True* for all values whose distance to the minimum value of the exercises' start target range is greater than that to the minimum value of the end target range. Depending on whether the target end values are greater than the target start values, the filter is applied to the minima or maxima of the current body part. Consequently, the inversed filter is applied to the other extrema. In Listing 4.4 line 5-10 the filtering can be observed. The defined lambda filter is used as a conditional index of the extrema numpy arrays, removing all values that result in a returned *False* value of the filter or inversed filter.

```
1     ...
2         # Check if distance to Exercise START target angle is greater than Exercise END target
 angle
3         # in order to remove falsy maxima/minima
4         def _dist_filter(x): return abs(angles_savgol[x] - min(ex_targets[AngleTargetStates.
START.value])) > abs(angles_savgol[x] - min(ex_targets[AngleTargetStates.END.value]))
5         if target_end_greater_start:
6             maxima = maxima[_dist_filter(maxima)]
7             minima = minima[np.invert(_dist_filter(minima))]
```

```
        else:
            maxima = maxima[np.invert(_dist_filter(maxima))]
            minima = minima[_dist_filter(minima)]
        ...
```

Listing 4.4: Target distance filter for extrema of the find_iteration_keypoints function from hma.movement_analysis.exercise_evaluator.py

After the distance filtering, the remaining extrema are applied to a binary matrix for starting and turning frames. The rows of these binary matrices represent a body part and the columns represent a frame of the examined sequence. Corresponding extrema are represented by a 1, all other frames are represented by a 0 in these matrices.
After extrema for all prioritised body parts have been identified and filtered by the described distance filter, the remaining extrema for these body parts must be confirmed by checking whether all or most of the body part sequences have extrema in a specified window range. Therefore, the *_confirm_extrema* function shown in Listing 4.5 is used. The function receives three parameters. An *extrema matrix* as described in the prior paragraph. A window size, that defines how many frames are grouped up when checking the number of found extrema. And a *confirm extrema threshold*, that determines how many body parts must have an extremum in the window range.
The function moves the defined window over the whole matrix, checking whether the sum of values in a window row is at least 1.0 as this means that there is at least one extremum of a body part (Listing 4.5 line 9-11). If at least as many rows contain an extremum in the current window as the defined *confirm_extrema_thres* value, an extremum is confirmed (Listing 4.5 line 13). Further, the mean frame of the first and last extremum in the window is added to the *confirmed_extrema* list and the *extrema matrix* values in that window are set to 0.0 after that.

```
  def _confirm_extrema(self, extrema_matrix: np.ndarray, w_size: int, confirm_extrema_thresh:
int) -> np.ndarray:
        confirmed_extrema = []
        for column_idx in range(0, extrema_matrix.shape[1]):
            w_start = column_idx
            w_end = w_start + w_size
            window = extrema_matrix[:, w_start:w_end]

            # Check how many window rows include at least one extremum
            w_row_extrema = 0
            for w_row in window:
```

```
11                  w_row_extrema += 1 if (np.sum(w_row) >= 1.0) else 0
12              # If enough window rows include an extremum
13              if w_row_extrema >= confirm_extrema_thresh:
14                  # Find first extremum in window
15                  extrema_in_window = np.argwhere(window > 0)
16                  first_window_extremum = np.min(extrema_in_window[:, 1])
17                  last_window_extremum = np.max(extrema_in_window[:, 1])
18                  # Use average index between first and last extremum in window as confirmed
    extremum index
19                  confirmed_extremum = int((first_window_extremum + last_window_extremum)/2) +
    w_start
20                  confirmed_extrema.append(confirmed_extremum)
21                  # Remove all 1.0 values from the current window slice of the extrema_matrix
22                  extrema_matrix[:, w_start:w_end] = np.zeros(window.shape)
23          return np.array(confirmed_extrema)
```

Listing 4.5: The _confirm_extrema function from hma.movement_analysis.exercise_evaluator.py

Identifying Iterations

The last step of the subsequencing algorithm is described in the *Determine Iteration Key Frames* section of chapter 3.3.3. In this stage, the identified start/end and turning points are grouped to three key frame indices that represent a subsequence containing one iteration of the examined exercise sequence. Therefore, the *_confirm_iterations* function that is displayed in Listing 4.6 ensures that the start, turning and end frame indices meet the condition $start\_index < turning\_index < end\_index$.

The *_confirm_iterations* function receives two numpy arrays as arguments. The *confirmed_start_frames* and *confirmed_turning_frames*, which contain the sequences frame indices of the confirmed key frames identified in the prior section of this chapter. The first condition in Listing 4.6 line 7 checks whether the end frame index from the last iterations' *last_end_frame* is greater than the current start index and continues with the start frame index if the condition is met. This prevents overlapping iterations. As a next step only those confirmed turning frame indices are kept, which are greater than the current starting frame (Listing 4.6 line 11). Consequently, the *confirmed_turning_frames* numpy array only contains turning points which lie after the current start frame. If the *confirmed_turning_frames* array is empty after this, there are no more iterations to identify,

and the loop is exited (Listing 4.6 line 14-15) to return the current list of iterations. However, if at least one confirmed turning point is left the same procedure is repeated in Listing 4.6 line 18-21. Thus, the first turning frame index after the start frame index is compared to all confirmed start/end frames and all start/end frames that lie before that turning frame are removed. If the resulting *confirmed_end_frames* array is empty, there are no suitable end frames left and the loop is exited. However, if there is at least one end frame left, the first end frame in *confirmed_end_frames* is kept for the condition at line 7 and an iteration is appended to the list of iterations, which is returned after the loop is finished (Listing 4.6 line 24-27).

```
1       def _confirm_iterations(self, confirmed_start_frames: np.ndarray,
    confirmed_turning_frames: np.ndarray) -> list:
2           iterations = []
3           last_end_frame = None
4           # We Don't need to iterate over the last element because this can't be the start of a
    iteration anymore.
5           for sf_idx in range(0, len(confirmed_start_frames)-1):
6               # Ensure that next start frame is greater equal than last end frame
7               if last_end_frame is not None and last_end_frame > confirmed_start_frames[sf_idx]:
8                   continue
9               # Only keep turning frames that occur later than the current start frame
10              start_frame = confirmed_start_frames[sf_idx]
11              confirmed_turning_frames = confirmed_turning_frames[start_frame <
    confirmed_turning_frames]
12              # If there is no element in confirmed_turning_frames that is greater the current
    start_frame,
13              # we can exit the loop since following start_frames will be even higher
14              if len(confirmed_turning_frames) == 0:
15                  break
16              # If there are still elements left in confirmed_turning_frames, take the smallest
    one as our turning point
17              else:
18                  turning_frame = confirmed_turning_frames[0]
19                  confirmed_end_frames = confirmed_start_frames[turning_frame <
    confirmed_start_frames]
20                  if len(confirmed_end_frames) == 0:
21                      break
22                  # If there are still elements left in confirmed_end_frames, take the smallest
    one as our end point
23                  else:
24                      end_frame = confirmed_end_frames[0]
```

```
25                      last_end_frame = end_frame
26                      iterations.append(np.array([start_frame, turning_frame, end_frame]))
27          return np.array(iterations)
```

Listing 4.6: The _confirm_iterations function from hma.movement_analysis.exercise_evaluator.py

4.3 Conclusion

The presented implementations show how the concepts described in chapter 3.2 and 3.3 can be realised in python 3.7. However, these implementations only represent one working solution to demonstrate the explained concepts and may be further optimised in terms of structure, performance, robustness, scalability and flexibility. The next chapter deals with the evaluation and presents the results of this work.

5 Evaluation

In this chapter, the developed concepts and results regarding the main objectives of this thesis are presented and evaluated. The first section states the developed results, which are then observed and evaluated in the following section to check the quality of these solutions and identify problems that may occur during the process. Finally, the evaluations results are concluded.

5.1 Results

The results of this work are the methods and the prototyped implementations of these methods as well as a sport exercise motion tracking dataset to solve the main objectives of this thesis. These results allow to analyse sport exercise motion sequences in real world scenarios. A video that briefly states the results can be found on the attached data storage under *results_summary_video.mp4*.

This includes counting and identifying single iteration subsequences from motion sequences that consist of multiple repititons of a sport exercise. The subsequencing result is displayed in Figure 5.1, showing a trainee performing ten repetitions of a squat exercise to the left and identified squat iterations to the top right. Additionally, the smoothed knees and hips joint angle graphs are displayed in the bottom of the figure, including markings for joint angle extrema and identified key frames.

