Deep Learning based Auto-labeling for Sparse Underwater Freestyle Marker-based Optical Motion Capture using Qualisys Miqus M5U MoCap Cameras

by

© Neda Golpayegani

A thesis submitted to the School of Graduate Studies

in partial fulfilment of the requirements for the degree of

Master of Engineering

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

October 2024

St. John's Newfoundland and Labrador Canada

Abstract

This thesis proposes novel algorithms to automate cleaning and labeling of motion capture (MoCap) data specifically for underwater marker-based optical MoCap systems. Challenges related to sparse underwater freestyle MoCap data, captured using Qualisys Miqus M5U MoCap cameras, are explored using a dataset of 21 passive markers. A thorough review on MoCap denoising, recovery, alignment, and auto-labeling methods is conducted. The manual cleaning process using Qualisys Track Manager software and Automatic Identification of Markers function is explained. Then, a novel semi-supervised geometry-based labeling algorithm is developed based on distance and angle measurements with a visual evaluation of 100% accuracy. This algorithm includes sub algorithms for extraneous removal via norm differences, anomaly detection, pelvis detection based on Principal Component Analysis, recovery of missing markers, and detection of corresponding reappearing markers along with a side detection algorithm. Finally, a deep learning-based auto-labeling algorithm utilizing Long-Short-Term Memory is proposed, employing Hungarian label assignment and Procrustes analysis to label unlabeled data. The network accepts the 3D relative positions of markers, velocity, and acceleration. The ground truth and the training set are generated by the geometry-based algorithm and enhanced using data augmentation and transfer learning of simulated trajectories. The pelvis detection technique automates the alignment, and the extraneous removal algorithm enhances accuracy from 66% to 98%. These algorithms work effectively in the presence of outliers, extraneous, ghosts, and missing markers. Future work will evaluate the algorithm with more data and ghost markers and explore a more robust body side detection algorithm.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Stephen Czarnuch, who first motivated me to begin this master's degree, and provided invaluable guidance throughout its duration.

I thank the Faculty of Engineering and Applied Science at Memorial University.

I am grateful for the help I received from members of the Marine Institute of Memorial University, who provided me with Mocap data.

I am thankful for my family who have provided me with great support.

Content

List of Tables

List of Figures

List of Abbreviations

MoCap	Motion Capture
OMC	Optical Motion Capture
QTM	Qualisys Track Manager
AIM	Automatic Identification of Markers
PCA	Principal Component Analysis
LSTM	Long Short-Term Memory
3D	Three-Dimensional
C3D	Coordinate 3D
NN	Neural Network
SD	Standard Deviation
ICP	Iterative Closest Point
FOV	Field of View
6DOF	Six Degree of Freedom
Mokka	Motion Kinematic and Kinetic Analyzer
BTK	Biomechanical ToolKit
ROI	Region of Interest
ID	Identifier
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent

List of Symbols

Common

x, X	*x* Coordinate
y, Y	*y* Coordinate
z, Z	*z* Coordinate
NaN	Not a Number
s	Second
Hz	Hertz
º	Degree
*, ×	Multiplication
%	Percentage
Max, *max*	Maximum
Min, *min*	Minimum
L/R	Left/Right
NA	Not Applicable

Chapter 3

VDC	Volts Direct Current
ºC	Celsius Degree
ºF	Fahrenheit Degree
MP	Megapixel
fps	Frames Per Second

Chapter 4

ID_1	ID number 1
p	Point; any point in a 3D space
O	Origin = (0,0,0) coordinate
x_n, y_n, z_n	*x* Coordinate, *y* Coordinate, *z* Coordinate of n$_{th}$ point
x_1, y_1, z_1	*x* Coordinate, *y* Coordinate, *z* Coordinate of point 1
(matrix)’	Transpose a matrix: switch its rows and columns
Sp, W_1, W_2	Spine point, Waist point 1, Waist point 2
$LW, W_L; RW, W_R$	Left Waist point; right Waist points
Ch, St	Chest point, Stomach point
Sh_L, Sh_R	Left and right Shoulder points
Hi_R, Hi_L	Left and right Hip points
K_R, K_L	Left and right Knee points
A_R, A_L	Left and right Ankle points
H_R, H_L	Left and right Head points
E_R, E_L	Left and right Elbow points
Ha_R, Ha_L	Left and right Hand points
d	Distance between any two points in 3D space
d_1	Distance between two specific points
d_W	Distance between left and right waist points
d_{sh}	Distance between left and right shoulder points
d_{ss}	Distance between spine and stomach points

Md_W	Marker set (M) value for d_W
Δ	Tolerance: allowable deviation from a measurement
Δd	Tolerance in distance
Δd_M	Deviation from a defined distance in the Marker set (M)
θ	Angle; angle between any three points in 3D space
θ_1	Angle between three specific points
θ_{W1}	Angle of waist point 1
$\Delta\theta$	Tolerance in angle
$M\theta_{Sp}$	Marker set (M) value for spine angle (θ_{Sp})
$\Delta\theta_M$	Deviation from a defined angle in the Marker set (M)
$==, \neq$	Equal, Not Equal
$=, \leftarrow$	Assignment
$<, >$	Comparison Operators
#	Number
$\emptyset$	Empty Set
$\bigcup(sets)$	Union of sets
$\bigcap(sets)$	Intersection of sets
$length_{arm_{max}}$	The maximum length of arm
$length_{leg_{max}}$	The maximum length of leg
Triplet, Trio	A group of three points
Gap_1	The first gap in a trajectory
Sp_{trio1}	Spine point from Trio number 1

$frame_{cn}$	Common frames for Trio number n
f_{csn}	The first frame in common frames of Trio number n
$W2_{t1}$	Waist point 2 in Trio number 1
Algorithm 1	**Function** READC3D$(c3d)$
$points$	All Points in a C3D file
num_points	Number of Points
num_Labels	Number of Labels
num_frames	Number of Frames
Algorithm 2	**Function** INLIERDETECTION $(C3D)$
$detectedpoints$	Detected points
$intersectFlag$	Flag that shows if there is any intersection in trajectories
$mergedTrajectory$	Merged trajectory
$gapfilledTrajectory$	Gap filled trajectory
$assignedLabel$	Assigned label ID
$removedLabels$	IDs of Removed Labels
$newLabels$	New Labels vector
$newPoints$	New Points
$newC3D$	New C3D file
Algorithm 2	**Function** VECTORIZETRIO$(points, ID_1, ID_2, ID_3)$
$frame$	Frame
$\overrightarrow{P_1}$	Vector P_1: a vector between two points in 3D space

$P_{1_{mag}}$	Magnitude of vector P_1: Distance between two points
Algorithm 2	**Function** $\textsc{Dist}(points, ID_1, ID_2, selectedFrame)$
d_{sel}	Distance between two points in a selected frame
d_{mean}	Mean distance between two points across all frames
d_{std}	SD of distances between two points across all frames
Algorithm 2	**Function** $\textsc{AngleTrio}(points, ID_1, ID_2, ID_3)$
$\theta_{1_{mean}}$	Mean angle between three points across all frames
$\theta_{1_{std}}$	SD of angles between three points across all frames
Algorithm 4	**Function** $\textsc{PelvisDetection}(points)$
$probablePelvisPoints$	Probable pelvis points
$\mathtt{C}(points, 3)$	Combinations of three points
$probablePelvisTriplets$	Probable pelvis triplets = $\mathtt{C}(probablePelvisPoints, 3)$
$probTrio$	Probable Trios
num_Trios	Number of Trios
$frame_c$	Common frames for each Trio
f_{cs}	The first frame in common frames of a Trio
f_{cs}	The last frame in common frames of a Trio
Sp_{cs}	Spine point in the first frame of common frames of a Trio
A, B, C	Point 1, Point 2, and Point 3 in a Trio (Triangle)
AB, BC, AC	Vectors between point A and B, B and C, A and C
AB_{cs}	Vector AB in f_{cs}

d_{cs1}	Distance between two points in f_{cs} of Trio 1
$frame_{maj}$	The frame in which the majority axis was calculated
$flag_{maj}$	Flag for Majority Axis Orientation Criterion Met
d_{MWW}	Distance between left and right waist points in marker set
$pelvisPoints$	Pelvis points: Sp, RW, LW
$newC3D_{pelvis}$	New C3D file with a cleaned and labeled pelvis points
Algorithm 5	**Function** PCAMAJORITYAXIS$(points, frame_{maj}, Sp, W_1, W_2)$
$centroid$	Mean position of all points in 3D space in each frame
$pcaCoefficients$	PCA coefficients
$direction_{maj}$	Direction of majority axis = first PCA coefficient
$distW_1ToAxis_{maj}$	Distance of Waist point 1 to majority axis
$distW_2ToAxis_{maj}$	Distance of Waist point 2 to majority axis
$W_1W_2Symmetry_{maj}$	$distW_1ToAxis_{maj} - distW_2ToAxis_{maj}$
Δd_{SW}	Δ (distance between spine and waist point)
Δd_{maj}	Δ (distance between spine and majority axis)
Δsym_{maj}	Δ (distance between waists and majority axis)
$SWWLine$	Line connects spine to $midpoint\ of$ waistline
$\theta_{SWW_{maj}}$	angle between SWWLine and majority axis
$\theta_{WW_{maj}}$	angle between waistline and majority axis
$\Delta\theta_{SWW_{maj}}$	Δ (angle between SWWLine and majority axis)
$\Delta\theta_{WW_{maj}}$	Δ (angle between waistline and majority axis)

$spineFlag$	Flag indicating if spine orientation criterion is met
$symmetryFlag$	Flag for Waist Symmetry with Major Axis
$\theta_{WW_{maj}}Flag$	Flag indicating if waistline angle criterion is met
Algorithm 6	**Function** BODYSIDEDETECTION$(Sp, W_1, W_2, frame_{side})$
$frame_{side}$	$frame_{side} \leftarrow [f_{cs}, f_{ce}]$ in each Trio
$translationVector$	Translation vector
$translationMatrix$	Translation matrix
$W1_{cs}, W2_{cs}$	Waist point 1 and Waist point 2 in f_{cs} *of a Trio*
$W1_{ce}, W2_{ce}$	Waist point 1 and waist point 2 in f_{ce} *of a Trio*
$homogeneousW1_{ce}$	Homogeneous $W1_{ce}$
$translatedW1_{ce}$	Translated $homogeneousW1_{ce}$
$\mathrm{distW1_{cs}transW1_{ce}}$	Distance between $W1_{cs}$ and $traslatedW1_{ce}$
$\mathrm{distW2_{cs}transW1_{ce}}$	Distance between $W2_{cs}$ and $traslatedW1_{ce}$
Algorithm 7	**Function** SHOULDERDETECTION$(points, Sp, RW, LW, \Delta d, \Delta d_{sh}, \Delta\theta_{sh})$
$probableShoulderPoints$	Probable Shoulder Points
$\mathrm{C}(points, 2)$	Combinations of two points
$probShoulderPairs$	Probable shoulder pairs = $\mathrm{C}(probableShoulderPoints, 2)$
$probPair$	Probable Shoulder pairs
num_Pairs	Number of pairs
$shoulderPair$	Pairs of detected shoulder points
$newC3D_{shoulder}$	New C3D with cleaned and labeled shoulder points
Chapter 5	

FP	False Positive
FN	False Negative
TN	True Negative
F_1	F_1 score
Av	Average
E	Extraneous marker
D	Dataset (e.g., D1 = Dataset number 1)

MATLAB Functions

$size(V, dim)$	Length of dimension dim of a vector V
$norm(V)$	Euclidean norm (magnitude) of vector V
$pdist(set)$	Euclidean distance between pairs of observations in a set
$isoutlier(Arr, method)$	Find outliers in data using a specific method (e.g., “mean”)
$atan2d(y, x)$	Four-quadrant inverse tangent ($\tan^{-1}$) of y and x coordinates
$cross(V1, V2)$	Cross product of vector V1 and vector V2
$dot(V1, V2)$	Dot product of vector V1 and vector V2
$isnan(Array)$	Determine which array elements are NaN
$std(V)$	SD(V): Standard Deviation of observations vector V
$mean(Array)$	Average or mean value of an array

1. Introduction and Overview

1.1 Background and Statement of the Problem

The global market for 3D motion capture (MoCap) [1] technology is expected to achieve a valuation of USD 261.17 million by the year 2026 [2]. Among the various types of MoCap technologies available [3], marker-based optical motion capture (OMC) [4]–[7] technology has long been regarded as the ‘gold standard’ in terms of reliability and accuracy [3], [8]–[13]. OMC systems employ cameras to track the 3D positions of reflective markers placed at specific anatomical locations on the subject’s body. These systems are widely used in various fields such as medicine [14], [15], biomedical engineering [16], sports [9], [17], [18], ergonomics [19], [20], gait analysis [21], [22], robotics [23]–[28], biomechanics [29]–[32], gaming [33], [34], virtual/augmented reality [35], and computer graphics animation [36]–[41]. OMC applications [3] include diagnosing disorders [42], rehabilitation [43], [44], athlete performance analysis [45], mimicking human movements by robots [46]–[49], analyzing the impact of various activities on the human body [50], [51], and creating realistic animations [52], [53].

1.1.1 OMC Fundamentals

The 3D positions of the markers (i.e., the markers' orb centers) are captured at a specific frame rate (capturing frequency) and saved in various MoCap file formats [6] such as C3D [54]. The trajectory of a marker represents its movement path tracked in 3D space over time [55]. The position of this trajectory at any given moment is denoted by its X, Y, and Z coordinates. The “fill level” [56] represents the visibility of the marker’s trajectory within the measurement range, calculated by dividing the total number of tracked frames for a marker by the total captured frames and expressing it as a percentage. OMC markers can be passive (reflect the camera's light) or active (are battery-powered and emit their own light) [7], [13], [57], [58].

1.1.1.1 Passive Markers Characteristic: "Reappearing" Markers

In MoCap data, when a specific physical marker is tracked for the first time in its initial frame, it is assigned a random ID number. There is a distinction between active and passive markers in terms of their assigned IDs under occlusion conditions. Each active marker has a unique ID number that remains consistent [59], [60] even if it disappears and reappears due to occlusion, resulting in a distinct trajectory with potential gaps (Figure 1-1).

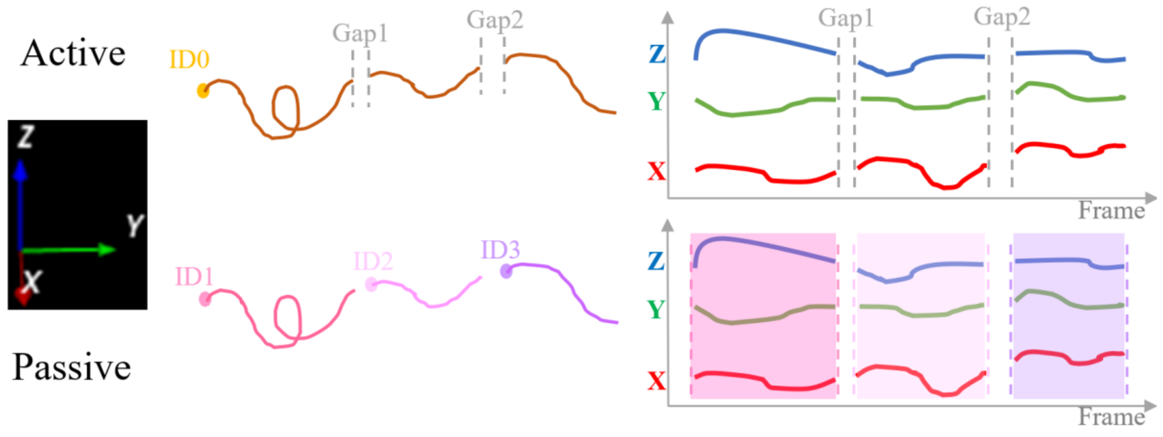


Figure 1-1: The effect of occlusion on Passive and Active Markers' Trajectories

When a passive marker, like ID1, disappears due to occlusion and then reappears, it is typically given a new ID, such as ID2, [59] since the passive system cannot recognize that these two points are the same. This leads to passive markers commonly having short segment trajectories (Figure 1-1). As a result, a MoCap file may contain multiple points with their corresponding trajectories associate to a specific physical marker. Therefore, post processing is needed to identify and merge the short segments associated with a specific physical marker to form its complete trajectory with gaps. While previous studies have described this characteristic without a specific term [12], [59], [61] or referred to it as "instances of occluded markers [16]," in this thesis, we term it as "reappearing markers."

1.1.1.2 OMC Errors and Challenges

OMC systems are prone to errors and noise [6] stemming from various factors [8], [62] such as calibration problems [63], environmental conditions [18], and occlusions [64]. Common types of noise in MoCap data include phantom markers [65] (which consist of outliers [59] and ghost markers [66]), extraneous markers [67], marker swapping [68] (i.e., mislabeling [6]), markers overlapping [69] (i.e., overlying [6]), and missing markers [70]. Phantom markers are not real markers; they arise from reflective surfaces or direct light sources, and they can be categorized into outliers and ghost markers. Outliers are inaccurate measurements that significantly deviate from the expected values. Ghost markers are virtual markers located close to valid markers. Extraneous markers are actual markers belonging to other objects. Marker swapping occurs when two markers' labels are switched due to their proximity or crossing paths in front of the camera, leading to misinterpretation by the capture system. Overlapped markers refer to points that are closer together than the marker size or marker measurement accuracy. Moreover, occlusion resulting from reduced visibility or self-occlusion, as well as markers dropped from the body, can lead to missing markers. This situation creates gaps in active markers' trajectories or generates reappearing markers in passive markers. Therefore, in the captured data file, there may be *M* points in each frame, where *M* can be less than, equal to, or more than *N* physical markers attached to the subject. Addressing these errors is crucial for further processing of MoCap data.

Marker set configuration and marker quantity also affect the MoCap data analysis [52], [69], [71], [72]. Increasing the number of markers can improve precision and reduce occlusion issues, but drawbacks include longer setup times, subject discomfort, high computational load, and potential marker interference. While a sparse dataset may reduce these drawbacks [72]–[74] it may compromise accuracy and worsen occlusion issues. Large marker spacing can also present challenges for statistical outlier detection.

1.1.1.3 MoCap Solving and Challenges

To utilize MoCap data in various applications such as action recognition [75], motion analysis [76], and pose estimation [77], the data must undergo a process known as MoCap "solving" [59], [78]–[82]. This process includes cleaning [83], [84] (denoising [8] and recovery of missing markers [85]), alignment [86], and labeling [87]. The goal of MoCap solving is to transform raw, unlabeled, noisy mocap data into cleaned, labeled data. In this processed data, each frame contains only *N* unique labeled points corresponding to *N* physical markers attached to the subject. Each point follows a distinct complete trajectory across all captured frames.

"Manual cleaning" [77]–[79] is commonly utilized in MoCap solving, where a technician opens a MoCap file like a C3D file using suitable software such as open-source Mokka software [88], motion kinematic and kinetic analyzer, or commercial options like Qualisys Track Manager (QTM) [89] and Vicon Nexus [90]. The technician then proceeds to rectify errors frame by frame. Despite the automatic labeling functions in these commercial software tools (e.g., AIM; Automatic Identification of Markers [91] in QTM or Vicon Nexus [92]), aids in manual cleaning, manual intervention remains necessary [86]. Hence, due to the time-consuming and laborious [78] manual cleaning process, there is a high demand for automated approaches [80].

Early automated MoCap solving methods struggled to generalize to real-world data due to their reliance on assumptions, constraints, empirical parameters, and hand-crafted features [93]–[96]. To overcome these limitations, data-driven approaches such as machine learning and deep learning have been employed [78]–[83]. However, the scarcity of cleaned and labeled MoCap data [59] hinders the effectiveness of these methods as they require large training datasets. This scarcity is due to the high cost and restrictions of traditional MoCap systems [97], as well as the time-consuming nature of manual cleaning MoCap data. It is important to highlight that the Motion-X [98] dataset, which comprises 144.2 hours of motion data, alongside the AMASS [99] dataset—recognized as the largest existing OMC dataset with 45 hours of data—are both considerably smaller in scale compared to the video datasets commonly employed in the OMC field [59]. Some techniques have used simulated data [59], [67] to address this issue, but they still necessitate manual intervention and may struggle with generalizing to unseen data, even when utilizing diverse motion datasets [59] like AMASS.

1.1.2 Underwater OMC

Underwater OMC technology enables the capture of underwater biomechanical movements for various applications [100]–[103] including underwater biomechanics [104]–[109], swimming and sports performance analysis [45], [110]–[121], rehabilitation utilizing underwater treadmills and gait analysis [122]–[125], underwater robotics [126]–[135] and underwater animation, filming, and virtual reality [136]–[139]. Beyond human applications, these systems also extend their utility to non-human subjects [140], [141] including underwater animal biomechanical analysis [142], [143], tracking marine vessels and structures [144][145], observing movements of oil pipelines [146], monitoring underwater autonomous vehicles [147]–[149], and analyzing objects in towing tanks [150].

1.1.2.1 Underwater OMC Challenges

Capturing underwater OMC data faces increased challenges compared to other environments [112], [123], [151]–[158]. Factors such as surface reflections [11], [151], [152] and the unique properties of water result in various types of noise, which compromise data accuracy [18] and reliability. Water's different optical properties [159]–[161] compared to air can distort captured data, resulting in inaccuracies in marker positions and the presence of ghost markers. The presence of bubbles [34], [71], [151], [152], [162] all around the swimmer and water's ability to absorb light [152], [154] also impacts the quality of optical markers detection. Additionally, suspended particles and impurities in water scatter light [154], further contributing to noisy data. The behavior of light underwater differs significantly from that in air, leading to reduced visibility [102], [151], resulting in high occlusion. This high occlusion causes missing or incorrectly tracked data, making passive marker tracking more challenging. Water's unpredictable currents, waves, and turbulence [154], [157] can cause subjects to move erratically, resulting in motion blur, outliers, and difficulties in tracking markers. Moreover, the use of waterproof equipment introduces technical limitations that may affect marker detection and data capture accuracy.

Marker displacement or detachment in water, especially for swimmers, can occur due to movement, water resistance, and friction between the skin and garment. Securely attaching markers to a person to prevent detachment during motion significantly increases setup time [7], [10], [71]. Consequently, re-tracking [6], [59] to reduce errors and decrease editing time, a common practice in on-land MoCap [8], becomes highly time-consuming for underwater MoCap. Therefore, underwater MoCap may be very noisy, making post-cleaning more challenging than in other environments.

The sparsity of underwater MoCap datasets presents a significant challenge, leading to unique difficulties in cleaning underwater MoCap data due to their inherent noisiness compared to other environments. Increasing the number of markers can extend setup time and impact the swimmer's performance [71]. Conversely, reducing the number of markers and working with sparse datasets that have large marker spacing can also complicate statistical outlier detection, especially in the presence of a higher proportion of outliers relative to the small number of valid markers in a frame.

Freestyle [157] and complex [163] movements underwater pose additional challenges such as creating severe self-occlusion. Also, the kinematics of movement are different in water conditions compared to land movements due to properties of water such as higher density and buoyancy [123], [124]. Buoyancy in water allows for effortless floating and deep diving due to a microgravity environment [103], enabling a wider range of motion and freedom of movement not achievable on land. Some underwater movements mimic those of aquatic animals, like dolphin kicks or turtle-like treading motions [114], [131].

Underwater MoCap solving techniques face greater challenges due to the aforementioned issues and the scarcity of swimming and underwater MoCap data. This scarcity is attributed to the distinctive characteristics of aquatic environments. Even image-based MoCap techniques face difficulties when dealing with aquatic data [97]. To address this issue, the SwimXYZ [97] synthetic dataset was developed to enhance the applicability of image-based MoCap methods in swimming. It comprises synthetic monocular videos that are meticulously annotated with accurate ground truth 2D and 3D joint information. The dataset includes a total of 11,520 videos, amounting to 3.4 million frames. These videos exhibit variations in camera angles, subjects, water conditions, lighting scenarios, and types of motion. Additionally, SwimXYZ offers 240 synthetic swimming motion sequences in SMPL format [164], showcasing diverse body shapes and movements.

However, to the best of our knowledge, there are currently no OMC datasets publicly available for underwater swimming actions, which includes intricate movements like treading water, orientations that are impossible out of water (e.g., floating on or under the water), or unpatterned maneuvers. Even datasets containing actions at the surface of the water (e.g., swimming strokes such as backstroke, breaststroke, butterfly, and front crawl) are rare and limited in size, and these actions are not entirely underwater [165].

1.1.2.2 Underwater Qualisys Miqus M5U MoCap Camera

Underwater OMC systems are in high demand due to their capability to efficiently capture and digitize motion compared to traditional video methods that require extensive time for manually digitizing anatomical landmarks through image-based techniques [166].

In 2009, Qualisys launched the first commercially available underwater MoCap camera [167], leading to a reduction in measurement and analysis time compared to video techniques [166]. Once calibrated, these systems accurately track 3D marker positions. A

2009 study [166] assessed the accuracy of the Qualisys underwater passive marker system for swimming stroke tracking, revealing a low RMS error in angle measurement (0.2°) and an average RMS error of 1.2 mm over three lengths, which is negligible for biomechanical analyses. The data from Qualisys closely aligned with that of a land-based motion analysis passive marker system using markers on an L-shaped frame. Qualisys released a new underwater MoCap camera, 7+u, in 2019 [167]. This series [168], known as Oqus [169], [170], has since been discontinued and replaced by the newer Arqus model [171]. Researchers have evaluated the accuracy of Qualisys underwater MoCap cameras [18], [121], [162], [172]. The most recent study [151] assessed the underwater human movement error using six Qualisys 7+ cameras that established an underwater capture volume of 8x2x2 meters. All cameras were synchronised, recording marker locations at a frequency of 100Hz. The calibration error, as provided by Qualisys QTM software, was -1.82 mm. Average error levels were found to be acceptable in two trials (1.23mm ± 8.23mm and 1.34mm ± 9.65mm), although errors increased at the ends and top of the capture domain. By concentrating on a specific area with higher accuracy, the error was minimized to 0.53mm (± 1.45mm).

However, these cameras have been too large to fit into the smaller capture volumes required for applications like aquatic treadmills with shorter distances between the pool walls and the subject, and they need a wider viewing angle [100], [122]. The increasing demand for applications such as aquatic therapy [123], [124] in clinical rehabilitation, emphasizing its advantages over land-based therapies, along with the benefits of exercise using an underwater treadmill [123], [124], has driven the development of specialized underwater MoCap cameras.

On May 16, 2019, Qualisys unveiled the Miqus M3u and M5u underwater MoCap cameras, the smallest ever MoCap solutions designed specifically for underwater measurement [167]. These cameras, along with the Miqus underwater color video camera [173], provide high-resolution, high-speed, and extremely low latency capabilities for accurate underwater measurements in confined spaces [140]. The new Miqus U cameras offer a wider field-of-view and lighter weight compared to previous models [174], making them easier to deploy in smaller tanks or pools. This advancement has opened up new possibilities for gait analysis and underwater rehabilitation [100], as well as animation of realistic underwater motion [167]. The Miqus M5U offers the highest resolution, and a maximum capture distance of 17 meters [174]. Qualisys cameras above water reflect infrared light. In contrast, the Miqus M5U emits visible light at 455 nm to illuminate passive markers, as infrared light is absorbed in water. The strobe light from the Miqus M5U appears blue to the naked eye. Currently, Qualisys underwater markers are passive and do not include active marker devices for underwater use.

1.2 Thesis Objectives

The aim of this thesis is to analyze underwater OMC using Qualisys Miqus M5U MoCap cameras with passive markers. This research addresses the aforementioned limitations, such as the lack of an underwater MoCap dataset and the need for automatic MoCap solving approaches, in order to streamline the time-consuming manual cleaning process.

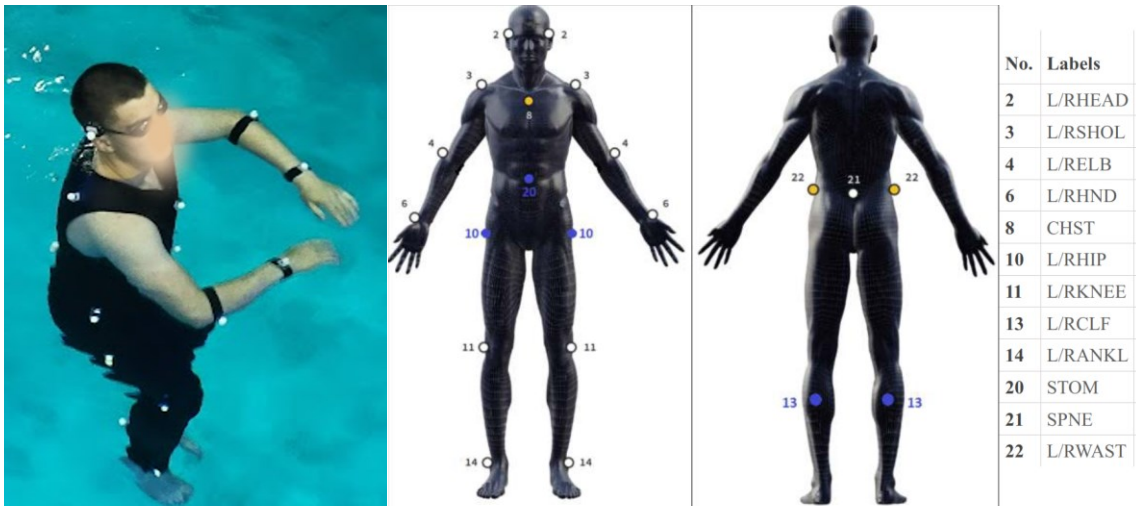
The proposed algorithms are versatile and applicable to OMC systems using passive or active markers, making them suitable for a wide range of actions. In this thesis, we apply them in one of the most challenging scenarios of underwater sparse freestyle MoCap using passive markers, which, to the best of our knowledge, has not been studied before.

1.3 System Overview

In this section, we will first introduce the captured data. Following that, we will describe the thesis chapters, which encompass our developed algorithms and contributions.

1.3.1 Capturing Underwater MoCap Data

We captured underwater MoCap data using seven Qualisys Miqus M5U underwater MoCap cameras installed at different locations around a four-meter-deep, 18 metre by 14 meter pool at the Memorial University Marine Institute Offshore Safety and Survival Centre. Although the Qualisys sports marker set [175] recommends a minimum of 41 markers, we opted to use only 21 reflective passive markers to explore the challenges associated with a sparse marker set. Figure 1-2, shows the locations of these markers and their corresponding label names. After calibration, data were recorded at 100Hz using QTM software and exported to .c3d files. Our QTM software version is “2022.2 build 7710”.



No.	Labels
2	L/RHEAD
3	L/RSHOL
4	L/RELB
6	L/RHND
8	CHST
10	L/RHIP
11	L/RKNEE
13	L/RCLF
14	L/RANKL
20	STOM
21	SPNE
22	L/RWAST

Figure 1-2: Marker set: Location of 21 passive markers and their labels.

1.3.2 Thesis Structure

This thesis comprises six chapters, presented in manuscript style, with titles listed in Table 1-1. The details of each chapter will be introduced in the following lines.

Table 1-1: Thesis Structure

Chapters	Title
1	Introduction and Overview
2	A Survey on Solving Marker-based Motion Capture Approaches
3	Marker-based Underwater Optical Motion Capture Data Preparation
4	Semi-supervised Geometry-Based Labeling of Sparse Underwater Optical MoCap Data
5	Deep Learning based Auto-Labelling Underwater Sparse Freestyle MoCap Data
6	Conclusion and Future Work

Chapter 1 presents the background and problem statement of OMC systems, focusing on underwater OMC. It covers the fundamentals, applications, and challenges faced by these systems. Furthermore, it introduces the captured data and outlines the thesis structure, detailing our developed algorithms, innovations, and contributions to the field.

Chapter 2 conducts a comprehensive literature review on MoCap solving approaches, which encompasses denoising, recovery, alignment, and auto-labeling approaches.

Chapter 3 outlines the preparation steps for marker-based underwater OMC systems. This includes setting up cameras, attaching markers to the subject, calibrating the cameras, recording the session, creating a marker set, and MoCap manual cleaning process using QTM software. It also describes features of this software, such as the AIM model facilitating the manual cleaning process. The cleaned C3D data file produced can serve as ground truth or a training dataset for the deep learning auto-labeling algorithm discussed in Chapter 5. Additionally, this chapter addresses the challenges of MoCap manual cleaning, emphasizing the need for automated algorithms to simplify this tedious process.

Chapter 4 demonstrates a novel semi-supervised geometry based MoCap labeling algorithm which we developed to streamline the laborious manual cleaning procedure described in Chapter 3. This system extracts distances and angles from a marker set to identify valid labels and applies an innovative extraneous removal algorithm based on the difference of the norms, along with other denoising and outliers' removal methods utilizing norm, velocity, acceleration, and jerk profiles. Additionally, it includes a method for recovering missing or dropped markers and a pelvis detection algorithm based on Principal Component Analysis (PCA) [176]. Moreover, side detection is employed to identify corresponding labels for reappearing markers that result from the use of passive markers. The evaluation was conducted visually, achieving 100% accuracy in detecting valid markers despite the presence of outliers, extraneous markers, ghost markers, and missing or dropped markers due to occlusion. We will explore the use of more ghost markers and enhance our pelvis detection procedure with automatic side detection.

Chapter 5 proposes a deep-learning based auto-labeling algorithm that accurately labels MoCap data utilizing Long short-term memory (LSTM) [177], employing Hungarian analysis [178] for label assignment and Procrustes analysis [179] to assign labels to unlabeled data in a post-processing step. The cleaned MoCap data generated by the algorithm described in Chapter 4 serves as both the ground truth and the training dataset. However, this dataset is insufficiently large to effectively train the deep learning network. To tackle the challenge posed by a limited training dataset, two distinct strategies are employed. The first approach involves data augmentation, which enhances the existing cleaned dataset by introducing random noise and gaps. The second strategy involves generating simulated trajectories, which are then combined with real data using transfer learning [180]. These synthetic datasets are produced utilizing a marker set that is developed with the OpenSim software [181], a tool designed for modeling musculoskeletal structures and simulating dynamic movements. The core algorithm in Chapter 5 builds upon the source code from [67], with several key enhancements. Notably, we replace their manual data alignment method with a PCA-based pelvis detection technique introduced in Chapter 4. Clouthier *et al.* [67] also faced accuracy issues due to extraneous markers; we resolve this by implementing the extraneous marker removal algorithm from Chapter 4, boosting our accuracy from 66% to 98%. Additionally, we streamline the input for the LSTM model from five to three data points, using only the 3D relative positions of markers instead of including velocity and acceleration.

Chapter 6 summarizes the findings of the thesis and proposes avenues for future research expansion. Upcoming work will involve collecting more underwater MoCap data to assess our algorithm's performance in different underwater actions and noises. The feasibility of adding ghost markers near valid markers will also be evaluated. There are also plans to conduct freestyle underwater action recognition using labeled data, as well as exploring a more robust solution for body side detection within a subset of pelvis detection.

1.4 Thesis Contributions' Summary

In summary, this thesis outlines the following technical contributions:

1. The first cleaned and labelled underwater freestyle swimming MoCap datasets using Qualisys QTM software and an AIM model;
2. A characterization of the challenges of utilizing a sparse MoCap Dataset, a topic rarely explored in previous research;
3. A comprehensive literature review on MoCap solving approaches;
4. A semi-supervised geometry-based labeling algorithm, incorporating an innovative norm-based denoising method that considers velocity, acceleration, and jerk profiles;
5. An innovative extraneous removal algorithm based on the difference of the norms;
6. A novel pelvis detection technique using PCA, implemented along with a method to recover dropped markers; and
7. An auto-labeling algorithm based on LSTM, the Hungarian label assignment, and Procrustes alignment, incorporating a geometry-based method for initial alignment, ground truth and training set creation, and an enhancement of the accuracy of the results.

1.5 Co-authorship Statement

I am the principal author of all manuscripts presented in this thesis, including the thesis as a whole. I developed the methods and analyzed the results in all manuscripts, with Dr. Stephen Czarnuch providing guidance, revisions and conceptualizing the study. The underwater motion data were captured using seven Qualisys Miqus M5U underwater cameras installed at different locations around a four-meter-deep, 18-metre by 14-meter pool at the Memorial University Marine Institute Offshore Safety and Survival Centre.

References

[1] M. R. Das and R. R. A, “A Review on Human Motion Capture,” *SSRN Electron. J.*, pp. 151–157, 2021, doi: 10.2139/ssrn.3794164.

[2] Proquest, “Global 3D Motion Capture Market,” 2020. https://www.proquest.com/docview/2347080425/fulltext/CFAA8230641F4C15PQ/1?accountid=12378&sourcetype=Wire Feeds (accessed Jul. 26, 2024).

[3] M. Menolotto, D. S. Komaris, S. Tedesco, B. O’flynn, and M. Walsh, “Motion capture technology in industrial applications: A systematic review,” *Sensors (Switzerland)*, vol. 20, no. 19, pp. 1–25, 2020, doi: 10.3390/s20195687.

[4] G. B. Guerra-filho, “Optical motion capture: Theory and implementation,” *J. Theor. Appl. Informatics*, vol. 12, pp. 61--89, 2005, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.7248

[5] J. F. Nunes, P. M. Moreira, and J. M. R. S. Tavares, “Human motion analysis and simulation tools: A survey,” *Handb. Res. Comput. Simul. Model. Eng.*, pp. 359–387, 2015, doi: 10.4018/978-1-4666-8823-0.ch012.

[6] M. Kitagawa and B. Windsor, *MoCap for Artists Workflow and Techniques for Motion Capture*, no. 0. Elsevier Inc, 2008.

[7] E. Batis and M. Bylund, “Designing a Tool for Assisting in the Setup of Optical Motion Capture Systems,” 2017.

[8] P. Skurowski and M. Pawlyta, “Detection and Classification of Artifact Distortions in Optical Motion Capture Sequences,” *Sensors*, vol. 22, no. 11, pp. 1–29, 2022, doi: 10.3390/s22114076.

[9] X. Suo, W. Tang, and Z. Li, “Motion Capture Technology in Sports Scenarios: A Survey,” *Sensors*, vol. 24, no. 9, pp. 1–15, 2024, doi: 10.3390/s24092947.

[10] L. Wade, L. Needham, P. McGuigan, and J. Bilzon, “Applications and limitations of current markerless motion capture methods for clinical gait biomechanics,” *PeerJ*, vol. 10, pp. 1–27, 2022, doi: 10.7717/peerj.12995.

[11] N. Giulietti, A. Caputo, P. Chiariotti, and P. Castellini, “SwimmerNET: Underwater 2D Swimmer Pose Estimation Exploiting Fully Convolutional Neural Networks,” *Sensors*, vol. 23, no. 4, pp. 1–17, 2023, doi: 10.3390/s23042364.

[12] M. H. Song and R. I. Godøy, “How fast is your body motion? Determining a sufficient frame rate for an optical motion tracking system using passive markers,” *PLoS One*, vol. 11, no. 3, pp. 1–14, 2016, doi: 10.1371/journal.pone.0150993.

[13] S. C. Puthenveetil *et al.*, “Comparison of Marker-Based and Marker-Less Systems for Low-Cost Human Motion Capture,” vol. 2B:, no. 33rd Computers and Information in Engineering Conference, 2013, doi: https://doi.org/10.1115/detc2013-12653.

[14] I. A. Mesquita, P. F. P. da Fonseca, A. R. V. Pinheiro, M. F. P. Velhote Correia, and C. I. C. da Silva, “Methodological considerations for kinematic analysis of upper limbs in healthy and poststroke adults Part II: a systematic review of motion capture systems and kinematic metrics,” *Top. Stroke Rehabil.*, vol. 26, no. 6, pp. 464–472, 2019, doi: 10.1080/10749357.2019.1611221.

[15] G. Giarmatzis *et al.*, “Understanding Post-Stroke Movement by Means of Motion Capture and Musculoskeletal Modeling: A Scoping Review of Methods and Practices,” *BioMed*, vol. 2, no. 4, pp. 409–421, 2022, doi: 10.3390/biomed2040032.

[16] I. Takeda, A. Yamada, and H. Onodera, “Artificial Intelligence-Assisted motion capture for medical applications: a comparative study between markerless and passive marker motion capture,” *Comput. Methods Biomech. Biomed. Engin.*, vol. 24, no. 8, pp. 864–873, 2021, doi: 10.1080/10255842.2020.1856372.

[17] S. Noiumkar and S. Tirakoat, “Use of optical motion capture in sports science: A case study of golf swing,” *Proc. - 2013 Int. Conf. Informatics Creat. Multimedia, ICICM 2013*, pp. 310–313, 2013, doi: 10.1109/ICICM.2013.58.

[18] E. van der Kruk and M. M. Reijne, “Accuracy of human motion capture systems for sport applications; state-of-the-art review,” *Eur. J. Sport Sci.*, vol. 18, no. 6, pp. 806–819, 2018, doi: 10.1080/17461391.2018.1463397.

[19] S. Salisu, N. I. R. Ruhaiyem, T. A. E. Eisa, M. Nasser, F. Saeed, and H. A. Younis, “Motion Capture Technologies for Ergonomics: A Systematic Literature Review,” *Diagnostics*, vol. 13, no. 15, pp. 1–16, 2023, doi: 10.3390/diagnostics13152593.

[20] F. Rybnikár, I. Kaˇcerová, P. Hoˇrejší, and M. Šimon, “Ergonomics Evaluation Using Motion Capture Technology—Literature Review,” 2023.

[21] N. Rizaldy, F. Ferryanto, A. Sugiharto, and A. I. Mahyuddin, “Evaluation of action sport camera optical motion capture system for 3D gait analysis,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1109, no. 1, p. 012024, 2021, doi: 10.1088/1757-899x/1109/1/012024.

[22] R. M. Kanko, E. K. Laende, E. M. Davis, W. S. Selbie, and K. J. Deluzio, “Concurrent assessment of gait kinematics using marker-based and markerless motion capture,” *J. Biomech.*, vol. 127, 2021, doi: 10.1016/j.jbiomech.2021.110665.

[23] O. E. Ramos, N. Mansard, O. Stasse, C. Benazeth, S. Hak, and L. Saab, “Dancing

humanoid robots," *IEEE Robot. Autom. Mag.*, vol. 22, no. 4, pp. 16–26, 2015, doi: 10.1109/MRA.2015.2415048.

[24] I. Maroger, O. Stasse, and B. Watier, "Walking human trajectory models and their application to humanoid robot locomotion," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 3465–3472, 2020, doi: 10.1109/IROS45743.2020.9341118.

[25] K. Miura *et al.*, "Robot motion remix based on motion capture data - Towards human-like locomotion of humanoid robots," *9th IEEE-RAS Int. Conf. Humanoid Robot. HUMANOIDS09*, pp. 596–603, 2009, doi: 10.1109/ICHR.2009.5379535.