Consequently, the identified subsequences are then used to rate the trainees exercise performance respectively. To do so, the calculated joint angles of the motion sequence are compared to predefined target angles of the performed exercise with respect to the current target state. The initial target state is always the *END* state and represents the pose after which the motion should be inverted to reach the start pose again. The target pose switches from *END* to *START* after the *turning frame* of the examined sequence has been reached. So the turning frame, determined by the subsequencing procedure, which also

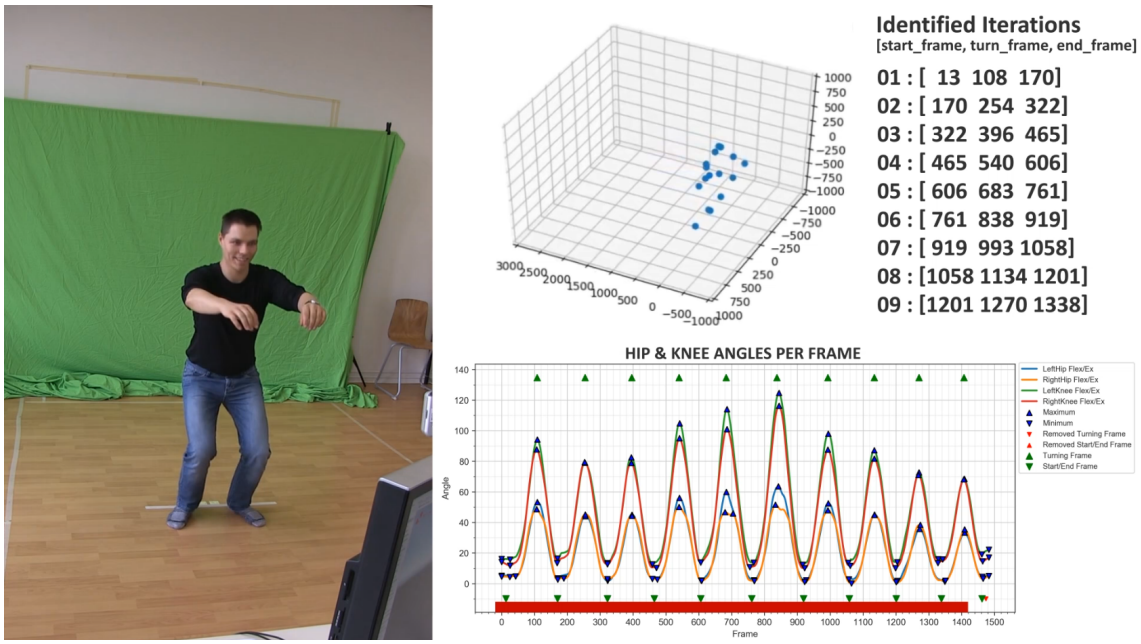

Figure 5.1: Subsequencing result presentation showing a trainee performing squats (left), the 3-D skeleton representation of the trainees motion (top-center), joint angle graphs with marked extrema and iteration key frames (bottom) and identified squat iterations including their key frame indices (top-right)

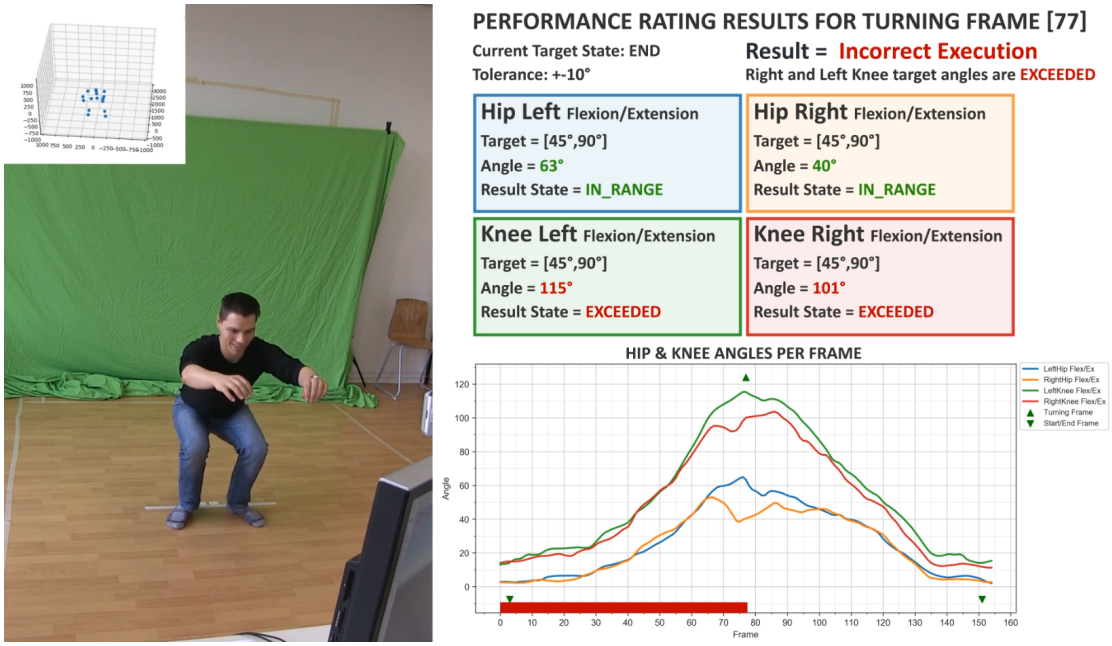

Figure 5.2: Trainees' squat exercise performance rating result determined for the turning frame of the motion. Shows the performing trainee and his tracked skeleton (left) as well as predefined exercise parameters and rating results (right).

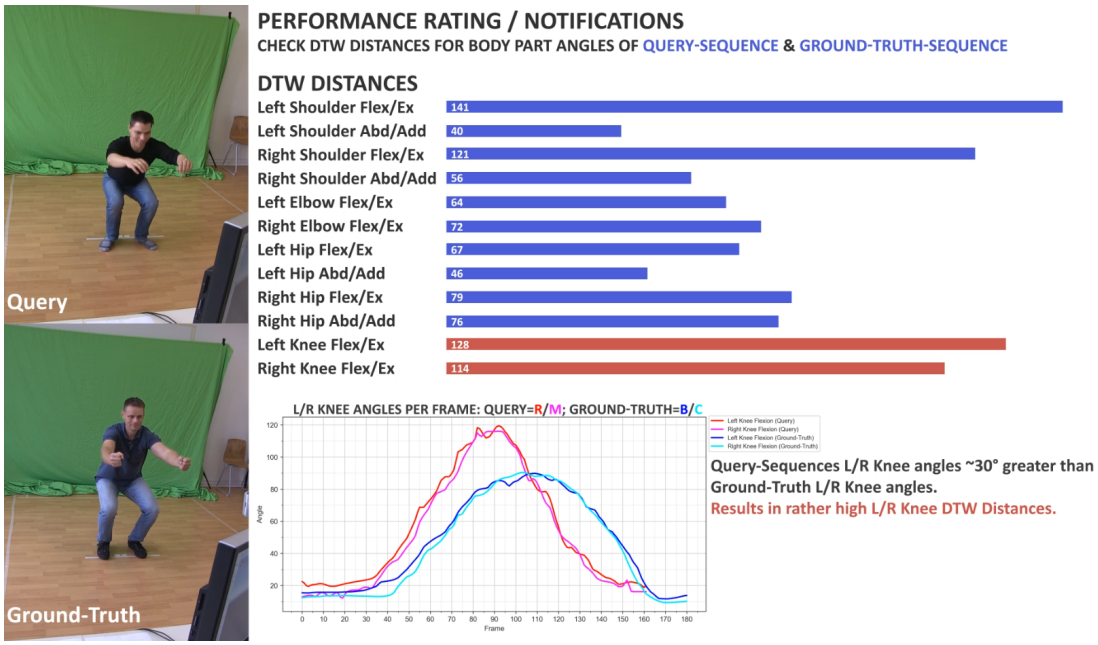

Figure 5.3: Trainees' squat exercise performance rating by comparing joint angle sequences to the joint angle sequences of an exercises' ground-truth motion sequence. Shows the performing trainee of the query sequence (top-left), the performing trainee of the ground-truth sequence, the DTW distances for each body part angle comparison (right) and graphs for the knees joint angles of both sequences.

works for single iteration sequences, is used as additional input for the rating procedure. Figure 5.2 shows the performance rating results for a squat execution. On the left hand side, the trainees pose at the turning frame of his motion is shown. The right hand side shows that the current target state is the *END* state and that the defined tolerance for the squat exercise is ten degrees. Further the target angle ranges, the current joint angles and the result states for prioritised joint angles of the squat exercise are displayed. The final execution performance result is located in the top right corner. In this case, the execution is incorrect due to exceeded knee angles. A correct execution requires to never exceed any of the prioritised angles during the motion.

In addition to the rating procedure stated in the last paragraph, another exercise performance rating method has been developed that directly utilises the DTW algorithm in order to determine an aligned distance between joint angle sequences of selected body parts. This method aims to identify body part motions in a query sequence which differ from those of the ground-truth sequence. Figure 5.3 visualises such comparison results. In this case, the comparison of the knees flexion angle sequences resulted in rather high DTW distances,

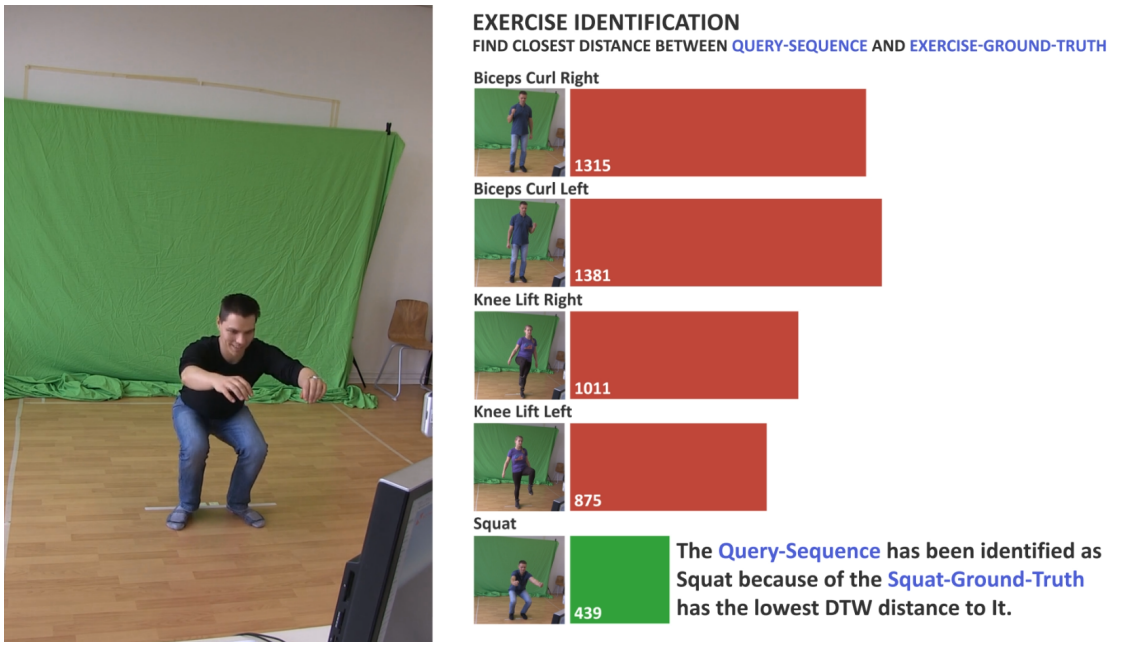

Figure 5.4: Result visualisation for identification of the exercise that is performed in a query sequence. Showing the query sequence performing trainee (left), the ground-truth sequences for five types of exercises as well as resulting DTW distances of the comparisons between the query sequence and ground-truth sequences respectively.

which indicates that the knees are bended differently. In the bottom of the figure, the knees joint angle graphs of both sequences are shown, which show that the knees maximum joint angles of both sequences differ by about 30 degrees. In general, the DTW distance results should lead to notifications that enable users to re-evaluate their body part motions or point out specific body parts which should be further examined in order to generate more specific performance feedback.

The last result feature identifies the type of exercise that is performed in a motion sequence by determining the DTW distances between a query sequence and multiple ground-truth sequences that represent one type of exercise respectively. In this method, a multidimensional DTW comparison is performed using all calculated joint angles of the sequences. Consequently, the resulting distance represents the difference of the compared sequences based on all available joint angles. The ground-truth sequence that results in the lowest DTW distance represents the type of exercise performed in the query sequence. Figure 5.4 shows the DTW distances after comparing the query sequence, that is performed by the trainee on the left hand side, to the exercise representing ground-truth sequences on the right hand side. In this example, the squat exercise has been identified correctly as its

ground-truth sequence resulted in the lowest DTW distance.
Additionally to the former presented results, a dataset including 1546 sequence files has been recorded. The dataset consists of eleven different exercises performed by eight trainees and is further separated in single and multi iteration sequences. The dataset has been used to test and evaluate the developed and implemented methods. Moreover a result video can be found on the attached data storage under *results_summary_video.mp4*. The video briefly presents the developed results and includes video clips of performed exercises from the recorded dataset.

5.2 Observations

5.2.1 Subsequencing Motion Sequences

In order to evaluate the subsequencing procedure described in 3.3 six exercises of the dataset presented in 3.1.1 have been selected. The selected exercises are *squat*, *overhead press*, *biceps curl left*, *biceps curl right*, *knee lift left* and *knee lift right*. The dataset contains sequences that consist of ten repetitions of a particular exercise. The amount of available sequences per exercise is listed in the *No. of Ten i Seqs* column of Table 3.1.
The goal is to find the parameter setup of the procedure which delivers the best results. The result of each particular exercise is determined by the *expected iterations* to *identified iterations* ratio. A perfect result for an exercise is considered a ratio of 1.000. The overall result across all examined exercises is then determined by the mean Euclidean distance from the perfect *expected iterations* to *identified iterations* ratio across all exercises. So the smaller the distance, the better the result. Consequently, a perfect overall result would be a mean distance of 0.0. The result values are rounded to three decimal places. The number of expected iterations equals the number of sequences including ten repetitions of an exercise multiplied by ten.
The procedure has four parameters that may influence the results, which are the *Savitzky Golay Filter window*, the *Savitzky Golay Filter order*, the *argrelextrema function order* and the *extrema grouping window size*. The parameters have initially been set to the following values as stated in chapter 4.2: *Savitzky Golay Filter window = 51*, *Savitzky Golay Filter order = 3*, *argrelextrema function order = 10*, *extrema grouping window size = 30*. The initial value setup has been chosen as it produced good results for a small number of test sequences while implementing the prototype.

The goal is now to find the parameter setup which results in the smallest average absolute *expected iterations* to *identified iterations* ratio distance. The start parameter values will be set to the initial setup, which has been explained in the last paragraph.

Savitzky Golay Filter Window

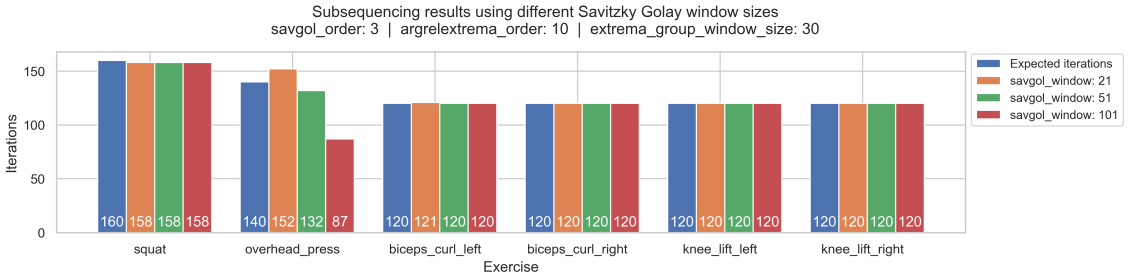

Figure 5.5: Subsequencing results bar chart grouped by exercises using Savitzky Golay window sizes 21, 51 and 101. Other parameters are set to: Savitzky Golay order=3, argrelextrema order=10, extrema group window size=30

Exercise	savgol_win = 21	savgol_win = 51	savgol_win = 101
squat	0.988	0.988	0.988
overhead press	1.086	0.943	0.621
biceps curl left	1.008	1.000	1.000
biceps curl right	1.000	1.000	1.000
knee lift left	1.000	1.000	1.000
knee lift right	1.000	1.000	1.000
Avg Abs Distance	0.106	0.069	0.391

Table 5.1: Subsequencing result ratios per exercise and average absolute distance using Savitzky Golay window sizes 21, 51 and 101. Other parameters are set to: Savitzky Golay order=3, argrelextrema order=10, extrema group window size=30

In this section, the results using different Savitzky Golay Filter window values are presented. In addition to the initial value of 51, a smaller size of 21 and a greater size of 101 have been used. The result can be observed in Figure 5.5. It can be seen that the results for different window sizes are similar for all exercises except the overhead press. The number of identified iterations for the overhead press exercise increases if the Savitzky Golay window is smaller. For the window size of 21, the resulting ration is 1.086. Thus, more iterations have been identified than expected.

All result ratios are listed in Table 5.1. A Savitzky Golay window size of 51 has resulted in the best average absolute distance across all examined exercises. But it can be observed that only the overhead press results using a window size of 101 differ from the results using a window size of 51. Also, the window size of 21 resulted in ratios very close to 1.000 but it is conspicuous that more iterations have been identified than have been actually performed. In this scenario the initial configuration performed best, resulting in an average absolute distance of 0.069.

Extrema Grouping Window

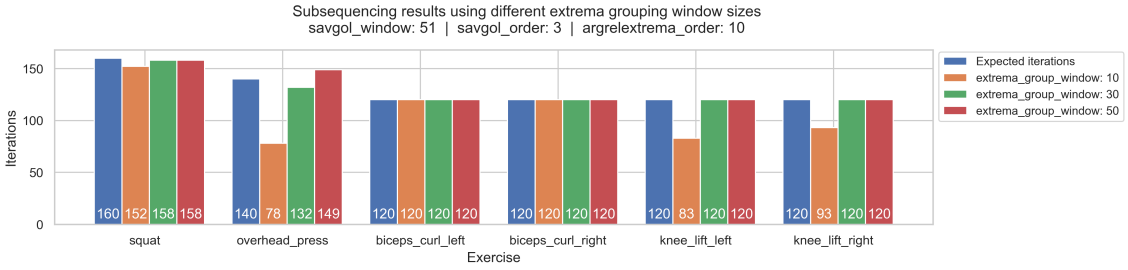

Figure 5.6: Subsequencing results bar chart grouped by exercises using extrema grouping windows 10, 30 and 50. Other parameters are set to: Savitzky Golay order=3, argrelextrema order=10, Savitzky Golay window size=51

Exercise	grouping_win = 10	grouping_win = 30	grouping_win = 50
squat	0.950	0.988	0.988
overhead press	0.557	0.943	1.064
biceps curl left	1.000	1.000	1.000
biceps curl right	1.000	1.000	1.000
knee lift left	0.692	1.000	1.000
knee lift right	0.775	1.000	1.000
Avg Abs Distance	1.026	0.069	0.076

Table 5.2: Subsequencing result ratios per exercise and average absolute distance using extrema grouping windows 10, 30 and 50. Other parameters are set to: Savitzky Golay order=3, argrelextrema order=10, Savitzky Golay window size=30

In this section, the results using different extrema grouping window size values are presented. Apart from the initial value 30, the window sizes 10 and 50 are tested. The results are visualised in Figure 5.6. The resulting *expected iterations* to *identified iterations* ratios for window sizes of 30 and 50 are the same for all exercises but the overhead press. A extrema

grouping window of size 10 results in significantly lower ratios. Consequently, the average absolute ratio distance of 1.026 is very high compared to the distances 0.069 and 0.076 of the other window sizes, as can be seen in Table 5.2.
Similar to the test for different Savitzky Golay window sizes, the overhead press results are the worst with an average ratio of 0.855 across all extrema grouping window sizes. In addition to that, the initial configuration delivered the best result with an average absolute distance of 0.069, shortly before the distance of 0.076 for a window size of 50.