[26] L. González, J. C. Álvarez, A. M. López, and D. Álvarez, "Metrological evaluation of human–robot collaborative environments based on optical motion capture systems†," *Sensors*, vol. 21, no. 11, 2021, doi: 10.3390/s21113748.

[27] M. Field, D. Stirling, F. Naghdy, and Z. Pan, "Motion capture in robotics review," *2009 IEEE Int. Conf. Control Autom. ICCA 2009*, pp. 1697–1702, 2009, doi: 10.1109/ICCA.2009.5410185.

[28] A. Ude, C. G. Atkeson, and M. Riley, "Programming full-body movements for humanoid robots by observation," *Rob. Auton. Syst.*, vol. 47, no. 2–3, pp. 93–108, 2004, doi: 10.1016/j.robot.2004.03.004.

[29] G. Nagymáté and R. M. Kiss, "Application of OptiTrack motion capture systems in human movement analysis," *Recent Innov. Mechatronics*, vol. 5, no. 1., pp. 1–9, 1970, doi: 10.17667/riim.2018.1/13.

[30] J. Ramberg, "Method development for capturing drivers posture," CHALMERS UNIVERSITY OF TECHNOLOGY, 2016. [Online]. Available: http://publications.lib.chalmers.se/records/fulltext/238933/238933.pdf

[31] B. Lewis, C. J. Nycz, G. S. Fischer, and K. K. Venkatasubramanian, *Authentication-based on biomechanics of finger movements captured using optical motion-capture*, vol. 11241 LNCS. Springer International Publishing, 2018. doi: 10.1007/978-3-030-03801-4_16.

[32] A. M. Aurand, J. S. Dufour, and W. S. Marras, "Accuracy map of an optical motion capture system with 42 or 21 cameras in a large measurement volume," *J. Biomech.*, vol. 58, pp. 237–240, 2017, doi: 10.1016/j.jbiomech.2017.05.006.

[33] C. Bregler, "Motion Capture Technology for Entertainment," *Online*, no. November, pp. 156–158, 2007.

[34] T. Baker, "The History of Motion Capture Within The Entertainment Industry," *Metropolia.fi*, pp. 15–20, 2020, [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/336908/taryn_mocap_thesis.pdf?sequence=2&isAllowed=y

[35] N. Sadoughi, Y. Liu, and C. Busso, "MSP-Avatar corpus: Motion capture recordings to study the role of discourse functions in the design of intelligent virtual agents," *2015 11th IEEE Int. Conf. Work. Autom. Face Gesture Recognition, FG 2015*, vol. 2015-Janua, pp. 1–6, 2015, doi: 10.1109/FG.2015.7284885.

[36] M. Y. Zhang, "Application of performance motion capture technology in film and television performance animation," *Appl. Mech. Mater.*, vol. 347–350, pp. 2781–2784, 2013, doi: 10.4028/www.scientific.net/AMM.347-350.2781.

[37] S. Salonen, "MOTION CAPTURE IN 3D ANIMATION," no. May, 2021.

[38] M. Shields, "The Way of Motion Capture: The Innovations of 'Avatar.'" https://filmschoolrejects.com/avatar-the-way-of-water-motion-capture/ (accessed

Jul. 01, 2024).

[39] A. Menache, *Understanding Motion Capture for Computer Animation (Second Edition)*. 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123814968000032

[40] B. Rosenhahn, K. R., and M. D., *Human Motion Understanding, Modelling, Capture and Animation*. 2008.

[41] R. Sengupta, “Was This Motion Captured ?,” 2011.

[42] U. G. Longo *et al.*, “Optical Motion Capture Systems for 3D Kinematic Analysis in Patients with Shoulder Disorders,” *Int. J. Environ. Res. Public Health*, vol. 19, no. 19, 2022, doi: 10.3390/ijerph191912033.

[43] A. C. Alarcón-Aldana, M. Callejas-Cuervo, and A. P. L. Bo, “Upper limb physical rehabilitation using serious videogames and motion capture systems: A systematic review,” *Sensors (Switzerland)*, vol. 20, no. 21, pp. 1–22, 2020, doi: 10.3390/s20215989.

[44] Z. Yang *et al.*, “A Novel Methodology for Extracting and Evaluating Therapeutic Movements in Game-Based Motion Capture Rehabilitation Systems,” *J. Med. Syst.*, vol. 42, no. 12, 2018, doi: 10.1007/s10916-018-1113-4.

[45] B. Pueo, J. M. Jimenez-olmedo, U. D. A. España, and B. Pueo, “Application of motion capture technology for sport performance analysis,” vol. 2041, pp. 241–247, 2017.

[46] M. Popescu, D. Mronga, I. Bergonzani, S. Kumar, and F. Kirchner, “Experimental Investigations into Using Motion Capture State Feedback for Real-Time Control of a Humanoid Robot,” *Sensors*, vol. 22, no. 24, pp. 1–12, 2022, doi:

10.3390/s22249853.

[47] S. Muench, J. Kreuziger, M. Kaiser, and R. Dillmann, “Robot Programming by Demonstration ({RPD}) - {U}sing Machine Learning and User Interaction Methods for the Development of Easy and Comfortable Robot Programming Systems,” *Proc. Intl Symp. Ind. Robot.*, no. June, pp. 685–693, 1994.

[48] M. Field, Z. Pan, D. Stirling, and F. Naghdy, “Human motion capture sensors and analysis in robotics,” *Ind. Rob.*, vol. 38, no. 2, pp. 163–171, 2011, doi: 10.1108/01439911111106372.

[49] L. Boutin, A. Eon, S. Zeghloul, and P. Lacouture, “An auto-adaptable algorithm to generate human-like locomotion for different humanoid robots based on motion capture data,” *IEEE/RSJ 2010 Int. Conf. Intell. Robot. Syst. IROS 2010 - Conf. Proc.*, pp. 1256–1261, 2010, doi: 10.1109/IROS.2010.5652230.

[50] G. Hernandez *et al.*, “Machine Learning Techniques for Motion Analysis of Fatigue from Manual Material Handling Operations Using 3D Motion Capture Data,” *2020 10th Annu. Comput. Commun. Work. Conf. CCWC 2020*, pp. 300–305, 2020, doi: 10.1109/CCWC47524.2020.9031222.

[51] F. Asadi and N. Arjmand, “Marker-less versus marker-based driven musculoskeletal models of the spine during static load-handling activities,” *J. Biomech.*, vol. 112, p. 110043, 2020, doi: 10.1016/j.jbiomech.2020.110043.

[52] M. Loper, N. Mahmoody, and M. J. Blackz, “MoSh: Motion and shape capture from sparse markers,” *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–13, 2014, doi: 10.1145/2661229.2661273.

[53] S. Sharma, S. Verma, M. Kumar, and L. Sharma, “Use of Motion Capture in 3D

Animation: Motion Capture Systems, Challenges, and Recent Trends,” *Proc. Int. Conf. Mach. Learn. Big Data, Cloud Parallel Comput. Trends, Prespectives Prospect. Com. 2019*, pp. 289–294, 2019, doi: 10.1109/COMITCon.2019.8862448.

[54] B. Motion, “The C3D File Format A Technical User Guide,” p. 134, 2021.

[55] Qualisys, “What is a trajectory.” https://www.qualisys.com/my/qacademy/#!/tutorials/what-is-a-trajectory (accessed Jun. 30, 2024).

[56] Qualisys, “Identifying gaps.” https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/filling_gaps_in_your_data/identifying_gaps.htm?Highlight=fill level (accessed Jul. 03, 2024).

[57] Qualisys, “High-quality passive & active mocap markers,” *Qualisys*. https://www.qualisys.com/accessories/markers/ (accessed Jan. 11, 2024).

[58] Qualisys, “Markers.” https://docs.qualisys.com/getting-started/content/4_what_comes_with_your_system/4_what_comes_with_your_qualisys_system/markers.htm (accessed Jul. 28, 2024).

[59] N. Ghorbani and M. J. Black, “SOMA: Solving Optical Marker-Based MoCap Automatically,” *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 11097–11106, 2021, doi: 10.1109/ICCV48922.2021.01093.

[60] E. Ceseracciu, “NEW FRONTIERS OF MARKERLESS MOTION CAPTURE APPLICATION TO SWIM BIOMECHANICS AND GAIT ANALYSIS,” 2011.

[61] E. Monaco, “Automatic Labelling of 3D Motion Capture Markers using Neural Networks,” Università degli Studi di Padova Dipartimento, 2022.

[62] P. Skurowski and M. Pawlyta, “On the noise complexity in an optical motion capture

facility," *Sensors (Switzerland)*, vol. 19, no. 20, pp. 1–30, 2019, doi: 10.3390/s19204435.

[63] A. R. Jensenius, K. Nymoen, S. A. Skogstad, and A. Voldsund, "A study of the noise-level in two infrared marker-based motion capture systems," *Proc. 9th Sound Music Comput. Conf. SMC 2012*, pp. 11–14, 2012.

[64] X. Chen and J. Davis, "Camera Placement Considering Occlusion for Robust Motion Capture," *Comput. Graph. Lab. Stanford Univ. Tech. Rep*, vol. 2, no. 2.2, p. 2, 2000, [Online]. Available: http://graphics.stanford.edu/papers/OcclusionMetric/occlusion_metric.pdf

[65] Qualisys, "Before calibrating." https://docs.qualisys.com/getting-started/content/8_calibration_series/8a_how_to_calibrate/before_calibrating.htm?Highlight=phantom (accessed Jul. 29, 2024).

[66] S. Alexanderson, C. O'Sullivan, and J. Beskow, "Real-time labeling of non-rigid motion capture marker sets," *Comput. Graph.*, vol. 69, pp. 59–67, 2017, doi: 10.1016/j.cag.2017.10.001.

[67] A. L. Clouthier, G. B. Ross, M. P. Mavor, I. Coll, A. Boyle, and R. B. Graham, "Development and Validation of a Deep Learning Algorithm and Open-Source Platform for the Automatic Labelling of Motion Capture Markers," *IEEE Access*, vol. 9, pp. 36444–36454, 2021, doi: 10.1109/ACCESS.2021.3062748.

[68] M. Perepichka, D. Holden, S. P. Mudur, and T. Popa, "Robust marker trajectory repair for MOCAP using kinematic reference," *Proc. - MIG 2019 ACM Conf. Motion, Interact. Games*, 2019, doi: 10.1145/3359566.3360060.

[69] P. Acevedo, B. Rekabdar, and C. Mousas, "Optimizing retroreflective marker set for

motion capturing props," *Comput. Graph.*, vol. 115, pp. 181–190, 2023, doi: 10.1016/j.cag.2023.07.021.

[70] P. Skurowski and M. Pawlyta, "Gap reconstruction in optical motion capture sequences using neural networks," *Sensors*, vol. 21, no. 18, pp. 1–26, 2021, doi: 10.3390/s21186115.

[71] G. Ascenso, "Development of a non-invasive motion capture system for swimming biomechanics," 2021.

[72] M. Schröder, J. Maycock, and M. Botsch, "Reduced marker layouts for optical motion capture of hands," *Proc. 8th ACM SIGGRAPH Conf. Motion Games, MIG 2015*, no. October, pp. 7–16, 2015, doi: 10.1145/2822013.2822026.

[73] G. Liu, J. Zhang, W. Wang, and L. McMillan, "Human motion estimation from a reduced marker set," *Proc. Symp. Interact. 3D Graph.*, vol. 2006, no. March, pp. 35–42, 2006, doi: 10.1145/1111411.1111418.

[74] S. Washino, D. L. Mayfield, G. A. Lichtwark, H. Mankyu, and Y. Yoshitake, "Swimming performance is reduced by reflective markers intended for the analysis of swimming kinematics," *J. Biomech.*, vol. 91, pp. 109–113, 2019, doi: 10.1016/j.jbiomech.2019.05.017.

[75] A. Gupta, J. Martinez, J. J. Little, and R. J. Woodham, "3D pose from motion for cross-view action recognition via non-linear circulant temporal encoding," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 2601–2608, 2014, doi: 10.1109/CVPR.2014.333.

[76] G. Hernandez, "USING DEEP LEARNING FOR MOTION ANALYSIS OF 3D MOTION CAPTURE DATA FOR FORECASTING MOTION AND FATIGUE,"

Texas State University, 2021.

[77] V. Joukov, J. F. S. Lin, K. Westermann, and D. Kulić, “Real-Time Unlabeled Marker Pose Estimation via Constrained Extended Kalman Filter,” *Springer Proc. Adv. Robot.*, vol. 11, pp. 762–771, 2020, doi: 10.1007/978-3-030-33950-0_65.

[78] K. Chen, Y. Wang, S. H. Zhang, S. Z. Xu, W. Zhang, and S. M. Hu, “MoCap-solver: A neural solver for optical motion capture data,” *ACM Trans. Graph.*, vol. 40, no. 4, 2021, doi: 10.1145/3450626.3459681.

[79] D. Holden, “Robust solving of optical motion capture data by denoising,” *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–12, 2018, doi: 10.1145/3197517.3201302.

[80] X. Pan *et al.*, “A Locality-based Neural Solver for Optical Motion Capture,” 2023, doi: 10.1145/3610548.3618148.

[81] G. Albanis, N. Zioulis, S. Thermos, A. Chatzitofis, and K. Kolomvatsos, “Noise-in, Bias-out: Balanced and Real-time MoCap Solving,” 2023, [Online]. Available: http://arxiv.org/abs/2309.14330

[82] J. Tang, L. Li, J. Hou, H. Xin, and X. Yu, “A Divide-and-conquer Solution to 3D Human Motion Estimation from Raw MoCap Data,” *Proc. - 2023 IEEE Conf. Virtual Real. 3D User Interfaces Abstr. Work. VRW 2023*, pp. 767–768, 2023, doi: 10.1109/VRW58643.2023.00226.

[83] U. Mall, G. R. Lal, S. Chaudhuri, and P. Chaudhuri, “A Deep Recurrent Framework for Cleaning Motion Capture Data,” no. Figure 1, 2017, [Online]. Available: http://arxiv.org/abs/1712.03380

[84] E. Martini, S. Member, A. Calanca, and N. Bombieri, “Denoising and Completion Filters for Human Motion Software : a Survey with Code,” pp. 0–14, 2023, doi:

10.36227/techrxiv.22956482.v1.

[85] H. Yasin, S. Ghani, and B. Kruger, “An Effective and Efficient Approach for 3D Recovery of Human Motion Capture Data,” *Sensors*, vol. 23, p. 3664, 2023, doi: 10.3390/s23073664.

[86] S. Ghorbani, A. Etemad, and N. F. Troje, *Auto-labelling of Markers in Optical Motion Capture by Permutation Learning*, vol. 11542 LNCS. Springer International Publishing, 2019. doi: 10.1007/978-3-030-22514-8_14.

[87] S. Xia, L. Su, X. Fei, and H. Wang, “Toward accurate real-time marker labeling for live optical motion capture,” *Vis. Comput.*, vol. 33, no. 6–8, pp. 993–1003, 2017, doi: 10.1007/s00371-017-1400-y.

[88] Biomechanical-toolkit.github.io, “Mokka,” *Biomechanical-toolkit.github.io*. https://biomechanical-toolkit.github.io/mokka/ (accessed Jan. 06, 2024).

[89] Qualisys, “Qualisys Track Manager,” *Qualisys*, 2011. https://www.qualisys.com/software/qualisys-track-manager/ (accessed Dec. 10, 2023).

[90] Vicon, “Vicon,” *Vicon*. https://www.vicon.com/ (accessed Dec. 10, 2023).

[91] Qualisys, “Using AIM models.” https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/using_aim_models/using_aim_models.htm (accessed Aug. 01, 2024).

[92] Vicon, “Automated labeling,” *Vicon*. https://documentation.vicon.com/nexus/v2.0/desktop/NexusWsN/Labeling/Automated_labeling.htm (accessed Jul. 06, 2024).

[93] A. Aristidou, J. Cameron, and J. Lasenby, “Real-time estimation of missing markers

in human motion capture," *2nd Int. Conf. Bioinforma. Biomed. Eng. iCBBE 2008*, pp. 1343–1346, 2008, doi: 10.1109/ICBBE.2008.665.

[94] S. Rajko and G. Qian, "Real-time automatic kinematic model building for optical motion capture using a markov random field," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4796 LNCS, pp. 69–78, 2007, doi: 10.1007/978-3-540-75773-3_8.

[95] M. Ringer and J. Lasenby, "A procedure for automatically estimating model parameters in optical motion capture," *Image Vis. Comput.*, vol. 22, no. 10 SPEC. ISS., pp. 843–850, 2004, doi: 10.1016/j.imavis.2004.02.011.

[96] J. Meyer, M. Kuderer, J. Muller, and W. Burgard, "Online marker labeling for fully automatic skeleton Tracking in optical motion capture," *Proc. - IEEE Int. Conf. Robot. Autom.*, no. May 2014, pp. 5652–5657, 2014, doi: 10.1109/ICRA.2014.6907690.

[97] G. Fiche, V. Sevestre, C. Gonzalez-Barral, S. Leglaive, and R. Séguier, "SwimXYZ: A large-scale dataset of synthetic swimming motions and videos," *Proc. - MIG 2023 16th ACM SIGGRAPH Conf. Motion, Interact. Games*, vol. 2030, pp. 1–10, 2023, doi: 10.1145/3623264.3624440.

[98] J. Lin *et al.*, "Motion-X: A Large-scale 3D Expressive Whole-body Human Motion Dataset," *Adv. Neural Inf. Process. Syst.*, vol. 36, no. NeurIPS, pp. 1–26, 2023.

[99] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. Black, "AMASS: Archive of motion capture as surface shapes," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-Octob, pp. 5441–5450, 2019, doi: 10.1109/ICCV.2019.00554.

[100] Qualisys, "Swimming," *Qualisys*. https://www.qualisys.com/life-

sciences/swimming/ (accessed Nov. 27, 2023).

[101] A. P. Silvatti, P. Cerveri, T. Telles, F. A. S. Dias, G. Baroni, and R. M. L. Barros, "Quantitative underwater 3D motion analysis using submerged video cameras: Accuracy analysis and trajectory reconstruction," *Comput. Methods Biomech. Biomed. Engin.*, vol. 16, no. 11, pp. 1240–1248, 2013, doi: 10.1080/10255842.2012.664637.

[102] F. A. Magalhaes, Z. Sawacha, R. Di Michele, M. Cortesi, G. Gatta, and S. Fantozzi, "Effectiveness of an automatic tracking software in underwater motion analysis," *J. Sport. Sci. Med.*, vol. 12, no. 4, pp. 660–667, 2013.

[103] Y. Bernal *et al.*, "Development of underwater motion capture system for space suit mobility assessment," *Proc. Hum. Factors Ergon. Soc.*, vol. 2017-Octob, pp. 945–949, 2017, doi: 10.1177/1541931213601718.

[104] C. Long *et al.*, "Lower Limb Muscle Activation in Young Adults Walking in Water and on Land," *Appl. Sci.*, vol. 14, no. 12, p. 5044, 2024, doi: 10.3390/app14125044.

[105] C. Long, "Comparing Lower-Limb Muscle Activity During Gait Performed in Water Versus on Land by," UTAH STATE UNIVERSITY, 2023.

[106] B. Worley, "Acute Effects of Multi-Joint Eccentric Exercise on Lower Extremity Muscle Activation Measured During Land and Water Walking," UTAH STATE UNIVERSITY, 2024.

[107] B. H. Olstad, C. Zinner, J. R. Vaz, J. M. H. Cabri, and P. L. Kjendlie, "Muscle activation in world-champion, world-class, and national breaststroke swimmers," *Int. J. Sports Physiol. Perform.*, vol. 12, no. 4, pp. 538–547, 2017, doi: 10.1123/ijspp.2015-0703.

[108] B. H. Olstad, J. R. Vaz, C. Zinner, J. M. H. Cabri, and P. L. Kjendlie, “Muscle coordination, activation and kinematics of world-class and elite breaststroke swimmers during submaximal and maximal efforts,” *J. Sports Sci.*, vol. 35, no. 11, pp. 1107–1117, 2017, doi: 10.1080/02640414.2016.1211306.

[109] B. H. Olstad, “A new approach for identifying phases of the breaststroke wave kick and calculation of feet slip using 3D automatic motion tracking,” *BMS Proc.*, pp. 195–199, 2014.

[110] M. Nakashima, R. Kanie, T. Shimana, Y. Matsuda, and Y. Kubo, “Development of a comprehensive method for musculoskeletal simulation in swimming using motion capture data,” *Proc. Inst. Mech. Eng. Part P J. Sport. Eng. Technol.*, vol. 237, no. 2, pp. 85–95, 2023, doi: 10.1177/1754337119838395.

[111] H. Suito, N. Tsujimoto, H. Shinkai, S. Sano, H. Nunome, and Y. Ikegami, “the Effect of Fatigue on the Underwater Arm Stroke Motion in the 100 M Front Crawl,” *J. Biomech.*, vol. 40, no. December, p. S772, 2007, doi: 10.1016/s0021-9290(07)70760-1.

[112] J. Yang, T. Li, Z. Chen, and X. Li, “Research on the Method of Underwater Swimming Motion Capture,” *J. Phys. Conf. Ser.*, vol. 1982, no. 1, pp. 1–4, 2021, doi: 10.1088/1742-6596/1982/1/012075.

[113] C. Connaboy, S. Coleman, G. Moir, and R. Sanders, “Measures of reliability in the kinematics of maximal undulatory underwater swimming,” *Med. Sci. Sports Exerc.*, vol. 42, no. 4, pp. 762–770, 2010, doi: 10.1249/MSS.0b013e3181badc68.

[114] S. Veiga, J. Lorenzo, A. Trinidad, R. Pla, A. Fallas-Campos, and A. de la Rubia, “Kinematic Analysis of the Underwater Undulatory Swimming Cycle: A Systematic

and Synthetic Review," *Int. J. Environ. Res. Public Health*, vol. 19, no. 19, 2022, doi: 10.3390/ijerph191912196.

[115] T. Tanaka, S. Hashizume, T. Sato, and T. Isaka, "Competitive-Level Differences in Trunk and Foot Kinematics of Underwater Undulatory Swimming," *Int. J. Environ. Res. Public Health*, vol. 19, no. 7, 2022, doi: 10.3390/ijerph19073998.

[116] G. R. D. Bernardina, A. G. P. Andrade, T. Monnet, P. Cerveri, and A. P. Silvatti, "Simultaneous In-Air and Underwater 3d Kinematic Analysis of Swimmers Feasibility and Reliability of Action Sport Cameras," *J. Biomech.*, 2023, doi: 10.2139/ssrn.4529173.

[117] A. F. Panaite, S. Rosca, and R. Sibişanu, "Pose and motion capture technologies," *MATEC Web Conf.*, vol. 342, p. 05004, 2021, doi: 10.1051/matecconf/202134205004.

[118] D. P. Born, T. Stöggl, A. Petrov, D. Burkhardt, F. Lüthy, and M. Romann, "Analysis of Freestyle Swimming Sprint Start Performance After Maximal Strength or Vertical Jump Training in Competitive Female and Male Junior Swimmers," *J. Strength Cond. Res.*, vol. 34, no. 2, pp. 323–331, 2020, doi: 10.1519/JSC.0000000000003390.

[119] P. Chainok *et al.*, "Biomechanical Features of Backstroke to Breaststroke Transition Techniques in Age-Group Swimmers," *Front. Sport. Act. Living*, vol. 4, no. March, pp. 1–11, 2022, doi: 10.3389/fspor.2022.802967.

[120] J. Ribeiro *et al.*, "Biomechanics, energetics and coordination during extreme swimming intensity: effect of performance level," *J. Sports Sci.*, vol. 35, no. 16, pp. 1614–1621, 2017, doi: 10.1080/02640414.2016.1227079.

[121] G. R. D. Bernardina, P. Cerveri, R. M. L. Barros, J. C. B. Marins, and A. P. Silvatti, "Action sport cameras as an instrument to perform a 3D underwater motion analysis," *PLoS One*, vol. 11, no. 8, pp. 1–14, 2016, doi: 10.1371/journal.pone.0160490.

[122] S. L. Raghu, R. T. Conners, C. kwon Kang, D. B. Landrum, and P. N. Whitehead, "Kinematic analysis of gait in an underwater treadmill using land-based Vicon T 40s motion capture cameras arranged externally," *J. Biomech.*, vol. 124, no. June, p. 110553, 2021, doi: 10.1016/j.jbiomech.2021.110553.

[123] S. L. Raghu, C. kwon Kang, P. Whitehead, A. Takeyama, and R. Conners, "Static accuracy analysis of Vicon T40s motion capture cameras arranged externally for motion capture in constrained aquatic environments," *J. Biomech.*, vol. 89, pp. 139–142, 2019, doi: 10.1016/j.jbiomech.2019.04.029.

[124] K. Abdul Jabbar, S. Kudo, K. W. Goh, and M. R. Goh, "Comparison in three dimensional gait kinematics between young and older adults on land and in shallow water," *Gait Posture*, vol. 57, no. July 2016, pp. 102–108, 2017, doi: 10.1016/j.gaitpost.2017.05.021.

[125] J. Lauer, A. H. Rouard, and J. P. Vilas-Boas, "Upper limb joint forces and moments during underwater cyclical movements," *J. Biomech.*, vol. 49, no. 14, pp. 3355–3361, 2016, doi: 10.1016/j.jbiomech.2016.08.027.

[126] L. Barbieri, F. Bruno, A. Gallo, M. Muzzupappa, and M. L. Russo, "Design, prototyping and testing of a modular small-sized underwater robotic arm controlled through a Master-Slave approach," *Ocean Eng.*, vol. 158, pp. 253–262, 2018, doi: 10.1016/j.oceaneng.2018.04.032.

[127] C. Chung and M. Nakashima, “Swimming humanoid robot ‘SWUMANOID’ as an experimental platform for research of human swimming,” *J. Robot. Mechatronics*, vol. 26, no. 2, pp. 265–266, 2014, doi: 10.20965/jrm.2014.p0265.

[128] M. Nakashima and Y. Tsunoda, “Improvement of crawl stroke for the swimming humanoid robot to establish an experimental platform for swimming research,” *Procedia Eng.*, vol. 112, pp. 517–521, 2015, doi: 10.1016/j.proeng.2015.07.235.

[129] M. Nakashima and C. Tsai, “Realization and Swimming Performance of the Butterfly Stroke by a Swimming Humanoid Robot Department of Systems and Control Engineering , Department of Mechanical Engineering ,” *J. Aero Aqua Bio-Mechanisms*, vol. 6, no. 1, pp. 9–15, 2017.

[130] F. RAZI and M. NAKASHIMA, “Preliminary Study of Backstroke by the Swimming Humanoid Robot,” *Proc. JSME Annu. Conf. Robot. Mechatronics*, vol. 2017, no. 0, pp. 2A2-D05, 2017, doi: 10.1299/jsmermd.2017.2a2-d05.

[131] Y. Ishii, S. Nishikawa, R. Niiyama, and Y. Kuniyoshi, “Development of a Musculoskeletal Humanoid Robot as a Platform for Biomechanical Research on the Underwater Dolphin Kick,” *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 3285–3291, 2018, doi: 10.1109/IROS.2018.8593912.

[132] M. Nakashima, T. Koga, and H. Takagi, “Measurement of propulsive forces in swimming by using a swimming humanoid robot,” *Int. Conf. Control. Autom. Syst.*, vol. 2021-Octob, no. Iccas, pp. 1780–1783, 2021, doi: 10.23919/ICCAS52745.2021.9649981.

[133] Y. Ukai and J. Rekimoto, “Swimoid: A swim support system using an underwater buddy robot,” *ACM Int. Conf. Proceeding Ser.*, pp. 170–177, 2013, doi:

10.1145/2459236.2459265.

[134] Y. Ukai and J. Rekimoto, “Swimoid: Interacting with an underwater buddy robot,” *ACM/IEEE Int. Conf. Human-Robot Interact.*, vol. 2, pp. 243–244, 2013, doi: 10.1109/HRI.2013.6483592.

[135] E. Kelasidi, P. Liljebäck, K. Y. Pettersen, and J. T. Gravdahl, “Biologically Inspired Swimming Snake Robots: Modeling, Control and Experimental Investigation,” *Ieee Robot. Autom. Mag.*, vol. XX, no. Xx, p. 1, 2015.

[136] Ecency, “Underwater Motion Capture Technology.” https://ecency.com/hive-174578/@abhay2695/underwater-motion-capture-technology-06a5c57dd4612 (accessed Jul. 10, 2024).

[137] S. Takahashi and S. Kuriyama, “Animations of Real Swimming via Motion Reconstruction,” 2011.

[138] Y. Hosokawa, D. Urata, A. Doi, T. Takata, and Y. Abe, “The motion capturing of female divers under water and the trial production of motion viewers for developing a virtual diving experience learning system,” *Artif. Life Robot.*, vol. 22, no. 3, pp. 346–356, 2017, doi: 10.1007/s10015-017-0359-0.

[139] C. Losee, “The Bathysphere Motion Capture and Immersive Projection,” THE UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL, 2010.

[140] Qualisys, “Cameras for underwater motion capture,” *Qualisys*. https://www.qualisys.com/cameras/underwater/ (accessed Jul. 15, 2024).

[141] Qualisys, “Marine vessels & structures.” https://www.qualisys.com/engineering/marine-vessels-and-structures/ (accessed Jul. 26, 2024).

[142] K. J. Nankervis *et al.*, “Effect of speed and water depth on limb and back kinematics in Thoroughbred horses walking on a water treadmill,” *Vet. J.*, vol. 300–302, p. 106033, 2023, doi: 10.1016/j.tvjl.2023.106033.

[143] X. Meng, J. Pan, and H. Qin, “Motion capture and retargeting of fish by monocular camera,” *Proc. - 2017 Int. Conf. Cyberworlds, CW 2017 - Coop. with Eurographics Assoc. Int. Fed. Inf. Process. ACM SIGGRAPH*, vol. 2017-Janua, pp. 80–87, 2017, doi: 10.1109/CW.2017.16.

[144] H. Enshaei, R. Birmingham, and E. Mesbahi, “Identification of Influential Parameters in a Ship’S Motion Responses: a Route To Monitoring Dynamic Stability,” *Trans. R. Inst. Nav. Archit. Part A Int. J. Marit. Eng.*, vol. 154, no. A1, pp. A43–A51, 2012, doi: 10.5750/ijme.v154iA1.874.

[145] Y. Yoshimura, K. Takase, H. Fukui, H. Suzuki, and S. Hirabayashi, “Simulation of Ship Drift Motion with a Simplified Mathematical Model under the Wind,” *J. Japan Soc. Nav. Archit. Ocean Eng.*, vol. 31, no. 0, pp. 47–57, 2020, doi: 10.2534/jjasnaoe.31.47.

[146] J. P. Jhan, J. Y. Rau, and C. M. Chou, “Underwater 3D rigid object tracking and 6-DOF estimation: A case study of giant steel pipe scale model underwater installation,” *Remote Sens.*, vol. 12, no. 16, pp. 1–14, 2020, doi: 10.3390/RS12162600.

[147] S. Lack, E. Rentzow, and T. Jeinsch, “Control of a small Underwater Vehicle Manipulator System - a highly automated Pick and Place Experiment *,” no. February, 2024.

[148] N. Bauschmann, D. A. Duecker, T. L. Alff, R. C. Hochdahl, and R. Seifried,

"Towards Full Actuation: Reconfigurable Micro Underwater Robots," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 6192–6199, 2023, doi: 10.1109/IROS55552.2023.10341621.

[149] O. Tortorici, C. Péraud, C. Anthierens, and V. Hugel, "Automated Deployment of an Underwater Tether Equipped with a Compliant Buoy–Ballast System for Remotely Operated Vehicle Intervention," *J. Mar. Sci. Eng.*, vol. 12, no. 2, 2024, doi: 10.3390/jmse12020279.

[150] J. K. Colling, S. J. Kang, E. Dehdashti, S. Husain, H. Masoud, and G. G. Parker, "Free-Decay Heave Motion of a Spherical Buoy," *Fluids*, vol. 7, no. 6, 2022, doi: 10.3390/fluids7060188.

[151] I. M. Thompson, J. Banks, D. Hudson, and M. Warner, "Assessment of error levels across the domain of a three dimensional underwater motion capture methodology," *40th Int. Soc. Biomech. Sport. Conf.*, pp. 703–706, 2022.

[152] T. Monnet, M. Samson, A. Bernard, L. David, and P. Lacouture, "Measurement of three-dimensional hand kinematics during swimming with a motion capture system: A feasibility study," *Sport. Eng.*, vol. 17, no. 3, pp. 171–181, 2014, doi: 10.1007/s12283-014-0152-4.

[153] C. Kirmizibayrak, J. Honorio, X. Jiang, R. Mark, and J. K. Hahn, "Digital Analysis and Visualization of Swimming Motion," *Int. J. Virtual Real.*, vol. 10, no. 3, pp. 9–16, 2011, doi: 10.20870/ijvr.2011.10.3.2817.

[154] A. Marouchos, M. Sherlock, and J. Cordell, "Challenges in underwater image capture," *Ocean. 2018 MTS/IEEE Charleston, Ocean 2018*, pp. 0–4, 2019, doi: 10.1109/OCEANS.2018.8604647.

[155] E. Ceseracciu *et al.*, "Markerless analysis of front crawl swimming," *J. Biomech.*, vol. 44, no. 12, pp. 2236–2242, 2011, doi: 10.1016/j.jbiomech.2011.06.003.

[156] S. Ceccon *et al.*, "Motion analysis of front crawl swimming applying CAST technique by means of automatic tracking," *J. Sports Sci.*, vol. 31, no. 3, pp. 276–287, 2013, doi: 10.1080/02640414.2012.729134.

[157] M. A. Hidayat Yani, S. Bayu Aji, I. F. Ariyanti, S. Sukaridhoto, M. A. Zainuddin, and A. Basuki, "Implementation of Motion Capture System for Swimmer Athlete Monitoring," *IES 2019 - Int. Electron. Symp. Role Techno-Intelligence Creat. an Open Energy Syst. Towar. Energy Democr. Proc.*, pp. 400–405, 2019, doi: 10.1109/ELECSYM.2019.8901554.

[158] N. J. Cronin, T. Rantalainen, J. P. Ahtiainen, E. Hynynen, and B. Waller, "Markerless 2D kinematic analysis of underwater running: A deep learning approach," *J. Biomech.*, vol. 87, pp. 75–82, 2019, doi: 10.1016/j.jbiomech.2019.02.021.

[159] Y. H. Kwon and J. B. Casebolt, "Effects of light refraction on the accuracy of camera calibration and reconstruction in underwater motion analysis," *Sport. Biomech.*, vol. 5, no. 2, pp. 315–340, 2006, doi: 10.1080/14763140608522881.

[160] Z. Zhu, X. Li, Z. Wang, L. He, B. He, and S. Xia, "Development and research of a multi-medium motion capture system for underwater intelligent agents," *Appl. Sci.*, vol. 10, no. 18, 2020, doi: 10.3390/APP10186237.

[161] K. Hu, C. Weng, Y. Zhang, J. Jin, and Q. Xia, "An Overview of Underwater Vision Enhancement: From Traditional Methods to Recent Deep Learning," *J. Mar. Sci. Eng.*, vol. 10, no. 2, 2022, doi: 10.3390/jmse10020241.

[162] G. R. D. Bernardina, P. Cerveri, R. M. L. Barros, J. C. B. Marins, and A. P. Silvatti, "In-air versus underwater comparison of 3D reconstruction accuracy using action sport cameras," *J. Biomech.*, vol. 51, pp. 77–82, 2017, doi: 10.1016/j.jbiomech.2016.11.068.

[163] F. Ferryanto, A. I. Mahyuddin, and M. Nakashima, "Markerless Optical Motion Capture System for Asymmetrical Swimming Stroke," *J. Eng. Technol. Sci.*, vol. 54, no. 5, 2022, doi: 10.5614/j.eng.technol.sci.2022.54.5.3.

[164] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A skinned multi-person linear model," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 1–16, 2015, doi: 10.1145/2816795.2818013.

[165] Qualisys, "Swimming Technique: dual media motion capture," *Qualisys*. https://qfl.qualisys.com/#!/project/swimming-techniques (accessed Dec. 02, 2023).

[166] S. Kudo and M. K. Lee, "Validity of underwater motion capture system for swimming," 2009.

[167] Qualisys, "Qualisys launches smallest-ever underwater mocap solution," 2019. https://press.qualisys.com/posts/pressreleases/qualisys-launches-smallest-ever-underwater-mo (accessed Jul. 23, 2024).

[168] Qualisys, "5+, 6+ and 7+ series." https://www.qualisys.com/cameras/5-6-7/ (accessed Jul. 23, 2024).

[169] Qualisys, "Products tagged 'oqus.'" https://www.qualisys.com/product-tag/oqus/ (accessed Jul. 23, 2024).

[170] Qualisys, "Introducing the Miqus M5 from Qualisys: Breaking new ground in motion capture technology," 2018.

https://meltwater.pressify.io/publication/5cc6fcf444c3c10004a9f290/5cb6dab997b1be1000d4c19a (accessed Jul. 23, 2024).

[171] Qualisys, “Arqus.” https://www.qualisys.com/cameras/arqus/ (accessed Jul. 23, 2024).

[172] G. R. D. Bernardina, R. G. Silva, P. Cerveri, R. M. L. de Barros, and A. P. Silvatti, “Accuracy of Sport Action Cameras for 3D Underwater Motion Analysis,” *ISBS - Conf. Proc. Arch.*, pp. 505–508, 2014, [Online]. Available: https://ojs.ub.uni-konstanz.de/cpa/article/view/5971

[173] Qualisys, “Compare our motion capture cameras.” https://www.qualisys.com/cameras/ (accessed Jul. 24, 2024).

[174] Qualisys, “Motion capture for underwater measurements.” https://www.qualisys.com/cameras/underwater/#tech-specs (accessed Jul. 24, 2024).

[175] Qualisys, “Qualisys Sports Marker Set,” *Qualisys*. https://cdn-content.qualisys.com/2022/07/Sports-Marker-Set.pdf (accessed Dec. 05, 2023).

[176] Wikipedia, “Principal component analysis.” https://en.wikipedia.org/wiki/Principal_component_analysis

[177] Wikipedia, “Long short-term memory.” https://en.wikipedia.org/wiki/Long_short-term_memory (accessed Jul. 29, 2024).

[178] H. W. Kuhn, “The Hungarian method for the assignment problem,” *50 Years Integer Program. 1958-2008 From Early Years to State-of-the-Art*, vol. 2, no. 1, pp. 29–47, 2010, doi: 10.1007/978-3-540-68279-0_2.

[179] S. Chatterjee, “Procrustes Problems,” *Technometrics*, vol. 47, no. 3, pp. 376–376,

2005, doi: 10.1198/tech.2005.s296.

[180] Wikipedia, “Transfer Learning.” https://en.wikipedia.org/wiki/Transfer_learning

[181] SimTK, “OpenSim,” *SimTK*. https://simtk.org/frs/index.php?group_id=91 (accessed Jan. 05, 2024).

2. A Survey on Solving Marker-based Motion Capture Approaches

Abstract

This survey article delves deeply into marker-based motion capture (MoCap), a technology utilized for recording the three-dimensional (3D) movement of objects or individuals. This technology finds applications in various fields such as virtual reality, animation, robotics, and biomechanics. It involves affixing markers to the surface of an object or the skin of a performer, and then tracking their positions in real-time using cameras. This article aims to provide an overview of various methods and techniques used to solve marker-based MoCap problems, including cleaning (denoising and recovery), alignment, and auto-labeling. The discussion revolves around the challenges and limitations posed by these algorithms, as well as the ongoing research aimed at addressing these issues. One such endeavor involves creating a natural movement during the gap-filling procedure. Moreover, other factors affecting MoCap data are studied. These factors include those that affect 3D reconstruction, leading to occlusion and tracking errors. Examples of such factors are marker and camera placement constraints, such as marker visibility, the number of markers used, and the symmetry of the marker set. The article concludes with potential future directions and developments in marker-based MoCap technology.

2.1 Introduction

Marker-based optical motion capture (OMC) [1] systems have precisely revolutionized the field of motion capture (MoCap) and analysis by tracking active or passive markers attached to the subject's body [2]. However, MoCap systems are prone to errors due to many factors, such as poor calibration, noisy environments, and occlusion [3]. Denoising, recovery, alignment, and auto-labeling are the main pillars of MoCap solving [4].

Denoising and recovery are commonly encountered problems in MoCap systems. Although their objectives may differ slightly, the underlying technology used to address these tasks is often very similar. These problems involve two main tasks: removing noise from the captured data and reconstructing missing markers. In some articles, a combination of denoising and recovery is called cleaning [5]. A recent survey studied denoising and completion filters for 3D skeleton-based human motion analysis from marker-based or marker-less MoCap systems and their assumptions. We focus on marker-based optical MoCap approaches not included or extensively discussed in that survey [6].

Denoising algorithms are designed to eliminate various types of noise [3], such as outliers, ghost markers, extraneous markers, swapping markers, overlapping markers, and spikes. These algorithms identify and remove invalid points from captured data, reducing manual effort to clean the data and enabling accurate motion analysis.

Recovery algorithms are designed to identify and reconstruct missing marker trajectories caused by gaps [3], often due to occlusion or self-occlusion, where markers become obstructed from the cameras' view. Solutions have been developed to address this issue by filling in these gaps, aiming to create a more natural motion and behavior while retrieving missing markers. This ensures precise motion analysis for various applications.

Aligning and maintaining subject direction consistency [7] in MoCap data is crucial for accurate analysis and interpretation of movement patterns. Subject direction consistency ensures that the captured movements are correctly attributed to the intended subjects, preventing errors in data processing and analysis. By addressing these issues, researchers can enhance the reliability and validity of MoCap studies, leading to more accurate results.

Auto-labeling methods [8] streamline the tedious task of manually cleaning and labeling MoCap data. The process includes several stages: data cleaning through denoising and recovering missing markers, aligning and labeling the data, and possibly post-processing to correct mislabeled, missing, or unlabeled data.

The paper is structured as follows: Section 2.2 covers denoising algorithms, Section 2.3 explains recovery methods, Section 2.4 presents aligning approaches, Section 2.5 discusses auto-labeling methods, and Section 2.6 concludes the article.