Investigating Overhead Press results

The evaluation results for the overhead press exercise differ significantly from the results of the other exercises, which can be observed in Figure 5.5, Figure 5.6, Table 5.1 and Table 5.2. On the one hand the resulting *expected iterations* to *identified iterations* ratios are the worst compared to all exercises. On the other hand, the results vary much more than for the other exercises across different parameter values.
In order to investigate the reason for these results, the subsequencing procedure of an overhead press sequence has been visualised for the best performing setup (ratio=0.943) in Figure 5.7. The subsequencing procedure identified 11 iterations for that sequence. It can be observed, that the orange graph representing the right shoulders *abduction/adduction* angles drops drastically from whenever the angle is close to 180 degrees. The reason for that phenomenon is that the *abduction/adduction* angle calculation is able to determine an angle range from -180 to 180. So whenever the trainee's shoulder abduction should be greater than 180 degrees, it jumps to a value that is slightly higher than -180 degrees. This leads to very significant minima in the right shoulder angle graph displayed in Figure 5.7. Subsequently, those minima may lead to start/end key frames if undesired extrema of the other body parts are within the extrema grouping window range. If such false start/end key frames in between two turning key frames, this will result in a false iteration. Even though this is a major issue related to the angle calculations, the subsequencing procedure is still able to identify most of the iterations correctly.

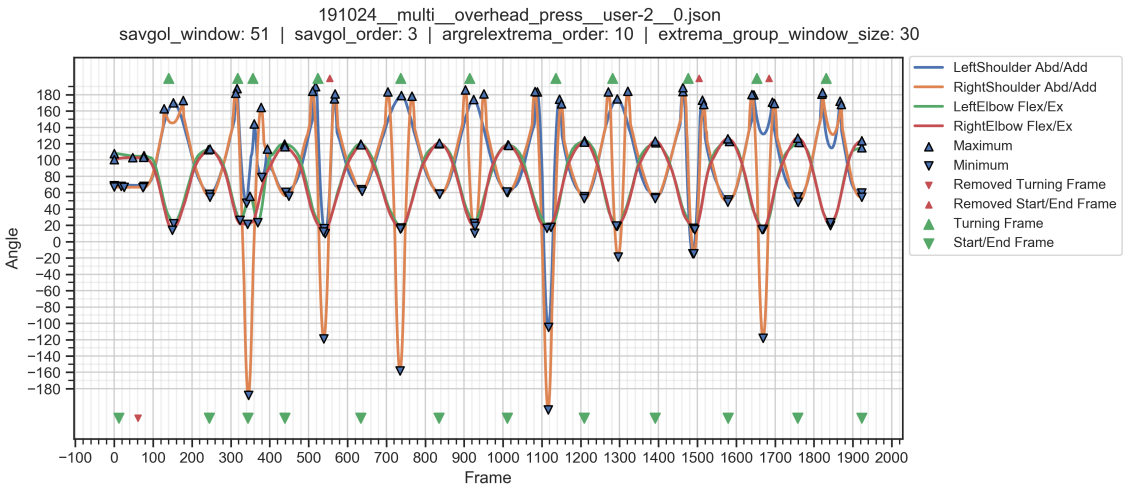

Figure 5.7: Marked extrema of smoothed prioritised joint angle data of a motion sequence that includes ten iterations of a squat exercise after removing extrema markings that are not close enough to their respective target angle. Enhanced by marked frames that represent confirmed keypoint candidates that have been removed and other marked frames that represent the final key frames of single iterations. Subsequencing parameters are set to: Savitzky Golay window size=51, Savitzky Golay order=3, argrelextrema order=10, extrema grouping window=30

5.2.2 Matching Motion Sequences

In order to evaluate the DTW distance based motion sequence matching procedure described in 3.4 five exercises of the dataset presented in 3.1.1 have been selected. The selected exercises are *squat*, *biceps curl left*, *biceps curl right*, *knee lift left* and *knee lift right*. The overhead press exercise sequences are not used due to the discovered angle calculation issues stated in chapter 5.2.1. The test dataset described in chapter 3.1.1 contains single iteration exercises as listed in Table 3.1.

The goal in this chapter is to evaluate whether the DTW based matching method comparing clinically meaningful joint angles of a query sequence to those of ground-truth sequences is able to identify the performed exercise correctly. The result is expressed as the percentage of correctly identified exercises. A correct identification is accounted if the DTW distance to the ground-truth sequence of the correct exercise is less than the distances to all other ground-truth sequences. The ground-truth sequence for each exercise is selected by hand on a random basis but making sure the sequence has no major tracking issues and the execution is assessed to be correct.

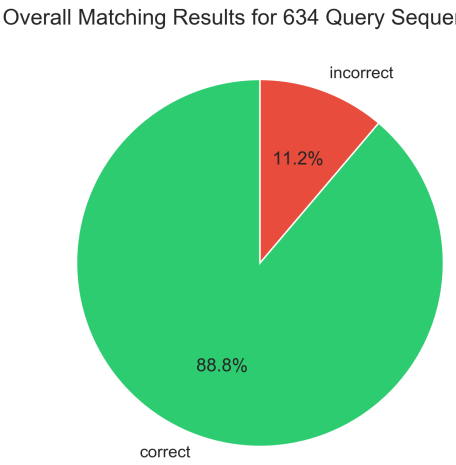

Figure 5.8: Overall matching result for all single iteration sequences of squat, biceps curl left, biceps curl right, knee lift left and knee lift right exercises.

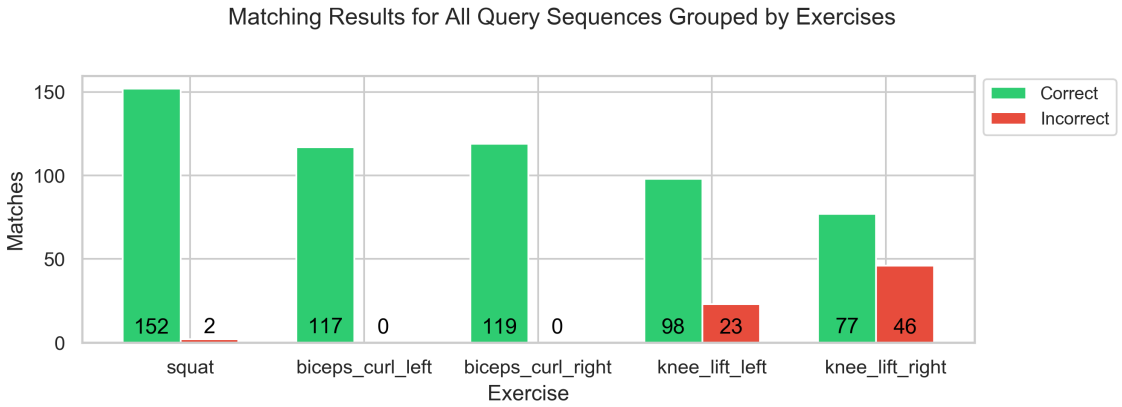

Figure 5.9: Matching results grouped by exercises for all single iteration sequences of squat, biceps curl left, biceps curl right, knee lift left and knee lift right exercises.

Exercise	Correct Matches	Incorrect Matches
squat	152 (98.7 %)	2 (1.3 %)
biceps curl left	117 (100.0 %)	0 (0.0 %)
biceps curl right	119 (100.0 %)	0 (0.0 %)
knee lift left	96 (80.7 %)	23 (19.3 %)
knee lift right	77 (62.6 %)	46 (37.4 %)
Overall	563 (88.8 %)	71 (11.2 %)

Table 5.3: Correct and incorrect matches listed by exercises for all single iteration sequences of squat, biceps curl left, biceps curl right, knee lift left and knee lift right exercises.

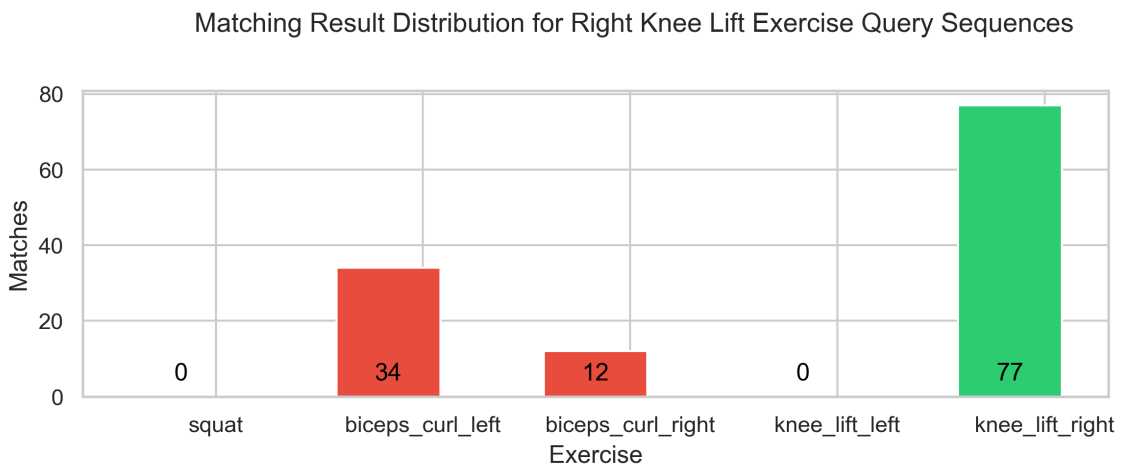

Figure 5.10: Closest distance result distribution after comparing 123 right knee lift query sequences to one ground-truth sequence per exercise.