2.2 MoCap Data Denoising

The relevant literature commonly refers to denoising as occlusion gap filling [9]–[11]. They often treat missing values as common noise and attempt to prevent the occurrence of artifacts by controlling the environment and ensuring precise calibration [12]. We will address these articles that solely focus on recovering missing markers in the next section. Some articles address denoising and recovery together, referring to it as cleaning [5], [12], [13]. In this section, we concentrate on denoising and cleaning techniques aimed at mitigating various types of noise, including outliers [14], ghost markers [15], extraneous markers [7], swapped markers [16], overlapping markers [17], spikes [18], and jitters [19]. Denoising approaches can be divided into three categories: filter-based algorithms, matrix low-rank theory algorithms, and data-driven algorithms. Filter-based methods can be further categorized into three types: low-pass, Kalman, and space-time filters. A recent systematic review [6] concluded that low-pass and Kalman filters were the most commonly used. Commercial MoCap software like Qualisys track manager [20] employ low-pass filters such as Butterworth and moving average filters [21], [22]. However, most articles on MoCap solving have focused on solutions that account for noise [7], [8] rather than directly addressing noise removal.

In [12], incorrect intervals were explicitly identified for further cleaning. They proposed a marker-wise, skeleton-free method based on two fundamental assumptions: the rigid body model and the correlation of marker trajectories. Different artifacts were classified into simple gap, sudden changes (single peak, heavy noise, rectangular distortion), and slow value change (“Figure 2-1”).

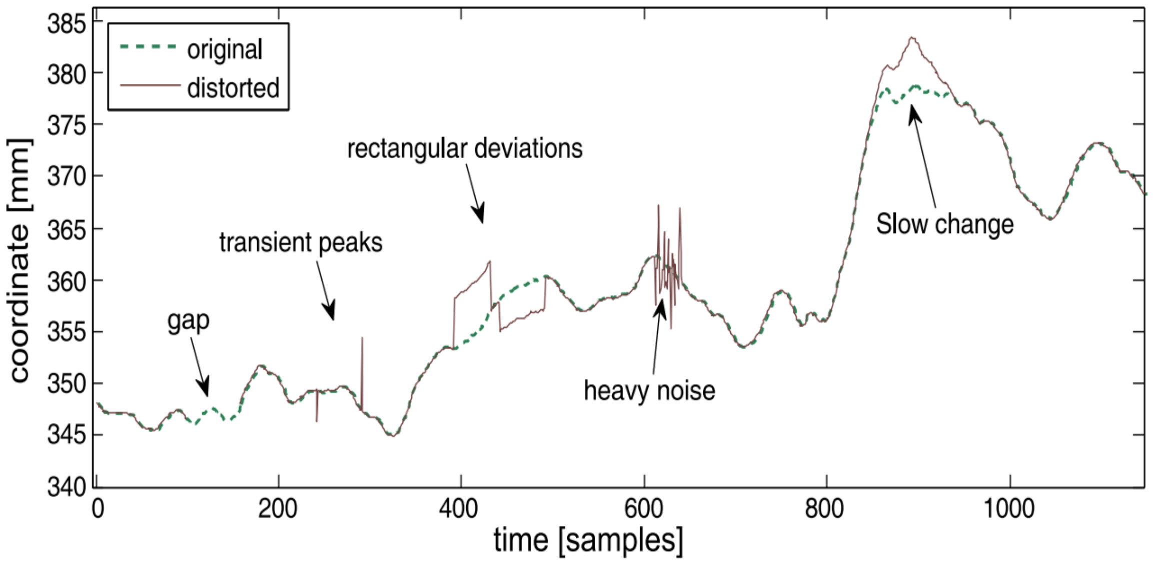


Figure 2-1: [12]*: “Figure 2. Identified types of distortions inpainted into exemplary data—the first coordinate of the first marker (head) of the IM subject.”*

We provide a more detailed explanation of this article [12], as it identified only one study that was similar to theirs [16]. In that study, invalid keyframes were explicitly identified using a robust method, which subsequently generated new kinematically correct paths through the application of a neural network (NN). However, this previous research did not include any identification of the type of distortion present.

In [12], after detecting and classifying the artifacts, each one was handled using different methods. They utilized reconstruction methods including Savitzky–Golay filtering (using a 13th order polynomial over a window of 101 samples), linear interpolation, spline interpolation, and predictions generated by a NN. Treating sudden changes relied on derivative analysis, median and Savitzky-Golay low-pass filtering (as predicting models), respectively, for short-term and long-term distortions, with stats-based thresholding and mathematical morphology. Slow change detection was based on the hysteresis thresholding of residuals with backward regrowing of identified segments. They employed neighbour-based predictors—initially, they assumed a polynomial predictor based on the least squares method, which they gave up in favor of a feed-forward NN using functional body mesh representation. The deviation of a trajectory from the prediction was used as a criterion for classification. The overall efficiency relied on the quality of model predictors, indicated by the standard deviation (SD) of residuals (e.g., three-sigma rule), which facilitated effective outlier detection. Additionally, it depended on accurately approximating a marker's real location based on its own or neighboring markers' past, present, or future positions. Using synthetically distorted sequences (presumed only one distortion at a time), performance of the approach was comparable to human operators. Companion results were additionally acquired in the experiment to compare the results obtained to the two generic existing methods of anomaly detection in the time series: three-sigma move, which employs mean and variance moving, and the Hampel filter, which is based on the moving median and median absolute deviation, which are more robust measures. Both the three-sigma move and Hampel filter are methods that consider each coordinate separately, unlike more sophisticated methods of anomaly detection based on machine learning, such as clustering, one-class support vector machines, or autoencoders, that can find anomalous frames instead of inter-marker dependencies. The criteria for evaluation were classification rates, presented as a confusion matrix, F1-score, and Matthews correlation coefficient. They assumed the root-mean-square error as the measure of quality. They concluded that identification of slow change was as difficult as expected. They analyzed alternative techniques, including simple feedforward and nonlinear autoregressive exogenous NNs, ridge, lasso, and support vector regression. However, the results were either poor or impractical (due to long training times), or both. They couldn't compare the efficiencies to the other solutions, as their work was the first proposal in this area. The only comparable method to them [16] was publicly unavailable, and the distortions were not classified. However, they conducted an indirect comparison of the efficiency of their solution against that of human operators and industrial software. The proposed solution was less effective than experienced human operators but significantly outperformed novice operators. Furthermore, the automatic repairing algorithms available in modern software, such as Vicon Nexus, have the potential to increase the number of artifacts. Detecting distortions for all three coordinates of a marker jointly was left for future work.

Quantifying various types of noise using Allan variance was discussed in [23], including white noise, random walk, blue noise, and flicker, with significant contribution to the first two and least to violet noise. Environmental long-term correlated noise and periodic distortion were noted. They suggested Butterworth or Woltring filters to remove white noise and found longer-term distortion removal, like flicker or random walk, challenging.

In a study by [24], the author compared moving averages, B-spline smoothing, and the Kalman filter. The B-spline-based least square method generated high-quality continuous outputs with minimal parameter adjustments for various motion signals, even in the presence of outliers or missing data. Moreover, for studies involving the extraction of signal features such as velocity and acceleration, reliable derivative values can be obtained through B-spline smoothing. Moreover, they employed low-pass filters to eliminate shaky movements caused by high-frequency noise. However, the process was offline.

Kalman and moving average filters are suitable for online applications due to their localized calculations. However, they may struggle with sudden local spikes. Methods like the Gaussian low-pass filter and the Kalman filter (optimized for unknown trajectories) often process each degree of freedom separately, and the filtered output may appear unnatural due to the lack of spatial-temporal characteristics. A modified k-means algorithm was claimed to outperform the standard filtering algorithms, such as mean and median, by completely removing noise with both spike and Gaussian characteristics [18].

A data-driven approach in [25] utilized multichannel singular spectrum analysis to eliminate outliers and noise. This technique employed singular value decomposition on the trajectory matrix derived from time series data to develop space-time filters. It was applied to the grand lag covariance matrix, which captures spatial-temporal correlations within a defined window, facilitating the extraction of spatial-temporal patterns. However, post-processing was required to meet kinematic constraints.

A data-driven method in [26] presented statistical models to classify movements as natural or unnatural by decomposing the motion into parts based on joint rotations and linear and angular velocities. A cyclic motion-specific approach in [27] identified male or female walking, using principle component analysis (PCA) for dimension reduction, and then fitted sinusoids to the resulting components. On the other hand, [28] provided a multi-resolution of epitomes as a statistical model of natural motions to choose various timescales flexibly. Given only positive (natural) training data, the epitomic analysis generated motion epitomes to compute a degree of naturalness score. Although these methods use kinematics to detect unnatural intervals, they invalidate all their markers.

To address this issue, a data-driven approach [29] automatically detected and fixed the erroneous joint rotations based on the self-similarity of human motion data. The individual frames or poses weren't examined because joint angles are relative measures and are spatially invariant regardless of the global pose, so the absolute marker positions were not needed. A motion-texture map was defined with temporal joints' rotation angles as rows and single pose-frames as columns. Motion words were short sequences of joint transformations around a motion frame. The outliers and erroneous motions for multiple performers were detected simultaneously and replaced by similar correct motions by comparing each motion word with its k-nearest neighbours using dynamic time warping. A movement digression map indicated unusual movements in time on specific joints. However, unlike the kinematic level solution presented in [19], this joint level method fails with more complicated errors such as marker swapping.

In [16], the benefits of both joint level [29] and kinematic level [19] approaches were leveraged to detect invalid intervals, such as marker swapping, by looking at poses' differences on a per-joint basis. A NN and a linear blend skinning from a cleaned series of joint transforms were used reconstruct markers. Preset numbers of frames were removed before and after a gap until the slope difference between the original and the reference path was minimized to a preset threshold. Short gaps were filled using polynomial and cubic Hermite splines with positional and velocity constraints to create a natural filling. The limitations of this approach include the experimentally selected preset parameters, the loss of small details, and the introduction of offsets in the marker paths.

3D human motion search and retrieval techniques were discussed in [30]. They developed a data-driven method to address corrupted, noisy, or missing markers in MoCap data. To leverage prior information, they created a knowledge base from an existing clean MoCap dataset. For efficient searching and retrieval of similar poses, they constructed a kd-tree and a parallel nearest neighbor search strategy.

In [31], a data-driven denoising method sparsely selected the most correlated subset of motion bases for clean motion reconstruction, considering Gaussian noise and outliers. They divided each human pose into five partitions termed poselets to construct motion dictionaries. Another approach [32] reconstructed missing markers by using sparse representation. They proposed a presentation coefficient weighted update algorithm to mitigate the limited capacity problem of the training set. These approaches' representations are coarse; therefore, [14] divided each pose into five parts termed the partlets to obtain a more fine-grained representation.

Unlike data-driven approaches, which rely on motion from a large data set, the noisy low-rank completion methods have the advantage of requiring no training data set to recover noisy MoCap sequence [33]. The low-rank structural characteristics of the motion data matrix were explored in [9] to complete the missing walking motion data using the singular value decomposition method. This method is unsuitable for long motion sequences with different poses.

In [34], a new robust non-linear matrix factorization method was proposed that is robust to sparse noise and outliers. They constructed a dictionary for the data space by factoring in a kernelized feature space. Then, a noisy matrix was decomposed as the sum of a sparse noise matrix and a clean data matrix in a low-dimensional nonlinear manifold.

In [10], a method was discussed for denoising MoCap data using filtered subspace clustering and low-rank matrix approximation. Using a filtered subspace clustering technique, the noisy MoCap sequence was divided into disjoint piecewise motions. Each piecewise motion shares a similar low-dimensional subspace representation. The accelerated proximal gradient algorithm was then utilized to find a complete low-rank matrix approximation for each noisy piecewise motion. A moving average smoothed the moving trajectories between the connected motions. Finally, the entire noisy MoCap data were restored by concatenating all the recovered piecewise motions in sequence. This method does not require prior knowledge about the structure or auxiliary data sets for training priors.

Most of these methods are robust in the presence of different types of noise. However, the following deep learning-based methods are more robust but sometimes deviate from the real motion.

In [5], two deep, bidirectional, recurrent, long short-term memory (LSTM) NNs were used for real-time cleaning of a wide variety of noise types (in joint angles and positions) and long gaps with a single trained model. The approach is not noise-specific or action-specific. However, noisy and clean motion pairs and unlabeled action-type examples should be available in the training set.

Another online method, with a simpler network and requiring less data, was presented in [35]. LSTM-based and one-time-window-based models were used to remove position noise and fill in missing parts of the pose. The approach was validated on synthetically noisy data which were injected with Gaussian additive noise (similar to [36]) into the input during training.

To address different noise types in Kinect data and time-consuming postprocessing smoothing step in [19], a perceptual-based noise-agnostic 3D skeleton motion data refinement method was presented in [37] based on a bidirectional recurrent autoencoder. They improved the refined motion data's kinematic information, bone-length consistency, and smoothness when the noisy data and target clean data had different topologies, which was unsolved in their previous work [38]. However, the types of noise weren't specified, and the refined motion can still be somewhat noisy due to poor reproduction accuracy.

Correcting mislabeled markers was addressed as a post-processing step in auto-labeling approaches. In [39], a sequence of unlabelled (shuffled) 3D trajectories as input was processed with a data-driven auto-labeling approach by applying permutation learning to each frame. The resulting labeled frames were concatenated to form trajectories again. Then, a temporal consistency constraint was used to correct mislabelled markers.

Recent articles [4], [7], [8], [40], [41] have proposed deep learning-based MoCap solving methods that address the presence of outliers, ghost markers, and extraneous markers in MoCap data.

With enough samples, data-driven approaches can automatically learn the time and space domain and the complexity and diversity in motion sequences. However, they may need help with unseen motions and obtaining samples for long sequences with multiple motion semantics.

2.3 MoCap Data Recovery

Previous work focused on gap filling and reconstruction, sometimes referred to as denoising tasks, while others considered cleaning [5] as denoising and reconstruction techniques [6]. Various researchers have focused on different aspects of the problem at hand. Some have explored the detection of erroneous intervals [12], while others have concentrated on achieving a natural appearance in reconstruction [26]–[28]. Additionally, some researchers have investigated sparse representation [31], [32], skeleton-based methods [42], methods based on low-rank matrix completion [43], truncated singular value decomposition [9], PCA via truncated nuclear norm regularization [44], truncated nuclear norm regularization [45], graph-based methods [46], data-driven methods such as kd-tree recovery [30], and deep learning based methods such as [35]. In [47], a denoising autoencoder was trained to predict the original uncorrupted skeleton. MoCap solving methods [4] involve marker tracking, denoising, and reconstruction. This section focuses exclusively on articles related to reconstruction methods.

Interpolation is a common method for filling gaps in marker trajectories, employing techniques like linear or spline interpolation that maintain the spatial-temporal characteristics of human motion [24], [42]. While effective for short gaps, these methods struggle with longer sequences due to the high correlation between adjacent trajectories.

In [48], a data-driven marker-based method suitable for large gaps with multiple missing body parts across different actors and motion styles was proposed. In the preprocessing step, all MoCap data from the prior database was first normalized with respect to global position and orientation. In addition, linear marker velocities and accelerations were stored as a simple optimization scheme without a bone length constraint. Subsequently, similar examples from the database were retrieved by a spatial indexing structure (kd-tree) based on the search for nearest neighbours. However, they assumed the same marker set for the motions in the database and the tested motion to be cleaned. Additionally, it was assumed that valid markers, which are the set of markers containing reliable positional information, are provided for each frame of the input motion to be completed.

In [39], a data-driven auto-labeling approach was proposed. They used a feed-forward deep residual NN for permutation learning. First, each trajectory was defined as the sequence of tracked marker locations, which ends with a gap or when the recording stops. Therefore, in each motion sample, the movement of each marker might be presented in multiple trajectories over time. They exploited the temporal consistency of each trajectory to correct the wrong predictions for each marker during the trajectory.

In [49], a new recovery process was proposed that combined statistical and kinematic information to address the issues with low-rank matrix completion and sparse coding that ignored kinematics and used a learned dictionary in a complete feature space. Inspired by coupled dictionary learning and locally linear embedding, they learned a dictionary of complete–incomplete training frame pairs, preserving the statistical information. They then used the kinematic information, including smoothness and bone-length constraints, to recover motions from incomplete frames using sparse representations and a learned dictionary via two gradient-based optimization models.

In [50], a Kalman filter-based real-time approach, suitable for long gaps, was proposed. They combined the prediction algorithm, using previous markers' positions, with information from neighbouring markers belonging to the same limb segment, with a rigid body assumption, to handle the cases that they couldn't handle in their previous work [51], where all markers on a limb were occluded, or one or two markers were not visible in a large gap. Real-time skeleton fitting was done without any pre-defined skeleton model by estimating the time center of rotation between two marker sets, using the Procrustes method [52] to calculate the limb orientation relative to a reference frame. However, they still assumed the presence of rigid limbs with at least three markers placed on each limb. Another Kalman filter-based approach compared their result with them [53]. In [54], the unscented Kalman filter was used as an alternative method to the extended Kalman filter [55] to provide a more accurate estimation of the distribution of the state random variable through sampling techniques.

In [56], a data-driven, piecewise linear modeling for long gaps was suggested by characterizing a k-means clustered hierarchy of low-dimensional local linear models using PCA to model motion sequences of a training set. Frames of a new sequence were classified by a random forest classifier into distinct local linear models extracted from the training set. Random forests involve growing and merging decision trees to form predictive models. The final prediction was made by voting from all the trees in the forest. The highest rank was selected to identify each frame's appropriate local linear model. The recovery was done by minimizing the least squared error using marker positions and principal components. However, the numerical stability was a significant challenge.

In [57], a locally weighted PCA regression method was proposed to address the issue in [56] and their previous work [58]. To their knowledge, it was the first least square method with sparsity constraints. They analyzed 3D skeletal motion data to address the "missing marker problem" in [59], a marker-based method that used PCA and segment coordination patterns in multi-limb motion data. This method was improved in [60] to remove noise.

In [61], a gap reconstruction method was proposed using NN without requiring massive training sequences to form a predictive model. Instead, they considered each sequence separately and tried to reconstruct the gaps in individual trajectories based on their own data. Their assumption was valid if the motion was correlated and most of the sequence was correct, like other common regression methods, starting with the least squares.

In [62], they combined low-rank matrix completion of the measured data with a group sparsity before the marker trajectories were mapped in the frequency domain. Compared to most existing approaches, the proposed methodology is fully unsupervised and does not need the user's training data or kinematic information.

2.4 MoCap Data Alignment

Registering human body scans aligns them with a common template. Alignment of articulated shapes like human bodies typically uses iterative closest point (ICP) to find marker correspondences in successive frames [63]–[68]. However, ICP-based methods are only effective when markers are present and the difference between successive frames is small. Most body alignment methods focus on aligning a template to different body shapes in a canonical pose [69]. In [70], LED markers were used instead of infrared markers to align an image with the camera pixel center based on the intensity distribution. Still, this method is not applicable to marker-based optical passive MoCap data.

In [71], a real-time online marker labeling algorithm was introduced to address the challenges of missing and ghost markers by utilizing point correspondence and graph matching methods, employing a soft graph matching model with the Hungarian algorithm [72] for finding the global optimal matching.

In [73], the alignments were regularized using an articulated 3D model of human shape and pose. Unlike the simple articulated model for initialization [74], [75], they used a richer, learned body shape model to register many different bodies in different poses accurately.

In [76], the skeleton tracking was initialized using a T-pose and adjusted by scaling the person’s size and aligning the skeleton to the subject’s limb. The observed markers were labeled, and the skeleton configuration was optimized in an expectation-maximization-like procedure. First, they used the highest marker observation to determine the person's height. Then, they matched the skeleton and person’s sizes by scaling. They identified specific points belonging to the legs and arms based on human anatomy and calculated their first principal axes. The skeleton model was aligned with these axes through least-squares optimization to establish the initial skeleton configuration. The optimization-based alignment made the initialization method robust to deviations from the ideal T-pose.

In [77], a fully automatic optical motion tracking method was introduced using a model-based inverse kinematics approach. The Hungarian method calculated associations between model markers and MoCap markers, while occlusions were handled using a posture interpolation step. In the initialization step, ghost markers were minimized to align models to motion data points, and all valid markers were visible in the first frame. The Euclidean cluster algorithm in the Point Cloud Library [78] was used in the automatic clustering step to cluster closely located motion data points. The algorithm worked well if the average distance between objects was larger than their size. Otherwise, manual correction might be necessary. This configuration could be saved and loaded for subsequent automated trials.

In [79], the skeleton was automatically replaced with a 3D body model by solving for marker locations relative to the body. They estimated the body shape and pose using sparse marker data without 3D scans. Despite the noisy treatment of non-rigid motions of soft tissue, they captured them from small marker sets (with 67 markers and missing fine details) to create a more realistic animation. The shape basis was learned from deformations of training body shapes using PCA. The pose-dependent component of the model was learned from a large set of scans, with various poses aligned using the technique in [73]. They optimized body shape and marker placement parameters using Powell's dogleg method [80] with Gauss-Newton Hessian approximation.

In [39], an auto-labeling method was presented using a differentiable permutation learning model. In preprocessing, the centroid of the marker array was calculated for each frame and subtracted from marker locations for translation invariance. PCA was used for orientation invariance by aligning the subject direction with the largest principal component with the z-axis. Rotations around the z-axis were invariant by aligning the second principal component with the x-axis. Subject size was normalized by scaling in three spatial dimensions. For training the model, they used Sinkhorn normalization [81] to convert any unconstrained non-negative matrix to a design structure matrix.

In [19], a state-of-the-art data-driven skeletal MoCap-solving technique was introduced that used a forward NN. Before training, as an assumption, the characters' height was normalized using a scaling factor computed from T-pose or extracted from MoCap software during calibration. A local reference frame was found to ensure accurate character representation in data-driven techniques. This involved using rigid body alignment [63] to describe the data without prior knowledge of joint transforms. They calculated the mean location of selected markers around the torso relative to a chosen joint (e.g., spine joints) to fit a rigid body into the data. This process was repeated for all poses in the dataset, resulting in a set of reference frames used to transform every pose into the local space. After training, the rigid body found during the preprocessing step was fitted to the subset of non-occluded markers to find a local reference frame, and the marker positions were transformed into it. Then, they fed the NN with these transformed markers to produce the joint transforms, which were subsequently converted back to the global reference frame.

MoCap-Solver [4] used separate NNs to produce skeletons to solve motions and reconstruct clean markers. They discussed that the alignment algorithm in [19] lacked robustness due to using a rigid-body registration algorithm to align poses in a local reference frame, and the precision was highly sensitive to corruption in specific markers, which limited its practical usage. To mitigate excessive dependence on a small number of key markers and enhance robustness, they normalized markers based on learning a pose-dependent marker reliability function. They automatically selected the most reliable frame for alignment and a global orientation. Their algorithm consistently outperformed the proposed method in [19] on both synthetic and real-world data.

In [13], a locality-based learning method using a graph NN was proposed to clean and solve labeled MoCap data. To expedite the training process, they aligned markers, similar to [4], to remove their global transformations by calculating the local coordinate systems of wrist and waist markers.

In [11], the first deep unsupervised human body reconstruction technique was introduced which utilized a denoising autoencoder to estimate missing landmarks and predict the body surface from a sparse set of landmarks. The impact of global orientation was addressed by focusing on data normalization. Their previous work [82] subtracted input landmarks from the mean point to achieve translation invariance. In this article, they reached a rotation invariant network, without scaling, by aligning all landmarks in the dataset to a reference set of landmarks using Procrustes analysis to transform landmarks by computing a rotation matrix and translation vector. Then, they used the aligned data to train the cascading network as before. Finally, they transformed the estimated surface to its original orientation at test time.

In [7], an auto-labeling approach was performed using a LSTM NN in conjunction with Hungarian label assignment. Initially, the marker coordinates were rotated around the vertical axis to ensure that the subject faced the positive x-direction at the beginning of the trial. During the training phase, this rotation process was automatically executed based on the acromion (shoulders) markers. However, for the test data, the rotation angle was calculated and input manually. Furthermore, Procrustes analysis was utilized to identify unlabeled markers by aligning local marker coordinates within the marker set with measured markers through scaling, rotation, and translation. This alignment process was used to assign labels to body segments that possessed a minimum of three markers, thereby leaving one or more markers unlabeled.

2.5 MoCap Data Auto-Labelling

Previously, there have been numerous efforts to automate the cleaning and solving of MoCap data. Early methods (e.g., moving average filter and low-rank matrix [10], Kalman smoothing and rank matrix completion [53], unscented Kalman filter and inverse kinematic [50], automatic kinematic model building based on Markov random field [83], multiple interacting articulated targets [84], AdaBoost [85], skeleton-based body models [86]–[88], and inverse kinematic [77]) mainly relied on rules based on empirical parameters and hand-crafted features. Although these approaches could produce acceptable outcomes for specific patterns and noise under assumptions and constraints, they consistently faced difficulties adapting to real-world data with intricate situations.

Data-driven methods have been employed to address the limitations above by learning from a large database, such as kd-tree [48], local PCA [56], self-similarity [29], sparse encoding [31], [15], model averaging [42], graph matching [71] [89], and deep learning-based approaches [90]–[92].

Most current research focuses on repairing occluded markers and solving motion. In [93], based on NNs, a data-driven real-time marker-based was used for finger marker recovery and tracking. In [94], labeling finger markers while simultaneously detecting occlusions and false observations (ghost markers) was done. In [16], a robust data-driven marker trajectory repair was proposed using kinematic reference. They compared their neural solver with those generated by commercial software. In [11], reconstruction and denoising autoencoder was used to estimate missing landmarks. Then, they used a cascading network to regress skinned multi-person linear body parameters [95] based on estimated joint positions using an attention model. However, using the model was challenging for characters with varying skeleton topologies.

In [39], an auto-labeling method with no manual initialization was presented using permutation learning model. For training the model, they used the Adam optimizer, a cross-entropy loss function, and Sinkhorn normalization to convert any unconstrained non-negative matrix to a design structure matrix. They added occlusion (missing markers) in their training data to improve the result. During execution, predicting a single permutation matrix was framed as a bipartite matching problem. Consequently, the Hungarian algorithm was applied to the cost matrix to determine the optimal solution. However, they were limited to a fixed number of markers in a restricted setup, with a subject-specific calibration stage, a limited range of motions (walk, jog, jump, and sit), and a unique marker set. Due to requiring high-quality MoCap training data, they faced challenges in their scalability to new scenarios.

In [7], an LSTM-based auto-labeling method was introduced in combination with Hungarian label assignment. Simulation trajectory and transfer learning were utilized to enhance the training set.

In [19], a skeletal MoCap solving method based on a forward NN with residual blocks was proposed. Using linear blend skinning, they used a set of marker configurations to synthetically reconstruct marker locations in a large skeletal MoCap database (e.g., CMU [96]). They calculated the mean and SD for joint transformations, marker configurations, and mean and covariance for marker locations. Pre-weighted local offsets and their mean and SD were also computed to feed into the NN to distinguish characters with different body proportions or marker placements. Then, the data were corrupted by a custom noise function to create training data. After training, outliers were eliminated and considered occlusions to be translation and rotation invariant. The network was fed with transformed markers to generate joint transforms, which were then converted back to the global reference frame. A Savitzky-Golay filter was used to remove jittery movements (quickly appearing or disappearing). Finally, in the retargeting stage, they utilized singular value decomposition to orthogonalize the rotational parts of joint transforms and extracted the local joint transformations using a Jacobian inverse kinematics solver.

Mocap-Solver [4] utilized distinct NNs for motion solving, achieving state-of-the-art results. However, it faced issues with outliers, occlusions, and complex movements. A heuristic removal of ghost markers restricted to a single body shape, and depended on high-quality real MoCap training data, limiting its scalability to new data.

In [8], an auto-labeling raw MoCap data was proposed using a NN. They removed outliers and ghost markers without calibration and with minimum user intervention. Unlike [4], which worked on labeled data and a single body shape and needed to compute the global orientation of the body, they worked with unlabeled points and dealt with varied body shapes. The assumption of graph isomorphism between MoCap frames and labels in Sinkhorn normalization was relaxed to allow for inexact matching between the labels and points using an optimal transport solution [97]. Due to a lack of real data, they created extensive simulated noisy training data and ground truth MoCap markers using AMASS [98]. They used the method proposed in [79] to fit SMPL-X [95] bodies to the labeled data to find body parameters and accurate marker placement on the body. They used dustbins

[99] to deal with missing and ghost markers. The dataset used for model selection and validation consisted of 215 sequences across four subjects with 40 markers on average. Their results outperformed prior works [77], [39], [100] under the same conditions. However, as with any learning-based method, they could not generalize for unseen motions.

In [13], the limitations of the state-of-the-art data-driven methods (e.g., [4], [19]) were classified into three problems. First, using skinning functions ignores the complexity of marker motions, and solving errors may lead to additional errors. Second, they often overlook the detailed correlations between markers by using a single fully connected network structure to encode all markers uniformly, leading to incorrect solutions for specific movements. Third, all methods must assume or model data noise using random sampling per frame, which overlooks long gaps and intense occlusion, decreasing accuracy.

To address the above issues, [13] proposed a locality-based learning method using a graph NN to clean and solve MoCap data and tracking errors. Unlike [4], they accurately reconstructed occluded markers by hand-crafted and learned intrinsic priors based on neighbouring markers' distance and a bidirectional LSTM network. Outliers, due to tracking errors, were detected by acceleration curves and replaced by simple spline interpolation. The training involved masking to simulate occluded and noisy markers commonly found in real data. Their alignment was similar to [4]. While the method in [8] assigned unlabeled markers to specific body parts, their method solved motions with labeled markers. Additionally, unlike [4] and [19], where the lack of quantification of local marker features hindered the successful resolution of motions, they extracted local features by constructing a heterogeneous graph that differentiated markers and joints as distinct node types, frame-by-frame and improved the accuracy by using graph convolution.

The placement of markers affects motion reconstruction, leading to labeling problems, unnatural animation, and inaccuracy. Labeling involves interpreting marker set data over timeframes to reconstruct captured models like humans or objects. This issue is investigated due to the time-consuming and costly labeling process for larger datasets, requiring commercial software and licenses (e.g., Qualisys and Vicon software). In [17], an optimal marker set configuration was proposed to improve the quality of MoCap data affected by factors like volume shape, motion between frames, ghost points, and self-occlusion of markers. They used a reversible-jump Markov chain Monte Carlo method to optimize the data, considering marker and camera placement constraints such as marker visibility, number of markers, symmetry of the marker set, and marker overlap. Researchers have also explored the optimal placement of cameras for MoCap systems [17], [68], [101]–[104].

2.6 Conclusion

We provided a comprehensive overview of various methods employed to tackle challenges associated with marker-based MoCap solving, including cleaning (denoising and recovery), alignment, and auto-labeling. While these systems address certain issues, they remain prone to inaccuracies due to their reliance on training data and limitations of feature-based methods. Thus, a more general solution applicable to various MoCap actions is still needed.

References

[1] G. B. Guerra-filho, “Optical motion capture: Theory and implementation,” *J. Theor. Appl. Informatics*, vol. 12, pp. 61--89, 2005, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.7248

[2] M. Menolotto, D. S. Komaris, S. Tedesco, B. O’flynn, and M. Walsh, “Motion capture technology in industrial applications: A systematic review,” *Sensors (Switzerland)*, vol. 20, no. 19, pp. 1–25, 2020, doi: 10.3390/s20195687.

[3] M. Kitagawa and B. Windsor, *MoCap for Artists Workflow and Techniques for Motion Capture*, no. 0. Elsevier Inc, 2008.

[4] K. Chen, Y. Wang, S. H. Zhang, S. Z. Xu, W. Zhang, and S. M. Hu, “MoCap-solver: A neural solver for optical motion capture data,” *ACM Trans. Graph.*, vol. 40, no. 4, 2021, doi: 10.1145/3450626.3459681.

[5] U. Mall, G. R. Lal, S. Chaudhuri, and P. Chaudhuri, “A Deep Recurrent Framework for Cleaning Motion Capture Data,” no. Figure 1, 2017, [Online]. Available: http://arxiv.org/abs/1712.03380

[6] E. Martini, S. Member, A. Calanca, and N. Bombieri, “Denoising and Completion Filters for Human Motion Software : a Survey with Code,” pp. 0–14, 2023, doi: 10.36227/techrxiv.22956482.v1.

[7] A. L. Clouthier, G. B. Ross, M. P. Mavor, I. Coll, A. Boyle, and R. B. Graham, “Development and Validation of a Deep Learning Algorithm and Open-Source Platform for the Automatic Labelling of Motion Capture Markers,” *IEEE Access*, vol. 9, pp. 36444–36454, 2021, doi: 10.1109/ACCESS.2021.3062748.

[8] N. Ghorbani and M. J. Black, "SOMA: Solving Optical Marker-Based MoCap Automatically," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 11097–11106, 2021, doi: 10.1109/ICCV48922.2021.01093.

[9] R. Lai, P. Yuen, and K. Lee, "Motion capture data completion and denoising by singular value thresholding," *Proc. Eurographics Assoc.*, pp. 1–4, 2011, [Online]. Available: http://www.comp.hkbu.edu.hk/~yqlai/images/egfinal.pdf

[10] X. Liu, Y. M. Cheung, S. J. Peng, Z. Cui, B. Zhong, and J. X. Du, "Automatic motion capture data denoising via filtered subspace clustering and low rank matrix approximation," *Signal Processing*, vol. 105, pp. 350–362, 2014, doi: 10.1016/j.sigpro.2014.06.009.

[11] M. Madadi, H. Bertiche, and S. Escalera, "Deep Unsupervised 3D Human Body Reconstruction from a Sparse set of Landmarks," *Int. J. Comput. Vis.*, vol. 129, no. 8, pp. 2499–2512, 2021, doi: 10.1007/s11263-021-01488-2.

[12] P. Skurowski and M. Pawlyta, "Detection and Classification of Artifact Distortions in Optical Motion Capture Sequences," *Sensors*, vol. 22, no. 11, pp. 1–29, 2022, doi: 10.3390/s22114076.

[13] X. Pan *et al.*, "A Locality-based Neural Solver for Optical Motion Capture," 2023, doi: 10.1145/3610548.3618148.

[14] Y. Feng *et al.*, "Mining Spatial-Temporal Patterns and Structural Sparsity for Human Motion Data Denoising," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2693–2706, 2015, doi: 10.1109/TCYB.2014.2381659.

[15] S. Alexanderson, C. O'Sullivan, and J. Beskow, "Real-time labeling of non-rigid motion capture marker sets," *Comput. Graph.*, vol. 69, pp. 59–67, 2017, doi:

10.1016/j.cag.2017.10.001.

[16] M. Perepichka, D. Holden, S. P. Mudur, and T. Popa, “Robust marker trajectory repair for MOCAP using kinematic reference,” *Proc. - MIG 2019 ACM Conf. Motion, Interact. Games*, 2019, doi: 10.1145/3359566.3360060.

[17] P. Acevedo, B. Rekabdar, and C. Mousas, “Optimizing retroreflective marker set for motion capturing props,” *Comput. Graph.*, vol. 115, pp. 181–190, 2023, doi: 10.1016/j.cag.2023.07.021.

[18] J. Barca, G. Rumantir, and R. Li, “Noise Filtering of New Motion Capture Markers Using Modified K-Means,” vol. 96, no. April 2008, pp. 79–98, 2008, doi: 10.1007/978-3-540-76827-2.

[19] D. Holden, “Robust solving of optical motion capture data by denoising,” *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–12, 2018, doi: 10.1145/3197517.3201302.

[20] Qualisys, “Qualisys Track Manager,” *Qualisys*, 2011. https://www.qualisys.com/software/qualisys-track-manager/ (accessed Dec. 10, 2023).

[21] Qualisys, “Smoothing your data,” *Qualisys*. https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/smoothing_your_data/smoothing_your_data.htm (accessed Jan. 12, 2024).

[22] Qualisys, “Smoothing types,” *Qualisys*. https://docs.qualisys.com/getting-started/content/37_trajectory_editor_series/37c_how_to_use_the_trajectory_editor_-_smoothing/smoothing_types.htm?Highlight=smoothing types (accessed Aug. 11, 2024).

[23] P. Skurowski and M. Pawlyta, “On the noise complexity in an optical motion capture

facility," *Sensors (Switzerland)*, vol. 19, no. 20, pp. 1–30, 2019, doi: 10.3390/s19204435.

[24] M. M. Ardestani and H. Yan, "Noise Reduction in Human Motion-Captured Signals for Computer Animation based on B-Spline Filtering," *Sensors*, vol. 22, no. 12, 2022, doi: 10.3390/s22124629.

[25] H. Lou and J. Chai, "Example-based human motion denoising," *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 5, pp. 870–879, 2010, doi: 10.1109/TVCG.2010.23.

[26] L. Ren, A. Patrick, A. A. Efros, J. K. Hodgins, and J. M. Rehg, "A data-driven approach to quantifying natural human motion," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1090–1097, 2005, doi: 10.1145/1073204.1073316.

[27] N. F. Troje, "Decomposing biological motion: A framework for analysis and synthesis of human gait patterns," *J. Vis.*, vol. 2, no. 5, pp. 371–387, 2002, doi: 10.1167/2.5.2.

[28] W. Kim and J. M. Rehg, "Detection of unnatural movement using epitomic analysis," *Proc. - 7th Int. Conf. Mach. Learn. Appl. ICMLA 2008*, pp. 271–276, 2008, doi: 10.1109/ICMLA.2008.138.

[29] A. Aristidou, D. Cohen-Or, J. K. Hodgins, and A. Shamir, "Self-similarity analysis for motion capture cleaning," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 297–309, 2018, doi: 10.1111/cgf.13362.

[30] H. Yasin, S. Ghani, and B. Kruger, "An Effective and Efficient Approach for 3D Recovery of Human Motion Capture Data," *Sensors*, vol. 23, p. 3664, 2023, doi: 10.3390/s23073664.

[31] J. Xiao, Y. Feng, M. Ji, X. Yang, J. J. Zhang, and Y. Zhuang, "Sparse motion bases

selection for human motion denoising," *Signal Processing*, vol. 110, pp. 108–122, 2015, doi: 10.1016/j.sigpro.2014.08.017.

[32] J. Xiao, Y. Feng, and W. Hu, "Predicting missing markers in human motion capture using l1-sparse representation," *Comput. Animat. Virtual Worlds*, vol. 22, no. April, pp. 221–228, 2011, doi: 10.1002/cav.413.

[33] W. Hu, X. Zhu, T. Wang, Y. Yi, and G. Yu, "Discrete subspace structure constrained human motion capture data recovery," *Appl. Soft Comput.*, vol. 129, p. 109617, 2022, doi: 10.1016/j.asoc.2022.109617.

[34] J. Fan, C. Yang, and M. Udell, "Robust non-linear matrix factorization for dictionary learning, denoising, and clustering," *IEEE Trans. Signal Process.*, vol. 69, pp. 1755–1770, 2021, doi: 10.1109/TSP.2021.3062988.

[35] T. Kucherenko, J. Beskow, and H. Kjellström, "A Neural Network Approach to Missing Marker Reconstruction in Human Motion Capture," 2018, [Online]. Available: http://arxiv.org/abs/1803.02665

[36] Y. Zhu, "Denoising method of motion capture data based on neural network," *J. Phys. Conf. Ser.*, vol. 1650, no. 3, 2020, doi: 10.1088/1742-6596/1650/3/032068.

[37] S. J. Li, H. S. Zhu, L. P. Zheng, and L. Li, "A Perceptual-Based Noise-Agnostic 3D Skeleton Motion Data Refinement Network," *IEEE Access*, vol. 8, pp. 52927–52940, 2020, doi: 10.1109/ACCESS.2020.2980316.

[38] S. Li, Y. Zhou, H. Zhu, W. Xie, Y. Zhao, and X. Liu, "Bidirectional recurrent autoencoder for 3D skeleton motion data refinement," *Comput. Graph.*, vol. 81, pp. 92–103, 2019, doi: 10.1016/j.cag.2019.03.010.

[39] S. Ghorbani, A. Etemad, and N. F. Troje, *Auto-labelling of Markers in Optical*

Motion Capture by Permutation Learning, vol. 11542 LNCS. Springer International Publishing, 2019. doi: 10.1007/978-3-030-22514-8_14.

[40] G. Albanis, N. Zioulis, S. Thermos, A. Chatzitofis, and K. Kolomvatsos, "Noise-in, Bias-out: Balanced and Real-time MoCap Solving," 2023, [Online]. Available: http://arxiv.org/abs/2309.14330

[41] K. Zhou, Z. Cheng, H. P. H. Shum, F. W. B. Li, and X. Liang, "StgAE: Spatial-temporal graph auto-encoder for hand motion denoising," *Proc. - 2021 IEEE Int. Symp. Mix. Augment. Reality, ISMAR 2021*, pp. 41–49, 2021, doi: 10.1109/ISMAR52148.2021.00018.

[42] M. Tits, J. Tilmanne, and T. Dutoit, "Robust and automatic motion-capture data recovery using soft skeleton constraints and model averaging," *PLoS One*, vol. 13, no. 7, pp. 1–21, 2018, doi: 10.1371/journal.pone.0199744.

[43] C. H. Tan, J. Hou, and L. P. Chau, "Human motion capture data recovery using trajectory-based matrix completion," *Electron. Lett.*, vol. 49, no. 12, pp. 752–754, 2013, doi: 10.1049/el.2013.0442.

[44] B. Hong, L. Wei, Y. Hu, D. Cai, and X. He, "Online robust principal component analysis via truncated nuclear norm regularization," *Neurocomputing*, vol. 175, no. PartA, pp. 216–222, 2015, doi: 10.1016/j.neucom.2015.10.052.

[45] W. Hu, Z. Wang, S. Liu, X. Yang, G. Yu, and J. J. Zhang, "Motion Capture Data Completion via Truncated Nuclear Norm Regularization," *IEEE Signal Process. Lett.*, vol. 25, no. 2, pp. 258–262, 2018, doi: 10.1109/LSP.2017.2687044.

[46] W. Yin, H. Yin, D. Kragic, and M. Bjorkman, "Graph-based normalizing flow for human motion generation and reconstruction," *2021 30th IEEE Int. Conf. Robot*

Hum. Interact. Commun. RO-MAN 2021, pp. 641–648, 2021, doi: 10.1109/RO-MAN50785.2021.9515316.

[47] S. Lohit, R. Anirudh, and P. Turaga, “Recovering trajectories of unmarked joints in 3d human actions using latent space optimization,” *Proc. - 2021 IEEE Winter Conf. Appl. Comput. Vision, WACV 2021*, pp. 2341–2350, 2021, doi: 10.1109/WACV48630.2021.00239.

[48] J. Baumann, B. Krüger, A. Zinke, and A. Weber, “Data-driven completion of motion capture data,” *VRIPHYS 2011 - 8th Work. Virtual Real. Interact. Phys. Simulations*, no. January, pp. 111–118, 2011, doi: 10.2312/PE/vriphys/vriphys11/111-118.

[49] G. Xia, H. Sun, G. Zhang, and L. Feng, “Human motion recovery jointly utilizing statistical and kinematic information,” *Inf. Sci. (Ny).*, vol. 339, pp. 189–205, 2016, doi: 10.1016/j.ins.2015.12.041.

[50] A. Aristidou and J. Lasenby, “Real-time marker prediction and CoR estimation in optical motion capture,” *Vis. Comput.*, vol. 29, no. 1, pp. 7–26, 2013, doi: 10.1007/s00371-011-0671-y.

[51] A. Aristidou, J. Cameron, and J. Lasenby, “Real-time estimation of missing markers in human motion capture,” *2nd Int. Conf. Bioinforma. Biomed. Eng. iCBBE 2008*, pp. 1343–1346, 2008, doi: 10.1109/ICBBE.2008.665.

[52] S. Chatterjee, “Procrustes Problems,” *Technometrics*, vol. 47, no. 3, pp. 376–376, 2005, doi: 10.1198/tech.2005.s296.

[53] M. Burke and J. Lasenby, “Estimating missing marker positions using low dimensional Kalman smoothing,” *J. Biomech.*, vol. 49, no. 9, pp. 1854–1858, 2016, doi: 10.1016/j.jbiomech.2016.04.016.

[54] D. Gomes, V. Guimarães, and J. Silva, “A fully-automatic gap filling approach for motion capture trajectories,” *Appl. Sci.*, vol. 11, no. 21, 2021, doi: 10.3390/app11219847.

[55] K. Dorfmüller-Ulhaas, “Robust Optical User Motion Tracking Using a Kalman Filter,” *10th ACM Symp. Virtual Real. Softw. Technol.*, no. May, 2003, [Online]. Available: https://opus.bibliothek.uni-augsburg.de/opus4/frontdoor/index/index/year/2007/docId/584

[56] G. Liu and L. McMillan, “Estimation of missing markers in human motion capture,” *Vis. Comput.*, vol. 22, no. 9–11, pp. 721–728, 2006, doi: 10.1007/s00371-006-0080-9.

[57] H. D. Kieu, H. Yu, Z. Li, and J. J. Zhang, “Locally weighted PCA regression to recover missing markers in human motion data,” *PLoS One*, vol. 17, no. 8 August, pp. 1–13, 2022, doi: 10.1371/journal.pone.0272407.

[58] Z. Li, H. Yu, H. D. Kieu, T. L. Vuong, and J. J. Zhang, “PCA-Based robust motion data recovery,” *IEEE Access*, vol. 8, no. 1, pp. 76980–76990, 2020, doi: 10.1109/ACCESS.2020.2989744.

[59] P. A. Federolf, “A novel approach to solve the ‘missing marker problem’ in marker-based motion analysis that exploits the segment coordination patterns in multi-limb motion,” *PLoS One*, vol. 8, no. 10, 2013, doi: 10.1371/journal.pone.0078689.

[60] Ø. Gløersen and P. Federolf, “Predicting missing marker trajectories in human motion data using marker intercorrelations,” *PLoS One*, vol. 11, no. 3, 2016, doi: 10.1371/journal.pone.0152616.

[61] P. Skurowski and M. Pawlyta, “Gap reconstruction in optical motion capture

sequences using neural networks," *Sensors*, vol. 21, no. 18, pp. 1–26, 2021, doi: 10.3390/s21186115.

[62] K. Kamali, A. A. Akbari, C. Desrosiers, A. Akbarzadeh, M. J. D. Otis, and J. C. Ayena, "Low-rank and sparse recovery of human gait data," *Sensors (Switzerland)*, vol. 20, no. 16, pp. 1–13, 2020, doi: 10.3390/s20164525.

[63] T. Theodoridis and J. Kraemer, "A Method for Registration of 3-D Shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992, doi: 10.1109/34.121791.

[64] A. Tagliasacchi, M. Schröder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly, "Robust articulated-ICP for real-time hand tracking," *Eurographics Symp. Geom. Process.*, vol. 34, no. 5, pp. 101–114, 2015, doi: 10.1111/cgf.12700.

[65] S. Ji, Y. Ren, Z. Ji, X. Liu, and G. Hong, "An improved method for registration of point cloud," *Optik (Stuttg).*, vol. 140, pp. 451–458, 2017, doi: 10.1016/j.ijleo.2017.01.041.

[66] Y. Sahillioğlu and L. Kavan, "Scale-Adaptive ICP," *Graph. Models*, vol. 116, no. May, 2021, doi: 10.1016/j.gmod.2021.101113.

[67] R. Marin, S. Melzi, E. Rodolà, and U. Castellani, "FARM: Functional Automatic Registration Method for 3D Human Bodies," *Comput. Graph. Forum*, vol. 39, no. 1, pp. 160–173, 2020, doi: 10.1111/cgf.13751.

[68] A. Chatzitofis, D. Zarpalas, P. Daras, and S. Kollias, "DeMoCap : Low-Cost Marker-Based Motion Capture," pp. 3338–3366, 2021, doi: 10.1007/s11263-021-01526-z.

[69] D. A. Hirshberg *et al.*, "Evaluating the Automated Alignment of 3D Human Body

Scans," no. October, pp. 76–86, 2011, doi: 10.15221/11.076.

[70] I. Stancic, T. G. Supuk, and A. Panjkota, "Design, development and evaluation of optical motion-tracking system based on active white light markers," *IET Sci. Meas. Technol.*, vol. 7, no. 4, pp. 206–214, 2013, doi: 10.1049/iet-smt.2012.0157.

[71] S. Xia, L. Su, X. Fei, and H. Wang, "Toward accurate real-time marker labeling for live optical motion capture," *Vis. Comput.*, vol. 33, no. 6–8, pp. 993–1003, 2017, doi: 10.1007/s00371-017-1400-y.

[72] H. W. Kuhn, "The Hungarian method for the assignment problem," *50 Years Integer Program. 1958-2008 From Early Years to State-of-the-Art*, vol. 2, no. 1, pp. 29–47, 2010, doi: 10.1007/978-3-540-68279-0_2.

[73] D. A. Hirshberg, M. Loper, E. Rachlin, and M. J. Black, "Coregistration: Simultaneous alignment and modeling of articulated 3D shape," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7577 LNCS, no. PART 6, pp. 242–255, 2012, doi: 10.1007/978-3-642-33783-3_18.

[74] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H. P. Seidel, "A statistical model of human pose and body shape," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 337–346, 2009, doi: 10.1111/j.1467-8659.2009.01373.x.

[75] S. Wuhrer, C. Shu, and P. Xi, "Landmark-free posture invariant human shape correspondence," *Vis. Comput.*, vol. 27, no. 9, pp. 843–852, 2011, doi: 10.1007/s00371-011-0557-z.

[76] J. Meyer, M. Kuderer, J. Muller, and W. Burgard, "Online marker labeling for fully automatic skeleton Tracking in optical motion capture," *Proc. - IEEE Int. Conf.*

Robot. Autom., no. May 2014, pp. 5652–5657, 2014, doi: 10.1109/ICRA.2014.6907690.

[77] J. Maycock, T. Röhlig, M. Schröder, M. Botsch, and H. Ritter, "Fully automatic optical motion tracking using an inverse kinematics approach," *IEEE-RAS Int. Conf. Humanoid Robot.*, vol. 2015-Decem, pp. 461–466, 2015, doi: 10.1109/HUMANOIDS.2015.7363590.

[78] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," *Robot. Autom. (ICRA), 2011 IEEE Int. Conf.*, pp. 1–4, 2011, doi: 10.1073/pnas.74.3.1167.

[79] M. Loper, N. Mahmoody, and M. J. Blackz, "MoSh: Motion and shape capture from sparse markers," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–13, 2014, doi: 10.1145/2661229.2661273.

[80] K. Robinson and S. Gatehouse, *(book)Numerical Optimization(2rd)*, vol. 17, no. 2. 2006.

[81] R. P. Adams and R. S. Zemel, "Ranking via Sinkhorn Propagation," pp. 1–12, 2011, [Online]. Available: http://arxiv.org/abs/1106.1925

[82] M. Madadi, H. Bertiche, and S. Escalera, "SMPLR : Deep SMPL reverse for 3D human pose and shape recovery," *arXiv:1812.10766v2*, 2019.

[83] S. Rajko and G. Qian, "Real-time automatic kinematic model building for optical motion capture using a markov random field," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4796 LNCS, pp. 69–78, 2007, doi: 10.1007/978-3-540-75773-3_8.

[84] Q. Yu, Q. Li, and Z. Deng, "Online motion capture marker labeling for multiple interacting articulated targets," *Comput. Graph. Forum*, vol. 26, no. 3, pp. 477–483,

2007, doi: 10.1111/j.1467-8659.2007.01070.x.

[85] J. L. Jiménez Bascones, M. Graña, and J. M. Lopez-Guede, “Robust labeling of human motion markers in the presence of occlusions,” *Neurocomputing*, vol. 353, pp. 96–105, 2019, doi: 10.1016/j.neucom.2018.05.132.

[86] M. Ringer and J. Lasenby, “A procedure for automatically estimating model parameters in optical motion capture,” *Image Vis. Comput.*, vol. 22, no. 10 SPEC. ISS., pp. 843–850, 2004, doi: 10.1016/j.imavis.2004.02.011.

[87] A. Cuevas, J. Rodriguez-Navarro, and A. Susín, “Auto-labeling as a minimization problem with virtual occlusions,” *18th Spanish Comput. Graph. Conf. CEIG 2008*, vol. 5, pp. 133–139, 2008.

[88] C. Schönauer, T. Pintaric, and H. Kaufmann, “Full Body Motion Capture A Flexible Marker-Based Solution,” 2011.

[89] J. Li, D. Xiao, K. Li, and J. Li, “Graph matching for marker labeling and missing marker reconstruction with bone constraint by LSTM in optical motion capture,” *IEEE Access*, vol. 9, pp. 34868–34881, 2021, doi: 10.1109/ACCESS.2021.3060385.

[90] S. Han, B. Liu, R. Wang, Y. Ye, C. D. Twigg, and K. Kin, “Online optical marker-based hand tracking with deep labels,” *ACM Trans. Graph.*, vol. 37, no. 4, 2018, doi: 10.1145/3197517.3201399.

[91] J. Bütepage, M. J. Black, D. Kragic, and H. Kjellström, “Deep representation learning for human motion prediction and classification,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1591–1599, 2017, doi: 10.1109/CVPR.2017.173.

[92] S. Graßhof, M. Bastholm, and S. S. Brandt, “Neural Network-Based Human Motion

Predictor and Smoother," *SN Comput. Sci.*, vol. 4, no. 6, 2023, doi: 10.1007/s42979-023-02195-0.

[93] D. Pavllo, T. Porssut, B. Herbelin, and R. Boulic, "Real-Time Marker-Based Finger Tracking with Neural Networks," *25th IEEE Conf. Virtual Real. 3D User Interfaces, VR 2018 - Proc.*, pp. 651–652, 2018, doi: 10.1109/VR.2018.8446173.

[94] S. Alexanderson, C. O'Sullivan, and J. Beskow, "Robust online motion capture labeling of finger markers," *Proc. - Motion Games 2016 9th Int. Conf. Motion Games, MIG 2016*, pp. 7–13, 2016, doi: 10.1145/2994258.2994264.

[95] G. Pavlakos *et al.*, "Expressive body capture: 3D hands, face, and body from a single image," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 10967–10977, 2019, doi: 10.1109/CVPR.2019.01123.

[96] CMU, "CMU Graphics Lab Motion Capture Database," *CMU*. http://mocap.cs.cmu.edu/ (accessed Dec. 16, 2023).

[97] M. Berger *et al.*, *Optimal transport – Old and New*. 2009.

[98] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. Black, "AMASS: Archive of motion capture as surface shapes," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-Octob, pp. 5441–5450, 2019, doi: 10.1109/ICCV.2019.00554.

[99] P. E. Sarlin, D. Detone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning Feature Matching with Graph Neural Networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4937–4946, 2020, doi: 10.1109/CVPR42600.2020.00499.

[100] S. Holzreiter, "Autolabeling 3D tracks using neural networks," *Clin. Biomech.*, vol. 20, no. 1, pp. 1–8, 2005, doi: 10.1016/j.clinbiomech.2004.04.006.

[101] X. Chen and J. Davis, "Camera Placement Considering Occlusion for Robust Motion Capture," *Comput. Graph. Lab. Stanford Univ. Tech. Rep*, vol. 2, no. 2.2, p. 2, 2000, [Online]. Available: http://graphics.stanford.edu/papers/OcclusionMetric/occlusion_metric.pdf

[102] P. Rahimian and J. K. Kearney, "Optimal camera placement for motion capture systems in the presence of dynamic occlusion," *Proc. ACM Symp. Virtual Real. Softw. Technol. VRST*, vol. 13-15-Nove, no. May, pp. 129–138, 2015, doi: 10.1145/2821592.2821596.

[103] P. Rahimian and J. K. Kearney, "Optimal Camera Placement for Motion Capture Systems," vol. 23, no. 3, pp. 1209–1221, 2017.

[104] P. Eichelberger *et al.*, "Analysis of accuracy in optical motion capture – A protocol for laboratory setup evaluation," *J. Biomech.*, vol. 49, no. 10, pp. 2085–2088, 2016, doi: 10.1016/j.jbiomech.2016.05.007.

3. Marker-based Underwater Optical Motion Capture Data Preparation

Abstract

This article presents the steps of motion capture (MoCap) data preparation for marker-based underwater optical motion capture (OMC) systems, including attaching markers to the subject, setting up the capturing volume by placing cameras, camera calibration, recording the session, creating a marker set, and manual cleaning the exported C3D data by denoising, recovering missing markers, and labeling, which are steps of MoCap solving process. The cleaned C3D output file can be utilized in various applications, such as ground truth data for evaluating automatic MoCap solving algorithms or as a training or validation dataset for machine learning and deep learning algorithms. We utilized the Qualisys Track Manager (QTM) software to capture MoCap data using Qualisys underwater Miqus M5U MoCap cameras, the first commercially available OMC cameras for aquatic environments. We present a manual cleaning procedure for a subset of our raw dataset, called *Dataset A*. The cleaned output, *Dataset A_cleaned* is used to generate a QTM automatic identification of markers (AIM) model, which is then applied to the raw *Dataset A* for comparison with the manual cleaned output. We discuss the challenges of manual cleaning and using the AIM model, highlighting the need for automated algorithms to streamline this laborious process. However, manual cleaning and verification of data remains unavoidable.

3.1 Introduction

Marker-based optical motion capture (OMC) [1] systems utilize cameras to measure the 3D locations of reflective markers on the subject's body. This technology has been applied in various environments, including underwater settings, where challenges from light refraction and distortion, among other issues unique to underwater environments, introduce new challenges [2]. Established commercial tools from MoCap companies such as Qualisys [3] and Vicon [4] are available to support MoCap with proprietary equipment. These MoCap systems are designed for data collection, management, and analysis for MoCap applications [5].

Qualisys Track Manager (QTM) [6], [7] is the software developed by Qualisys for motion capture. It is compatible with various Qualisys MoCap cameras [8], including the Miqus M5U underwater MoCap camera [9], specifically designed for underwater measurement. It can be used in underwater applications [10], [11], such as in-water rehabilitation using underwater treadmills [12], underwater animation [13], and swimming performance analysis [14], as shown in Figure 3-1.

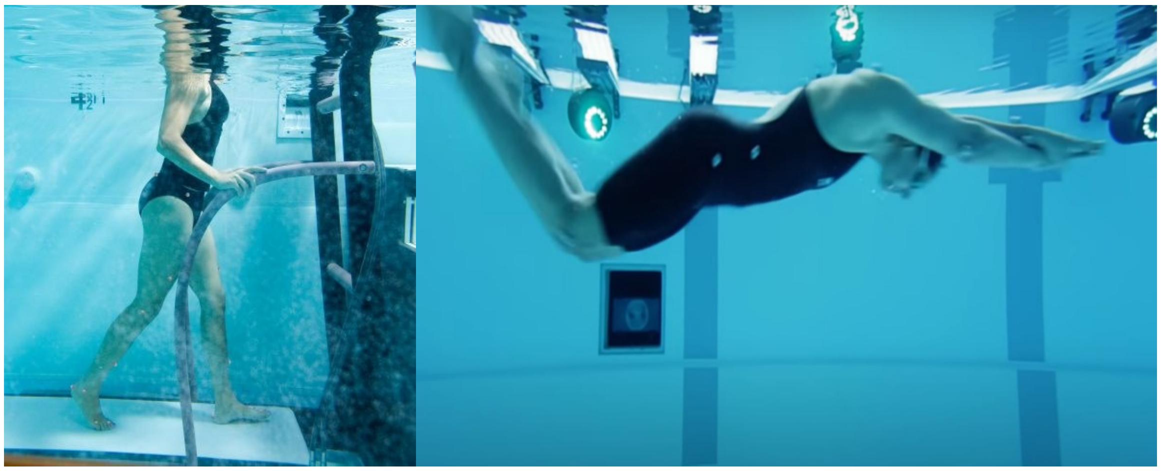


Figure 3-1: In-water rehabilitation and swimming performance.

This article outlines the steps for preparing marker-based underwater OMC data using 7 Qualisys Miqus M5U underwater MoCap cameras and QTM software. The process includes attaching 21 passive markers to the swimmer suit, camera placement and calibration, recording data at 100Hz, and manually cleaning the captured raw C3D file.

The paper is organized as follows. Section 3.2 provides a review of the research utilizing the Qualisys MoCap system for different underwater applications. Section 3.3 describes our setup for capturing underwater MoCap data, including reflective marker attachment, camera placement, calibration, and recording data using QTM software. Section 3.4 describes noise in underwater MoCap data. Section 3.5 introduces MoCap data cleaning and editing software tools such as commercial software, Mokka software [15], biomechanical analysis software, and MATLAB and Python C3D libraries. Sections 3.6 and 3.7 explain the marker set labels and the marker set creation using OpenSim [16], open-source biomechanical modelling software. Section 3.8 explains our process for manual cleaning of our raw MoCap *Dataset A*, including denoising, recovery, and labeling methods using QTM software. Section 3.9 presents our data cleaning procedure using the QTM automatic identification of markers (AIM) model [17]–[19]. Sections 3.10 introduces a brief introduction of some other QTM features such as skeleton solver [20]–[25], and rigid body and Euler angles [26]–[28]. Section 3.11 concludes the paper.

3.2 Underwater Application

To the best of our knowledge, there are currently no published research studies using Qualisys Miqus underwater MoCap cameras, specifically Miqus M5U, for human kinematics analysis within OMC systems. Since their introduction in 2019 [11], these cameras have been used primarily for tracking marine vessels and structures due to their compact size and wide-angle capabilities, which make them ideal for confined spaces [29]. Their applications include monitoring underwater autonomous vehicles [30], robotics [31], objects in towing tanks, animal biomechanical analysis [32], and movements of oil pipelines [33]. Tortorici *et al.* [30] utilized five Miqus M5U cameras operating at 180 frames per second to track remotely operated vehicles markers. Lack *et al.* [34] used 8 Miqus M5U to control a small underwater vehicle manipulator systems. Researchers [35]–[37] measured the kinematics and surface electromyography of human lower limbs using a Miqus underwater video camera.

There are a number of studies on swimming performance and underwater gait analysis using other Qualisys underwater MoCap cameras [9], such as the Oqus [38], which is not suitable for confined spaces. The Oqus brand has been discontinued [39] and replaced by the newer Arqus [40] camera model. Lauer *et al.* [41] utilized 12 Qualisys underwater MoCap cameras to measure upper limb joint forces and moments during underwater cyclical movements. The study found that the upper limb joint load was within 5% of the swimmer's body weight. This suggests that low-load aquatic exercises could be beneficial in reducing joint stress in aquatic therapy and rehabilitation. Olstad et al. [42]–[44] examined the kinematics of breaststroke swimmers utilizing Qualisys underwater MoCap Oqus 3 and 4 cameras functioning at 100 Hz, which recorded retro-reflective passive markers. They employed QTM software versions 2.6 and 2.8. Ribeiro *et al.* [45] investigated how swimmers with different speeds organize factors such as biomechanics, energy, and coordination during extreme-intensity swims. Kinematics parameters were assessed using seven land plus eight underwater cameras (Oqus 3+ and Oqus Underwater) operating at 60 Hz. Abdul Jabbar *et al.* [46] researched the effect of aquatic medium and age on the lower limb's joint angles. Underwater gait kinematic data were acquired at 100 Hz using 8 Oqus underwater MoCap cameras. Washino *et al.* [47] suggested that swimming performance is reduced, at least in part, due to additional drag caused by reflective markers. They utilized 25 Qualisys reflective markers. However, they recorded the MoCap data by a digital video camera positioned about 5 meters above the water and 30 meters away from the swimming lane. Chainok *et al.* [48] sought to identify biomechanical features of transitioning from backstroke to breaststroke using a dual-media MoCap system, which included 12 land and 11 underwater cameras (Oqus 3 and 4 series) along with 51 spherical retroreflective markers. Tanaka *et al.* [49] compared foot and trunk kinematic parameters during underwater undulatory swimming between faster and slower swimmers using 8 Qualisys underwater cameras and 12 retro-reflective markers. Nakashima *et al.* [50] developed of a method for musculoskeletal simulation in swimming using dual media with 18 Qualisys underwater cameras and 48 reflective markers. Qualisys [10] offers a promotional video of their underwater systems and software, demonstrating swimming biomechanics. However, these studies all notably focus on the subject at the surface of the water, completing only predictable and pre-defined activities (i.e., swimming strokes).

3.3 Capturing Underwater MoCap Data

The marker-based OMC system setup involves attaching markers on the body, placing cameras in the volume of interest, and camera calibration. We captured underwater MoCap data using seven Qualisys Miqus M5U underwater MoCap cameras installed at different locations around a four-meter-deep, 18-meter by 14-meter pool at the Memorial University

Marine Institute Offshore Safety and Survival Centre, as shown in Figure 3-2. A total of 21 passive (reflective) markers were attached to the swimmer suit and body, as shown in Figure 3-3. After calibration, data were recorded at 100Hz using QTM software and exported to C3D files. The subsequent sections detail visualizing markers attached to the swimmer's body, identifying the area of interest, positioning the cameras for optimal results, performing camera calibration, and recording MoCap data.

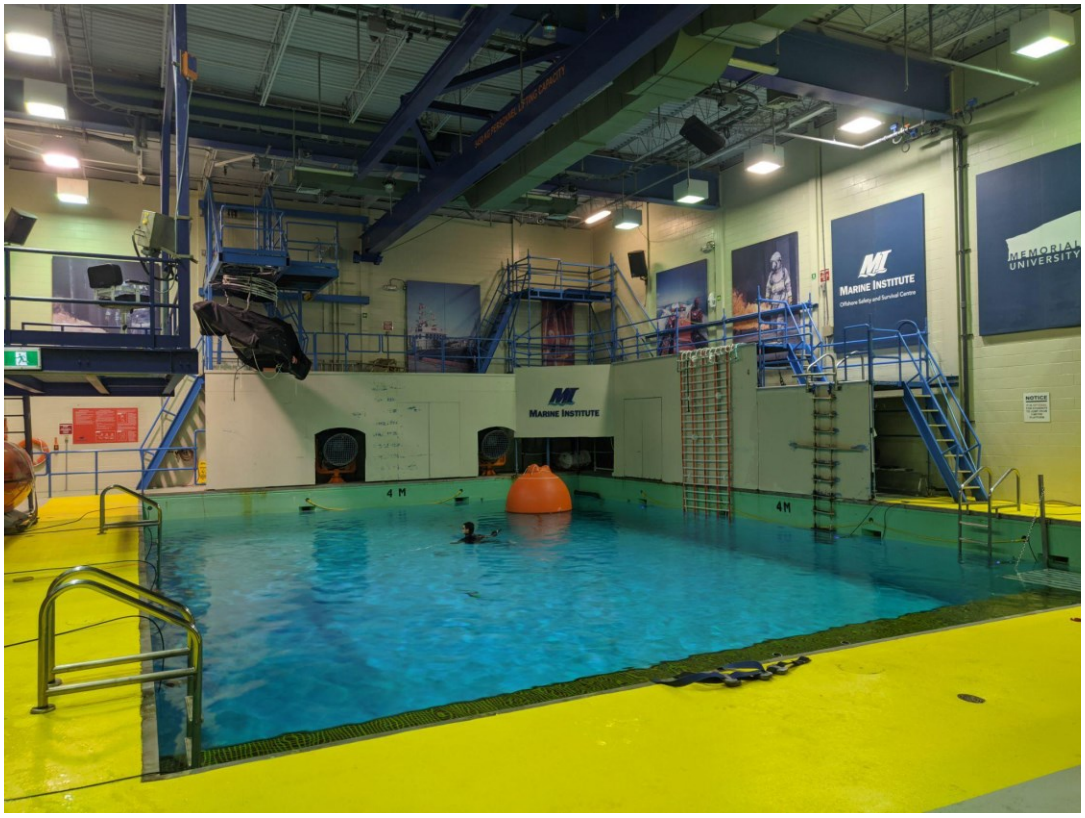


Figure 3-2: A 4-meter deep pool utilized for capturing MoCap data.

3.3.1 Marker Attachment

In this section, we will first provide a visualization of the anatomical locations where we attached the 21 Qualisys passive markers to the subjects' bodies. Following that, we will introduce the Qualisys markers specifications.

3.3.1.1 Passive Marker Anatomical Locations

We attached 21 reflective super-spherical MoCap markers on the swimmer's body and suit based on the Qualisys sports marker set [51], as shown in Figure 3-3.

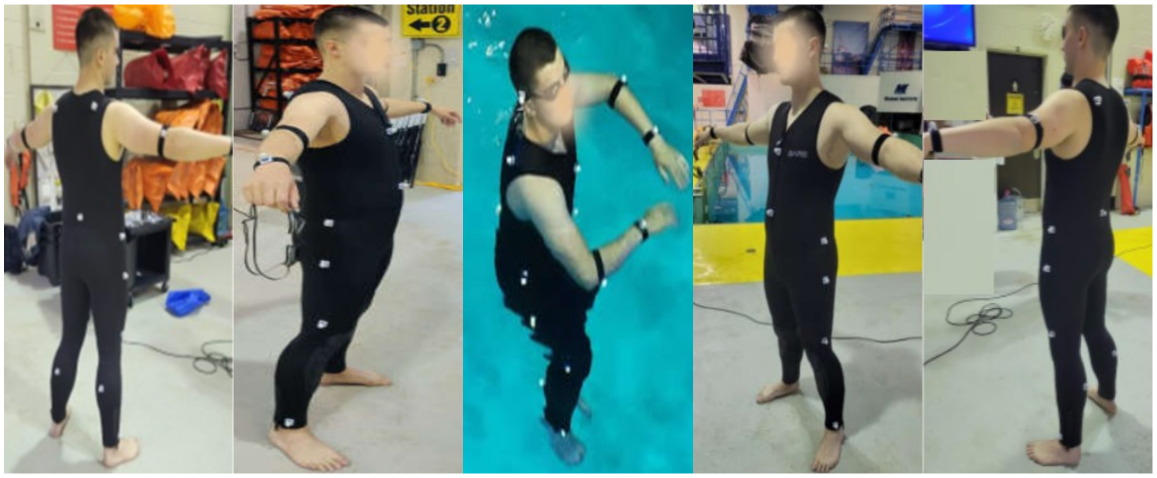


Figure 3-3: 21 Passive Marker Anatomical Locations.

3.3.1.2 Qualisys Passive Markers

Qualisys offers different types of passive markers for underwater OMC tracking systems [52], including hand-coated underwater markers, super-spherical underwater markers, super-spherical MoCap markers, and retro-reflective underwater tape. The selection of marker size depends on several factors: (1) the setup, which includes available space on the object and streaming characteristics; (2) the distance between the markers and cameras, influenced by camera type—such as the Arqus UW [40], which allows for smaller markers at greater distances due to its strong strobe; and (3) water quality, which can affect visibility. Generally, larger markers provide a greater reflective surface area, making them easier to detect in various underwater conditions. Ordinary passive markers are ineffective underwater as their retro-reflective properties diminish. Hand-coated underwater markers are water-resistant, threaded markers with a spherical body covered in retro-reflective tape, with a minimum size of 12 mm. They can remain submerged permanently. However, in some cases, the taped markers may break apart and appear as multiple markers. There for, super-spherical underwater markers, that can be for a long-time underwater, are recommended, with a minimum size of 14 mm. If a smaller size is needed, super-spherical MoCap markers can be used, with minimum size of 6.5 mm; however, they will fail after extended periods (over 1 hour) of submersion. Moreover, retro-reflective underwater tape serves as an alternative to ordinary spherical underwater markers. It is particularly useful in scenarios where the use of ordinary markers could interfere with the movement of the object being tracked. Figure 3-4 illustrates the dimensions and weight specifications of the Qualisys passive super-spherical markers [53], [54], which we utilize.

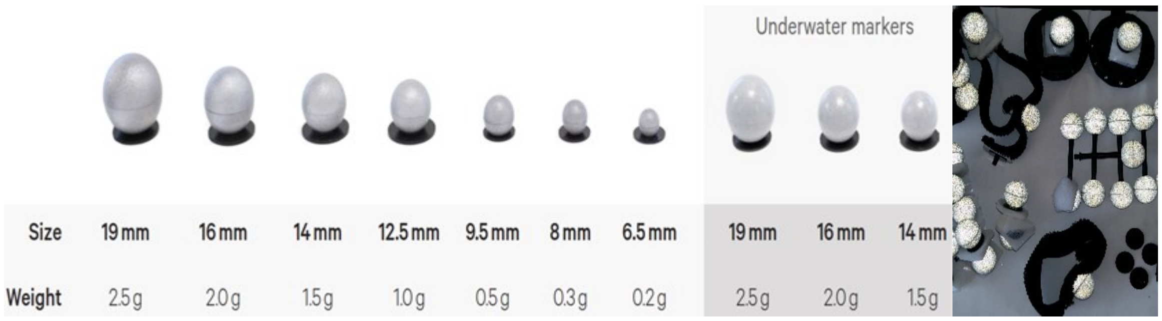


Figure 3-4: Qualisys passive super-spherical markers.

3.3.2 Qualisys Miqus M5U Underwater MoCap Camera

Qualisys Miqus M5u underwater MoCap camera, which was released on 16 May 2019 as the smallest underwater MoCap Camera [11], offers the widest field-of-view and a maximum capture distance of 17 meters among other Qualisys underwater MoCap cameras [55]. Figure 3-5 and Table 3-1 show the Miqus M5U camera we used for our study and its specifications.

Figure 3-5: Miqus M5U Camera

Table 3-1: Qualisys Miqus M5U Camera Specifications

Pixels	4 MP
Resolution	2048 * 2048
Frame rate (full FOV)	180 fps
Underwater FOV	51° * 51°
Max capture distance (with 19 mm marker)	17 m
Weight	2.5 kg / 5.5 lbs
Dimensions	250 × 110 × 110 mm (9.8 × 4.3 × 4.3 in)
Buoyancy	Neutral
Operating voltage	24 VDC
Operating temperature	0-35°C (32-95°F)
Underwater housing	Stainless steel and acrylic

3.3.3 Cameras' Placement in the Volume of Interest

The volume of interest [56] is where a system records motion using cameras. A sample visualization of our pool is depicted in Figure 3-6, created with QTM software. It shows seven cameras positioned around the pool: four on one side and three on the opposite side, along with the "L-frame" calibration fixture and the swimmer. Figure 3-7 shows a "birds eye view" of part of the pool. To reconstruct 3D MoCap data, each marker should be visible to at least two cameras (preferably three or more) such that, following system calibration, the 3D geometry of the markers can be calculated from the 2D images. It is best to position the cameras as far back as possible from the volume to maximize camera view overlap.

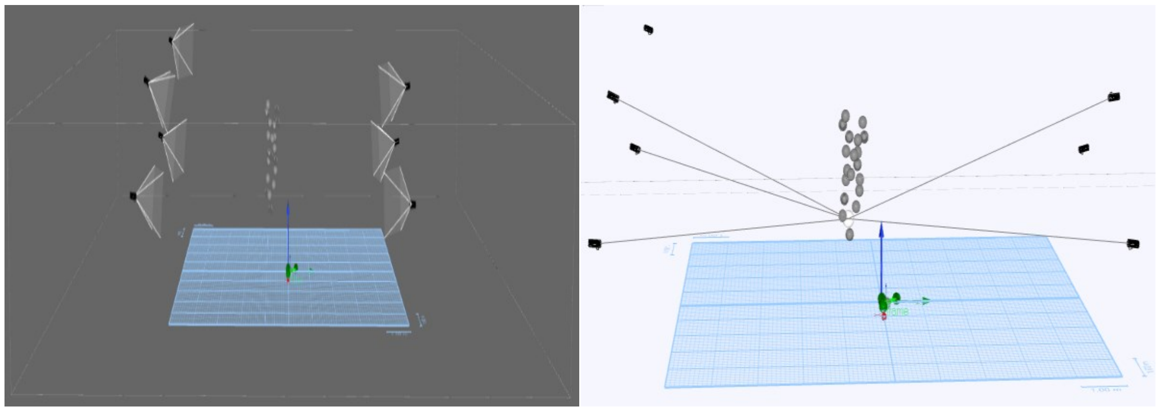


Figure 3-6: The Volume of Interest

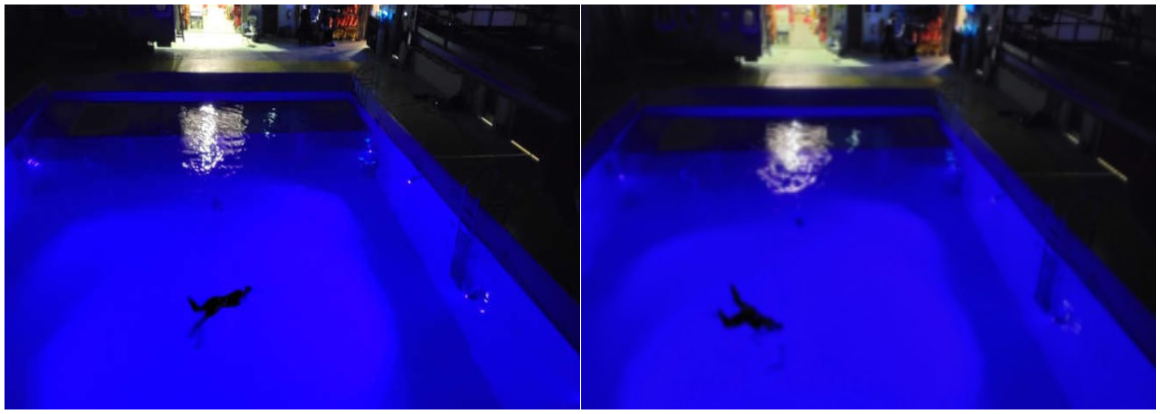


Figure 3-7: Underwater cameras' placement

3.3.4 QTM Software

Qualisys Track Manager (QTM) [6] is a powerful proprietary software designed by Qualisys to integrate seamlessly with Qualisys MoCap cameras. This sophisticated tracking software facilitates the capture, cleaning, editing, labeling, and analysis of MoCap data across various dimensions, including 2D, 3D, and six degrees of freedom (6DOF) in real-time, ensuring fast and precise measurements with minimal latency. QTM also features a 3D video overlay function that works with any Qualisys video camera calibrated with the MoCap system. Two important features of QTM software are:

- ✓ Automatic Identification of Markers (AIM) [17] function enables the identification of markers to facilitate manual data cleaning, regardless of the marker set, by creating an AIM model and feeding it with sample motion data.
- ✓ Skeleton solver [20] is a proven and very robust inverse kinematics solver that can deal with occluded markers in challenging multi-character takes. By using the pre-defined marker set, applying the pre-trained AIM model, stand in a T-pose to fit the skeleton, the real-time solver will start streaming skeletal data such as rigid body joint angles expressed in Euler angles [27].

Furthermore, the QTM software supports integration with various plugins for Unity [57], Unreal Engine [58], iClone [59], MotionBuilder [60], and Maya [61]. QTM is also compatible with major force plates and EMG systems used in biomechanical research. Additionally, QTM offers features like real-time streaming with QTM Connect, real-time server and SDK for seamless integration with other applications, C3D import/export, MATLAB and LabVIEW [62] export, and exporting data to tab-separated value files for further analysis in programs like Microsoft Excel.

The version of QTM software that we use is “2022.2 build 7710”.

3.3.5 Calibration

Before MoCap recording, the Qualisys system must be calibrated to accurately define the cameras’ coordinates relative to each other and the environment. Qualisys Calibration kit [63], [64] includes a carbon fiber wand, and a folding L-frame, as shown in Figure 3-8. The L-frame sets the global coordinate system’s orientation during calibration, defining the X, Y, and Z axes with its corner as the origin. It can be removed after calibration.

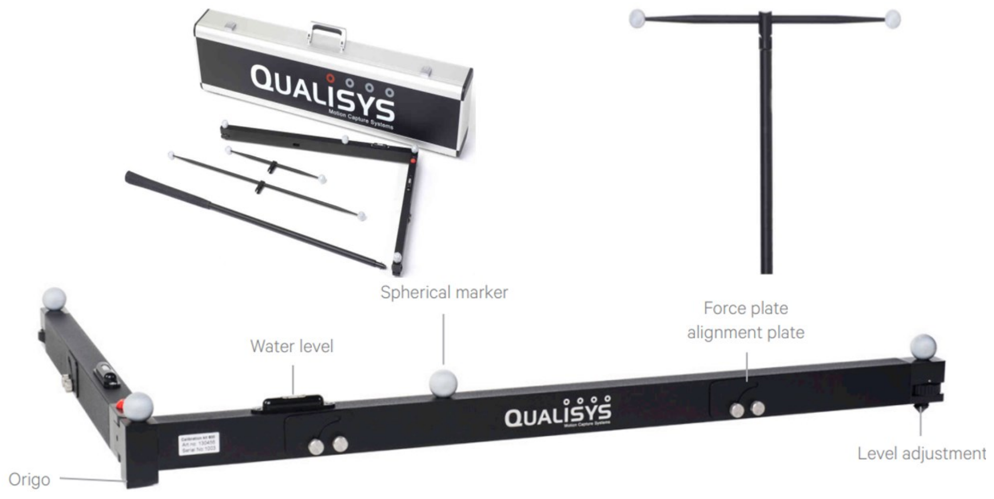


Figure 3-8: Qualisys Calibration Kit: Wand (top right), L-frame (bottom)

Calibration [65], [66] involves waving and twisting a calibration wand continuously in various directions within the volume to ensure it enters all cameras' fields of view as concurrently as possible. The QTM software will indicate if the moving speed is too fast. The software accurately determines each camera's position relative to the L-frame. It is important to keep moving during calibration, cover the entire volume of interest, and avoid waving the wand in areas outside the volume. Additionally, it is crucial to avoid hitting the wand on anything. The calibration procedure done by our swimmer is shown in Figure 3-9.

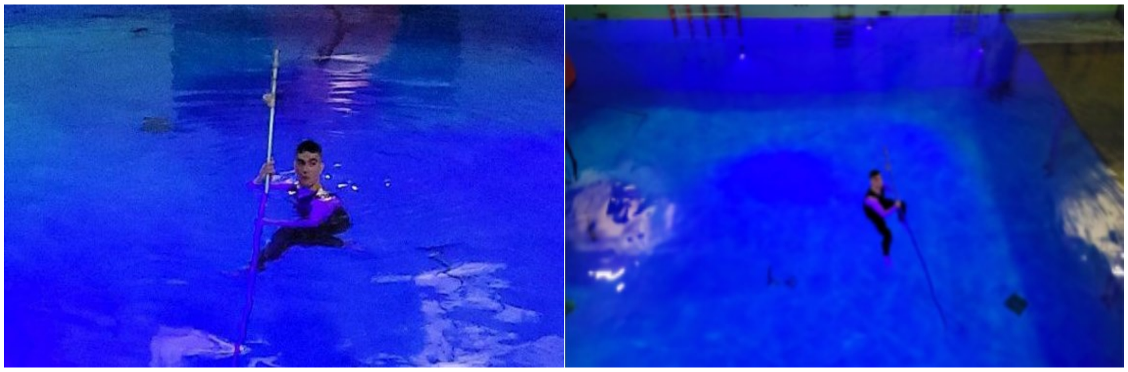


Figure 3-9: Calibration wand and the Calibration procedure by our swimmer.

3.3.6 Recording MoCap Data and Exported File Types

After calibration, data were recorded at a frequency of 100 Hz using QTM. This recorded data can be exported into various file formats [67], including C3D [68], TSV [69], AVI [70], FBX [71], or MATLAB, allowing for further processing in other applications. For our purposes, we export the data into a C3D file [72].

3.4 Noise in Underwater MoCap Data

Underwater MoCap is prone to noise due to surface reflections, water's unique properties, and water's unpredictable waves and turbulence, as depicted in the Figure 3-10. The noisy image and the corresponding MoCap data are shown in Figure 3-11.