The overall result for the comparison of 634 sequences resulted in 563 (88.8%) correct and 71 (11.2%) incorrect matches as visualised in Figure 5.8. Figure 5.9 shows the distribution of the results across the exercises. It is striking that the amount of incorrect matches of the right and left knee lift exercises is considerably higher than those of the other exercises. Table 5.3 lists all matching results by their absolute value and percentage separated by exercise.

Observing the result distribution for the right knee lift exercise displayed in Figure 5.10 shows that all incorrect matches have been assigned to the left and right biceps curl exercises. Consequently, it is assumed that the right knee lift sequences which have been incorrectly matched to the left biceps curl ground-truth include left arm movement that is more similar to the left biceps curl ground-truth than to the right knee lift ground-truth. The left elbow angles of two ground-truths sequences and an incorrectly matched query sequence have been compared in Figure 5.11. It can be seen that the query sequences left elbow flexion angle has been moved much less than in both ground-truth sequences. As query sequence graph is very different to both ground-truth sequences there is no obvious indicator that confirms the assumption. In Figure 5.12 the same sequences have been compared for the left shoulder flexion angles. In this case, it can be observed that the graph of the left shoulder angles for the left biceps curl ground-truth sequence is significantly more similar to the query sequence than the graph of the right knee lift sequence. Based on these observations, it is concluded that the major difference in the left shoulder angles of the right knee lift query and ground-truth sequences lead to a resulting DTW distance that is greater than that of the left biceps curl ground-truth.

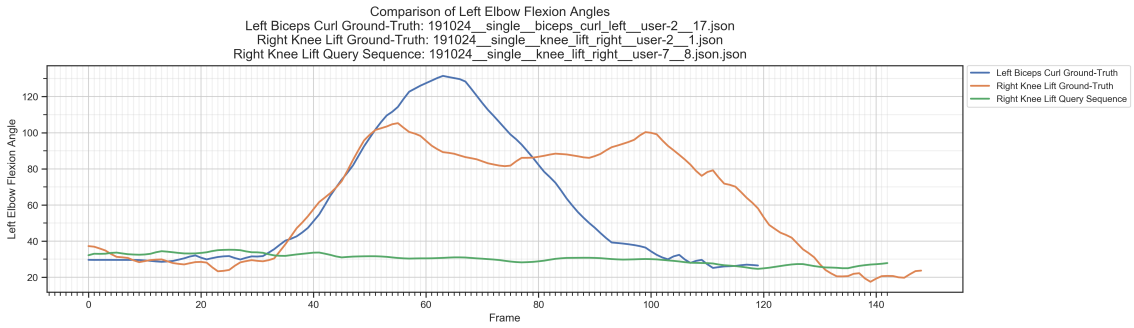

Figure 5.11: Comparison of left elbow flexion angles for the ground-truth sequences of the right knee lift and left biceps curl exercises and a query sequence of a right knee lift.

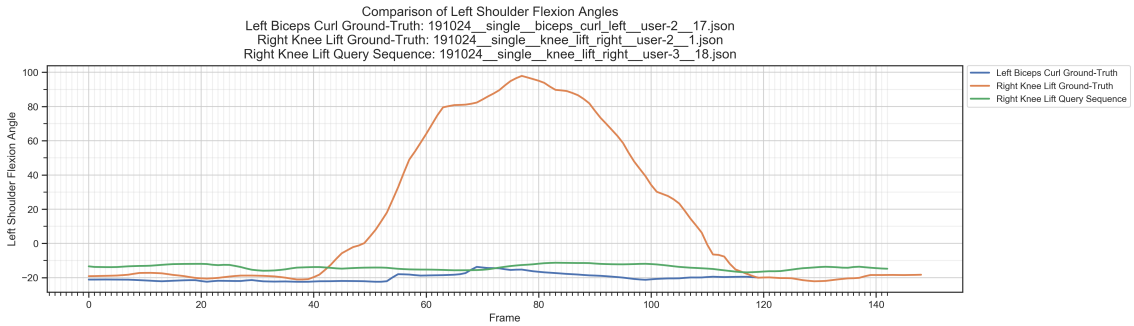

Figure 5.12: Comparison of left shoulder flexion angles for the ground-truth sequences of the right knee lift and left biceps curl exercises and a query sequence of a right knee lift.

5.2.3 Rating Execution of Exercises

The evaluation of the two exercise execution rating approaches as described in chapter 3.5 is done without the participation of a physician. Consequently, the correctness of executed exercises can not be reliably determined. Thus, the evaluation of both approaches does not include final determinations whether an exercise execution is correct or incorrect. However, both approaches are evaluated focussing on their specific outcome.

The *Comparing Joint Angles* method is supposed to return specific feedback whether predefined target angles for each body part have been met. Therefore, the validity of the returned feedback is checked whether it corresponds to what is expected when comparing the actual angles of a sequence with the predefined target angles.

The second approach provides information about the similarities of a query sequence and a ground-truth sequence with respect to specific body parts. In order to determine

whether this feedback in form of a DTW distance is a meaningful indicator to identify wrong executions the correlation between the difference of the maximum angles from ground-truths and query sequences and the returned DTW distances are examined.

Comparing Joint Angles

In order to evaluate the *Comparing Joint Angles* approach, ten single iteration exercise sequences of a squat exercise will be selected on a random basis. Then the joint angle values for all prioritised angles of those sequences are retrieved for their *turning frame* as determined by the subsequencing procedure described in chapter 3.3. The squats exercise file specifies the *left knee flexion*, *right knee flexion*, *left hip flexion* and *right hip flexion* as prioritised angles. The optimal range for all these angles is specified as 45 to 90 degrees with a tolerance of 10 degrees. So for this evaluation scenario, the determined result states shall fit the maximum angle for the *turning frame* of the sequence.

Table 5.4 shows the results of the evaluation. It can be observed that all 40 result states have been determined correctly. As the defined tolerance in the quats exercise file is 10, angles from 35 to 100 degrees are considered as *in range*. Angles below 35 shall lead to the result state *undercut* as it is the case for the left hip flexion angle of sequence number 10. Further, all angles greater than 100 degrees correctly resulted in *exceeded* states as occurred in the sequences number 1, 4 and 5.

Seq	Left Hip Flex		Right Hip Flex		Left Knee Flex		Right Knee Flex		Result
	Angle	State	Angle	State	Angle	State	Angle	State	
1	63.5°	in_range	60.3°	in_range	121.8°	exceeded	123.8°	exceeded	correct
2	49.0°	in_range	48.3°	in_range	89.3°	in_range	90.4°	in_range	correct
3	48.8°	in_range	49.3°	in_range	88.1°	in_range	88.9°	in_range	correct
4	65.2°	in_range	46.3°	in_range	114.3°	exceeded	103.5°	exceeded	correct
5	67.0°	in_range	63.8°	in_range	121.2°	exceeded	119.9°	exceeded	correct
6	46.9°	in_range	44.9°	in_range	98.7°	in_range	98.7°	in_range	correct
7	45.4°	in_range	44.8°	in_range	88.5°	in_range	85.5°	in_range	correct
8	43.0°	in_range	36.3°	in_range	72.14°	in_range	61.14°	in_range	correct
9	40.2°	in_range	40.1°	in_range	67.2°	in_range	65.8°	in_range	correct
10	32.2°	undercut	36.4°	in_range	69.4°	in_range	72.6°	in_range	correct

Table 5.4: Joint angles for prioritised angles of single iteration squat exercise sequences at the turning frame and the determined result state.

Quality Ratings from Similarities

To evaluate the exercise execution rating approach described in chapter 3.5.2, the correlation between two comparison values of a squat exercise ground-truth sequence and 144 squat exercise query sequences is determined. One of the values is the DTW distance between the two sequences concerning one prioritised joint angle. The other comparison value is the absolute difference between the maximum angles of the two sequences with regard to one prioritised joint angle, which is further simplified as the *maximum angle difference*.
The result of the evaluation shall show how strongly the 144 distances and *turning frame* angle differences correlate to each other. Therefore, the correlation coefficient is determined for each prioritised angle of the squat exercise, which are the *left hip*, *right hip*, *left knee* and *right knee*. A description about prioritised angles is given in chapter 3.1.3.
The correlation between the DTW distance and the maximum angle difference shall show how meaningful the DTW distance is in a scenario where a physician defines maximum angle guidelines for a patient. The DTW distance is meaningful if the correlation is strong as a high violation of a physicians maximum angle guidelines result in a high DTW distance. In contrast to this, it is meaningless with respect to the described scenario if the correlation is rather weak. A weak correlation is represented by a correlation coefficient near zero, a strong positive correlation by a correlation coefficient near one, and a strong negative correlation by a correlation coefficient near minus one.

The figures 5.13, 5.14, 5.15 and 5.16 display the correlation between maximum angle differences and DTW distances with respect to a prioritised joint angle. Additionally, a linear regression graph is shown, which visualises the relationship between both values. It can be observed, that there is a rather strong positive correlation for all examined joint angles. The left hip flexion maximum angle differences show the weakest correlation to the DTW distances represented by a correlation coefficient of 0.866. The left knee flexion shows the strongest correlation with a correlation coefficient of 0.957. The average correlation coefficient across all four prioritised body parts of the squat exercise is 0.904 and represents a strong positive correlation.