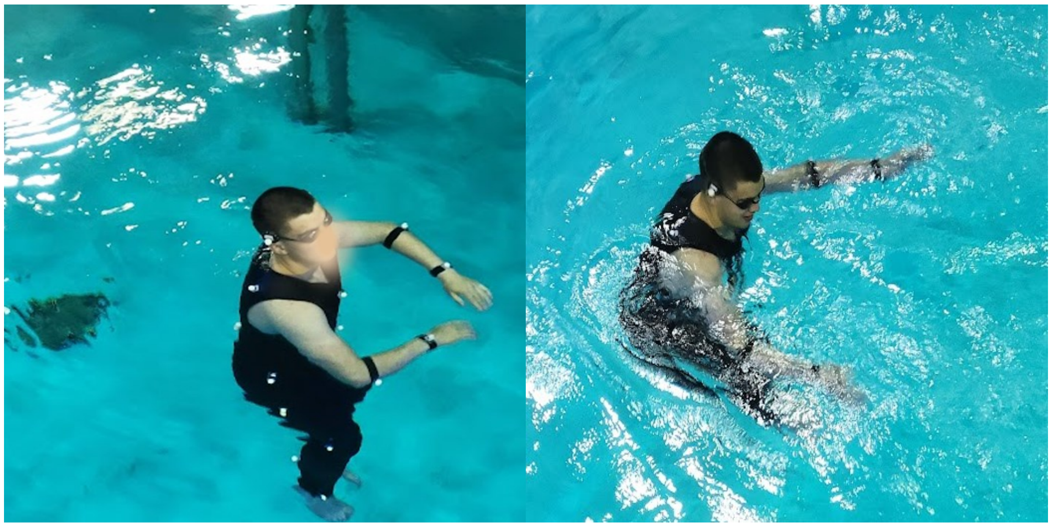


Figure 3-10: Noise in Underwater Data; Surface Reflection and Wave

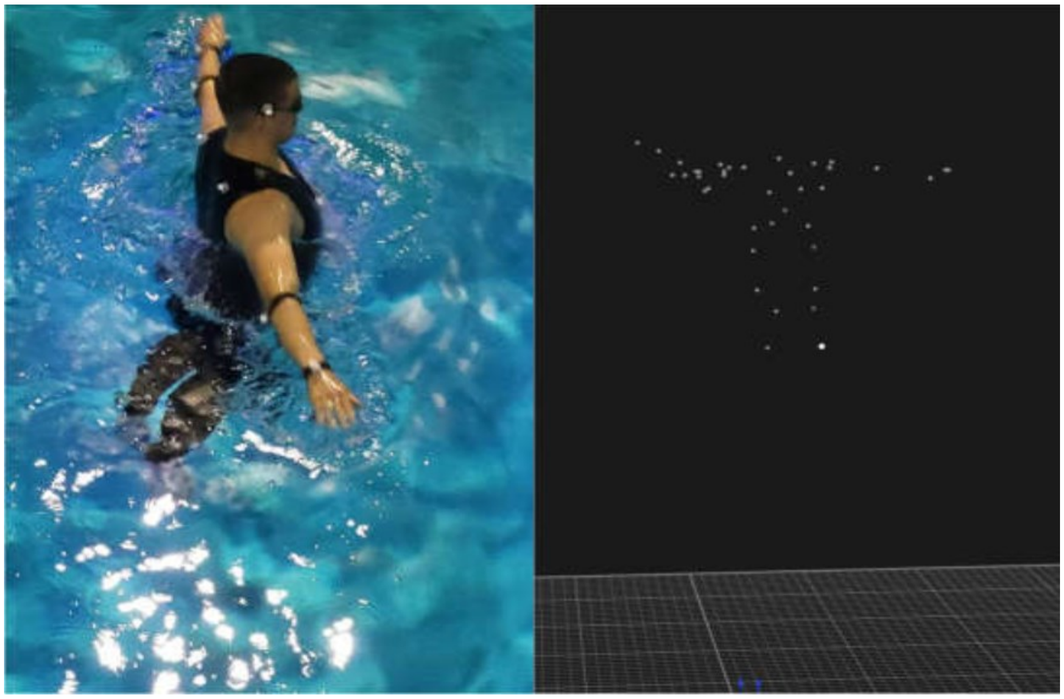


Figure 3-11: Noisy MoCap Data

3.5 MoCap Data Cleaning and Editing Tools

To utilize MoCap data, the noisy raw data must be cleaned using various methods, including commercial software, open-source software, and programming tools. The following sections define these editing tools.

3.5.1 Commercial MoCap Software

Commercial MoCap software, such as Qualisys QTM, Vicon Nexus, and OptiTrack [73] is utilized for recording, processing, editing, and animating motion data. These tools feature automatic labeling functions (e.g., AIM in QTM or Vicon Nexus) that assist with manual cleaning; however, some manual intervention is still required. These systems often integrate with hardware for real-time feedback. They necessitate a purchase.

3.5.2 Mokka Software

Motion Kinematic and Kinetic Analyzer (Mokka) [15] is an open-source software designed to analyze biomechanical data. It is compatible with various file formats, including C3D. It can be used for tasks such as manually cleaning raw MoCap data when commercial software is unavailable. The Mokka User Interface is shown in Figure 3-12.

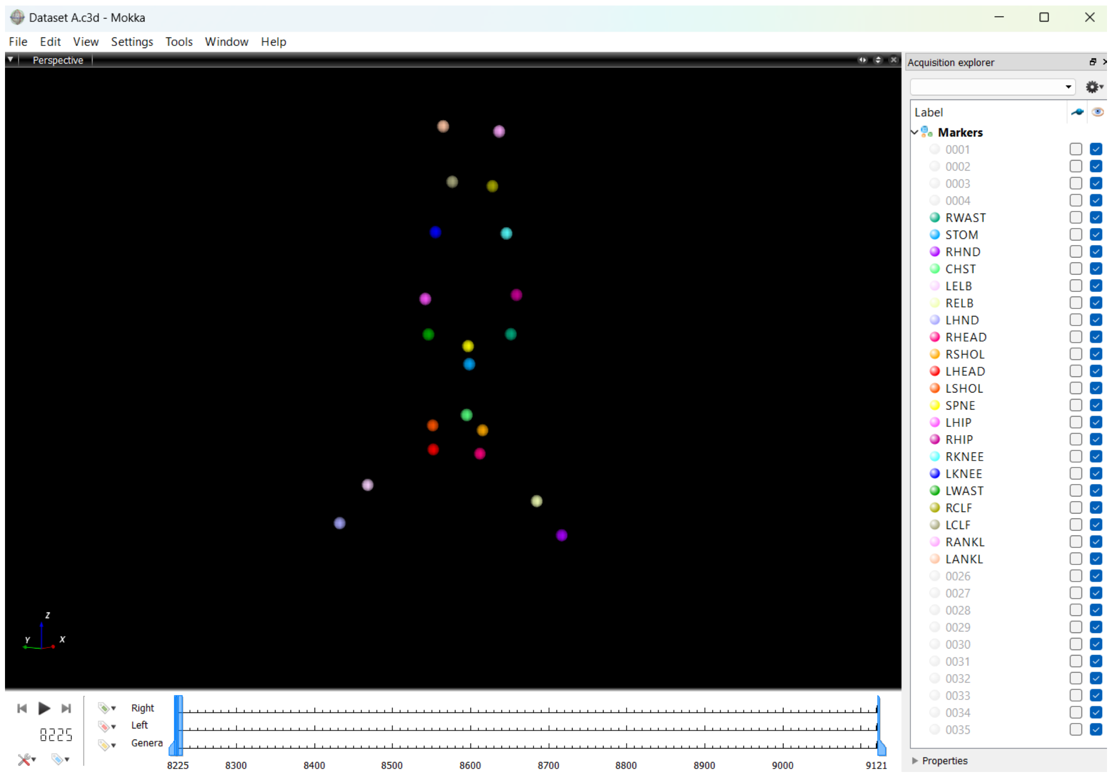


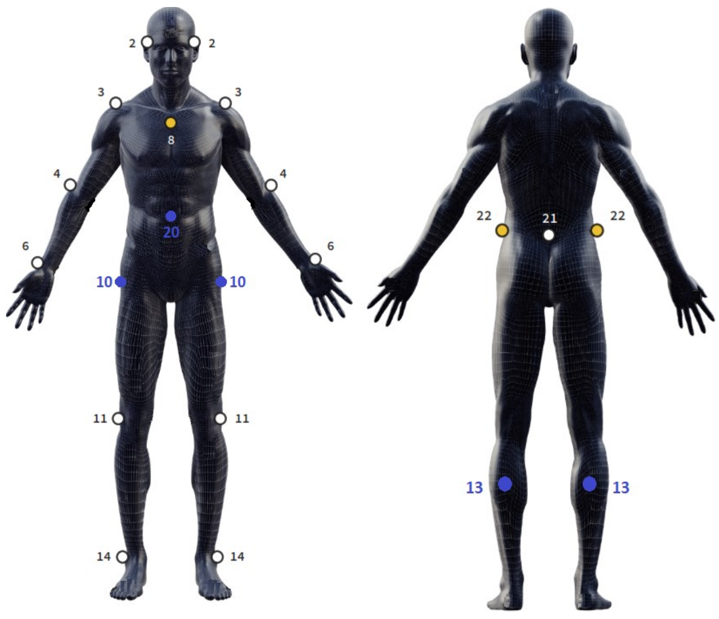
Figure 3-12: Mokka User Interface

3.5.3 Applications of MATLAB and Python in MoCap Data Cleaning

Programming software like MATLAB and Python can be used to generate semi-automatic or automatic MoCap cleaning and labeling algorithms. The EZC3D [74] library and Biomechanical ToolKit (BTK) [75] are tools for reading and modifying C3D files. We utilized the EZC3D library in both MATLAB and Python.

3.6 Marker Set

Our marker set consists of 21 passive markers attached to our swimmer's suit and body. The anatomical placement of the correspondence points, our given labels and their correspondence name in the Qualisys Sports Marker Set [51] are shown in Figure 3-13. The numbers in this figure are based on the Qualisys marker set. For example, our given label name to marker number 8 is "CHST" and its correspondence Qualisys name is "Chest".

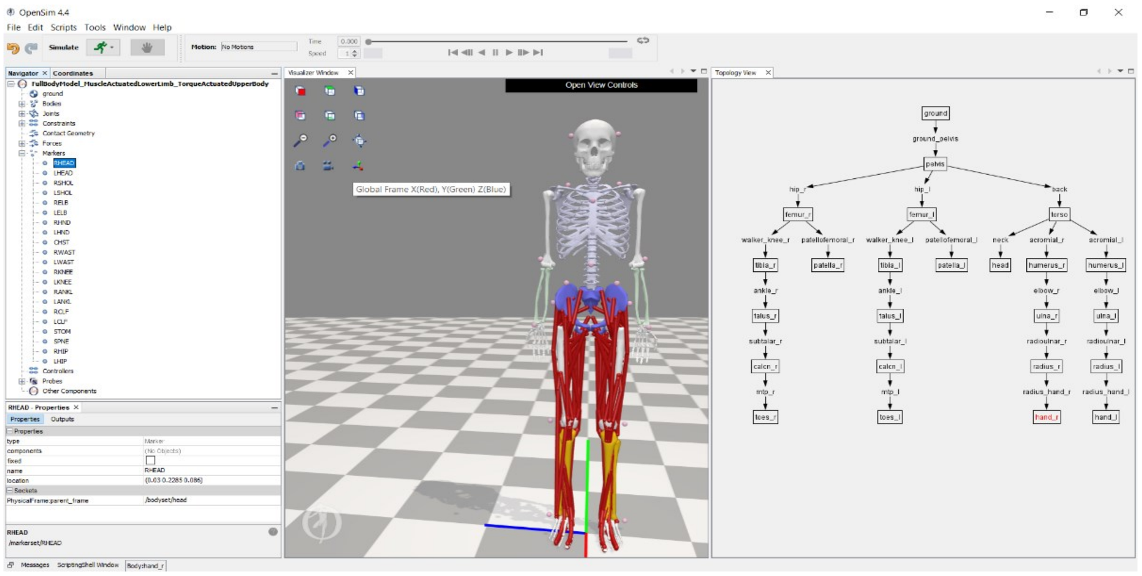


No.	Labels	Qualisys Name
2	L/RHEAD	HeadL/R
3	L/RSHOL	L/RShoulderTop
4	L/RELB	L/RElbowOut
6	L/RHND	L/RWristIn
8	CHST	Chest
10	L/RHIP	L/RThighFrontLow
11	L/RKNEE	L/RKneeOut
13	L/RCLF	L/RShinFrontHigh
14	L/RANKL	LAnkleOut
20	STOM	SpineThoracic12
21	SPNE	WaistBack
22	L/RWAST	WaistL/R

Figure 3-13: Anatomical locations of 21 markers

3.7 OpenSim Marker Set

OpenSim 4.4 [16] creates the marker set based on the musculoskeletal model [76]. The user interface contains a model, markers, and a topology tree, as shown in Figure 3-14. The markers are placed on Simbody based on the anatomical location described in Figure 3-13. The markers' local coordinates are defined in the "MarkerSet.xml" file.

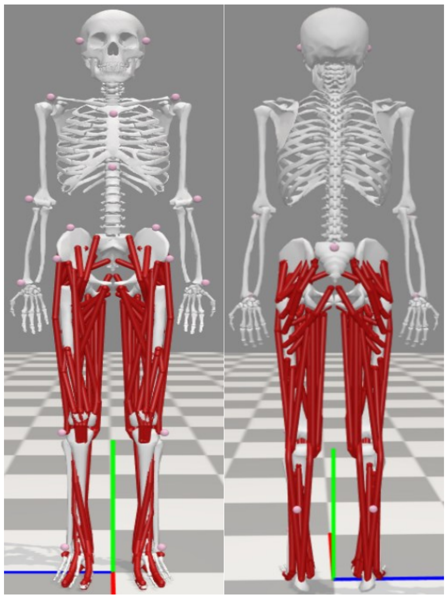


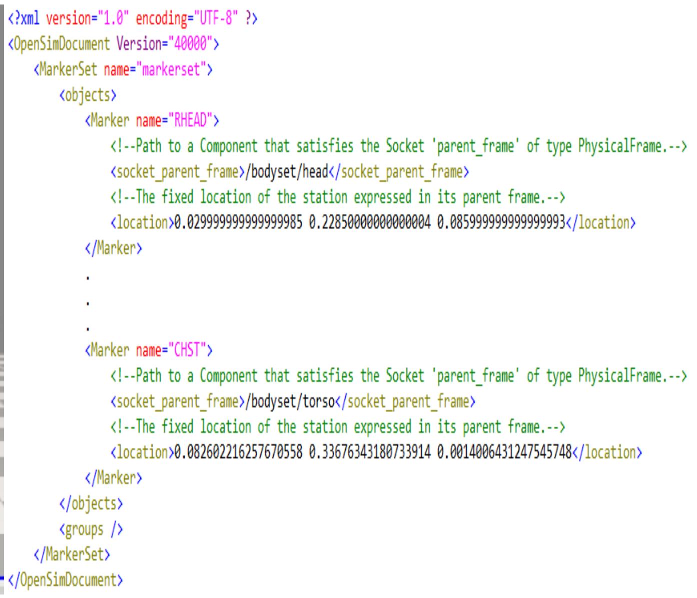
RHEAD - Properties

Properties	
type	Marker
components	(No Objects)
fixed	☐
name	RHEAD
location	(0.03 0.2285 0.086)
Sockets	
PhysicalFrame:parent_frame	/bodyset/head

RHEAD
/markerset/RHEAD

Figure 3-14: OpenSim User Interface





```
<?xml version="1.0" encoding="UTF-8" ?>
<OpenSimDocument Version="40000">
  <MarkerSet name="markerset">
    <objects>
      <Marker name="RHEAD">
        <!--Path to a Component that satisfies the Socket 'parent_frame' of type PhysicalFrame.-->
        <socket_parent_frame>/bodyset/head</socket_parent_frame>
        <!--The fixed location of the station expressed in its parent frame.-->
        <location>0.029999999999999985 0.22850000000000004 0.085999999999999993</location>
      </Marker>
      .
      .
      .
      <Marker name="CHST">
        <!--Path to a Component that satisfies the Socket 'parent_frame' of type PhysicalFrame.-->
        <socket_parent_frame>/bodyset/torso</socket_parent_frame>
        <!--The fixed location of the station expressed in its parent frame.-->
        <location>0.082602216257670558 0.33676343180733914 0.0014006431247545748</location>
      </Marker>
    </objects>
    <groups />
  </MarkerSet>
</OpenSimDocument>
```

Figure 3-15: OpenSim model markers placement and the output MarkerSet.xml file

The locations of 21 markers in front and back of the body model and part of the generated MarkerSet.xml file are shown in Figure 3-15. The colors of the x, y, and z axis are red, green, and blue.

3.8 Manual Cleaning Raw C3D MoCap Data using QTM software

The raw MoCap files often exhibit significant noise and contain gaps, particularly in underwater environments. The primary goal of manually cleaning MoCap data is to refine and label these datasets so that they can be effectively utilized for subsequent processing and various applications. The following sections describe our manual cleaning process for our captured *Dataset A* using QTM software, which includes labelling, denoising, and recovering missing markers. We also clean data using QTM AIM model and compare its results to the result of manual cleaning.

Our captured MoCap data contains 21 reflective passive markers based on our marker set shown in Figure 3-13. The cleaning process described for passive markers can also be applied to active markers or partially cleaned data. Figure 3-16 illustrates the noisy frame of our MoCap data alongside the corresponding cleaned and labeled MoCap data.

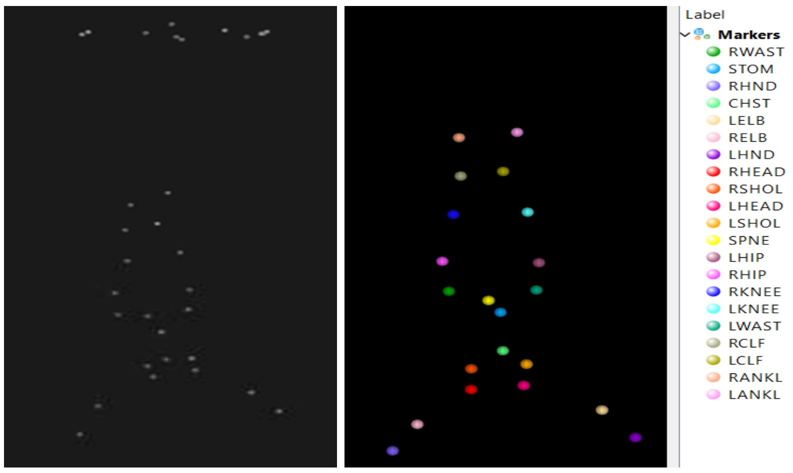
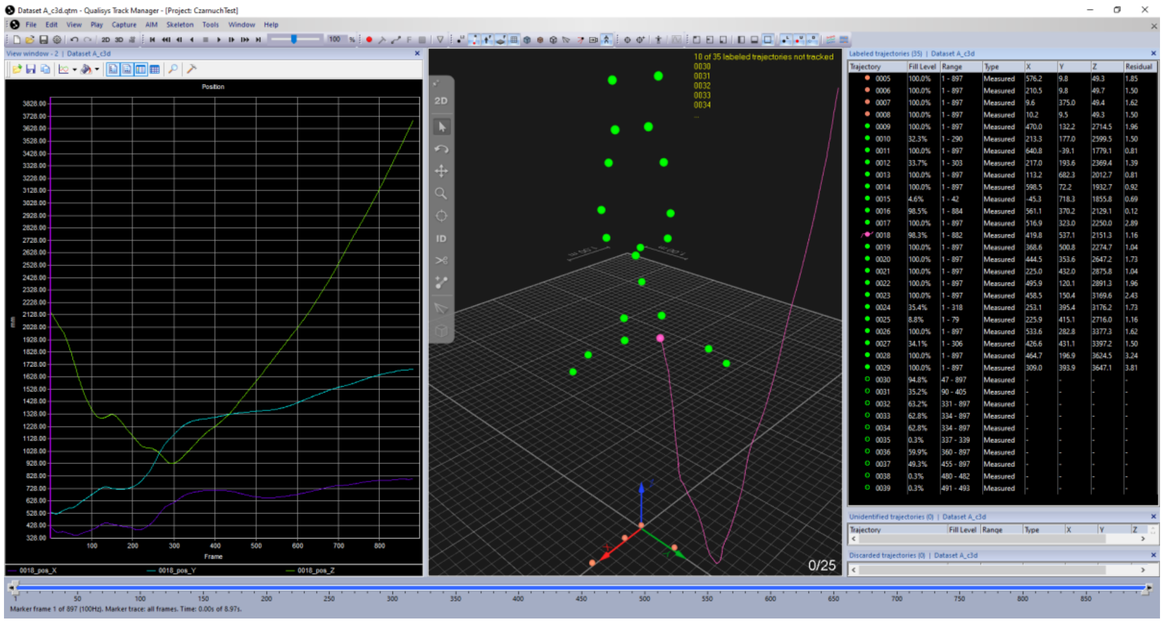


Figure 3-16: Noisy MoCap data and its corresponding cleaned and labeled data.

3.8.1 Raw C3D Dataset A

Our *Dataset A* is a noisy C3D file containing 897 frames. The sampling frequency of data is 100 Hz. For this dataset, our swimmer begins at the first frame in a downward posture similar to the inverse of "T-pose" as depicted in Figure 3-17, which showcases the QTM User Interface. For the trial, the swimmer submerges and breaststrokes downward toward the bottom of the pool head first, turns head up, and swims back to the surface. The first frame features 21 valid markers, which are physical markers affixed to the swimmers' bodies and are illustrated in green. Additionally, it includes 4 extraneous markers that are attached to the "L-frame," depicted in orange.



Dataset A_c3d.qtm - Qualisys Track Manager - [Project: CzarnuchTest]
File Edit View Play Capture AIM Skeleton Tools Window Help
View window - 2 | Dataset A_c3d
Position
mm
Frame
0018_pos_X
0018_pos_Y
0018_pos_Z
2D
ID
0/25

18 of 35 labeled trajectories not tracked
0030
0031
0032
0033
0034

Labeled trajectories (35) | Dataset A_c3d

Trajectory	Fill Level	Range	Type	X	Y	Z	Residual
0005	100.0%	1 - 897	Measured	576.2	9.8	49.3	1.85
0006	100.0%	1 - 897	Measured	210.5	9.8	49.7	1.50
0007	100.0%	1 - 897	Measured	9.6	375.0	49.4	1.62
0008	100.0%	1 - 897	Measured	10.2	9.5	49.3	1.50
0009	100.0%	1 - 897	Measured	470.0	132.2	2714.5	1.96
0010	32.3%	1 - 290	Measured	213.3	177.0	2599.5	1.50
0011	100.0%	1 - 897	Measured	640.8	-39.1	1779.1	0.81
0012	33.7%	1 - 303	Measured	217.0	193.6	2369.4	1.39
0013	100.0%	1 - 897	Measured	113.2	682.3	2012.7	0.81
0014	100.0%	1 - 897	Measured	598.5	72.2	1932.7	0.92
0015	4.6%	1 - 42	Measured	-45.3	718.3	1855.8	0.69
0016	98.5%	1 - 884	Measured	561.1	370.2	2129.1	0.12
0017	100.0%	1 - 897	Measured	516.9	323.0	2250.0	2.89
0018	98.3%	1 - 882	Measured	419.8	537.1	2151.3	1.16
0019	100.0%	1 - 897	Measured	368.6	500.8	2274.7	1.04
0020	100.0%	1 - 897	Measured	444.5	353.6	2647.2	1.73
0021	100.0%	1 - 897	Measured	225.0	432.0	2875.8	1.04
0022	100.0%	1 - 897	Measured	495.9	120.1	2891.3	1.96
0023	100.0%	1 - 897	Measured	458.5	150.4	3169.6	2.43
0024	35.4%	1 - 318	Measured	253.1	395.4	3176.2	1.73
0025	8.8%	1 - 79	Measured	225.9	415.1	2716.0	1.16
0026	100.0%	1 - 897	Measured	533.6	282.8	3377.3	1.62
0027	34.1%	1 - 306	Measured	426.6	431.1	3397.2	1.50
0028	100.0%	1 - 897	Measured	464.7	196.9	3624.5	3.24
0029	100.0%	1 - 897	Measured	309.0	393.9	3647.1	3.81
0030	94.8%	47 - 897	Measured	-	-	-	-
0031	35.2%	90 - 405	Measured	-	-	-	-
0032	63.2%	331 - 897	Measured	-	-	-	-
0033	62.8%	334 - 897	Measured	-	-	-	-
0034	62.8%	334 - 897	Measured	-	-	-	-
0035	0.3%	337 - 339	Measured	-	-	-	-
0036	59.9%	360 - 897	Measured	-	-	-	-
0037	49.3%	455 - 897	Measured	-	-	-	-
0038	0.3%	480 - 482	Measured	-	-	-	-
0039	0.3%	491 - 493	Measured	-	-	-	-

Unidentified trajectories (0) | Dataset A_c3d

Trajectory	Fill Level	Range	Type	X	Y	Z

Discarded trajectories (0) | Dataset A_c3d

Marker frame 1 of 897 (100Hz). Marker trace: all frames. Time: 0.00s of 8.97s.

Figure 3-17: QTM User Interface; Dataset A

The *Dataset A* contains a total of 35 tracked points. Given that there are only 21 valid markers, the discrepancy in numbers—specifically the additional 14 points—is attributed to factors such as noise or the phenomenon of "reappearing valid markers." This occurrence is typical of passive markers and will be elaborated upon in detail in the subsequent section.

Each point in a MoCap C3D file has a trajectory [77] that represents its movement path tracked in three-dimensional (3D) space over time. The trajectory of point number 0018 is illustrated in Figure 3-17, represented by the pink curve. Also, in this figure, the position of this trajectory at any given moment is plotted using its X, Y, and Z coordinates.

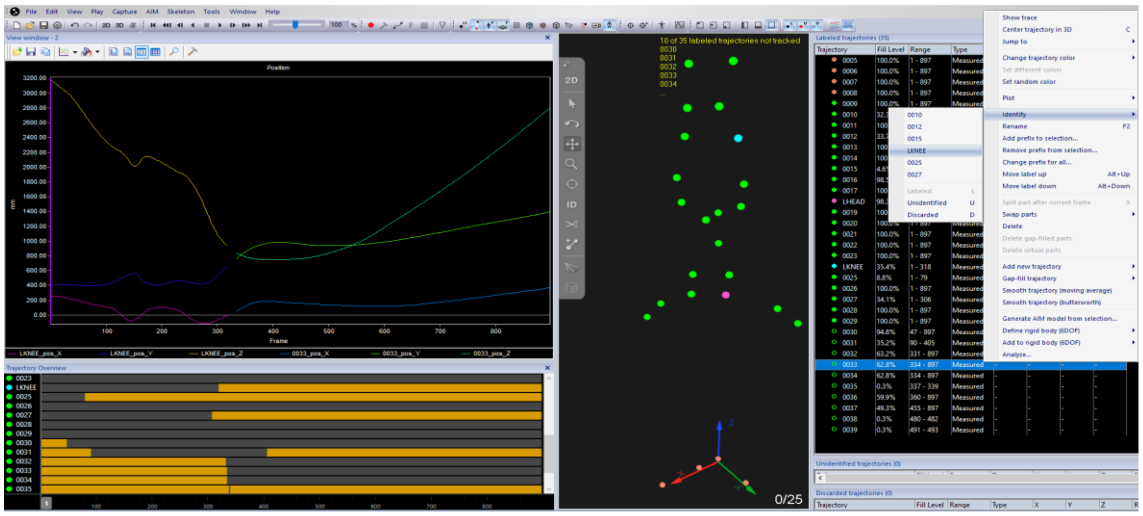
The term “Fill Level” [7] refers to the percentage of the capture period during which a trajectory is visible. For point number 0018, the fill level is 98.3%. This indicates that the total duration of its trajectory encompasses 882 frames. As illustrated in Figure 3-17 the trajectory for this point extends from frame 1 to frame 882. This means that this point is not detected by any cameras after frame 882 until the last captured frame, which is 897.

3.8.2 Identifying and Labeling Trajectories

After a motion capture recording, all recorded trajectories will be displayed in the right pane of QTM under either “Labeled trajectories” or “Unidentified trajectories”. Labeling a trajectory can be done manually by renaming each trajectory. For instance, point number 0018 can be labeled as “LHEAD” (left head marker), and point number 0024 can be labeled as “LKNEE” (left knee marker), as illustrated in Figure 3-18. In the subsequent section, we will describe the “reappearing” markers, which are characteristic of passive markers. Understanding this feature is essential before proceeding with the manual cleaning process.

3.8.2.1 Passive Markers Characteristic: “Reappearing” Markers

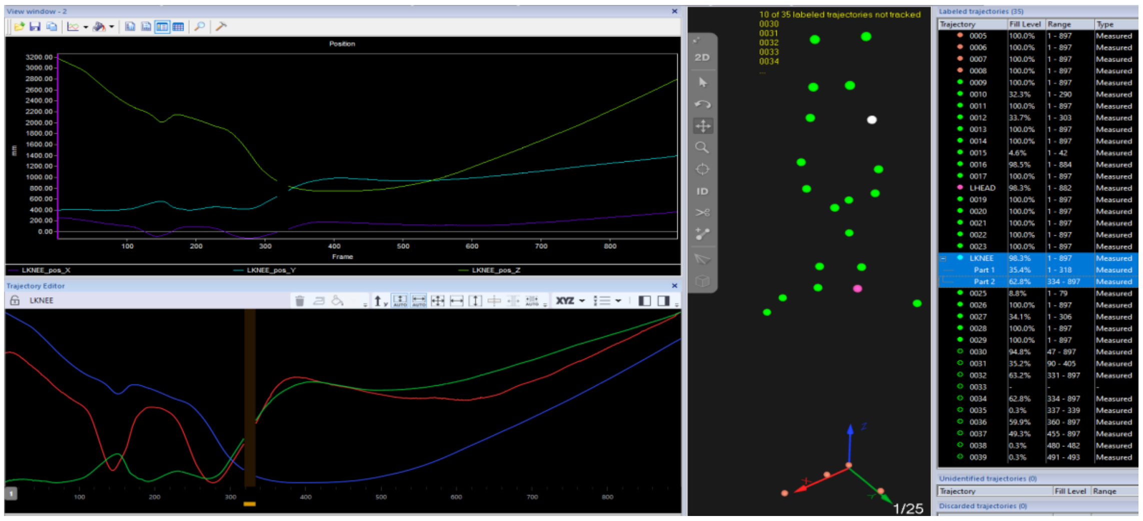
In MoCap data, a physical marker is assigned a random ID number when tracked for the first time. If a passive marker becomes occluded and later reappears, it is typically assigned a new ID because the tracking system can not recognize it as the same marker. Consequently, passive markers often exhibit short segment trajectories rather than continuous trajectories, with potential gaps. As a result, a MoCap file may have multiple points linked to different segments of the same physical marker's trajectory. Thus, the trajectory segments belong to each specific physical marker should be identified and merged into a complete trajectory with gaps. Figure 3-18 and Figure 3-19 illustrate before and after merging short trajectory segments of an example passive marker (i.e., 'LKNEE').



Labeled trajectories (35)

Trajectory	Fill Level	Range	Type
0005	100.0%	1 - 897	Measured
0006	100.0%	1 - 897	Measured
0007	100.0%	1 - 897	Measured
0008	100.0%	1 - 897	Measured
0009	100.0%	1 - 897	Measured
0010	32.3		
0011	100		
0012	33.7		
0013	100		
0014	100		
0015	4.6		
0016	98.5		
0017	100		
LHEAD	98.3		
0019	100		
0020	100.0%	1 - 897	Measured
0021	100.0%	1 - 897	Measured
0022	100.0%	1 - 897	Measured
0023	100.0%	1 - 897	Measured
LKNEE	35.4%	1 - 318	Measured
0025	8.8%	1 - 79	Measured
0026	100.0%	1 - 897	Measured
0027	34.1%	1 - 306	Measured
0028	100.0%	1 - 897	Measured
0029	100.0%	1 - 897	Measured
0030	94.8%	47 - 897	Measured
0031	35.2%	90 - 405	Measured
0032	63.2%	331 - 897	Measured
0033	62.8%	334 - 897	Measured
0034	62.8%	334 - 897	Measured
0035	0.3%	337 - 339	Measured
0036	59.9%	360 - 897	Measured
0037	49.3%	455 - 897	Measured
0038	0.3%	480 - 482	Measured
0039	0.3%	491 - 493	Measured

Figure 3-18: Passive Markers Characteristic; Reappearing Markers Trajectories



View window - 2
Position
Frame
LKNEE_pos_X
LKNEE_pos_Y
LKNEE_pos_Z
Trajectory Editor
LKNEE
10 of 35 labeled trajectories not tracked
1/25
Unidentified trajectories (0)
Discarded trajectories (0)

Labeled trajectories (35)

Trajectory	Fill Level	Range	Type
0005	100.0%	1 - 897	Measured
0006	100.0%	1 - 897	Measured
0007	100.0%	1 - 897	Measured
0008	100.0%	1 - 897	Measured
0009	100.0%	1 - 897	Measured
0010	32.3%	1 - 290	Measured
0011	100.0%	1 - 897	Measured
0012	33.7%	1 - 303	Measured
0013	100.0%	1 - 897	Measured
0014	100.0%	1 - 897	Measured
0015	4.6%	1 - 42	Measured
0016	98.5%	1 - 884	Measured
0017	100.0%	1 - 897	Measured
LHEAD	98.3%	1 - 882	Measured
0019	100.0%	1 - 897	Measured
0020	100.0%	1 - 897	Measured
0021	100.0%	1 - 897	Measured
0022	100.0%	1 - 897	Measured
0023	100.0%	1 - 897	Measured
LKNEE	98.3%	1 - 897	Measured
Part 1	35.4%	1 - 318	Measured
Part 2	62.8%	334 - 897	Measured
0025	8.8%	1 - 79	Measured
0026	100.0%	1 - 897	Measured
0027	34.1%	1 - 306	Measured
0028	100.0%	1 - 897	Measured
0029	100.0%	1 - 897	Measured
0030	94.8%	47 - 897	Measured
0031	35.2%	90 - 405	Measured
0032	63.2%	331 - 897	Measured
0033	-	-	-
0034	62.8%	334 - 897	Measured
0035	0.3%	337 - 339	Measured
0036	59.9%	360 - 897	Measured
0037	49.3%	455 - 897	Measured
0038	0.3%	480 - 482	Measured
0039	0.3%	491 - 493	Measured

Figure 3-19: Merging Reappearing Markers Trajectories

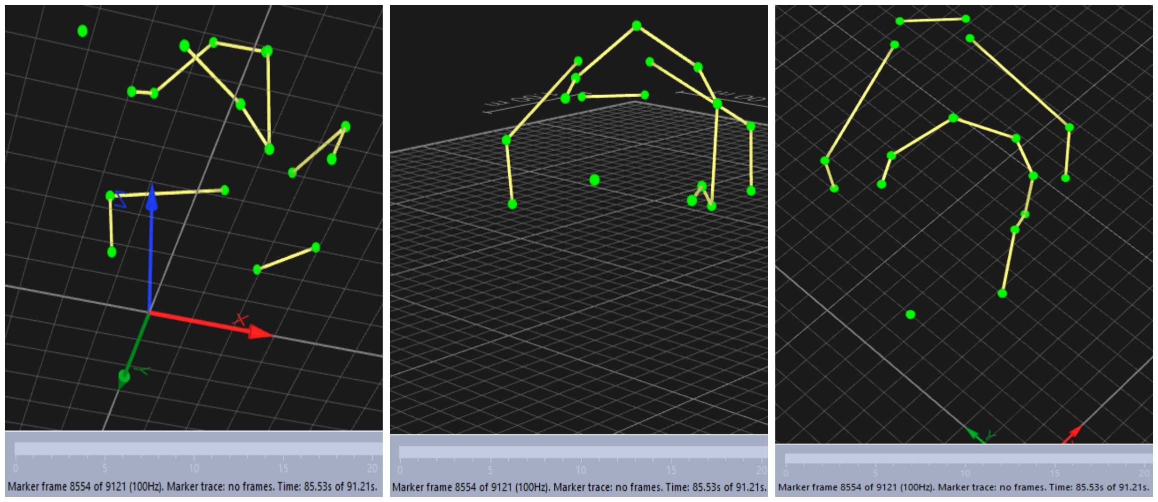
In Figure 3-18, we identified the marker “LKNEE” in the first frame and tracked it until frame 318, at which point it was lost due to occlusion. During a meticulous frame-by-frame manual review, we detected point number 0033 as left knee in frame 334. This indicates that the passive marker previously labeled as “LKNEE” (point number 0018), which was tracked from frame 1 to frame 318 and subsequently occluded, reappeared in frame 334 with a new identifier, which is 0033. To rectify this, we utilized the QTM feature “Identify” by right-clicking on point 0033 and linking it to the “LKNEE” marker. Consequently, as depicted in Figure 3-19, the “LKNEE” markers now consist of two parts and span from frame 1 to frame 897, with a gap occurring between frames 319 and 333.

Comparing the 3D coordinate plots in Figure 3-18 and Figure 3-19 provides a visual understanding of the characteristics of passive markers. In Figure 3-18, the plot consists of two distinct representations related to the points labeled “LKNEE” and “0033.” Conversely, Figure 3-19 illustrates a complete trajectory associated with the new “LKNEE” points, which integrates both segments into one cohesive path. The gaps observed in this trajectory indicate periods during which no left knee marker was captured in the dataset. Consequently, the physical marker “LKNEE” now possesses a unique trajectory within this cleaned dataset. Additionally, the trajectory editor in Figure 3-19 showcases the integrated “LKNEE” continuous trajectory along with the identified gap.

Trajectory overview in Figure 3-18 highlights a crucial aspect regarding the behavior of “reappearing” markers: the trajectories of these markers should not overlap, meaning there should be no intersections between them. The yellow segment preceding each marker ID in the trajectory editor indicates a gap in that marker’s trajectory. For instance, when the visibility duration of the marker labeled “LKNEE” concludes, it is represented in yellow, signifying that it has become occluded. At this point, if we examine marker “0033,” we observe that after several frames, the yellow section associated with this marker ends, and it becomes visible. This observation confirms that the trajectories of these two markers do not intersect at any point.

3.8.3 Bone Visualization

Manual Cleaning MoCap Data begins with the first frame by detecting valid markers and labeling them based on the marker set. Each frame must undergo a thorough review to ensure accurate labeling and to identify and remove any anomalies. To enhance visualization, the connections between markers can be established using bones, which facilitates the identification of irregularities. In this review, we need to adjust the angle of view to improve posture visualization. Figure 3-20 illustrates how bone connectivity and the angle of view influence the identification of valid markers. In the first and second images, it is challenging to discern what the green single marker represents. However, in the third image, it becomes clearer that this marker is the ankle marker.



Marker frame 8554 of 9121 (100Hz). Marker trace: no frames. Time: 85.53s of 91.21s.

Marker frame 8554 of 9121 (100Hz). Marker trace: no frames. Time: 85.53s of 91.21s.

Marker frame 8554 of 9121 (100Hz). Marker trace: no frames. Time: 85.53s of 91.21s.

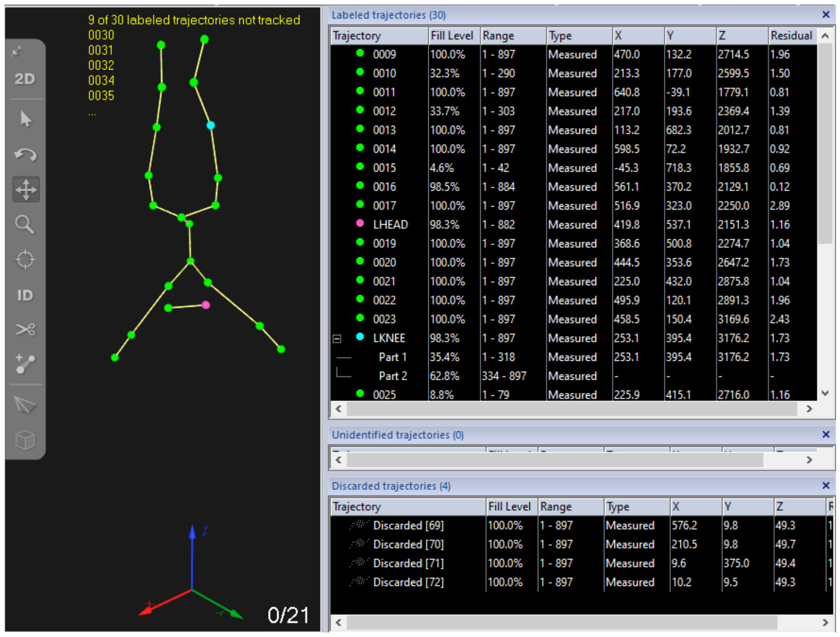
Figure 3-20: The Impact of Bone Connectivity and View Angle Adjustment on Visualization Enhancement.

3.8.4 Denoising

This section explains manual denoising of our *Dataset A*, which removes extraneous markers [78], outliers [79], ghost markers [80], and swapping markers [81]. Subsequent sections will define these types of noise and outline the methods used for their removal from the MoCap dataset using the QTM software.

3.8.4.1 Extraneous Removal

Extraneous markers refer to actual markers that are affixed to an object other than the primary subject, which in this context is our swimmer. For instance, as illustrated in Figure 3-17, there are four orange markers identified by IDs 0005, 0006, 0007, and 0008. These markers are attached to the "L-frame" calibration tool positioned on the floor of the pool. These markers can be easily removed by clicking on them to locate the corresponding ID in the trajectory column and deleting them. Once deleted, these markers will be relocated to the discarded trajectories section within the QTM software, as depicted in Figure 3-21.



9 of 30 labeled trajectories not tracked
0030
0031
0032
0034
0035
...

Labeled trajectories (30)

Trajectory	Fill Level	Range	Type	X	Y	Z	Residual
● 0009	100.0%	1 - 897	Measured	470.0	132.2	2714.5	1.96
● 0010	32.3%	1 - 290	Measured	213.3	177.0	2599.5	1.50
● 0011	100.0%	1 - 897	Measured	640.8	-39.1	1779.1	0.81
● 0012	33.7%	1 - 303	Measured	217.0	193.6	2369.4	1.39
● 0013	100.0%	1 - 897	Measured	113.2	682.3	2012.7	0.81
● 0014	100.0%	1 - 897	Measured	598.5	72.2	1932.7	0.92
● 0015	4.6%	1 - 42	Measured	-45.3	718.3	1855.8	0.69
● 0016	98.5%	1 - 884	Measured	561.1	370.2	2129.1	0.12
● 0017	100.0%	1 - 897	Measured	516.9	323.0	2250.0	2.89
● LHEAD	98.3%	1 - 882	Measured	419.8	537.1	2151.3	1.16
● 0019	100.0%	1 - 897	Measured	368.6	500.8	2274.7	1.04
● 0020	100.0%	1 - 897	Measured	444.5	353.6	2647.2	1.73
● 0021	100.0%	1 - 897	Measured	225.0	432.0	2875.8	1.04
● 0022	100.0%	1 - 897	Measured	495.9	120.1	2891.3	1.96
● 0023	100.0%	1 - 897	Measured	458.5	150.4	3169.6	2.43
⊟ ● LKNEE	98.3%	1 - 897	Measured	253.1	395.4	3176.2	1.73
Part 1	35.4%	1 - 318	Measured	253.1	395.4	3176.2	1.73
Part 2	62.8%	334 - 897	Measured	-	-	-	-
● 0025	8.8%	1 - 79	Measured	225.9	415.1	2716.0	1.16

Unidentified trajectories (0)

Discarded trajectories (4)

Trajectory	Fill Level	Range	Type	X	Y	Z
Discarded [69]	100.0%	1 - 897	Measured	576.2	9.8	49.3
Discarded [70]	100.0%	1 - 897	Measured	210.5	9.8	49.7
Discarded [71]	100.0%	1 - 897	Measured	9.6	375.0	49.4
Discarded [72]	100.0%	1 - 897	Measured	10.2	9.5	49.3

Figure 3-21: Extraneous Markers Removal

3.8.4.2 Outlier Removal

Outliers in MoCap data refer to invalid data points that deviate significantly from the rest of the markers. We can easily discard them from the dataset. Figure 3-22 shows examples of outliers in MoCap data. The first image highlights one outlier in yellow, while the second image displays two outliers marked in white and red.