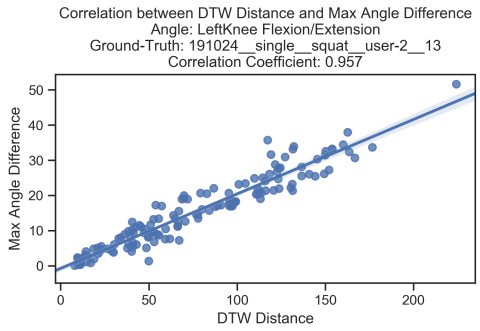

Figure 5.13: Correlation of the absolute differences between maximum left knee angles and DTW distances of a ground-truth squat sequence and 144 query squat sequences

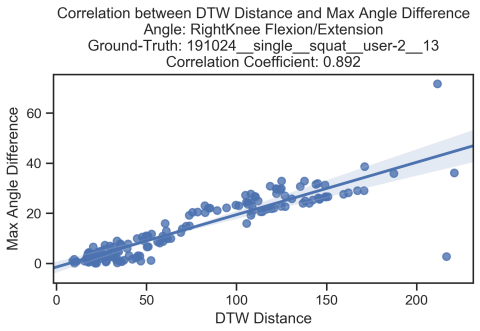

Figure 5.14: Correlation of the absolute differences between maximum right knee angles and DTW distances of a ground-truth squat sequence and 144 query squat sequences

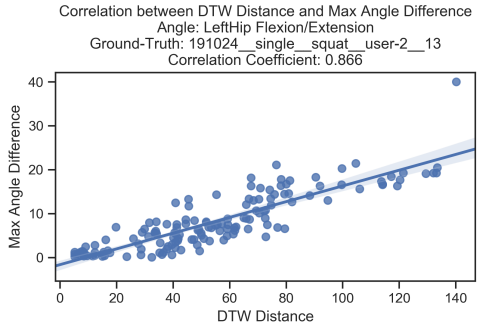

Figure 5.15: Correlation of the absolute differences between maximum left knee angles and DTW distances of a ground-truth squat sequence and 144 query squat sequences

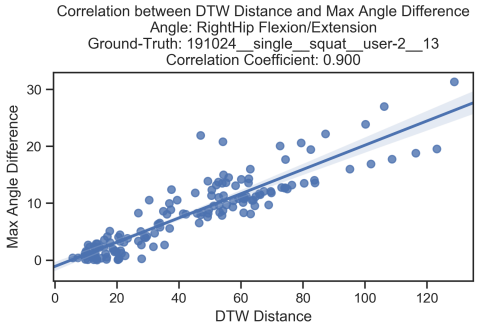

Figure 5.16: Correlation of the absolute differences between maximum right knee angles and DTW distances of a ground-truth squat sequence and 144 query squat sequences

5.3 Conclusion

5.3.1 Subsequencing Motion Sequences

Different parameters for the subsequencing procedure have been tested to find the best configuration. However, the initial configuration resulted in the best average absolute distance to the perfect *expected iterations* to *identified iterations* ratio of 1.000. The configuration led to 770 identified iterations compared to 780 expected iterations (98.718%) resulting in an average absolute distance of 0.069. However, it is important to note that the average absolute distance and the percentage of identified iterations are only an indicator of how well the procedure performs. It is still possible that unidentified iterations are compensated by mistakenly identified iterations.
Apart from that, the results for the overhead press exercise differ greatly from the other exercises caused by an issue with *abduction/adduction* angle calculations described in the last chapter. This leads to the assumption that exercises in which the occurrence of abduction angles greater than 180 degrees can not be excluded will not deliver the optimal results using this subsequencing procedure while using clinically meaningful joint angles.

5.3.2 Matching Motion Sequences

The test results using 634 query sequences of five different exercises shows that squat and biceps curl exercises are reliably identified by comparing them to the chosen ground-truth sequences using the DTW algorithm with medical joint angle input. Only two (0.5%) of these 390 sequences were matched incorrectly. However, the results also show that knee lift exercises are frequently matched with the ground-truths sequences of biceps curls due to similar arm movement. 69 (28.5%) of these 242 sequences were matched incorrectly.
The results lead to the conclusion, that exercise executions which differ from the ground-truth execution of that exercise might more likely be identified incorrectly. Especially if a body part is not moved at all when it should be or when the pool contains multiple similar exercises. It is assumed that a test scenario using a bigger pool of exercises including different variations of those exercises may influence the test results to the worse.

5.3.3 Rating Execution of Exercises

The evaluation results for both exercise rating approaches show that both approaches are suitable to deliver meaningful feedback, which can be further processed in order to determine whether an exercise is performed correctly or not.
The evaluation of the *Comparing Joint Angles* method shows, that the method provides reliable information about whether the reached joint angles lie in a predefined range. Based on that information, good and bad performances can be determined on a one frame basis and high-quality feedback with respect to each joint angle can be returned.
The evaluation of the *Quality Ratings from Similarities* approach shows, that a strong correlation between maximum angle differences and DTW distances with respect to a particular joint angle exists. Therefore, the DTW distance is interpreted as a viable indicator whether joint angles of a ground-truth sequence are met with respect to particular joint angles. Consequently, in a scenario where a physician guides a patient by stating target angles for an exercise along with recording a personalised ground-truth sequence in which those angles are met, the DTW distances of the *Quality Ratings from Similarities* methods allow to give feedback about whether the guidelines have been kept.

6 Conclusion and Outlook

6.1 Conclusion

The main objectives of this thesis have been to find viable solutions (A) to retrieve single iteration subsequences from repetitive exercise sequences, (B) to match motion sequences by determining a difference between them and (C) to automatically rate the execution of sport exercises. Additionally, (D) a 3-D motion sequence dataset of sport exercises should be created in order to evaluate the developed approaches for *A*, *B* and *C*.
This work contributes useful solutions to all main objectives. Therefore, clinically meaningful joint angles have been calculated as required by the project context in order to compare these angles to predefined exercise angles which represent an optimal execution. These joint angles, which were calculated from 3-D motion tracking input, have further been used as pose describing input data for all other developed approaches.

The first part of this thesis provides information about the main objectives and their relevance in a medical environment by briefly stating possible scenario in which the objectives *A*, *B* and *C* could be of high value and how the parts could work together. Moreover, possible challenges regarding the objectives are mentioned and a brief overview of the research context and its requirements is given as well as the major structure of this work is presented.
The second part presents related work and fundamental knowledge regarding the main objectives and human joint angle calculations, followed by brief discussions about the suitability of the presented approaches regarding the objectives and requirements of this thesis.
Continuing with the methodology, the main part of this work, where the approaches to fulfil the main objectives *A*, *B* and *C* are presented as well as the calculation of joint angles and defined representations of motion sequences and exercises. Thus, each procedure is structured into its main parts and explained in detail. It is important to note, that the

provided solutions, on the one hand, work on their own but on the other hand can not realise a scenario as stated in the second part of chapter 1.1 and visualised in Figure 1.1 due to the usage of clinically meaningful joint angles. The problem is, that the postprocessing step explained in chapter 3.2.3 is required to determine these angles. This postprocessing step already needs information about the target angles of a particular exercise in order to predict the most likely type of motion. However, the identification of the performed exercise represents step (4) in the stated scenario and requires a single iteration sequence, which should be retrieved in step (3) already. Though, step (3) already requires the resulting clinically meaningful angles after the mentioned postprocessing step. A possible solution to this could be the usage of clinically *meaningless* angles for the subsequencing procedure, which do not require a postprocessing step. Even though the stated procedure which connects the singular objectives of this work can not be realised with the presented methods, each method on its own represents a viable solution to the respective task.
To confirm the viability of each solution, all methods have been evaluated on their own using a dataset, which has been created in the context of this work and represents the fourth main contribution (D). The evaluation showed, that each of the developed methods represents a viable approach to tackle the main objectives of this work. Nevertheless, some problems and unfavourable circumstances could also be identified. One important problem has been found by examining the subsequencing results of the overhead press exercise. During the process, rapid abduction angle changes from near 180 degrees to -180 degrees were notices. This interfered with the subsequencing procedure, which is based on the identification of minima and maxima in joint angle sequences. Consequently, the current subsequencing method using clinically meaningful angles is not perfectly suited for exercises which may consist of abduction or adduction motions that result in angles greater than 180 degrees. For exercises, which did not include such motions, the method works very well. This again leads to the assumption, that a clinically *meaningless* representation of joint angles may resolve this problem.
Despite that, a potential disadvantage of the approach to match motion sequences has been assumed. The method is on the one hand very accurate distinguishing motions, which differ greatly in used body parts but shows weaknesses when comparing rather similar motions or motions that become similar when performed without motion of particular body parts. This led to the assumption, that the method is not perfectly viable to identify different variations of an exercise or reliably distinguish rather similar exercises. As the similarity of exercises increase in datasets containing more exercises, this could be a problem in real-world scenarios. However, it is also noticed, that the matching approach may still be

suited to retrieve a list of the most similar motion sequences from a database containing any amount of unlabeled motion sequences.
Furthermore, the evaluation of the two exercise rating approaches has shown, that both approaches are suited to return meaningful feedback regarding all body parts and joint angles to rate the execution of exercises based on these methods. Nevertheless, the method that utilises similarities of joint angle sequences seems more promising than the method in which tracked and predefined joint angles are directly compared. This is mainly due to the better flexibility of usable input data and easier potential customisability. On the one hand, the direct comparison method requires the determination of clinically meaningful joint angles in order to compare those to predefined target angles of exercises. On the other hand, the similarity does not necessarily require clinically meaningful angles but could potentially work with clinically *meaningless* angles or other types of input. Also, each variation of an exercise will require a separate exercise file for the comparison method, whereas the similarity method requires a ground-truth sequence as optimal execution reference. Instead of defining exercise files that fit the personal needs of each patient, the recording of the ground-truth performed by the physician is considered a better workflow. However, the returned feedback from the similarity method is not as suited as the feedback from the comparison method to identify injurious exercise executions caused by only slight angle differences to the correct execution. But still, the similarity methods feedback is viable as it enables to notify a user about single body parts which were moved differently to the ground-truth considering the whole motion, whereas the comparison method only provides information regarding single frames without taking the rest of the motion into consideration.