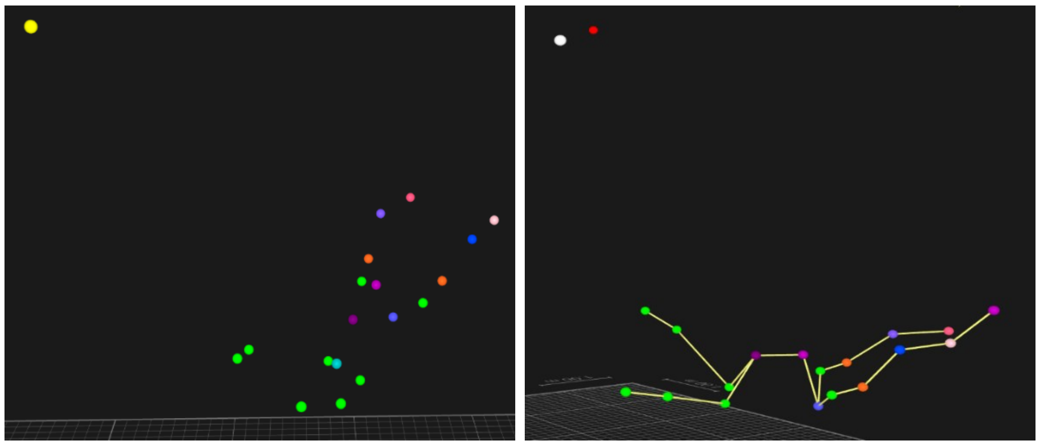


Figure 3-22: Outlier Removal; Yellow outlier (left), white and red outliers (right).

3.8.4.3 Ghost Markers Removal

Ghost markers are erroneous markers that are not associated with any physical object or body part and can be removed by discarding them. Detecting these ghost markers poses a challenge, as they often appear near valid markers. To effectively distinguish valid markers from ghost markers, it is essential to analyze frames preceding and succeeding the current frame. Additionally, factors such as marker fill levels should be considered, along with a comparison of the distances between the markers and the distances in the marker set. Adjusting the view angle and bone connectivity can further aid in clarifying which markers are valid. The images in Figure 3-23 show an example of a ghost marker, along with the effect of changing view angle and bone connectivity on distinguishing the ghost marker from the valid markers.

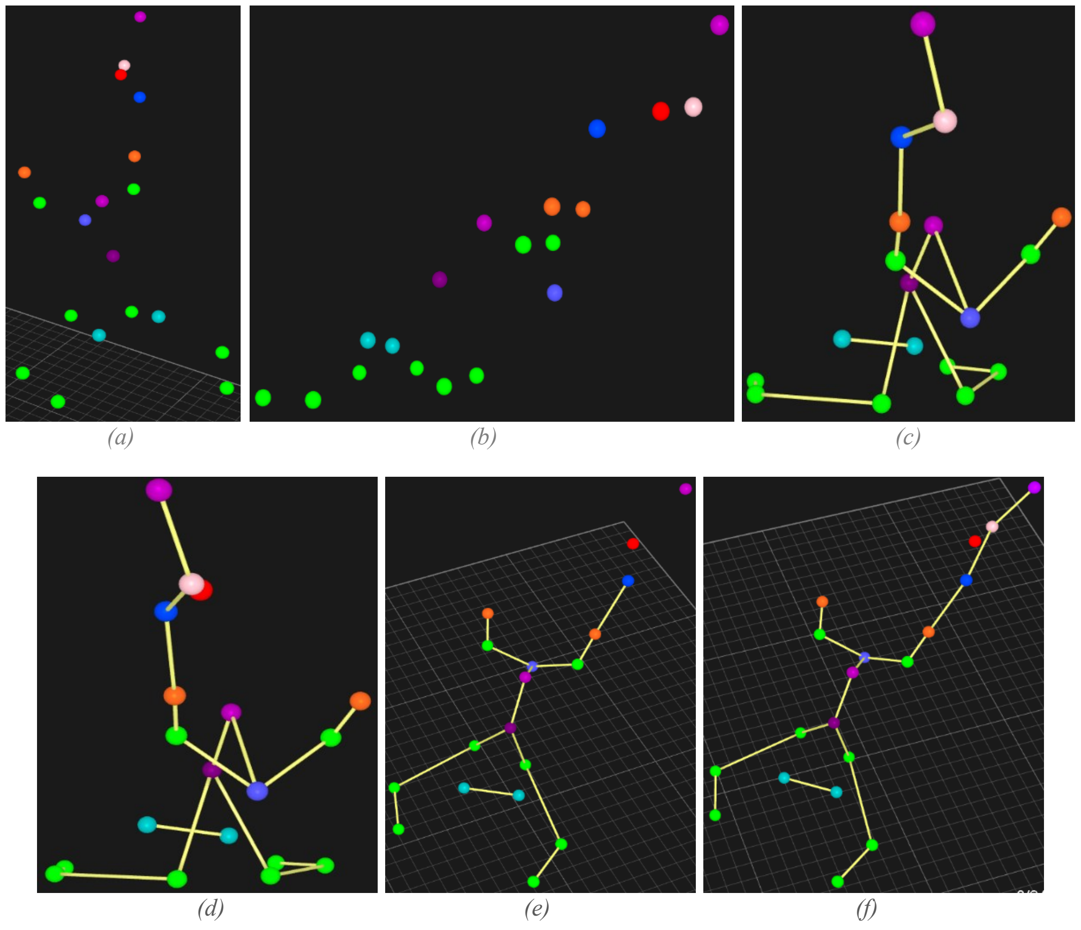


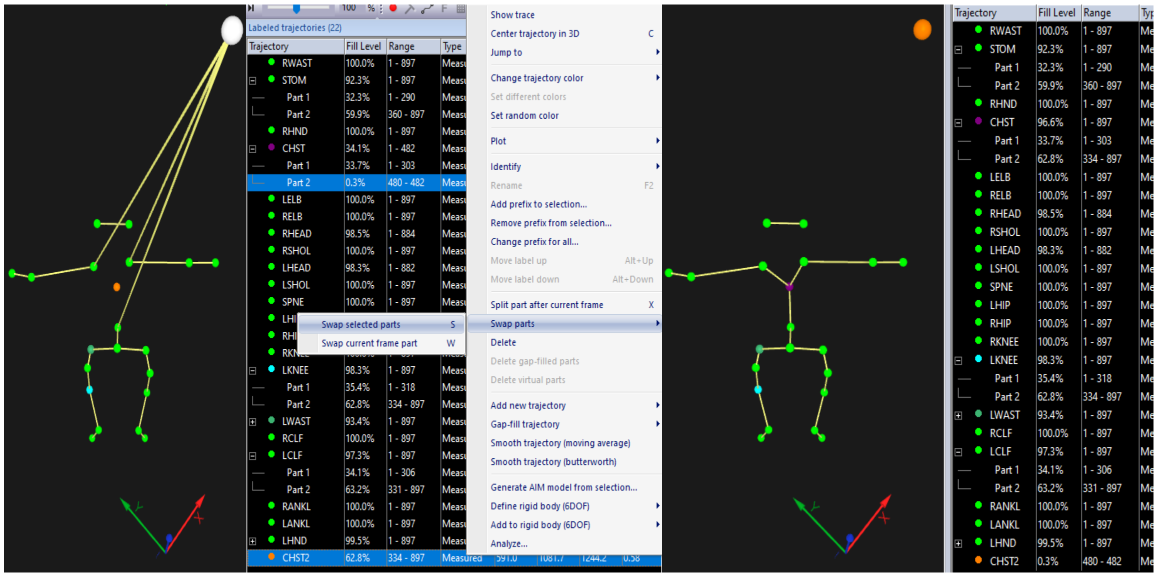
Figure 3-23: Challenges of Manual Cleaning for detecting ghost markers. (a) A ghost marker (red) appears to overlap with a valid marker (pink) from one perspective, (b) in another view, the red and pink markers are clearly separated. (c) Obscured red marker behind pink at an unfavorable angle. (d) View change reveals red marker, still appearing overlapped. (e) Optimal angle; valid (pink) marker is absent. (f) Tracking across frames distinguishes valid (pink) from ghost (red) based on bone length.

In the first image (a) of Figure 3-23, it is essential to differentiate between the two nearby points (red and light pink) to identify which one is a valid marker and which one is a ghost marker. The first four images in Figure 3-23 (from images (a) to image (d)) demonstrate that utilizing both bone connectivity and viewing angle simultaneously can aid in this differentiation. In image (a), there is a lack of bone connectivity, and the two points—red and light pink—appear to overlap. However, upon changing the viewing angle in the second image (b), it becomes evident that these two markers are not actually overlapping. Despite this clarification, it remains unclear which of the two markers is valid. The third image (c) introduces bone connectivity; however, due to an unfavorable viewing angle, the red point is obscured by the pink point. In image (d), a change in perspective reveals the red point, yet it still appears to overlap with the pink point.

After selecting an optimal viewing angle and enhancing visualization through bone connectivity, we analyze the frames preceding and succeeding the current frame to determine which of the two points is a valid marker and which one is a ghost marker. The fifth image (e) illustrates the previous frame; in this image, the viewing angle and bone connectivity are adequate for identifying that the red point could potentially be a calf marker, while the pink point has not yet appeared. In image six (f), both points are visible, and with good bone connectivity and viewing angle, we can also assess their fill levels. We can confidently conclude that the red point, exhibiting a low fill level, is indeed a ghost marker. To further validate our findings, we compared the respective distances of these two points to the knee and ankle points against those in the established marker set. This comparison allows us to accurately identify that the light pink point is a valid calf marker while confirming that the red point is a ghost marker.

3.8.4.4 Swapping Markers

Marker swapping [7], [81], [82] occurs when the labels of two markers are inadvertently exchanged due to their close proximity or overlapping paths in front of a camera. This error results in segments of the 3D trajectory being inaccurately assigned to the incorrect marker. For example, if a hand marker moves closely in front of a chest marker during motion capture, the tracking system may erroneously attribute data from one marker to the other. This issue is particularly common when two markers become occluded and later reappear; in the case of passive markers, this often leads to them being assigned new IDs. Furthermore, during manual cleaning processes, markers can be misidentified due to viewpoint issues that obscure their actual positions. An example is shown in Figure 3-24.



Trajectory	Fill Level	Range	Type				
RWAST	100.0%	1 - 897	Measu				
STOM	92.3%	1 - 897	Measu				
Part 1	32.3%	1 - 290	Measu				
Part 2	59.9%	360 - 897	Measu				
RHND	100.0%	1 - 897	Measu				
CHST	34.1%	1 - 482	Measu				
Part 1	33.7%	1 - 303	Measu				
Part 2	0.3%	480 - 482	Measu				
LELB	100.0%	1 - 897	Measu				
RELB	100.0%	1 - 897	Measu				
RHEAD	98.5%	1 - 884	Measu				
RSHOL	100.0%	1 - 897	Measu				
LHEAD	98.3%	1 - 882	Measu				
LSHOL	100.0%	1 - 897	Measu				
SPNE	100.0%	1 - 897	Measu				
LHI							
RHI							
RKN							
LKNEE	98.3%	1 - 897	Measu				
Part 1	35.4%	1 - 318	Measu				
Part 2	62.8%	334 - 897	Measu				
LWAST	93.4%	1 - 897	Measu				
RCLF	100.0%	1 - 897	Measu				
LCLF	97.3%	1 - 897	Measu				
Part 1	34.1%	1 - 306	Measu				
Part 2	63.2%	331 - 897	Measu				
RANKL	100.0%	1 - 897	Measu				
LANKL	100.0%	1 - 897	Measu				
LHND	99.5%	1 - 897	Measu				
CHST2	62.8%	334 - 897	Measured	591.0	1081.7	1244.2	0.58

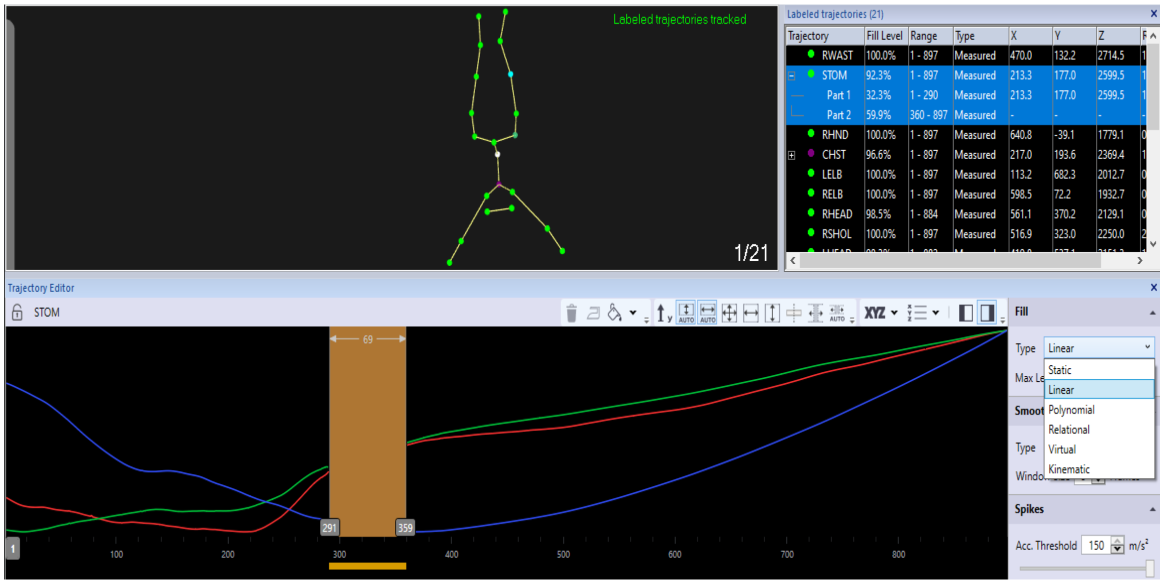
Trajectory	Fill Level	Range	Typ
RWAST	100.0%	1 - 897	Me
STOM	92.3%	1 - 897	Me
Part 1	32.3%	1 - 290	Me
Part 2	59.9%	360 - 897	Me
RHND	100.0%	1 - 897	Me
CHST	96.6%	1 - 897	Me
Part 1	33.7%	1 - 303	Me
Part 2	62.8%	334 - 897	Me
LELB	100.0%	1 - 897	Me
RELB	100.0%	1 - 897	Me
RHEAD	98.5%	1 - 884	Me
RSHOL	100.0%	1 - 897	Me
LHEAD	98.3%	1 - 882	Me
LSHOL	100.0%	1 - 897	Me
SPNE	100.0%	1 - 897	Me
LHIP	100.0%	1 - 897	Me
RHIP	100.0%	1 - 897	Me
RKNEE	100.0%	1 - 897	Me
LKNEE	98.3%	1 - 897	Me
Part 1	35.4%	1 - 318	Me
Part 2	62.8%	334 - 897	Me
LWAST	93.4%	1 - 897	Me
RCLF	100.0%	1 - 897	Me
LCLF	97.3%	1 - 897	Me
Part 1	34.1%	1 - 306	Me
Part 2	63.2%	331 - 897	Me
RANKL	100.0%	1 - 897	Me
LANKL	100.0%	1 - 897	Me
LHND	99.5%	1 - 897	Me
CHST2	0.3%	480 - 482	Me

Figure 3-24: Swapping Markers

In Figure 3-24, the first image displays an orange marker positioned at the chest location and labeled as “CHST2.” However, it becomes evident that the second part of the “CHST” label is not indicative of a chest marker; rather, it is an outlier. To rectify this discrepancy, we will swap the second part of “CHST” with “CHST2.” This can be accomplished by selecting both markers intended for swapping and right-clicking to access the option to either swap selected parts or swap parts only at the current marker frame. In this instance, we opted to swap selected parts, resulting in the corrected markers shown in the second image. As a final step, we should remove “CHST2,” since it has been identified as an outlier.

3.8.5 Recovery

After denoising the raw MoCap data, it is essential to recover any missing markers and fill in the gaps [83] within the trajectories of the valid markers. The identification of these gaps [84] can be accomplished by examining the Fill Level column in QTM. Once identified, the gaps can be addressed using the trajectory editor, which offers various fill types [85] such as Linear, Polynomial, Static, Relational, Virtual, and Kinematic. Each of these methods will be elaborated upon in subsequent sections.



Labeled trajectories (21)

Trajectory	Fill Level	Range	Type	X	Y	Z
● RWAST	100.0%	1 - 897	Measured	470.0	132.2	2714.5
● STOM	92.3%	1 - 897	Measured	213.3	177.0	2599.5
Part 1	32.3%	1 - 290	Measured	213.3	177.0	2599.5
Part 2	59.9%	360 - 897	Measured	-	-	-
● RHND	100.0%	1 - 897	Measured	640.8	-39.1	1779.1
● CHST	96.6%	1 - 897	Measured	217.0	193.6	2369.4
● LELB	100.0%	1 - 897	Measured	113.2	682.3	2012.7
● RELB	100.0%	1 - 897	Measured	598.5	72.2	1932.7
● RHEAD	98.5%	1 - 884	Measured	561.1	370.2	2129.1
● RSHOL	100.0%	1 - 897	Measured	516.9	323.0	2250.0

Type: Linear

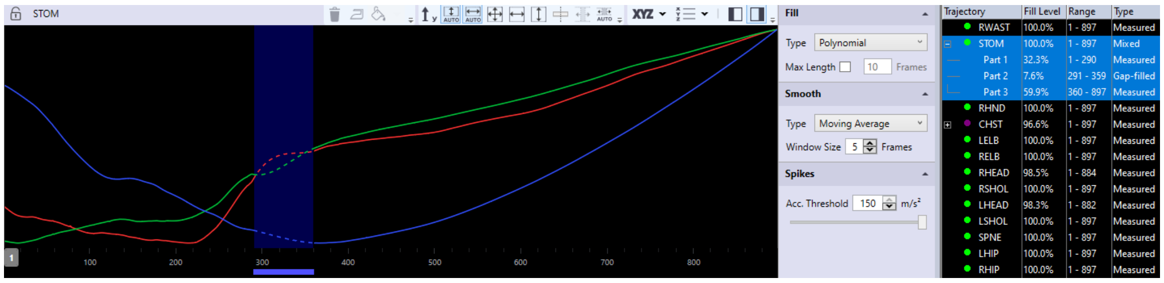
Acc. Threshold 150 m/s²

Figure 3-25: Filling Gaps; A gap (brown) in a "STOM" trajectory.

Figure 3-25 illustrates the "STOM" marker, for which we aim to fill the trajectory gap using various fill types. The gap begins at frame 291 and concludes at frame 359, resulting in a total gap size of 69 frames, as indicated in the trajectory editor. Consequently, we can calculate the fill level using the formula 100 * (897 - 69) / 897, which yields a fill level of 92.3%, as reflected in the Fill Level column next to "STOM." To fill selected gaps, click the Fill icon (paint bucket) in the trajectory editor. This will fill all chosen gaps based on the specified fill type in the settings sidebar.

3.8.5.1 Polynomial

The default fill type is Polynomial, which uses cubic polynomial interpolation to smoothly connect the X, Y, and Z curves across a gap. This method requires trajectory data on both sides of the gap. However, for larger gaps, the accuracy of the results may diminish; therefore, it is advisable to set a maximum number of frames for filling to ensure reliability. In instances where the gap exceeds the limits suitable for Polynomial filling, alternative methods such as Relational or Kinematic can be utilized, which will be discussed in subsequent sections. Figure 3-26 illustrates the application of polynomial gap filling on the "STOM" marker, achieving a complete fill level of 100% for this marker by incorporating the "Gap-filled" part2 from frames 291 to 359.



Fill

Type: Polynomial

Max Length ☐ 10 Frames

Smooth

Type: Moving Average

Window Size 5 Frames

Spikes

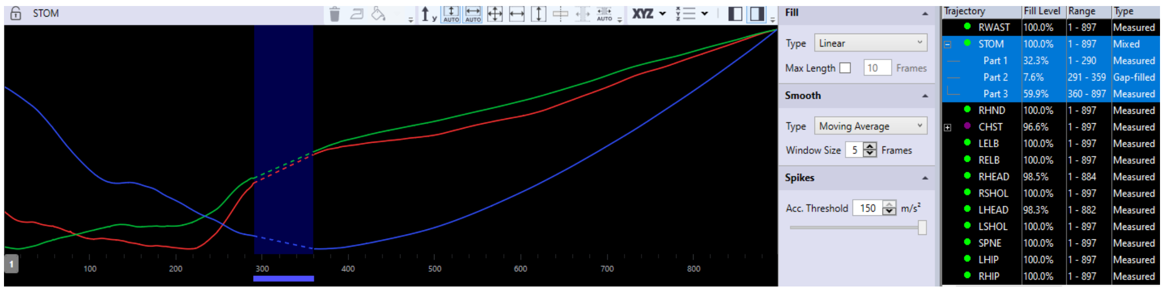
Acc. Threshold 150 m/s²

Trajectory	Fill Level	Range	Type
● RWAST	100.0%	1 - 897	Measured
● STOM	100.0%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	7.6%	291 - 359	Gap-filled
Part 3	59.9%	360 - 897	Measured
● RHND	100.0%	1 - 897	Measured
● CHST	96.6%	1 - 897	Measured
● LELB	100.0%	1 - 897	Measured
● RELB	100.0%	1 - 897	Measured
● RHEAD	98.5%	1 - 884	Measured
● RSHOL	100.0%	1 - 897	Measured
● LHEAD	98.3%	1 - 882	Measured
● LSHOL	100.0%	1 - 897	Measured
● SPNE	100.0%	1 - 897	Measured
● LHIP	100.0%	1 - 897	Measured
● RHIP	100.0%	1 - 897	Measured

Figure 3-26: Polynomial Gap Filling Type

3.8.5.2 Linear

Linear type fills gaps with a straight line connecting the X, Y, and Z coordinates on one side of the gap to those on the other side. This method is suitable for small gaps or constant velocity movement. If a gap occurs at the start of a trajectory, it will be filled with the first available data point following the gap. Conversely, if a gap appears at the end of a trajectory, it will be filled with the last data point recorded before the gap. A maximum frame length can be set for linear gap filling. Figure 3-27 displays the application of linear gap filling on the "STOM" marker, effectively completing its trajectory to 100%.



Fill

Type: Linear

Max Length ☐ 10 Frames

Smooth

Type: Moving Average

Window Size: 5 Frames

Spikes

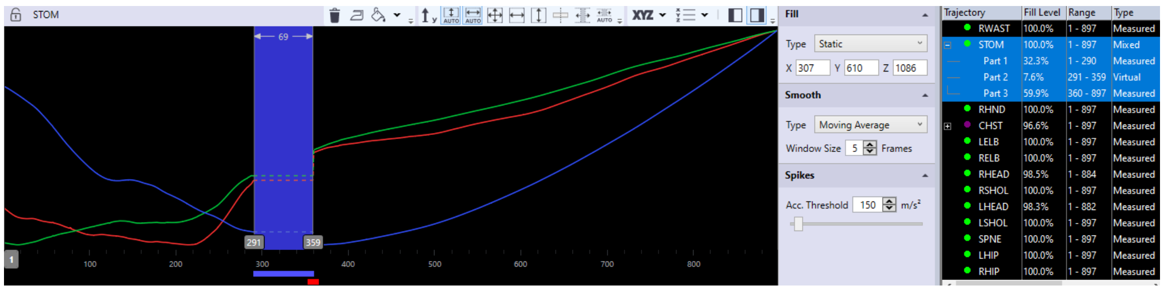
Acc. Threshold: 150 m/s²

Trajectory	Fill Level	Range	Type
● RWAST	100.0%	1 - 897	Measured
● STOM	100.0%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	7.6%	291 - 359	Gap-filled
Part 3	59.9%	360 - 897	Measured
● RHND	100.0%	1 - 897	Measured
● CHST	96.6%	1 - 897	Measured
● LELB	100.0%	1 - 897	Measured
● RELB	100.0%	1 - 897	Measured
● RHEAD	98.5%	1 - 884	Measured
● RSHOL	100.0%	1 - 897	Measured
● LHEAD	98.3%	1 - 882	Measured
● LSHOL	100.0%	1 - 897	Measured
● SPNE	100.0%	1 - 897	Measured
● LHIP	100.0%	1 - 897	Measured
● RHIP	100.0%	1 - 897	Measured

Figure 3-27: Linear Gap Filling Type

3.8.5.3 Static

Static is a virtual type that uses fixed values for X, Y, and Z throughout a gap. It is suitable for filling gaps in stationary objects with known coordinates. As it does not rely on data from the edges of the gap, it works even without surrounding data or in an empty trajectory. Figure 3-28 shows the application of this filling type to completely fill the gap in the "STOM" marker. The initial X, Y, Z coordinates at the start of the gap are treated as static values. This figure indicates that this filling method is not appropriate for such a gap.



Fill

Type: Static

X 307 Y 610 Z 1086

Smooth

Type: Moving Average

Window Size: 5 Frames

Spikes

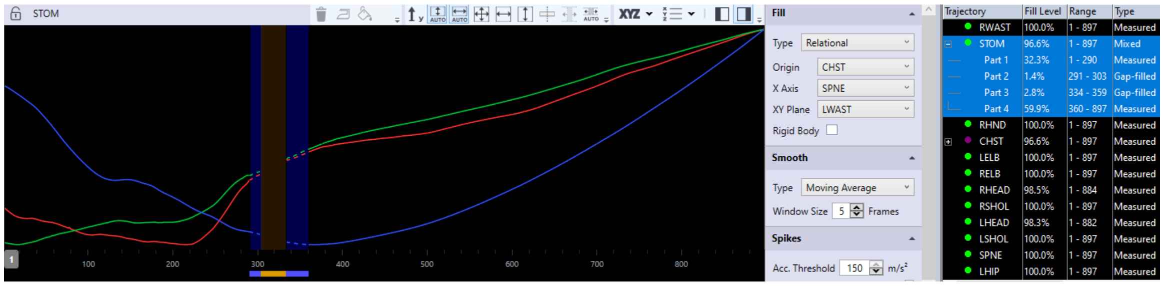
Acc. Threshold: 150 m/s²

Trajectory	Fill Level	Range	Type
● RWAST	100.0%	1 - 897	Measured
● STOM	100.0%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	7.6%	291 - 359	Virtual
Part 3	59.9%	360 - 897	Measured
● RHND	100.0%	1 - 897	Measured
● CHST	96.6%	1 - 897	Measured
● LELB	100.0%	1 - 897	Measured
● RELB	100.0%	1 - 897	Measured
● RHEAD	98.5%	1 - 884	Measured
● RSHOL	100.0%	1 - 897	Measured
● LHEAD	98.3%	1 - 882	Measured
● LSHOL	100.0%	1 - 897	Measured
● SPNE	100.0%	1 - 897	Measured
● LHIP	100.0%	1 - 897	Measured
● RHIP	100.0%	1 - 897	Measured

Figure 3-28: Static Gap Filling Type

3.8.5.4 Relational

The Relational type connects the curves on both sides of a gap based on the movement of surrounding markers. It is useful when tracking markers remain fixed with respect to each other, like a cluster. Up to three context markers should be selected to establish a local coordinate system for the missing data. An origin marker is necessary to define this system's origin, while an additional marker can set the X-axis and a third can define the XY plane. By checking "Rigid Body," QTM will treat these three context markers as a rigid body based on their average configuration during the gap. Figure 3-29 illustrates the outcome of filling the gap of the "STOM" marker using this filling type. We selected "CHST" as the origin, which has a filling level of 96.6%. Notably, the gap associated with "CHST" is entirely contained within the "STOM" gap. Consequently, by applying this filling method, we successfully filled the "STOM" trajectory to a level of 96.6%.



Fill
Type: Relational
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Rigid Body: ☐

Smooth
Type: Moving Average
Window Size: 5 Frames

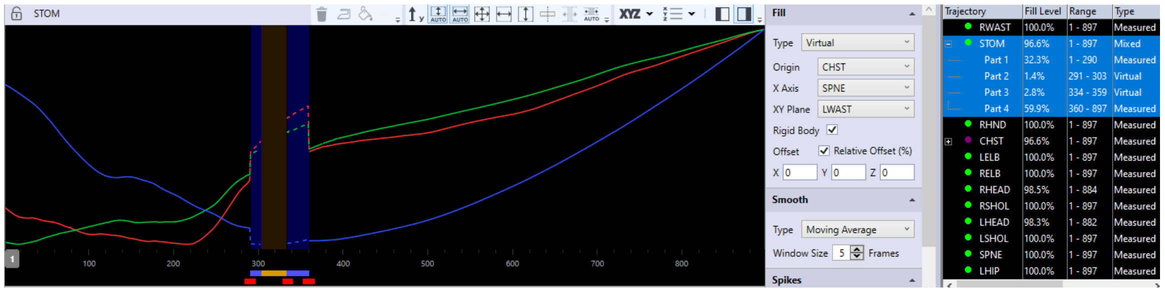
Spikes
Acc. Threshold: 150 m/s²

Trajectory	Fill Level	Range	Type
RWAST	100.0%	1 - 897	Measured
STOM	96.6%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	1.4%	291 - 303	Gap-filled
Part 3	2.8%	334 - 359	Gap-filled
Part 4	59.9%	360 - 897	Measured
RHND	100.0%	1 - 897	Measured
CHST	96.6%	1 - 897	Measured
LELB	100.0%	1 - 897	Measured
RELB	100.0%	1 - 897	Measured
RHEAD	98.5%	1 - 884	Measured
RSHOL	100.0%	1 - 897	Measured
LHEAD	98.3%	1 - 882	Measured
LSHOL	100.0%	1 - 897	Measured
SPNE	100.0%	1 - 897	Measured
LHIP	100.0%	1 - 897	Measured

Figure 3-29: Relational Gap Filling Type

3.8.5.5 Virtual

The Virtual type of gap-filling operates similarly to Relational gap-filling by addressing gaps based on the movement of surrounding markers; however, it distinguishes itself by being independent of the trajectory data at the edges of the gap. This independence allows the Virtual type to be utilized in scenarios involving empty trajectories or sections that lack surrounding data. As with Relational type, up to three context markers can be selected, and "Rigid Body" can be chosen to treat them as a rigid body. By default, the virtual trajectory will correspond to the trajectory designated as "Origin;" however, an offset can be applied to move the virtual trajectory to a different position. Similar to the Relational type, there exists a gap within the filled “STOM” trajectory associated with the “CHST” marker, as illustrated in Figure 3-30. Consequently, the resulting fill level is 96.3%.



Fill

Type: Virtual
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Rigid Body ✓
Offset ✓ Relative Offset (%)
X 0 Y 0 Z 0

Smooth

Type: Moving Average
Window Size 5 Frames

Spikes

Trajectory	Fill Level	Range	Type
RWAST	100.0%	1 - 897	Measured
STOM	96.6%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	1.4%	291 - 303	Virtual
Part 3	2.8%	334 - 359	Virtual
Part 4	59.9%	360 - 897	Measured
RHND	100.0%	1 - 897	Measured
CHST	96.6%	1 - 897	Measured
LELB	100.0%	1 - 897	Measured
RELB	100.0%	1 - 897	Measured
RHEAD	98.5%	1 - 884	Measured
RSHOL	100.0%	1 - 897	Measured
LHEAD	98.3%	1 - 882	Measured
LSHOL	100.0%	1 - 897	Measured
SPNE	100.0%	1 - 897	Measured
LHIP	100.0%	1 - 897	Measured

Figure 3-30: Virtual Gap Filling Type

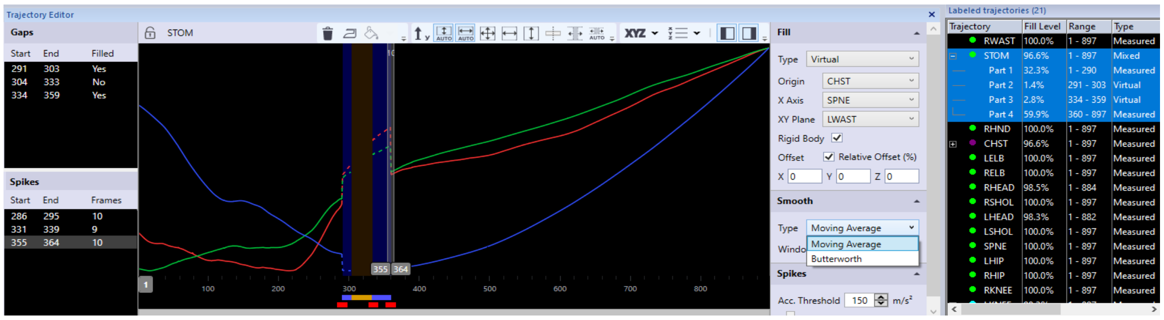
3.8.5.6 Kinematic

The Kinematic type fills gaps based on the movement of the skeleton segment or of the rigid body, suitable for tracking a rigid body or using the skeleton solver. Consequently, this specific method of gap filling is not applicable for the “STOM” marker.

3.8.6 Smoothing Spikes

Spikes [86] refer to discontinuities that occur between consecutive frames in a trajectory, characterized by sudden and significant changes in acceleration. These spikes can arise from various factors, with one primary cause being occlusion. Additionally, labeling errors, such as swapping markers, can contribute to their occurrence; these errors must be rectified before proceeding with further analysis. Any remaining spikes can be effectively identified and smoothed using a trajectory editor. QTM provides two types of smoothing [87]: moving average and Butterworth filtering, which will be elaborated upon in the following sections. Over-smoothing or filtering out crucial information should be avoided especially if acceleration changes are essential for the application. Figure 3-31 and Figure 3-32 illustrate the detected spikes and the smoothing process applied to them using the trajectory editor.

We can also click the Smooth icon (Iron icon) in the trajectory editor to smooth spikes. These spikes were generated using Virtual gap filling on the "STOM" marker, resulting in the creation of three distinct spikes. The blue bars show filled gap frames, red dots indicate data spikes, and yellow highlights the gap in the trajectory.


Trajectory Editor

STOM

Gaps

Start	End	Filled
291	303	Yes
304	333	No
334	359	Yes

Spikes

Start	End	Frames
286	295	10
331	339	9
355	364	10

Fill
Type: Virtual
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Rigid Body: ✓
Offset: ✓ Relative Offset (%)
X 0 Y 0 Z 0

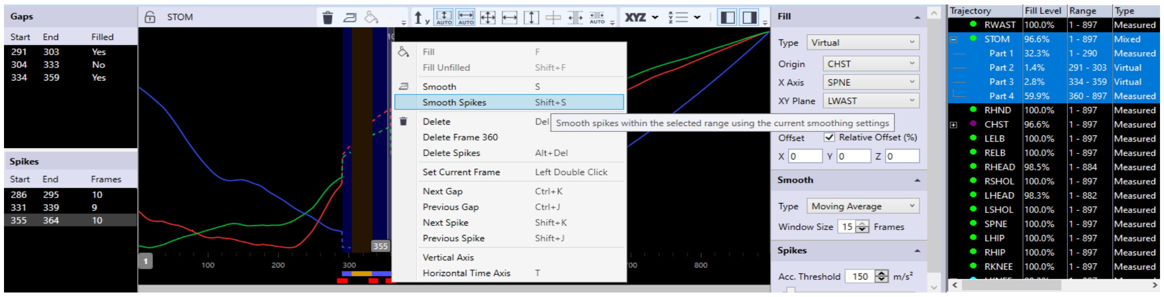
Smooth
Type: Moving Average
Moving Average
Butterworth

Spikes
Acc. Threshold 150 m/s²

Labeled trajectories (21)

Trajectory	Fill Level	Range	Type
RWAST	100.0%	1 - 897	Measured
STOM	96.6%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	1.4%	291 - 303	Virtual
Part 3	2.8%	334 - 359	Virtual
Part 4	59.9%	360 - 897	Measured
RHND	100.0%	1 - 897	Measured
CHST	96.6%	1 - 897	Measured
LELB	100.0%	1 - 897	Measured
RELB	100.0%	1 - 897	Measured
RHEAD	98.5%	1 - 884	Measured
RSHOL	100.0%	1 - 897	Measured
LHEAD	98.3%	1 - 882	Measured
LSHOL	100.0%	1 - 897	Measured
SPNE	100.0%	1 - 897	Measured
LHIP	100.0%	1 - 897	Measured
RHIP	100.0%	1 - 897	Measured
RKNEE	100.0%	1 - 897	Measured

Figure 3-31: Spikes Identification


STOM

Gaps

Start	End	Filled
291	303	Yes
304	333	No
334	359	Yes

Spikes

Start	End	Frames
286	295	10
331	339	9
355	364	10

Fill	F
Fill Unfilled	Shift+F
Smooth	S
Smooth Spikes	Shift+S
Delete	Del
Delete Frame 360	
Delete Spikes	Alt+Del
Set Current Frame	Left Double Click
Next Gap	Ctrl+K
Previous Gap	Ctrl+J
Next Spike	Shift+K
Previous Spike	Shift+J
Vertical Axis	
Horizontal Time Axis	T

Smooth spikes within the selected range using the current smoothing settings

Fill
Type: Virtual
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Offset: ✓ Relative Offset (%)
X 0 Y 0 Z 0

Smooth
Type: Moving Average
Window Size 15 Frames

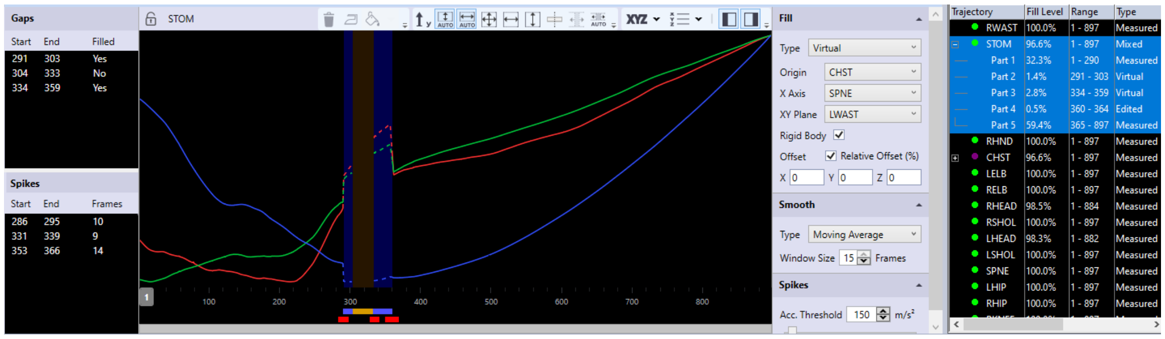
Spikes
Acc. Threshold 150 m/s²

Trajectory	Fill Level	Range	Type
RWAST	100.0%	1 - 897	Measured
STOM	96.6%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	1.4%	291 - 303	Virtual
Part 3	2.8%	334 - 359	Virtual
Part 4	59.9%	360 - 897	Measured
RHND	100.0%	1 - 897	Measured
CHST	96.6%	1 - 897	Measured
LELB	100.0%	1 - 897	Measured
RELB	100.0%	1 - 897	Measured
RHEAD	98.5%	1 - 884	Measured
RSHOL	100.0%	1 - 897	Measured
LHEAD	98.3%	1 - 882	Measured
LSHOL	100.0%	1 - 897	Measured
SPNE	100.0%	1 - 897	Measured
LHIP	100.0%	1 - 897	Measured
RHIP	100.0%	1 - 897	Measured
RKNEE	100.0%	1 - 897	Measured

Figure 3-32: Smoothing Spikes

3.8.6.1 Moving Average

The Moving Average type (Figure 3-33) is suitable for local spikes, as it averages the data in a customizable window (up to 15 frames) around the current frame.


STOM

Gaps

Start	End	Filled
291	303	Yes
304	333	No
334	359	Yes

Spikes

Start	End	Frames
286	295	10
331	339	9
353	366	14

Fill
Type: Virtual
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Rigid Body: ✓
Offset: ✓ Relative Offset (%)
X 0 Y 0 Z 0

Smooth
Type: Moving Average
Window Size 15 Frames

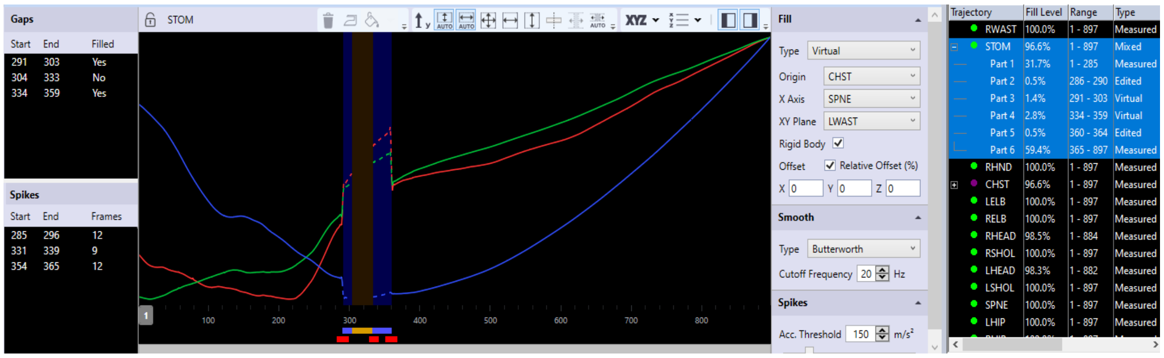
Spikes
Acc. Threshold 150 m/s²

Trajectory	Fill Level	Range	Type
RWAST	100.0%	1 - 897	Measured
STOM	96.6%	1 - 897	Mixed
Part 1	32.3%	1 - 290	Measured
Part 2	1.4%	291 - 303	Virtual
Part 3	2.8%	334 - 359	Virtual
Part 4	0.5%	360 - 364	Edited
Part 5	59.4%	365 - 897	Measured
RHND	100.0%	1 - 897	Measured
CHST	96.6%	1 - 897	Measured
LELB	100.0%	1 - 897	Measured
RELB	100.0%	1 - 897	Measured
RHEAD	98.5%	1 - 884	Measured
RSHOL	100.0%	1 - 897	Measured
LHEAD	98.3%	1 - 882	Measured
LSHOL	100.0%	1 - 897	Measured
SPNE	100.0%	1 - 897	Measured
LHIP	100.0%	1 - 897	Measured
RHIP	100.0%	1 - 897	Measured

Figure 3-33: Moving Average Smoothing Type

3.8.6.2 Butterworth

The Butterworth filter is particularly effective for extensive frame ranges that experience significant high-frequency noise, using a low-pass filter to attenuate information above the specified cutoff frequency. The initial cutoff frequency should be set at 2-3 times higher than the highest frequency to be retained and can be adjusted as necessary. Figure 3-34 and Figure 3-35 displays the application of Butterworth smoothing on the "STOM" marker, utilizing cutoff frequencies of 20 and 5, respectively. This demonstrates that a lower cutoff frequency results in a greater degree of signal smoothing.