In conclusion, this work provides viable approaches to identify subsequences of single iterations from sequences of repeated exercises, to match motion sequences based on their similarity and to produce feedback for performed sport exercises in order to rate the performance. Nevertheless, is has been observed, that clinically meaningful angles may not always represent the one optimal pose describing input as they are hard to reliably determine and prone to anomalies for certain motions, those of the shoulders due to their high range of motion.

6.2 Outlook

On the whole, this work provides viable approaches to deal with motion analysis tasks like subsequencing of repetitive motions, motion sequence matching and rating of sport exercise motion sequences. Even though, there is room for improvements regarding the developed methods as well as the used pose representation input data.
As mentioned in the conclusion of this work, clinically meaningful joint angles are hard to reliably determine and are still prone to anomalies which consequently affect all the developed methods. An interesting approach to further improve the developed methods could be to use a different joint angle representation, which is not necessarily clinically meaningful, like for example *Quaternions*.
Other than that, the developed approaches may still have room for various improvements, especially in terms of robustness and performance optimisation, which could be examined for each method respectively. Despite that, each method may be further optimised methodically, by carefully examining their advantages and disadvantages in various usage scenarios. For example, the subsequencing approach could be examined and optimised when used to retrieve subsequences from motion sequence streams as the algorithm is potentially suited to perform this task. Further, the approaches which are based on Dynamic Time Warping distances may be optimised by examining the addition of weight scales for specific body parts, which reflect the importance of particular body parts for certain motions.
Apart from optimising the particular methods of this work, more complex software could be developed based on the developed approaches. As the research context already indicates, the most obvious usage may be software that assists trainees or patients while performing sport exercises. But other than that, the methods may be suited to put them into other contexts than only sport exercises as the subsequencing, matching and one of the rating methods do not necessarily only work for sports exercises but possibly for various kinds of motions.

List of Figures

List of Tables

Listings

Acronyms

AR Augmented Reality. 7

DOF degrees of freedom. 15

DTW Dynamic Time Warping. 9

GCS Global Coordinate System. 14

ISB International Society of Biomechanics. 17

JCS Joint Coordinate System. 9

JSON JavaScript Object Notation. 29

LCS Local Coordinate System. 14

STC Standardization and Terminology Committee. 17

Bibliography

[1] K. Adistambha, C. H. Ritz, I. S. Burnett. “Motion classification using Dynamic Time Warping”. In: *2008 IEEE 10th Workshop on Multimedia Signal Processing*. 2008 IEEE 10th Workshop on Multimedia Signal Processing. Oct. 2008, pp. 622–627. DOI: 10.1109/MMSP.2008.4665151 (cit. on p. 20).

[2] D. Alexiadis, P. Kelly, P. Daras, N. O’Connor, T. Boubekeur, M. Ben Moussa. “Evaluating a dancer’s performance via Kinect-based skeleton trackinga dancer’s performance via kinect-based skeleton tracking”. In: MM’11 - Proceedings of the 2011 ACM Multimedia Conference and Co-Located Workshops. Nov. 28, 2011, pp. 659–662. DOI: 10.1145/2072298.2072412 (cit. on pp. 23, 24).

[3] P. Allgeuer, S. Behnke. “Fused Angles and the Deficiencies of Euler Angles”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Madrid: IEEE, Oct. 2018, pp. 5109–5116. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8593384. URL: https://ieeexplore.ieee.org/document/8593384/ (visited on 09/03/2019) (cit. on p. 14).

[4] M. Andriluka, L. Pishchulin, P. Gehler, B. Schiele. “2D Human Pose Estimation: New Benchmark and State of the Art Analysis”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2014 (cit. on p. 5).

[5] C. Anglin, U. P. Wyss. “Review of arm motion analyses”. In: *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine* 214.5 (May 1, 2000), pp. 541–555. ISSN: 0954-4119. DOI: 10.1243/0954411001535570. URL: https://doi.org/10.1243/0954411001535570 (visited on 09/04/2019) (cit. on pp. 14, 16).

[6] J. Ben-Arie, Zhiqian Wang, P. Pandit, S. Rajaram. “Human activity recognition using multidimensional indexing”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.8 (Aug. 2002), pp. 1091–1104. DOI: 10.1109/TPAMI.2002.1023805 (cit. on p. 20).

[7] C. Bregler. “Learning and recognizing human dynamics in video sequences”. In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. ISSN: 1063-6919. June 1997, pp. 568–574. DOI: 10.1109/CVPR.1997.609382 (cit. on pp. 19, 20).

[8] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, Y. Sheikh. “OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields”. In: *arXiv preprint arXiv:1812.08008*. 2018 (cit. on p. 5).

[9] K.-h. Chang, M. Y. Chen, J. Canny. “Tracking Free-Weight Exercises”. In: *UbiComp 2007: Ubiquitous Computing*. Ed. by J. Krumm, G. D. Abowd, A. Seneviratne, T. Strang. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, pp. 19–37. ISBN: 978-3-540-74853-3. DOI: 10.1007/978-3-540-74853-3_2 (cit. on p. 19).

[10] O. Chomat, J. Crowley. “Recognizing Motion Using Local Appearance”. In: (Mar. 2, 2000) (cit. on p. 20).

[11] G. K. Cole, B. M. Nigg, J. L. Ronsky, M. R. Yeadon. “Application of the Joint Coordinate System to Three-Dimensional Joint Attitude and Movement Representation: A Standardization Proposal”. In: *Journal of Biomechanical Engineering* 115.4 (1993), p. 344. ISSN: 01480731. DOI: 10.1115/1.2895496. URL: http://Biomechanical.asmedigitalcollection.asme.org/article.aspx?articleid=1399508 (visited on 08/22/2019) (cit. on p. 14).

[12] A. Corradini. “Dynamic Time Warping for Off-Line Recognition of a Small Gesture Vocabulary”. In: *Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS’01)*. RATFG-RTS ’01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 82–. URL: http://dl.acm.org/citation.cfm?id=882476.883586 (cit. on p. 21).

[13] D. Das, S. M. Busetty, V. Bharti, P. K. Hegde. “Strength Training: A Fitness Application for Indoor Based Exercise Recognition and Comfort Analysis”. In: *2017 16th IEEE International Conference on Machine Learning and Applications*

(ICMLA). 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA). Dec. 2017, pp. 1126–1129. DOI: 10.1109/ICMLA.2017.00012 (cit. on pp. 19, 23).

[14] D. Del Vecchio, R. Murray, P. Perona. "Decomposition of human motion into dynamics-based primitives with application to drawing tasks". In: *Automatica* 39 (Dec. 1, 2003), pp. 2085–2098. DOI: 10.1016/S0005-1098(03)00250-4 (cit. on p. 19).

[15] E. Grood, W. Suntay. "A Joint Coordinate System for the Clinical Description of Three-Dimensional Motions: Application to the Knee". In: *Journal of biomechanical engineering* 105 (June 1, 1983), pp. 136–44. DOI: 10.1115/1.3138397 (cit. on pp. 9, 17).

[16] R. A. Güler, N. Neverova, 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 (cit. on p. 5).

[17] I. A. Kapandji. *The physiology of the joints: annotated diagrams of the mechanics of the human joints. 1: Upper limb. - 5. ed., compl. rev., repr. - 1989*. OCLC: 830896378. Edinburgh: Churchill Livingstone, 1989. ISBN: 978-0-443-02504-4 (cit. on pp. 10–13).

[18] E. Keogh, T. Palpanas, V. Zordan, D. Gunopulos, M. Cardle. "Indexing Large Human-Motion Databases". In: Proc Int Conf Very Large Data Bases. Vol. 30. Dec. 31, 2004, pp. 780–791. DOI: 10.1016/B978-012088469-8/50069-3 (cit. on pp. 18, 20).

[19] R. Krishnan, N. Björsell, E. Gutierrez-Farewik, C. Smith. "A survey of human shoulder functional kinematic representations". In: *Medical & Biological Engineering & Computing* 57 (Oct. 26, 2018). DOI: 10.1007/s11517-018-1903-3 (cit. on p. 9).

[20] *Kugelkoordinaten*. In: *Wikipedia*. Page Version ID: 190346530. July 11, 2019. URL: https://de.wikipedia.org/w/index.php?title=Kugelkoordinaten&oldid=190346530 (visited on 09/13/2019) (cit. on p. 37).

[21] C. Kümmel, P. Wesberg, S. Vorweg, O. Stamm, A. Steinert, J. Villwock, K. Hildebrand. "BewARe - Sensor-based Augmented Reality system for individualized endurance training for elderly people". In: IEEE VR 2019 Workshop - Applied VR for Enhanced Healthcare (cit. on p. 7).

[22] A. Kuzmanic, V. Zanchi. “Hand shape classification using DTW and LCSS as similarity measures for vision-based gesture recognition system”. In: *EUROCON 2007 - The International Conference on "Computer as a Tool"*. EUROCON 2007 - The International Conference on "Computer as a Tool". Sept. 2007, pp. 264–269. DOI: 10.1109/EURCON.2007.4400350 (cit. on p. 21).