Gaps

Start	End	Filled
291	303	Yes
304	333	No
334	359	Yes

Spikes

Start	End	Frames
285	296	12
331	339	9
354	365	12

Fill

Type: Virtual
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Rigid Body ✓
Offset ✓ Relative Offset (%)
X 0 Y 0 Z 0

Smooth

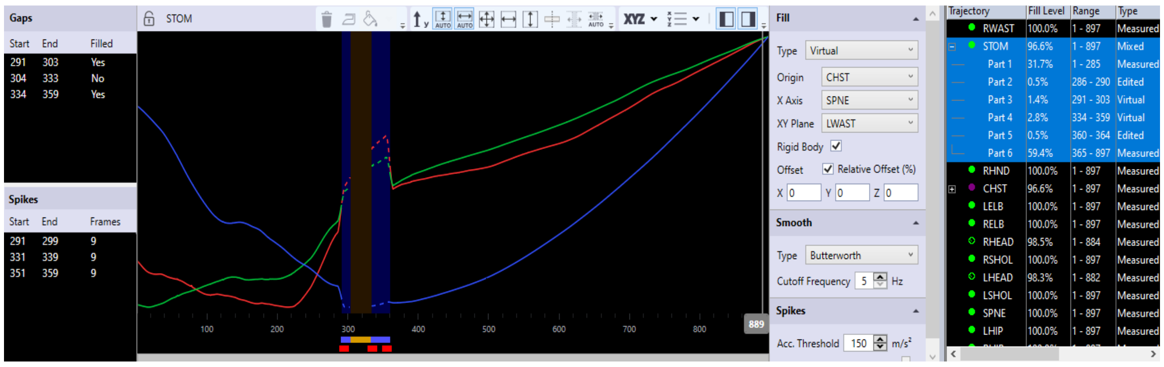
Type: Butterworth
Cutoff Frequency 20 Hz

Spikes

Acc. Threshold 150 m/s²

Trajectory	Fill Level	Range	Type
● RWAST	100.0%	1 - 897	Measured
● STOM	96.6%	1 - 897	Mixed
Part 1	31.7%	1 - 285	Measured
Part 2	0.5%	286 - 290	Edited
Part 3	1.4%	291 - 303	Virtual
Part 4	2.8%	334 - 359	Virtual
Part 5	0.5%	360 - 364	Edited
Part 6	59.4%	365 - 897	Measured
● RHND	100.0%	1 - 897	Measured
● CHST	96.6%	1 - 897	Measured
● LELB	100.0%	1 - 897	Measured
● RELB	100.0%	1 - 897	Measured
● RHEAD	98.5%	1 - 884	Measured
● RSHOL	100.0%	1 - 897	Measured
● LHEAD	98.3%	1 - 882	Measured
● LSHOL	100.0%	1 - 897	Measured
● SPNE	100.0%	1 - 897	Measured
● LHIP	100.0%	1 - 897	Measured

Figure 3-34: Butterworth Smoothing Type Response with Cutoff Frequency = 20 Hz



Gaps

Start	End	Filled
291	303	Yes
304	333	No
334	359	Yes

Spikes

Start	End	Frames
291	299	9
331	339	9
351	359	9

Fill

Type: Virtual
Origin: CHST
X Axis: SPNE
XY Plane: LWAST
Rigid Body ✓
Offset ✓ Relative Offset (%)
X 0 Y 0 Z 0

Smooth

Type: Butterworth
Cutoff Frequency 5 Hz

Spikes

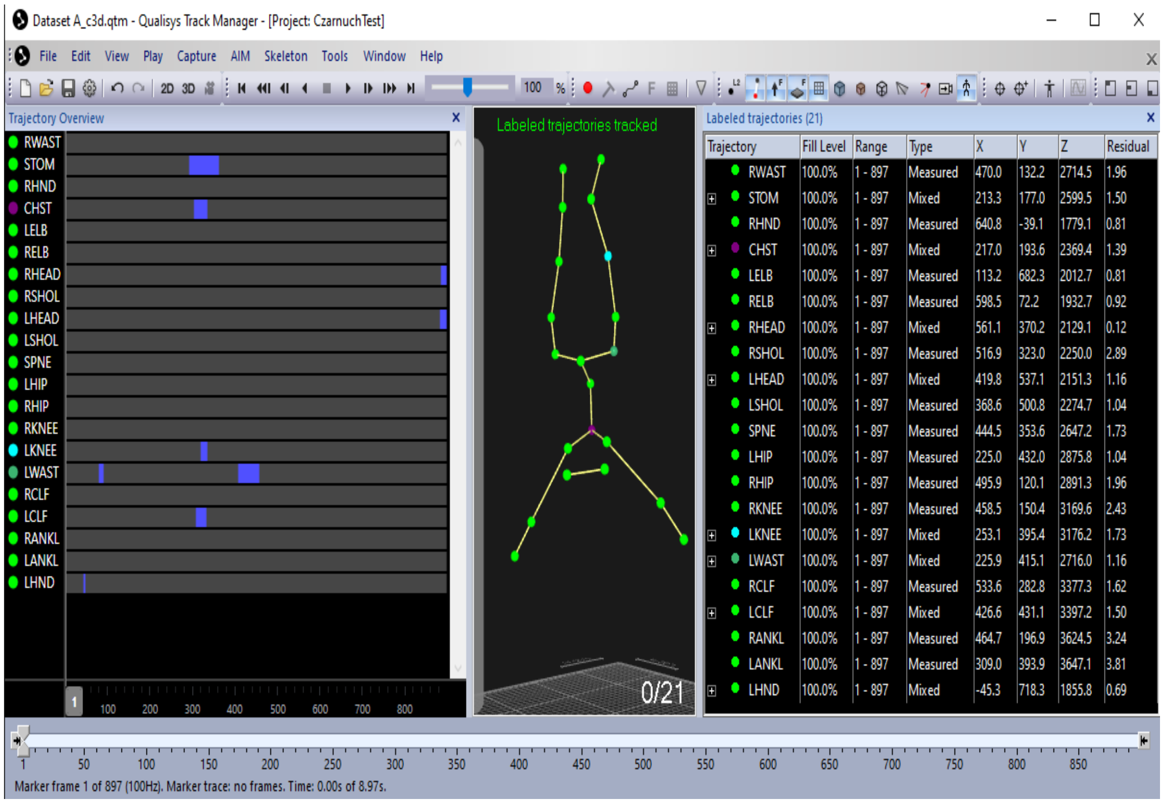
Acc. Threshold 150 m/s²

Trajectory	Fill Level	Range	Type
● RWAST	100.0%	1 - 897	Measured
● STOM	96.6%	1 - 897	Mixed
Part 1	31.7%	1 - 285	Measured
Part 2	0.5%	286 - 290	Edited
Part 3	1.4%	291 - 303	Virtual
Part 4	2.8%	334 - 359	Virtual
Part 5	0.5%	360 - 364	Edited
Part 6	59.4%	365 - 897	Measured
● RHND	100.0%	1 - 897	Measured
● CHST	96.6%	1 - 897	Measured
● LELB	100.0%	1 - 897	Measured
● RELB	100.0%	1 - 897	Measured
○ RHEAD	98.5%	1 - 884	Measured
● RSHOL	100.0%	1 - 897	Measured
○ LHEAD	98.3%	1 - 882	Measured
● LSHOL	100.0%	1 - 897	Measured
● SPNE	100.0%	1 - 897	Measured
● LHIP	100.0%	1 - 897	Measured

Figure 3-35: Butterworth Smoothing Type Response with Cutoff Frequency = 5 Hz

3.8.7 Manual Cleaning Output

The manual cleaning process applied to our *Dataset A* results in a final C3D output that is meticulously cleaned and accurately labeled with 21 markers, in accordance with the specified marker set. This output is free from any noise or gaps. Figure 3-36 illustrates the final output after manual cleaning. This output encompasses the skeleton visualization of the swimmer's posture in the first frame, the Trajectory Overview, and detailed information regarding the error correction for each label. The Type column specifies the nature of error correction, which can be categorized as either "Measured" or "Mixed." The trajectories classified as "Measured" were directly tracked by the QTM during the motion capture trial. The "Mixed" type may include segments that are "Gap-filled," resulting from gap filling processes, or "Edited," which pertains to smoothing adjustments. The number of parts shows the number of trajectory segments for each marker. In the Trajectory Overview, the blue bars indicate the frames where filled gaps are present for each label, effectively showcasing their size and locations.



Labeled trajectories (21)

Trajectory	Fill Level	Range	Type	X	Y	Z	Residual
RWAST	100.0%	1 - 897	Measured	470.0	132.2	2714.5	1.96
STOM	100.0%	1 - 897	Mixed	213.3	177.0	2599.5	1.50
RHND	100.0%	1 - 897	Measured	640.8	-39.1	1779.1	0.81
CHST	100.0%	1 - 897	Mixed	217.0	193.6	2369.4	1.39
LELB	100.0%	1 - 897	Measured	113.2	682.3	2012.7	0.81
RELB	100.0%	1 - 897	Measured	598.5	72.2	1932.7	0.92
RHEAD	100.0%	1 - 897	Mixed	561.1	370.2	2129.1	0.12
RSHOL	100.0%	1 - 897	Measured	516.9	323.0	2250.0	2.89
LHEAD	100.0%	1 - 897	Mixed	419.8	537.1	2151.3	1.16
LSHOL	100.0%	1 - 897	Measured	368.6	500.8	2274.7	1.04
SPNE	100.0%	1 - 897	Measured	444.5	353.6	2647.2	1.73
LHIP	100.0%	1 - 897	Measured	225.0	432.0	2875.8	1.04
RHIP	100.0%	1 - 897	Measured	495.9	120.1	2891.3	1.96
RKNEE	100.0%	1 - 897	Measured	458.5	150.4	3169.6	2.43
LKNEE	100.0%	1 - 897	Mixed	253.1	395.4	3176.2	1.73
LWAST	100.0%	1 - 897	Mixed	225.9	415.1	2716.0	1.16
RCLF	100.0%	1 - 897	Measured	533.6	282.8	3377.3	1.62
LCLF	100.0%	1 - 897	Mixed	426.6	431.1	3397.2	1.50
RANKL	100.0%	1 - 897	Measured	464.7	196.9	3624.5	3.24
LANKL	100.0%	1 - 897	Measured	309.0	393.9	3647.1	3.81
LHND	100.0%	1 - 897	Mixed	-45.3	718.3	1855.8	0.69

Marker frame 1 of 897 (100Hz). Marker trace: no frames. Time: 0.00s of 8.97s.

Figure 3-36: Manual Cleaning Output

3.9 Labeling using AIM Models

Automatic identification of markers (AIM) [17] helps automatically identify and labels trajectories based on a created model [18], [19] for a specific marker set. AIM models are also learning models, which implies that new trials can be added to the existing model to enhance its knowledge. The initial trial defines the connections between the markers; however, training [88] the existing model with new trials provides additional examples of distances and angles between markers. This process will improve the accuracy of applying [89] the trained model to future test subjects. The generated AIM model can be applied to any recording with the same marker set and similar motions.

In the following sections, we describe how to generate an AIM model for our marker set and apply this AIM model to our *Dataset A*. Then, we compare the results of our manual cleaning on *Dataset A* with the results of applying the AIM model to it. We also apply this AIM model to a new C3D dataset.

3.9.1 AIM model Generation and Application Procedure

For an AIM model to function effectively, it is crucial that we “teach” it using a dataset comprised of thoroughly cleaned and accurately labeled trials. This can be achieved by utilizing either manually cleaned and labeled archival trials or by recording specific trials tailored for this purpose and ensuring they are thoroughly cleaned.

To effectively record a trial for the creation of an AIM model, it is advisable to start with the subject in a “T-pose,” with arms extended sideways. This position facilitates easier identification of markers. It is essential that the subject performs the complete range of motion that is intended for tracking by the AIM model.

To create an AIM model using archival MoCap data or a specific trial, we must first manually clean and label the data and create the visual bone connections between markers. Once this is done, during the AIM model creation process, QTM will display the learned connectivity based on the training dataset, which differs from our bone connectivity. It is essential to review and correct this connectivity if needed before generating and saving the AIM model for applying it on a dataset. The summary of the procedure for creating an AIM model and applying it on a dataset, using QTM, is as follows:

1- Clean and label a C3D MoCap file and create bone connectivity in QTM.

2- Select "AIM" from the top menu bar and choose "Generate Model" (Figure 3-37).

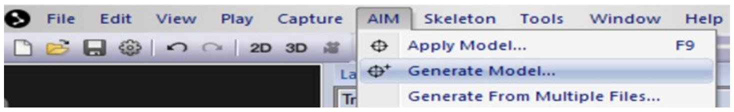
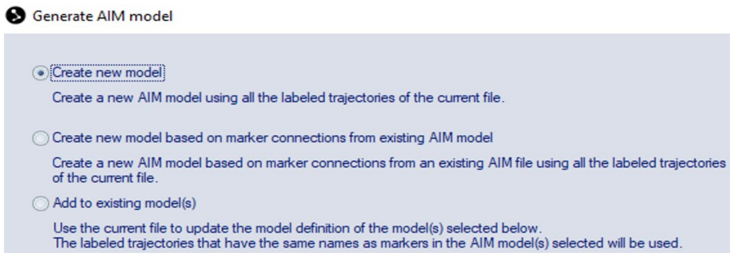


Figure 3-37: QTM "AIM" icon

3- In the opened window, select "Create a new model" (Figure 3-38).


Generate AIM model

(●) Create new model
Create a new AIM model using all the labeled trajectories of the current file.

( ) Create new model based on marker connections from existing AIM model
Create a new AIM model based on marker connections from an existing AIM file using all the labeled trajectories of the current file.

( ) Add to existing model(s)
Use the current file to update the model definition of the model(s) selected below.
The labeled trajectories that have the same names as markers in the AIM model(s) selected will be used.

Figure 3-38: Create a new AIM model

4- In "Verify and edit AIM structure bones" windows, rectify any incorrect connections.

5- Click "Next" again, enter a name for the AIM model, and Click "OK."

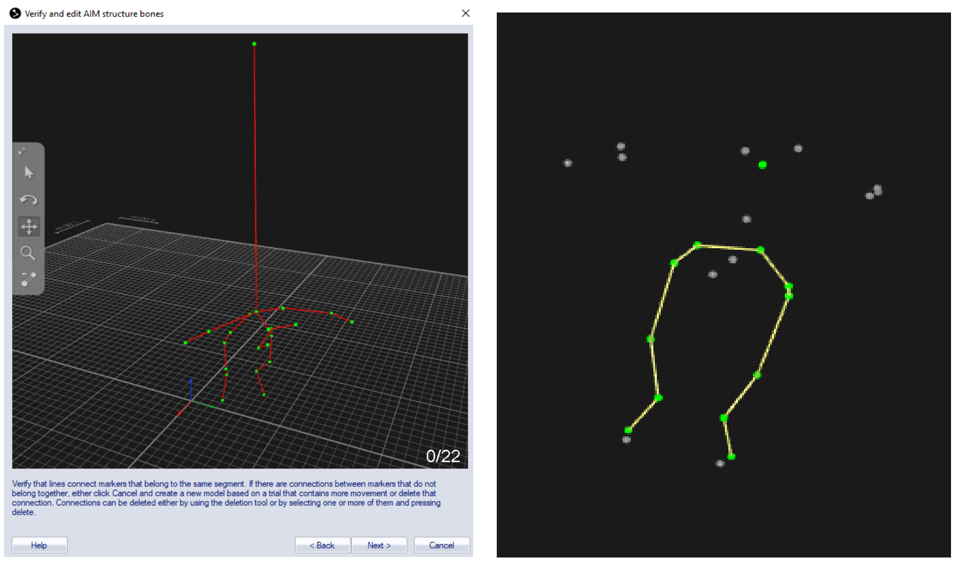
6- Click "Finish" in a message window that displays the successful generation.

7- Open a raw C3D file, choose "Apply Model" (Figure 3-37) and then the AIM model.

8- Check the result, correct any mislabeling, and fill in any gaps.

3.9.2 Result of Creating an AIM Model using a Noisy Trial

Figure 3-39 (a) shows that using a noisy trial to create an AIM model leads to incorrect red bone connections. When this flawed model is applied to the same unlabeled motion trial, it produces incomplete and incorrect results as shown in Figure 3-39 (b).



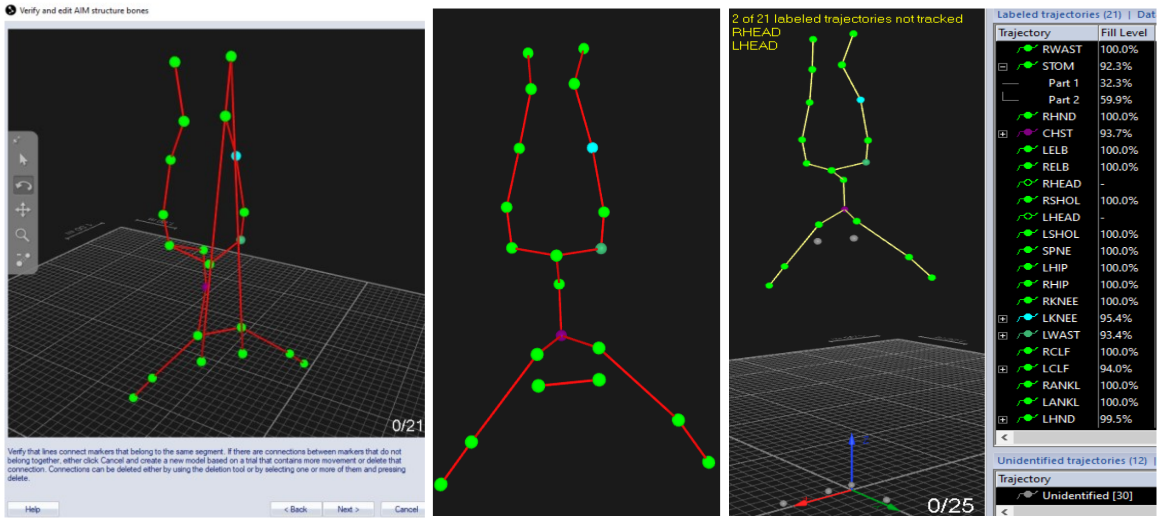
Verify that lines connect markers that belong to the same segment. If there are connections between markers that do not belong together, either click Cancel and create a new model based on a trial that contains more movement or delete that connection. Connections can be deleted either by using the deletion tool or by selecting one or more of them and pressing delete.

(a) (b)

Figure 3-39: Failure of AIM model; (a) Failure of an AIM model generation based on a noisy trial; (b) The result of applying an inaccurate AIM model

3.9.3 Results of Creating an AIM Model based on Initial Bad Connections

Figure 3-36 displays our cleaned and labeled *Dataset A* with bone connectivity which we use as a trial to create and teach an AIM model. Figure 3-40 (a) displays the preview model created by QTM based on the trained trial. Some connections are incorrect, which we have corrected in Figure 3-40 (b). We save the created AIM model, then apply it to our raw *Dataset A*. The result shown in Figure 3-40 (c) indicates that the head markers were not detected, likely due to their lack of connections to other markers. We will investigate this further in the next section by adjusting the initial connectivity.



Verify that lines connect markers that belong to the same segment. If there are connections between markers that do not belong together, either click Cancel and create a new model based on a trial that contains more movement or delete that connection. Connections can be deleted either by using the deletion tool or by selecting one or more of them and pressing delete.

2 of 21 labeled trajectories not tracked
RHEAD
LHEAD

Labeled trajectories (21)

Trajectory	Fill Level
RWAST	100.0%
STOM	92.3%
Part 1	32.3%
Part 2	59.9%
RHND	100.0%
CHST	93.7%
LELB	100.0%
RELB	100.0%
RHEAD	-
RSHOL	100.0%
LHEAD	-
LSHOL	100.0%
SPNE	100.0%
LHIP	100.0%
RHIP	100.0%
RKNEE	100.0%
LKNEE	95.4%
LWAST	93.4%
RCLF	100.0%
LCLF	94.0%
RANKL	100.0%
LANKL	100.0%
LHND	99.5%

Unidentified trajectories (12)

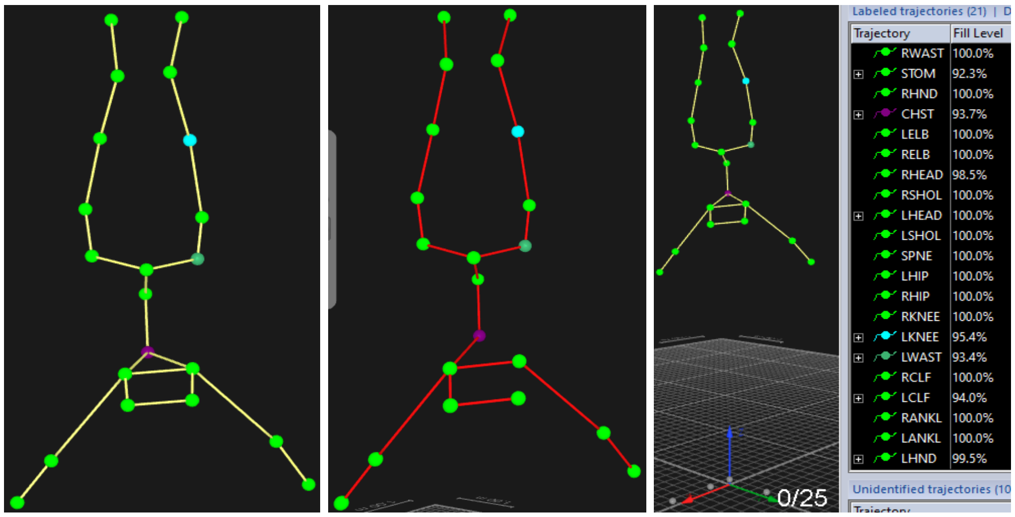
Trajectory
Unidentified [30]

(a) (b) (c)

Figure 3-40: AIM model labeling based on initial bad connections: (a) AIM model with wrong connections; (b) AIM model with corrected connections; (c) Result of applying the generated AIM model on Dataset A.

3.9.4 Results of Creating an AIM Model based on Initial Good Connections

Figure 3-41 (a), (b), and (c) illustrate the new bone connectivity in our cleaned *Dataset A*, the updated red connections between markers in the model preview, and the results of applying the AIM model to our raw *Dataset A*, which now successfully detects head markers. We now need to thoroughly examine the results frame by frame for accuracy.



Labeled trajectories (21)

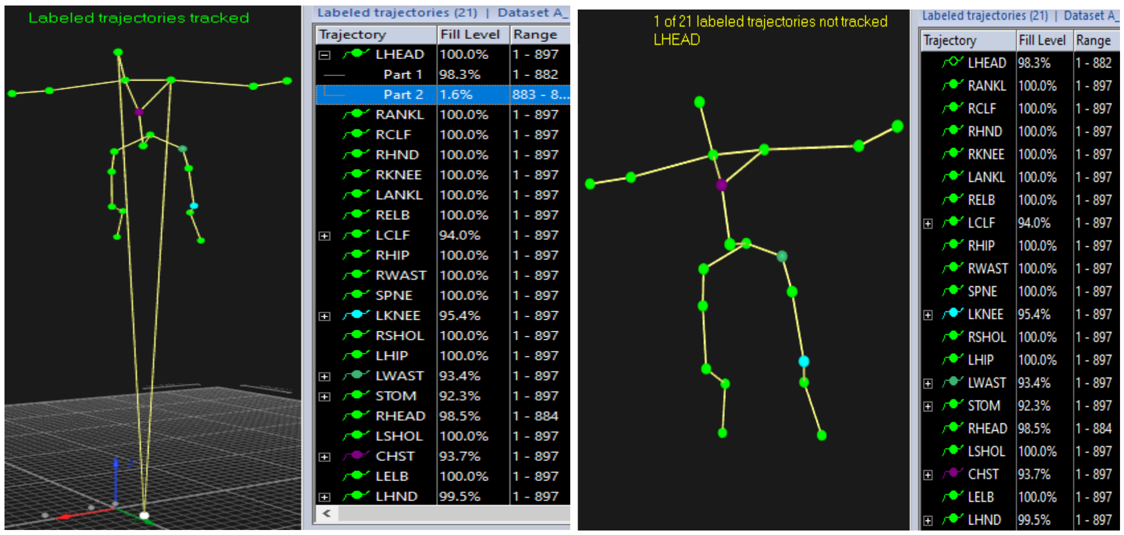
Trajectory	Fill Level
RWAST	100.0%
STOM	92.3%
RHND	100.0%
CHST	93.7%
LELB	100.0%
RELB	100.0%
RHEAD	98.5%
RSHOL	100.0%
LHEAD	100.0%
LSHOL	100.0%
SPNE	100.0%
LHIP	100.0%
RHIP	100.0%
RKNEE	100.0%
LKNEE	95.4%
LWAST	93.4%
RCLF	100.0%
LCLF	94.0%
RANKL	100.0%
LANKL	100.0%
LHND	99.5%

Unidentified trajectories (10

(a) (b) (c)

Figure 3-41: AIM model labeling based on initial good connections: (a) Initial bone connectivity; (b) AIM model with corrected connections; (c) The result of applying the generated AIM model on Dataset A.

During a frame-by-frame inspection of labeled *Dataset A* using the AIM model, we identified an incorrect labeling in frame 883, as illustrated in Figure 3-42 (a). The "LHEAD" marker Part 2 is mistakenly linked to the "LHEAD" marker, while it is evident from the image that this part is an extraneous marker. This likely occurred due to the occlusion of the "LHEAD" marker from frame 883 to frame 897. In previous frames, when the swimmer descends near the pool floor, "LHEAD" comes close to this extraneous marker. Therefore, it seems that during these occlusion frames, this extraneous marker was mistakenly detected as "LHEAD." Consequently, we have removed this erroneous Part 2 from the "LHEAD," and the corrected result is displayed in Figure 3-42 (b).


Labeled trajectories tracked
1 of 21 labeled trajectories not tracked
LHEAD

(a) Labeled trajectories (21) | Dataset A_

Trajectory	Fill Level	Range
LHEAD	100.0%	1 - 897
Part 1	98.3%	1 - 882
Part 2	1.6%	883 - 8...
RANKL	100.0%	1 - 897
RCLF	100.0%	1 - 897
RHND	100.0%	1 - 897
RKNEE	100.0%	1 - 897
LANKL	100.0%	1 - 897
RELB	100.0%	1 - 897
LCLF	94.0%	1 - 897
RHIP	100.0%	1 - 897
RWAST	100.0%	1 - 897
SPNE	100.0%	1 - 897
LKNEE	95.4%	1 - 897
RSHOL	100.0%	1 - 897
LHIP	100.0%	1 - 897
LWAST	93.4%	1 - 897
STOM	92.3%	1 - 897
RHEAD	98.5%	1 - 884
LSHOL	100.0%	1 - 897
CHST	93.7%	1 - 897
LELB	100.0%	1 - 897
LHND	99.5%	1 - 897

(b) Labeled trajectories (21) | Dataset A_

Trajectory	Fill Level	Range
LHEAD	98.3%	1 - 882
RANKL	100.0%	1 - 897
RCLF	100.0%	1 - 897
RHND	100.0%	1 - 897
RKNEE	100.0%	1 - 897
LANKL	100.0%	1 - 897
RELB	100.0%	1 - 897
LCLF	94.0%	1 - 897
RHIP	100.0%	1 - 897
RWAST	100.0%	1 - 897
SPNE	100.0%	1 - 897
LKNEE	95.4%	1 - 897
RSHOL	100.0%	1 - 897
LHIP	100.0%	1 - 897
LWAST	93.4%	1 - 897
STOM	92.3%	1 - 897
RHEAD	98.5%	1 - 884
LSHOL	100.0%	1 - 897
CHST	93.7%	1 - 897
LELB	100.0%	1 - 897
LHND	99.5%	1 - 897

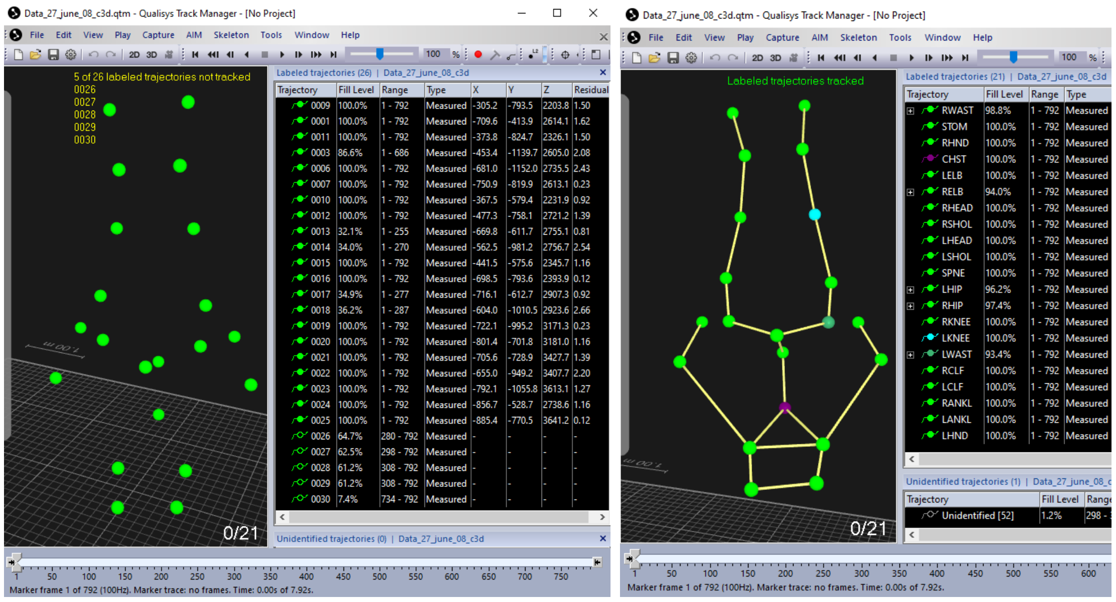
Figure 3-42: An incorrect labeling found during inspection: (a) Mislabeled "LHEAD" part2; (b) Corrected Labeling.

3.9.5 Comparison Manual Cleaning and Cleaning using an AIM Model

A comparison of the fill levels in Figure 3-36 and Figure 3-42 (b), which represent the results of manual cleaning and labeling of *Dataset A* using the AIM model, indicates that the AIM model only labels data without filling gaps.

3.9.6 Applying an Existing AIM model on a New C3D file

In Figure 3-43 (a) the AIM model is applied to a new C3D file using the same marker set and similar motion. This dataset is less noisy than *Dataset A*. The successful visual inspection indicates that the AIM model effectively labeled this dataset. However, the Fill Level suggests that further gap filling is still needed.


Data_27_june_08_c3d.qtm - Qualisys Track Manager - [No Project]
File Edit View Play Capture AIM Skeleton Tools Window Help
5 of 26 labeled trajectories not tracked
0026
0027
0028
0029
0030
0/21
Labeled trajectories tracked
Marker frame 1 of 792 (100Hz). Marker trace: no frames. Time: 0.00s of 7.92s.

Labeled trajectories (26) | Data_27_june_08_c3d

Trajectory	Fill Level	Range	Type	X	Y	Z	Residual
0009	100.0%	1 - 792	Measured	-305.2	-793.5	2203.8	1.50
0001	100.0%	1 - 792	Measured	-709.6	-413.9	2614.1	1.62
0011	100.0%	1 - 792	Measured	-373.8	-824.7	2326.1	1.50
0003	86.6%	1 - 686	Measured	-453.4	-1139.7	2605.0	2.08
0006	100.0%	1 - 792	Measured	-681.0	-1152.0	2735.5	2.43
0007	100.0%	1 - 792	Measured	-750.9	-819.9	2613.1	0.23
0010	100.0%	1 - 792	Measured	-367.5	-579.4	2231.9	0.92
0012	100.0%	1 - 792	Measured	-477.3	-758.1	2721.2	1.39
0013	32.1%	1 - 255	Measured	-669.8	-611.7	2755.1	0.81
0014	34.0%	1 - 270	Measured	-562.5	-981.2	2756.7	2.54
0015	100.0%	1 - 792	Measured	-441.5	-575.6	2345.7	1.16
0016	100.0%	1 - 792	Measured	-698.5	-793.6	2393.9	0.12
0017	34.9%	1 - 277	Measured	-716.1	-612.7	2907.3	0.92
0018	36.2%	1 - 287	Measured	-604.0	-1010.5	2923.6	2.66
0019	100.0%	1 - 792	Measured	-722.1	-995.2	3171.3	0.23
0020	100.0%	1 - 792	Measured	-801.4	-701.8	3181.0	1.16
0021	100.0%	1 - 792	Measured	-705.6	-728.9	3427.7	1.39
0022	100.0%	1 - 792	Measured	-655.0	-949.2	3407.7	2.20
0023	100.0%	1 - 792	Measured	-792.1	-1055.8	3613.1	1.27
0024	100.0%	1 - 792	Measured	-856.7	-528.7	2738.6	1.16
0025	100.0%	1 - 792	Measured	-885.4	-770.5	3641.2	0.12
0026	64.7%	280 - 792	Measured	-	-	-	-
0027	62.5%	298 - 792	Measured	-	-	-	-
0028	61.2%	308 - 792	Measured	-	-	-	-
0029	61.2%	308 - 792	Measured	-	-	-	-
0030	7.4%	734 - 792	Measured	-	-	-	-

Unidentified trajectories (0) | Data_27_june_08_c3d

(a)

Labeled trajectories (21) | Data_27_june_08_c3d

Trajectory	Fill Level	Range	Type
RWAST	98.8%	1 - 792	Measured
STOM	100.0%	1 - 792	Measured
RHND	100.0%	1 - 792	Measured
CHST	100.0%	1 - 792	Measured
LELB	100.0%	1 - 792	Measured
RELB	94.0%	1 - 792	Measured
RHEAD	100.0%	1 - 792	Measured
RSHOL	100.0%	1 - 792	Measured
LHEAD	100.0%	1 - 792	Measured
LSHOL	100.0%	1 - 792	Measured
SPNE	100.0%	1 - 792	Measured
LHIP	96.2%	1 - 792	Measured
RHIP	97.4%	1 - 792	Measured
RKNEE	100.0%	1 - 792	Measured
LKNEE	100.0%	1 - 792	Measured
LWAST	93.4%	1 - 792	Measured
RCLF	100.0%	1 - 792	Measured
LCLF	100.0%	1 - 792	Measured
RANKL	100.0%	1 - 792	Measured
LANKL	100.0%	1 - 792	Measured
LHND	100.0%	1 - 792	Measured

Unidentified trajectories (1) | Data_27_june_08_c

Trajectory	Fill Level	Range
Unidentified [52]	1.2%	298 -

(b)

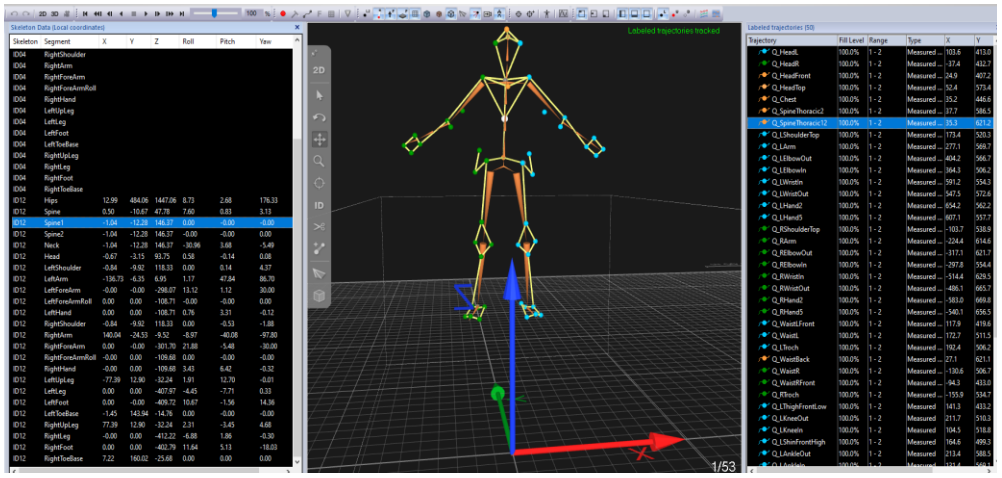
Figure 3-43: Applying AIM model on a new C3D file; (a) New C3D file; (b) The result of applying AIM model.

3.10 Other QTM Features

This section provides a brief overview of the skeleton solver and rigid bodies, which are crucial for animation and the development of six degrees of freedom (6DoF) models.

3.10.1 Skeleton Solver

The skeleton solver [20] calculates and displays skeleton data based on a specific marker set. This can be exported to TSV and FBX formats or streamed in real-time to animation software for retargeting [7]. The procedure includes labeling the skeleton markers, calibrating the skeleton [24], and acquiring the skeleton data (i.e., X, Y, and Z coordinates, and Roll, Pitch, and Yaw angels) [25], (Figure 3-44).



Skeleton Data (Local coordinates)

Skeleton	Segment	X	Y	Z	Roll	Pitch	Yaw
ID04	RightShoulder						
ID04	RightArm						
ID04	RightForeArm						
ID04	RightForeArmRoll						
ID04	RightHand						
ID04	LeftUpLeg						
ID04	LeftLeg						
ID04	LeftFoot						
ID04	LeftToeBase						
ID04	RightUpLeg						
ID04	RightLeg						
ID04	RightFoot						
ID04	RightToeBase						
ID12	Hips	12.99	484.06	1447.06	8.73	2.68	176.33
ID12	Spine	0.50	-10.67	47.78	7.60	0.83	3.13
ID12	Spine1	-1.04	-12.28	146.37	0.00	-0.00	-0.00
ID12	Spine2	-1.04	-12.28	146.37	-0.00	-0.00	0.00
ID12	Neck	-1.04	-12.28	146.37	-30.96	3.68	-5.49
ID12	Head	-0.67	-3.15	93.75	0.58	-0.14	0.08
ID12	LeftShoulder	-0.84	-9.92	118.33	0.00	0.14	4.37
ID12	LeftArm	-136.73	-6.35	6.95	1.17	47.84	86.70
ID12	LeftForeArm	-0.00	-0.00	-298.07	13.12	1.12	30.00
ID12	LeftForeArmRoll	0.00	0.00	-108.71	-0.00	-0.00	0.00
ID12	LeftHand	0.00	0.00	-108.71	0.76	3.31	-0.12
ID12	RightShoulder	-0.84	-9.92	118.33	0.00	-0.53	-1.88
ID12	RightArm	140.04	-24.53	-9.52	-8.97	-40.08	-97.80
ID12	RightForeArm	0.00	-0.00	-301.70	21.88	-5.48	-30.00
ID12	RightForeArmRoll	-0.00	0.00	-109.68	0.00	-0.00	-0.00
ID12	RightHand	-0.00	0.00	-109.68	3.43	6.42	-0.32
ID12	LeftUpLeg	-77.39	12.90	-32.24	1.91	12.70	-0.01
ID12	LeftLeg	0.00	0.00	-407.97	-4.45	-7.71	0.33
ID12	LeftFoot	0.00	-0.00	-409.72	10.67	-1.56	14.36
ID12	LeftToeBase	-1.45	143.94	-14.76	0.00	-0.00	-0.00
ID12	RightUpLeg	77.39	12.90	-32.24	2.31	-3.45	4.68
ID12	RightLeg	-0.00	0.00	-412.22	-6.88	1.86	-0.30
ID12	RightFoot	0.00	0.00	-402.79	11.64	5.13	-18.03
ID12	RightToeBase	7.22	160.02	-25.68	0.00	0.00	0.00

Labeled trajectories (50)

Trajectory	Fill Level	Range	Type	X	Y
Q_HeadL	100.0%	1 - 2	Measured ...	103.6	413.0
Q_HeadR	100.0%	1 - 2	Measured ...	-37.4	432.7
Q_HeadFront	100.0%	1 - 2	Measured ...	24.9	407.2
Q_HeadTop	100.0%	1 - 2	Measured ...	52.4	573.4
Q_Chest	100.0%	1 - 2	Measured ...	35.2	446.6
Q_SpineThoracic2	100.0%	1 - 2	Measured ...	37.7	586.5
Q_SpineThoracic12	100.0%	1 - 2	Measured ...	35.3	621.2
Q_LShoulderTop	100.0%	1 - 2	Measured ...	173.4	520.3
Q_LArm	100.0%	1 - 2	Measured ...	277.1	569.7
Q_LElbowOut	100.0%	1 - 2	Measured ...	404.2	566.7
Q_LElbowIn	100.0%	1 - 2	Measured ...	364.3	506.2
Q_LWristIn	100.0%	1 - 2	Measured ...	591.2	554.3
Q_LWristOut	100.0%	1 - 2	Measured ...	547.5	572.6
Q_LHand2	100.0%	1 - 2	Measured ...	654.2	562.2
Q_LHand5	100.0%	1 - 2	Measured ...	607.1	557.7
Q_RShoulderTop	100.0%	1 - 2	Measured ...	-103.7	538.9
Q_RArm	100.0%	1 - 2	Measured ...	-224.4	614.6
Q_RElbowOut	100.0%	1 - 2	Measured ...	-317.1	621.7
Q_RElbowIn	100.0%	1 - 2	Measured ...	-297.8	554.4
Q_RWristIn	100.0%	1 - 2	Measured ...	-514.4	629.5
Q_RWristOut	100.0%	1 - 2	Measured ...	-486.1	665.7
Q_RHand2	100.0%	1 - 2	Measured ...	-583.0	669.8
Q_RHand5	100.0%	1 - 2	Measured ...	-540.1	656.5
Q_WaistLFront	100.0%	1 - 2	Measured ...	117.9	419.6
Q_WaistL	100.0%	1 - 2	Measured ...	172.7	511.5
Q_LTroch	100.0%	1 - 2	Measured ...	192.4	506.2
Q_WaistBack	100.0%	1 - 2	Measured ...	27.1	621.1
Q_WaistR	100.0%	1 - 2	Measured ...	-130.6	506.7
Q_WaistRFront	100.0%	1 - 2	Measured ...	-94.3	433.0
Q_RTroch	100.0%	1 - 2	Measured ...	-155.9	534.7
Q_LThighFrontLow	100.0%	1 - 2	Measured	141.3	433.2
Q_LKneeOut	100.0%	1 - 2	Measured	211.7	510.3
Q_LKneeIn	100.0%	1 - 2	Measured	104.5	518.8
Q_LShinFrontHigh	100.0%	1 - 2	Measured	164.6	499.3
Q_LAnkleOut	100.0%	1 - 2	Measured	213.4	588.5
Q_LAnkleIn	100.0%	1 - 2	Measured	131.4	569.1

Figure 3-44: Skeleton Solver

3.10.2 Rigid Body and Euler Angles

Unlike our human MoCap application, the QTM software also supports tracking of rigid bodies, defined as bodies where the markers remain fixed relative to each other. Tracking rigid bodies [26] with six degrees of freedom (6DoF) [28] records the object's position as it moves both translationally (left/right, up/down, and forward/backward) and rotationally along its axes (yaw, pitch, and roll). These Euler angels describe the orientation of a rigid body with respect to a fixed coordinate system, as shown in Figure 3-45.