[23] A. LaViers, Y. Chen, C. Belta, M. Egerstedt. “Automatic Sequencing of Ballet Poses”. In: *IEEE Robotics Automation Magazine* 18.3 (Sept. 2011), pp. 87–95. DOI: 10.1109/MRA.2011.942118 (cit. on p. 19).

[24] C. Li, M. Fei, H. Hu, Z. Qi. “Free Weight Exercises Recognition Based on Dynamic Time Warping of Acceleration Data”. In: Communications in Computer and Information Science. Vol. 355. Sept. 1, 2013, pp. 178–185. DOI: 10.1007/978-3-642-37105-9_20 (cit. on p. 19).

[25] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, P. Dollár. “Microsoft COCO: Common Objects in Context”. In: *arXiv:1405.0312 [cs]* (May 1, 2014). arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312 (visited on 09/12/2019) (cit. on p. 5).

[26] B. A. MacWilliams, R. B. Davis. “Addressing Some Misperceptions of the Joint Coordinate System”. In: *Journal of Biomechanical Engineering* 135.5 (May 1, 2013). ISSN: 0148-0731. DOI: 10.1115/1.4024142. URL: /biomechanical/article/135/5/054506/365823/Addressing-Some-Misperceptions-of-the-Joint (visited on 08/23/2019) (cit. on pp. 9, 17).

[27] J. E. Mebius. “Derivation of the Euler-Rodrigues formula for three-dimensional rotations from the general formula for four-dimensional rotations”. In: *arXiv:math/0701759* (Jan. 25, 2007). arXiv: math/0701759. URL: http://arxiv.org/abs/math/0701759 (visited on 09/12/2019) (cit. on p. 36).

[28] “DTW-Based Motion Comparison and Retrieval”. In: *Information Retrieval for Music and Motion*. Ed. by M. Müller. Berlin, Heidelberg: Springer, 2007, pp. 211–226. ISBN: 978-3-540-74048-3. DOI: 10.1007/978-3-540-74048-3_10. URL: https://doi.org/10.1007/978-3-540-74048-3_10 (visited on 10/30/2019) (cit. on p. 21).

[29] M. Müller. *Information Retrieval for Music and Motion*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. ISBN: 978-3-540-74047-6 978-3-540-74048-3. DOI: 10.1007/978-3-540-74048-3. URL: http://link.springer.com/10.1007/978-3-540-74048-3 (visited on 10/31/2019) (cit. on pp. 20, 21).

[30] G. Neumann, W. Maass, J. Peters. "Learning complex motions by sequencing simpler motion templates". In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. the 26th Annual International Conference. Montreal, Quebec, Canada: ACM Press, 2009, pp. 1–8. ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553471. URL: http://portal.acm.org/citation.cfm?doid=1553374.1553471 (visited on 09/16/2019) (cit. on p. 20).

[31] S. A. Niyogi, E. H. Adelson. "Analyzing and Recognizing Walking Figures in XYT". In: 1994, pp. 469–474 (cit. on p. 20).

[32] L. Perumal. "Euler angles: conversion of arbitrary rotation sequences to specific rotation sequence". In: *Computer Animation and Virtual Worlds* 25.5 (2014), pp. 521–529. ISSN: 1546-427X. DOI: 10.1002/cav.1529. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.1529 (visited on 09/02/2019) (cit. on p. 14).

[33] R. Polana, R. Nelson. "Low level recognition of human motion (or how to get your man without finding his body parts)". In: Dec. 11, 1994, pp. 77–82. ISBN: 978-0-8186-6435-9. DOI: 10.1109/MNRAO.1994.346251 (cit. on p. 20).

[34] J. Rittscher, A. Blake. "Classification of human body motion". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Proceedings of the Seventh IEEE International Conference on Computer Vision. Vol. 1. Sept. 1999, 634–639 vol.1. DOI: 10.1109/ICCV.1999.791284 (cit. on p. 20).

[35] D. G. E. Robertson, G. E. Caldwell, J. Hamill, G. Kamen, S. N. Whittlesey. *Research methods in biomechanics*. Second edition. Champaign, Illinois: Human Kinetics, 2014. 428 pp. ISBN: 978-0-7360-9340-8 (cit. on pp. 9, 14, 15).

[36] H. Sakoe, S. Chiba. "Dynamic programming algorithm optimization for spoken word recognition". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26.1 (Feb. 1978), pp. 43–49. ISSN: 0096-3518. DOI: 10.1109/TASSP.1978.1163055 (cit. on p. 21).

[37] S. Salvador, P. Chan. “Toward Accurate Dynamic Time Warping in Linear Time and Space”. In: *Intell. Data Anal.* 11.5 (Oct. 2007), pp. 561–580. ISSN: 1088-467X. URL: http://dl.acm.org/citation.cfm?id=1367985.1367993 (visited on 10/30/2019) (cit. on pp. 22, 23).

[38] A. Savitzky, M. J. E. Golay. “Smoothing and Differentiation of Data by Simplified Least Squares Procedures.” In: *Analytical Chemistry* 36.8 (July 1, 1964), pp. 1627–1639. ISSN: 0003-2700. DOI: 10.1021/ac60214a047. URL: https://doi.org/10.1021/ac60214a047 (cit. on p. 19).

[39] R. W. Schafer. “What Is a Savitzky-Golay Filter? [Lecture Notes]”. In: *IEEE Signal Processing Magazine* 28.4 (July 2011), pp. 111–117. ISSN: 1053-5888, 1558-0792. DOI: 10.1109/MSP.2011.941097 (cit. on p. 19).

[40] P. Senin. “Dynamic Time Warping Algorithm Review”. In: (Jan. 1, 2009).

[41] C.-J. Su. “Personal Rehabilitation Exercise Assistant with Kinect and Dynamic Time Warping”. In: *International Journal of Information and Education Technology* 3 (Jan. 1, 2013), pp. 448–454. DOI: 10.7763/IJIET.2013.V3.316 (cit. on pp. 23, 24, 51).

[42] R. Tang, X.-D. Yang, S. Bateman, J. A. Jorge, A. Tang. “Physio@Home: Exploring Visual Guidance and Feedback Techniques for Physiotherapy Exercises”. In: *CHI*. 2015. DOI: 10.1145/2702123.2702401 (cit. on p. 13).

[43] C. Tappert, C. Suen, T. Wakahara. “The state of the art in online handwriting recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12.8 (Aug. 1990), pp. 787–808. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/34.57669 (cit. on p. 21).

[44] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, C. Schmid. “Learning from Synthetic Humans”. In: *CVPR*. 2017 (cit. on p. 5).

[45] A. Vieira, W. Schwartz, M. Campos, T. Lewiner. “Distance Matrices as Invariant Features for Classifying MoCap Data”. In: Proceedings - International Conference on Pattern Recognition. Jan. 1, 2012 (cit. on p. 6).

[46] P. Volny, D. Novak, P. Zezula. “Employing Subsequence Matching in Audio Data Processing”. In: (Apr. 2012) (cit. on p. 48).

[47] T. Wren, P. Mitiguy. “A Simple Method to Obtain Consistent and Clinically Meaningful Pelvic Angles from Euler Angles during Gait Analysis”. In: *Journal of applied biomechanics* 23 (Sept. 1, 2007), pp. 218–23. DOI: 10.1123/jab.23.3.218 (cit. on p. 9).

[48] G. Wu, P. R. Cavanagh. “ISB recommendations for standardization in the reporting of kinematic data”. In: *Journal of Biomechanics* 28.10 (Oct. 1, 1995), pp. 1257–1261. ISSN: 0021-9290. DOI: 10.1016/0021-9290(95)00017-C. URL: http://www.sciencedirect.com/science/article/pii/002192909500017C (visited on 08/23/2019) (cit. on p. 17).

[49] G. Wu, F. C. van der Helm, H. (DirkJan) Veeger, M. Makhsous, P. Van Roy, C. Anglin, J. Nagels, A. R. Karduna, K. McQuade, X. Wang, F. W. Werner, B. Buchholz. “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”. In: *Journal of Biomechanics* 38.5 (May 2005), pp. 981–992. ISSN: 00219290. DOI: 10.1016/j.jbiomech.2004.05.042. URL: https://linkinghub.elsevier.com/retrieve/pii/S002192900400301X (visited on 08/22/2019) (cit. on pp. 14, 16–18).

[50] G. Wu, S. Siegler, P. Allard, C. Kirtley, A. Leardini, D. Rosenbaum, M. Whittle, D. D. D’Lima, L. Cristofolini, H. Witte, O. Schmid, I. Stokes. “ISB recommendation on definitions of joint coordinate system of various joints for the reporting of human joint motion—part I: ankle, hip, and spine”. In: *Journal of Biomechanics* 35.4 (Apr. 2002), pp. 543–548. ISSN: 00219290. DOI: 10.1016/S0021-9290(01)00222-6. URL: https://linkinghub.elsevier.com/retrieve/pii/S0021929001002226 (visited on 08/23/2019) (cit. on pp. 17, 18).

[51] YassineMrabet. *English: Planes of human anatomy.* June 7, 2008. URL: https://commons.wikimedia.org/wiki/File%3AHuman_anatomy_planes.svg (visited on 09/05/2019) (cit. on pp. 6, 10).

[52] W. Zhao, M. A. Reinthal, D. D. Espy, X. Luo. “Rule-Based Human Motion Tracking for Rehabilitation Exercises: Realtime Assessment, Feedback, and Guidance”. In: *IEEE Access* 5 (2017), pp. 21382–21394. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2017.2759801 (cit. on pp. 23, 24).

All links were last followed on November 28, 2019.