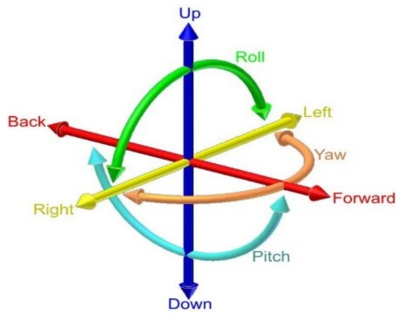


Figure 3-45: Euler Angles

3.11 Conclusion

The steps of manual cleaning demonstrated in this article thoroughly illustrate the challenges and time-consuming nature of manually cleaning MoCap data, particularly when dealing with large datasets that contain numerous frames, markers, noise, and complex actions. The reappearing characteristics of passive markers due to occlusion introduce additional complications, as they increase the number of points within a MoCap file. Furthermore, the potential short lifespan of these markers can lead them to be mistaken for noise movements. These challenges are intensified in underwater environments, where decreased visibility leads to an increased frequency of occlusion. During the manual cleaning process, it is necessary to frequently change views to accurately distinguish between markers. This is crucial because, for instance, markers may appear overlapped from one angle while being spaced apart in another view. Even with automatic marker identification tools such as the Qualisys AIM model found in commercial software—which aim to facilitate this process—manual intervention remains essential. This intervention requires cleaned and labeled training data. These challenges render manual cleaning of MoCap data both tedious and time-consuming. This situation motivates the exploration of semi-automatic or fully automatic approaches to streamline the process.

References

[1] G. B. Guerra-filho, “Optical motion capture: Theory and implementation,” *J. Theor. Appl. Informatics*, vol. 12, pp. 61--89, 2005, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.7248

[2] J. Yang, T. Li, Z. Chen, and X. Li, “Research on the Method of Underwater Swimming Motion Capture,” *J. Phys. Conf. Ser.*, vol. 1982, no. 1, pp. 1–4, 2021, doi: 10.1088/1742-6596/1982/1/012075.

[3] Qualisys, “Qualisys,” *Qualisys*. https://www.qualisys.com/ (accessed Dec. 10, 2023).

[4] Vicon, “Vicon,” *Vicon*. https://www.vicon.com/ (accessed Dec. 10, 2023).

[5] A. F. Panaite, S. Rosca, and R. Sibişanu, “Pose and motion capture technologies,” *MATEC Web Conf.*, vol. 342, p. 05004, 2021, doi: 10.1051/matecconf/202134205004.

[6] Qualisys, “Qualisys Track Manager,” *Qualisys*, 2011. https://www.qualisys.com/software/qualisys-track-manager/ (accessed Dec. 10, 2023).

[7] Qualisys, *Qualisys Track Manager User Manual*, 2022.1. 2022. [Online]. Available: https://cdn-content.qualisys.com/2022/07/QTM-user-manual.pdf

[8] Qualisys, “Compare our motion capture cameras.” https://www.qualisys.com/cameras/ (accessed Jul. 24, 2024).

[9] Qualisys, “Cameras for underwater motion capture,” *Qualisys*. https://www.qualisys.com/cameras/underwater/ (accessed Jul. 15, 2024).

[10] Qualisys, “Swimming,” *Qualisys*. https://www.qualisys.com/life-sciences/swimming/ (accessed Nov. 27, 2023).

[11] Qualisys, “Qualisys launches smallest-ever underwater mocap solution,” 2019. https://press.qualisys.com/posts/pressreleases/qualisys-launches-smallest-ever-underwater-mo (accessed Jul. 23, 2024).

[12] S. L. Raghu, R. T. Conners, C. kwon Kang, D. B. Landrum, and P. N. Whitehead, “Kinematic analysis of gait in an underwater treadmill using land-based Vicon T 40s motion capture cameras arranged externally,” *J. Biomech.*, vol. 124, no. June, p. 110553, 2021, doi: 10.1016/j.jbiomech.2021.110553.

[13] S. Takahashi and S. Kuriyama, “Animations of Real Swimming via Motion Reconstruction,” 2011.

[14] S. Veiga, J. Lorenzo, A. Trinidad, R. Pla, A. Fallas-Campos, and A. de la Rubia, “Kinematic Analysis of the Underwater Undulatory Swimming Cycle: A Systematic and Synthetic Review,” *Int. J. Environ. Res. Public Health*, vol. 19, no. 19, 2022, doi: 10.3390/ijerph191912196.

[15] Biomechanical-toolkit.github.io, “Mokka,” *Biomechanical-toolkit.github.io*. https://biomechanical-toolkit.github.io/mokka/ (accessed Jan. 06, 2024).

[16] SimTK, “OpenSim,” *SimTK*. https://simtk.org/frs/index.php?group_id=91 (accessed Jan. 05, 2024).

[17] Qualisys, “Using AIM models.” https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/using_aim_models/using_aim_models.htm (accessed Aug. 01, 2024).

[18] Qualisys, “How to create an AIM model.” https://www.qualisys.com/video-

tutorials/how-to-generate-an-aim-model/ (accessed Aug. 01, 2024).

[19] Qualisys, "Recording an AIM trial." https://docs.qualisys.com/getting-started/content/11_aim_series/11a_how_to_create_an_aim_model/recording_an_aim_trial.htm?Highlight=T pose (accessed Aug. 01, 2024).

[20] Qualisys, "Using the skeleton solver." https://docs.qualisys.com/getting-started/content/getting_started/getting_started_with_animation/using_the_skeleton_solver.htm (accessed Aug. 01, 2024).

[21] Qualisys, "How to use the skeleton solver for animation – Part 1." https://www.qualisys.com/video-tutorials/how-to-set-up-the-skeleton-solver/ (accessed Aug. 01, 2024).

[22] Qualisys, "How to use the skeleton solver for animation - Part 2." https://www.qualisys.com/my/qacademy/#!/tutorials/how-to-use-the-skeleton-solver-for-animation-part-2 (accessed Aug. 01, 2024).

[23] Qualisys, "Setting up an actor's skeleton." https://docs.qualisys.com/getting-started/content/36_skeleton_solver_series/36a_how_to_use_the_skeleton_solver_for_animation/setting_up_an_actors_skeleton.htm?Highlight=calibrating the skeleton (accessed Aug. 01, 2024).

[24] Qualisys, "Calibrating the skeleton." https://docs.qualisys.com/getting-started/content/36_skeleton_solver_series/36a_how_to_use_the_skeleton_solver_for_animation/calibrating_the_skeleton.htm (accessed Aug. 01, 2024).

[25] Qualisys, "Skeleton data in QTM." https://docs.qualisys.com/getting-started/content/36_skeleton_solver_series/36a_how_to_use_the_skeleton_solver_for_animation/skeleton_data_in_qtm.htm (accessed Aug. 01, 2024).

[26] Qualisys, “Rigid body overview.” https://docs.qualisys.com/getting-started/content/17_rigid_body_series/17a_how_to_track_rigid_bodies/rigid_body_overview.htm#:~:text=Your Qualisys system allows you,%2C pitch%2C and roll). (accessed Aug. 01, 2024).

[27] Qualisys, “Learn about 6DOF.” https://www.qualisys.com/webinars/learn-about-6dof/ (accessed Aug. 01, 2024).

[28] Qualisys, “Viewing the 6DOF data.” https://docs.qualisys.com/getting-started/content/17_rigid_body_series/17a_how_to_track_rigid_bodies/viewing_the_6dof_data.htm?Highlight=angle (accessed Aug. 01, 2024).

[29] Qualisys, “Marine vessels & structures.” https://www.qualisys.com/engineering/marine-vessels-and-structures/ (accessed Jul. 26, 2024).

[30] O. Tortorici, C. Péraud, C. Anthierens, and V. Hugel, “Automated Deployment of an Underwater Tether Equipped with a Compliant Buoy–Ballast System for Remotely Operated Vehicle Intervention,” *J. Mar. Sci. Eng.*, vol. 12, no. 2, 2024, doi: 10.3390/jmse12020279.

[31] N. Bauschmann, D. A. Duecker, T. L. Alff, R. C. Hochdahl, and R. Seifried, “Towards Full Actuation: Reconfigurable Micro Underwater Robots,” *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 6192–6199, 2023, doi: 10.1109/IROS55552.2023.10341621.

[32] K. J. Nankervis *et al.*, “Effect of speed and water depth on limb and back kinematics in Thoroughbred horses walking on a water treadmill,” *Vet. J.*, vol. 300–302, p. 106033, 2023, doi: 10.1016/j.tvjl.2023.106033.

[33] J. P. Jhan, J. Y. Rau, and C. M. Chou, “Underwater 3D rigid object tracking and 6-DOF estimation: A case study of giant steel pipe scale model underwater installation,” *Remote Sens.*, vol. 12, no. 16, pp. 1–14, 2020, doi: 10.3390/RS12162600.

[34] S. Lack, E. Rentzow, and T. Jeinsch, “Control of a small Underwater Vehicle Manipulator System - a highly automated Pick and Place Experiment *,” no. February, 2024.

[35] C. Long, “Comparing Lower-Limb Muscle Activity During Gait Performed in Water Versus on Land by,” UTAH STATE UNIVERSITY, 2023.

[36] B. Worley, “Acute Effects of Multi-Joint Eccentric Exercise on Lower Extremity Muscle Activation Measured During Land and Water Walking,” UTAH STATE UNIVERSITY, 2024.

[37] C. Long *et al.*, “Lower Limb Muscle Activation in Young Adults Walking in Water and on Land,” *Appl. Sci.*, vol. 14, no. 12, p. 5044, 2024, doi: 10.3390/app14125044.

[38] Qualisys, “Products tagged ‘oqus.’” https://www.qualisys.com/product-tag/oqus/ (accessed Jul. 23, 2024).

[39] Qualisys, “5+, 6+ and 7+ series.” https://www.qualisys.com/cameras/5-6-7/ (accessed Jul. 23, 2024).

[40] Qualisys, “Arqus.” https://www.qualisys.com/cameras/arqus/ (accessed Jul. 23, 2024).

[41] J. Lauer, A. H. Rouard, and J. P. Vilas-Boas, “Upper limb joint forces and moments during underwater cyclical movements,” *J. Biomech.*, vol. 49, no. 14, pp. 3355–3361, 2016, doi: 10.1016/j.jbiomech.2016.08.027.

[42] B. H. Olstad, "A new approach for identifying phases of the breaststroke wave kick and calculation of feet slip using 3D automatic motion tracking," *BMS Proc.*, pp. 195–199, 2014.

[43] B. H. Olstad, J. R. Vaz, C. Zinner, J. M. H. Cabri, and P. L. Kjendlie, "Muscle coordination, activation and kinematics of world-class and elite breaststroke swimmers during submaximal and maximal efforts," *J. Sports Sci.*, vol. 35, no. 11, pp. 1107–1117, 2017, doi: 10.1080/02640414.2016.1211306.

[44] B. H. Olstad, C. Zinner, J. R. Vaz, J. M. H. Cabri, and P. L. Kjendlie, "Muscle activation in world-champion, world-class, and national breaststroke swimmers," *Int. J. Sports Physiol. Perform.*, vol. 12, no. 4, pp. 538–547, 2017, doi: 10.1123/ijspp.2015-0703.

[45] J. Ribeiro *et al.*, "Biomechanics, energetics and coordination during extreme swimming intensity: effect of performance level," *J. Sports Sci.*, vol. 35, no. 16, pp. 1614–1621, 2017, doi: 10.1080/02640414.2016.1227079.

[46] K. Abdul Jabbar, S. Kudo, K. W. Goh, and M. R. Goh, "Comparison in three dimensional gait kinematics between young and older adults on land and in shallow water," *Gait Posture*, vol. 57, no. July 2016, pp. 102–108, 2017, doi: 10.1016/j.gaitpost.2017.05.021.

[47] S. Washino, D. L. Mayfield, G. A. Lichtwark, H. Mankyu, and Y. Yoshitake, "Swimming performance is reduced by reflective markers intended for the analysis of swimming kinematics," *J. Biomech.*, vol. 91, pp. 109–113, 2019, doi: 10.1016/j.jbiomech.2019.05.017.

[48] P. Chainok *et al.*, "Biomechanical Features of Backstroke to Breaststroke Transition

Techniques in Age-Group Swimmers," *Front. Sport. Act. Living*, vol. 4, no. March, pp. 1–11, 2022, doi: 10.3389/fspor.2022.802967.

[49] T. Tanaka, S. Hashizume, T. Sato, and T. Isaka, "Competitive-Level Differences in Trunk and Foot Kinematics of Underwater Undulatory Swimming," *Int. J. Environ. Res. Public Health*, vol. 19, no. 7, 2022, doi: 10.3390/ijerph19073998.

[50] M. Nakashima, R. Kanie, T. Shimana, Y. Matsuda, and Y. Kubo, "Development of a comprehensive method for musculoskeletal simulation in swimming using motion capture data," *Proc. Inst. Mech. Eng. Part P J. Sport. Eng. Technol.*, vol. 237, no. 2, pp. 85–95, 2023, doi: 10.1177/1754337119838395.

[51] Qualisys, "Qualisys Sports Marker Set," *Qualisys*. https://cdn-content.qualisys.com/2022/07/Sports-Marker-Set.pdf (accessed Dec. 05, 2023).

[52] Qualisys, "High-quality passive & active mocap markers," *Qualisys*. https://www.qualisys.com/accessories/markers/ (accessed Jan. 11, 2024).

[53] Qualisys, "Super-spherical mocap markers." https://www.qualisys.com/accessories/markers/super-spherical-markers/ (accessed Jul. 28, 2024).

[54] Qualisys, "Super-spherical underwater markers." https://www.qualisys.com/accessories/markers/underwater-markers-extra-durable/ (accessed Jul. 28, 2024).

[55] Qualisys, "Motion capture for underwater measurements." https://www.qualisys.com/cameras/underwater/#tech-specs (accessed Jul. 24, 2024).

[56] Qualisys, "Volume of interest," *Qualisys*. https://docs.qualisys.com/getting-

started/content/getting started/setting up your system/planning your volume/volume of interest.htm?Highlight=volume (accessed Jan. 11, 2024).

[57] Unity, “Unity.” https://unity.com/ (accessed Aug. 01, 2024).

[58] Unrealengine, “Unreal Engine.” https://www.unrealengine.com/en-US (accessed Aug. 01, 2024).

[59] Reallusion, “iClone.” https://www.reallusion.com/iclone/ (accessed Aug. 01, 2024).

[60] Autodesk, “What is MotionBuilder?” https://www.autodesk.com/products/motionbuilder/free-trial (accessed Aug. 01, 2024).

[61] Autodesk, “What is Maya?” https://www.autodesk.com/ca-en/products/maya/overview?term=1-YEAR&tab=subscription (accessed Aug. 01, 2024).

[62] Wikipedia, “LabVIEW.” https://en.wikipedia.org/wiki/LabVIEW (accessed Aug. 01, 2024).

[63] Qualisys, “Assembling the calibration kit.” https://docs.qualisys.com/getting-started/content/8_calibration_series/8a_how_to_calibrate/assembling_the_calibration_kit.htm (accessed Jul. 28, 2024).

[64] Qualisys, “Carbon fiber calibration kit.” https://www.qualisys.com/accessories/calibration-kits/carbon-fibre-calibration-kit/ (accessed Jul. 28, 2024).

[65] Qualisys, “Calibrating your system,” *Qualisys*. https://docs.qualisys.com/getting-started/content/getting_started/running_your_qualisys_system/calibrating_your_system/calibrating_your_system.htm (accessed Jan. 06, 2024).

[66] Qualisys, “How to calibrate.” https://www.qualisys.com/video-tutorials/how-to-calibrate/ (accessed Jul. 28, 2024).

[67] Qualisys, “Exporting and streaming data.” https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/exporting_and_streaming_data/exporting_and_streaming_data.htm?Highlight=export (accessed Aug. 01, 2024).

[68] B. Motion, “The C3D File Format A Technical User Guide,” p. 134, 2021.

[69] Wikipedia, “Tab-separated values (TSV).” https://en.wikipedia.org/wiki/Tab-separated_values (accessed Aug. 01, 2024).

[70] Wikipedia, “Audio Video Interleave (AVI).” https://en.wikipedia.org/wiki/Audio_Video_Interleave (accessed Aug. 01, 2024).

[71] Wikipedia, “FBX (Filmbox).” https://en.wikipedia.org/wiki/FBX#:~:text=FBX (Filmbox) is a proprietary,series of video game middleware. (accessed Aug. 01, 2024).

[72] Qualisys, “Exporting files to C3D.” https://docs.qualisys.com/getting-started/content/13_how_to_visualize_data_in_visual3d/exporting_files_to_c3d.htm?Highlight=export (accessed Aug. 01, 2024).

[73] Optitrack, “optiTrack.” https://optitrack.com/ (accessed Aug. 02, 2024).

[74] Github, “ezc3d.” https://github.com/pyomeca/ezc3d

[75] Biomechanical-toolkit.github.io, “BTK is evolving to become a bigger project!,” *Biomechanical-toolkit.github.io*. https://biomechanical-toolkit.github.io/news/2016/08/29/btk-evolution/ (accessed Jan. 06, 2024).

[76] S. A *et al.*, “OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement Ajay,” *PLOS Comput. Biol.*, vol. 14,

no. 7, 2018, doi: 10.3758/BF03326891.

[77] Qualisys, “What is a trajectory.” https://www.qualisys.com/my/qacademy/#!/tutorials/what-is-a-trajectory (accessed Jun. 30, 2024).

[78] A. L. Clouthier, G. B. Ross, M. P. Mavor, I. Coll, A. Boyle, and R. B. Graham, “Development and Validation of a Deep Learning Algorithm and Open-Source Platform for the Automatic Labelling of Motion Capture Markers,” *IEEE Access*, vol. 9, pp. 36444–36454, 2021, doi: 10.1109/ACCESS.2021.3062748.

[79] N. Ghorbani and M. J. Black, “SOMA: Solving Optical Marker-Based MoCap Automatically,” *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 11097–11106, 2021, doi: 10.1109/ICCV48922.2021.01093.

[80] S. Alexanderson, C. O’Sullivan, and J. Beskow, “Real-time labeling of non-rigid motion capture marker sets,” *Comput. Graph.*, vol. 69, pp. 59–67, 2017, doi: 10.1016/j.cag.2017.10.001.

[81] M. Kitagawa and B. Windsor, *MoCap for Artists Workflow and Techniques for Motion Capture*, no. 0. Elsevier Inc, 2008.

[82] Qualisys, “Swapping trajectories.” https://docs.qualisys.com/getting-started/content/26_labeling_series/26b_how_to_re-label__swap__and_split_trajectory_parts/swapping_trajectories.htm?Highlight=swap (accessed Aug. 11, 2024).

[83] Qualisys, “Filling gaps in your data.” https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/filling_gaps_in_your_data/filling_gaps_in_your_data.htm (accessed Aug. 11, 2024).

[84] Qualisys, "Identifying gaps." https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/filling_gaps_in_your_data/identifying_gaps.htm?Highlight=fill level (accessed Jul. 03, 2024).

[85] Qualisys, "Fill types." https://docs.qualisys.com/getting-started/content/37_trajectory_editor_series/37b_how_to_use_the_trajectory_editor_-_gap-filling/fill_types.htm (accessed Aug. 11, 2024).

[86] Qualisys, "Smoothing your data," *Qualisys*. https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/smoothing_your_data/smoothing_your_data.htm (accessed Jan. 12, 2024).

[87] Qualisys, "Smoothing types," *Qualisys*. https://docs.qualisys.com/getting-started/content/37_trajectory_editor_series/37c_how_to_use_the_trajectory_editor_-_smoothing/smoothing_types.htm?Highlight=smoothing types (accessed Aug. 11, 2024).

[88] Qualisys, "How to train an AIM model." https://www.qualisys.com/my/qacademy/#!/tutorials/how-to-train-an-aim-model (accessed Aug. 12, 2024).

[89] Qualisys, "How to apply AIM models." https://www.qualisys.com/my/qacademy/#!/tutorials/how-to-apply-aim-models (accessed Aug. 12, 2024).

4. Semi-supervised Geometry-based Cleaning and Labeling of Sparse Freestyle Underwater Optical MoCap Data

Abstract

This paper presents an interactive labeling method for underwater optical motion capture (MoCap) markers to simplify the tedious manual data cleaning process. Using a sparse freestyle MoCap dataset, containing only 21 passive physical markers, poses challenges when applying many traditional denoising solutions. Hence, a novel method is employed to remove extraneous markers based on the norm differences, and outliers are eliminated by detecting abnormalities in velocity, acceleration, and jerk profiles. The geometry-based algorithm identifies valid markers amid remaining noise and ghost markers by assuming a rigid body and incorporating user-defined tolerances to demonstrate deviations from rigidity concerning the angles and distances within the marker set. It first uses a Principal Component Analysis-based method to detect pelvis points, which are then used to identify other body part markers. This process includes recognizing corresponding passive reappearing markers and using a body side detection method to assign unique labels to each marker. A method is used to reconstruct dropped markers that were not captured in any frames. An evaluation was carried out visually, demonstrating a 100% detection accuracy for valid markers. This algorithm effectively streamlines the time-consuming manual cleaning process of MoCap data. In the future, we will explore the use of additional ghost markers and automatic side detection in the pelvis detection procedure.

4.1 Introduction

Marker-based optical motion capture (OMC) [1] systems are advanced technologies used to track the three-dimensional (3D) motion of markers attached to a subject's body. These systems are employed for a wide range of applications in diverse environments [2]. Qualisys underwater Miqus motion capture (MoCap) cameras are the first commercially available optical MoCap cameras for underwater use [3] for various applications such as in-water rehabilitation using underwater treadmills, underwater animation, and swimming performance analysis, by using Qualisys Track Manager (QTM) software [4]. However, MoCap systems are susceptible to errors stemming from various factors, including inadequate calibration, noisy environments, and occlusion [5]. Underwater settings present additional challenges compared to other environments due to factors such as surface reflections and reduced visibility [6], which contribute to increased noise levels and further occlusions. Therefore, manual cleaning [7] is an essential part of MoCap data processing, involving tasks such as denoising, recovering data, and labeling MoCap markers.

The challenge of manually cleaning underwater MoCap data is heightened by the occlusion effects associated with passive marker systems. In these systems, markers are commonly given an ID when first detected, but then given a random new ID when they reappear after tracking is lost (e.g., being occluded), leading to the creation of partial trajectories that are not connected instead of complete trajectories [8]. Therefore, post-processing is necessary to identify "reappearing" markers and merge short segments linked to a specific physical marker, creating a complete trajectory with gaps. Additionally, body side detection is required to differentiate corresponding markers on the left and right sides of the body. Meanwhile, active marker systems may experience gaps in trajectories due to occlusions which can be filled with gap-filling approaches.

The number of physical markers significantly impacts manual cleaning for underwater MoCap data [9], which experiences more occlusion and noise than other environments. Increasing the number of markers improves accuracy and reduces occlusion issues by providing redundant tracking points, enhancing post-processing capabilities. However, this comes with drawbacks such as longer setup times, discomfort for subjects, complex data processing, high computational loads, and potential marker interference from closely spaced markers that can lead to tracking inaccuracies or added noise. On the other hand, using fewer (and as a result, more sparse) markers with larger spacing can complicate statistical outlier detection, particularly when there is a higher ratio of outliers compared to valid markers.

Freestyle [10] underwater movement complicates manual cleaning due to the unique properties of water, which allow for intricate rotations, direction changes, and floating that are difficult to replicate on land. This complexity makes it hard to identify and predict marker movements specifically during gaps due to their unpredictable nature.

To tackle the challenges of manual cleaning in MoCap, researchers have turned to machine learning and deep learning techniques to automate this labor-intensive process [11]. These methods still necessitate training and ground truth datasets that have undergone manual cleaning and labeling [12]. While some methods utilize simulated data to eliminate the necessity for manual labeling to generate training sets, they still require manual labeling for MoCap datasets that exhibit significant variations in marker placement compared to their simulation marker set [8]. Moreover, these methods may not perfectly replicate real-world variations in human motion. Additionally, commercial software tools like QTM's Automatic Identification of Markers (AIM) [13] can speed up manual cleaning, but they require purchase and still need manual input. Free software such as Mokka [14] offers measurements between markers, yet it is difficult to manage all angles and distances simultaneously due to occlusions and the large number of markers and frames. Manual cleaning also often necessitates frequent view changes to accurately assess distances and movements.

To our knowledge, there are currently no publicly available OMC datasets for underwater freestyle actions. Additionally, datasets featuring surface-level swimming actions (e.g., backstroke, breaststroke, butterfly, and front crawl) [15] are scarce and limited in size, and these actions do not fully represent underwater or freestyle activities. Therefore, there is a need to create comprehensive MoCap datasets specifically for underwater freestyle actions to tackle the unique challenges posed by underwater conditions.

This paper proposes a semi-supervised geometry-based cleaning and labeling method implemented in MATLAB to streamline the challenging and tedious process of manual cleaning MoCap data. Our sparse marker set included 21 passive physical markers, whereas Qualisys recommends a minimum of 41 markers for human tracking [16], and universal datasets like AMASS contain marker sets ranging from 37 to 91 markers [17]. This decision was made due to the significantly high time required for underwater setup and with consideration for the comfort of the swimmer. Additionally, our research aimed to explore the challenges associated with sparse MoCap datasets. For example, the relatively large gap between our markers on the hip and knee made it challenging to denoise using common algorithms. Hence, a novel statistical denoising approach leverages the difference of norms to eliminate extraneous markers. Additionally, outliers are removed through repetitive checks to identify abnormalities in velocity, acceleration, and jerk profiles.

A geometry-based algorithm detects valid markers among remaining noise and ghost markers by assuming rigid body conditions. It compares angles and distances in datasets to those in the marker set, applying user-defined tolerances for rigidity deviations. First, pelvis points are detected by incorporating Principal Component Analysis (PCA) [18]. These detected pelvis points are subsequently utilized to identify markers for other body parts. This process involves identifying corresponding passive reappearing markers and employing a body side detection method to locate short trajectories associated with each physical marker. Subsequently, these short trajectories are merged, and the gaps between them are filled using linear interpolation. This results in a unique label for each physical marker along with its corresponding complete trajectory across all frames. Moreover, a trajectory is created using the triangulation method for a dropped marker that was not captured in any frames during data collection.

The visual evaluation demonstrated a 100% accurate detection of 21 valid markers on our 10 C3D files [19], [20], captured at 100 Hz, with totally 7792 frames of underwater freestyle MoCap data captured using 7 Qualisys Miqus M5U Underwater MoCap cameras. In the future, we explore additional ghost markers and automatic side detection.

The paper is structured as follows: Section 4.2 details explanation of MoCap cleaning and labeling problems that we aim to address in this article. Section 4.3 describes the proposed method along with its rationale and design considerations. Section 4.4 presents the experimental results and discussion, and Section 4.5 summarizes the article.

4.2 MoCap Cleaning and Labeling Problem

MoCap data captures the 3D positions of markers at a specific frame rate (i.e., capturing frequency). The trajectory [21] of a marker shows its movement path over time, represented by its x, y, and z coordinates at any moment. The “fill level” [22] indicates how visible a point is, calculated by dividing the number of frames where the point was tracked by the total frames captured during the MoCap session, expressed as a percentage. The total points in a MoCap file may differ from the number of physical markers due to factors like noise, dropped markers, and occlusion effects on passive markers. The following sections elaborate on these two issues: noise and ”reappearing” passive markers.

4.2.1 Noise

Each frame of a MoCap file, as shown in Figure 4-1, may contain valid markers (black) and various noise types: extraneous (blue), outlier (green), overlapped (yellow), ghost (red), and missing marker due to occlusion or dropped markers.

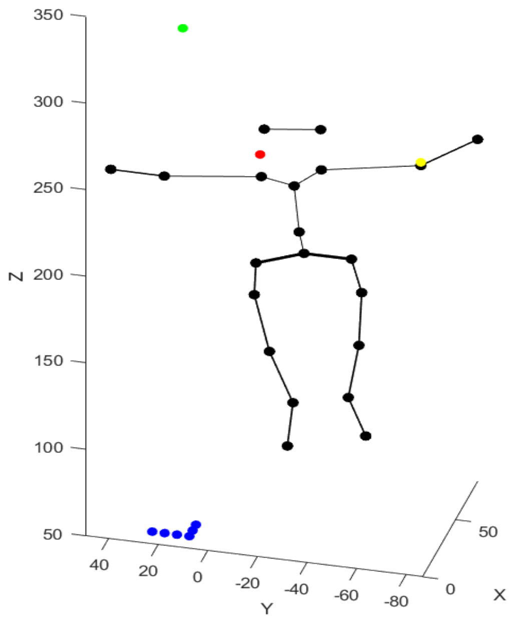


Figure 4-1: MoCap Noise Types; Outlier (green), Ghost (red), Extraneous (blue), Overlapped (yellow)

Extraneous markers are real markers from other objects, while outliers are inaccurate measurements that deviate significantly from expected values. Overlapping markers occur when points are closer than the marker size or measurement accuracy. Ghost markers are virtual markers near valid ones. Occlusion [8] typically causes gaps in active marker trajectories and leads to “reappearing” markers in passive systems.

4.2.2 Passive Marker Characteristics: Reappearing Markers

In MoCap data, a physical marker is assigned a random ID (e.g., ID_1) upon first tracking in a frame. Passive markers typically receive new IDs after occlusion, while active markers keep the same ID. Thus, an active marker has a unique ID and a complete trajectory with possible gaps due to occlusion. Conversely, a passive marker may be linked to multiple IDs, each corresponding to a short trajectory segment. Therefore, post-processing is needed to merge these segments into a complete trajectory with gaps and assign a single ID.

$$\begin{bmatrix} x_1 & NaN & . & . & . & x_m \\ y_1 & NaN & . & . & . & y_m \\ z_1 & NaN & . & . & . & z_m \end{bmatrix} \begin{bmatrix} NaN & NaN & . & . & . & x_m \\ NaN & NaN & . & . & . & y_m \\ NaN & NaN & . & . & . & z_m \end{bmatrix} \begin{bmatrix} x_1 & NaN & . & . & . & x_m \\ y_1 & NaN & . & . & . & y_m \\ z_1 & NaN & . & . & . & z_m \end{bmatrix} \dots \begin{bmatrix} x_1 & NaN & . & . & . & x_m \\ y_1 & NaN & . & . & . & y_m \\ z_1 & NaN & . & . & . & z_m \end{bmatrix}$$

Figure 4-2: MoCap C3D Data representation of an active marker

$$\begin{bmatrix} x_1 & NaN & . & . & . & x_n \\ y_1 & NaN & . & . & . & y_n \\ z_1 & NaN & . & . & . & z_n \end{bmatrix} \begin{bmatrix} NaN & NaN & . & . & . & x_n \\ NaN & NaN & . & . & . & y_n \\ NaN & NaN & . & . & . & z_n \end{bmatrix} \begin{bmatrix} NaN & NaN & . & . & . & x_n \\ NaN & NaN & . & . & . & y_n \\ NaN & NaN & . & . & . & z_n \end{bmatrix} \dots \begin{bmatrix} NaN & NaN & . & . & . & x_n \\ NaN & NaN & . & . & . & y_n \\ NaN & NaN & . & . & . & z_n \end{bmatrix}$$

Figure 4-3: MoCap C3D Data representation of a passive marker

Figure 4-2 and Figure 4-3 depict the contents of a MoCap file for an active and passive marker, highlighting the occlusion effect. Each matrix represents a single frame, with rows for 3D coordinates and columns for individual points. In Figure 4-2, active point 1 is occluded in frame 2, leading to its 3D coordinates being recorded as null values ([NaN, NaN, NaN]). When it reappears in frame 3, it retains ID_1 until the last frame. In contrast, in Figure 4-3, passive point 1 remains null from frame 2 onward.

An example is shown in Figure 4-4, showcasing a physical calf marker with the ID_{20} (orange) in the top left image. Its trajectory, as depicted in the bottom row image, remains visible from frame 1 to frame 253. Then, it disappears only to reappear again in frame 308 with a new ID_5 (yellow), remaining visible until the end at frame 460. Thus, post-processing is required to merge the trajectories of ID_{20} and ID_5, creating a complete trajectory from frame 1 to frame 460, with a gap between frames 253 and 308. Subsequently, label ID_{20} must be assigned to this trajectory and label ID_5 removed.

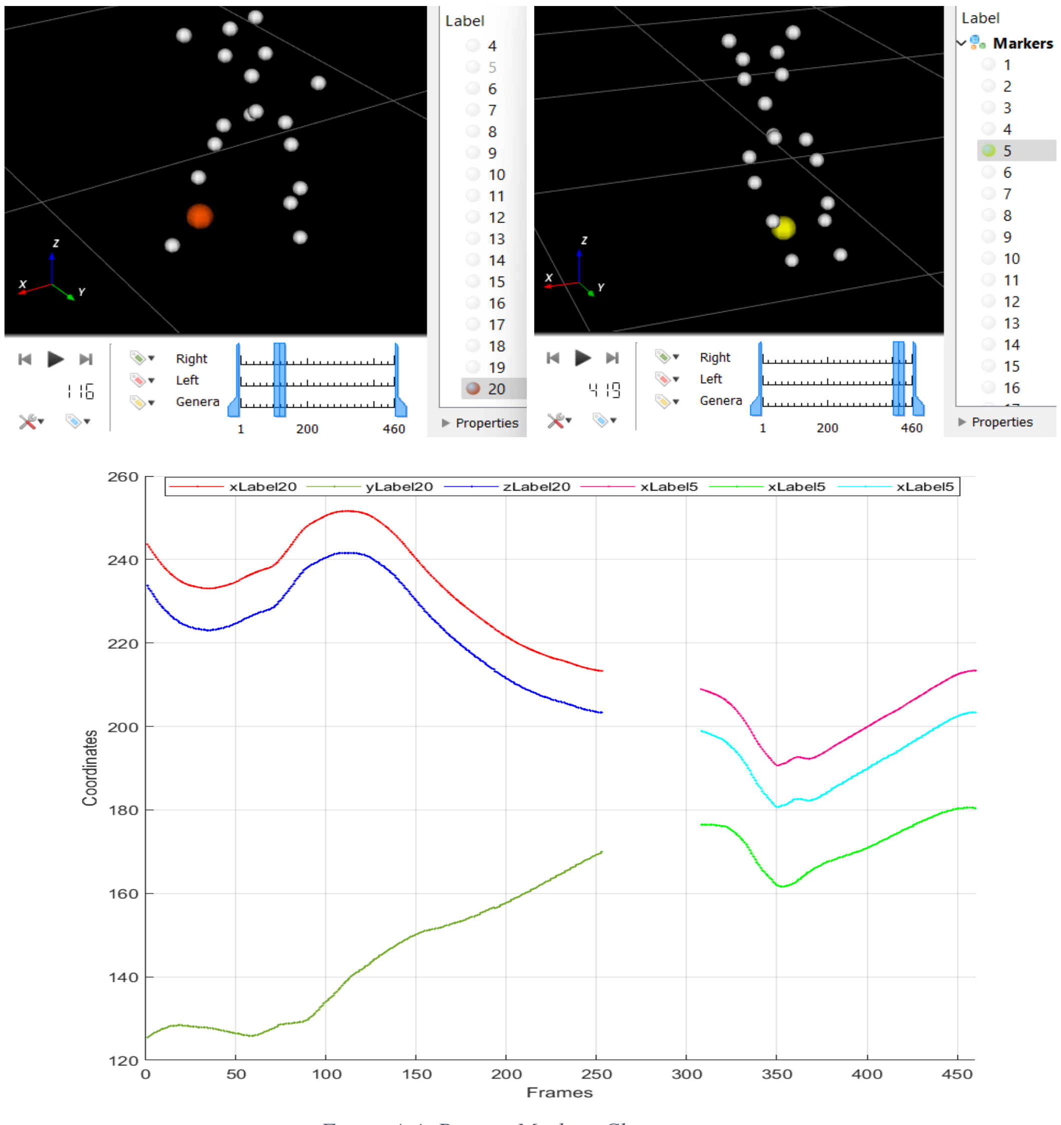


Figure 4-4: Passive Markers Characteristic

4.3 Experimental Methodology

The overview of our system is illustrated in Figure 4-5. The left flowchart outlines the sequential steps of the proposed algorithm, while the right image depicts our marker set, which comprises 21 distinct markers. The preprocessing and statistical processing phases are designed to reduce noise interference and computation load by minimizing data points. The geometry-based inlier detection method identifies valid markers using rigid body assumptions. As a result, the algorithm refines raw, noisy MoCap data into a cleaned and labeled dataset. The evaluation of the processed data was conducted visually. The subsequent sections detail the steps of our methodology.

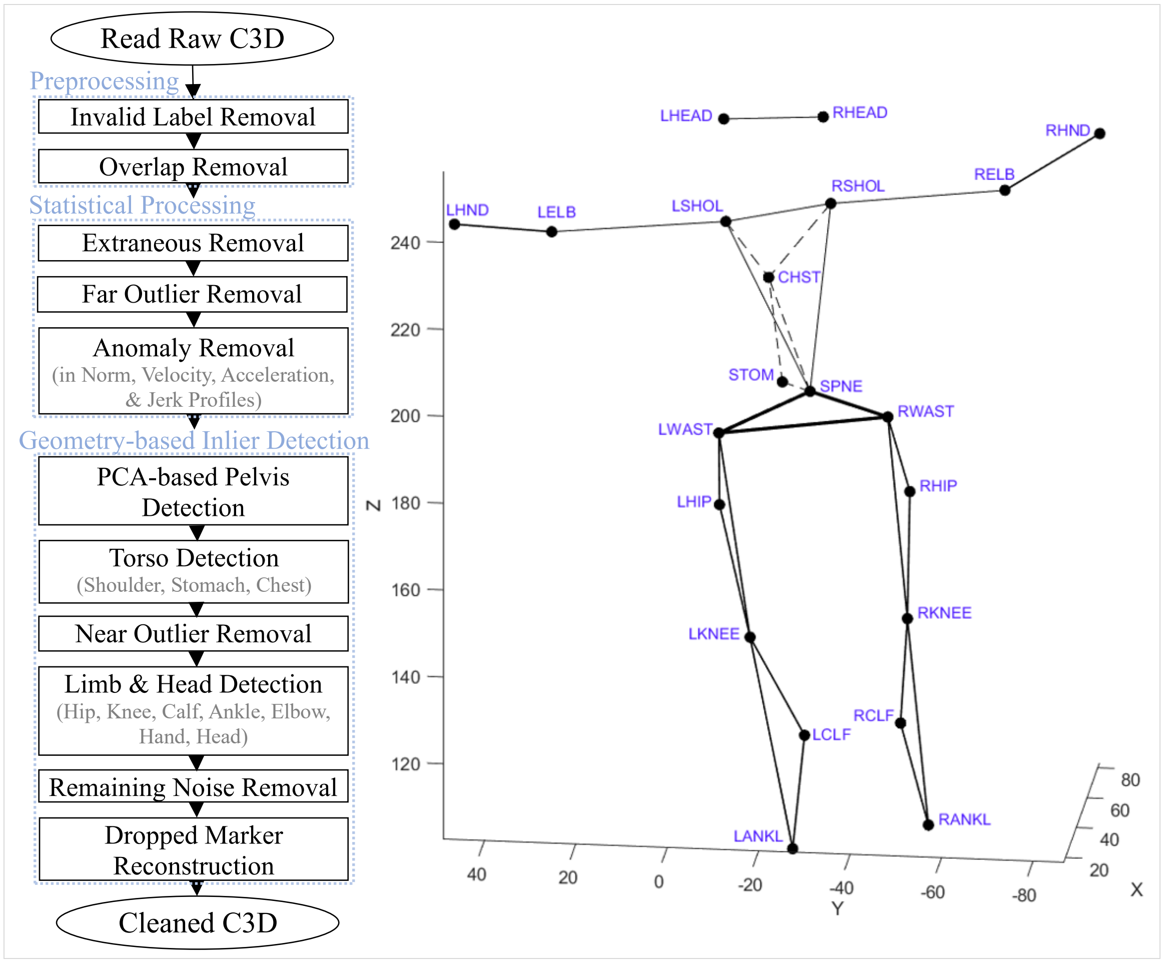


Figure 4-5: System Overview

4.3.1 Preprocessing

The following sections outline how to read C3D [19] MoCap files using MATLAB and eliminate invalid or overlapping points. The implementation was executed using MATLAB R2022b on an 11th Gen Intel i7 processor running at 2.80GHz, with 16.0 GB of RAM. To visualize C3D files, we utilize QTM [4] and Mokka [14].

4.3.1.1 Reading a C3D file using EZC3D Library in MATLAB

To process the data, we utilized the EZC3D library [23] in MATLAB to read C3D files. Algorithm 1 presents the pseudocode for extracting key information from a C3D file.

Algorithm 1 Read C3D File

```
 1:     addpath('\MATLAB\ezc3d')
 2:     $c3d \leftarrow$ ezc3dRead('$Test$.c3d')

 3: Function READC3D($c3d$)
 4:     $points \leftarrow$ c3d.data.points(:,:,:)
 5:     $//points(coordinates, points, frames)$
 6:     $num\_points \leftarrow$ c3d.header.points.size
 7:     $Labels \leftarrow$ c3d.parameters.POINT.LABELS.DATA
 8:     $num\_Labels \leftarrow$ size($Labels$, 1)
 9:     $num\_frames \leftarrow$ c3d.parameters.POINT.FRAMES.DATA
10:     $//num\_frames \leftarrow$ size($points$, 3)
11:     return $points, num\_points, Labels, num\_Labels, num\_frames$
12: End
```

4.3.1.2 Invalid Label Removal

Labels containing no information (i.e., null or NaN) across all frames were removed to decrease computational load. This issue may arise when a C3D file is exported from a larger C3D file without adequate trimming, resulting in more points than labels due to leftover points from the original file. Figure 4-2 and Figure 4-3 show an invalid point 2 (i.e., the second column of each matrix) that is null across all frames (i.e., [NaN, NaN, NaN]).

4.3.1.3 Overlap Removal

In each frame, we initially calculated the Euclidean distance between every pair of points (e.g., using the "pdist()" function in MATLAB). We defined a minimum distance criterion of 6mm, based on the minimum Qualisys marker size of 6.5 mm [24]. If any two markers were closer than 6mm, they were classified as overlapping. Figure 4-6 shows these overlapping markers in green and blue. Two markers (visible as large gray orbs above marker ID$_{19}$) visually appears to overlap; however, both are valid waist and hip markers that are close due to a treading motion resembling sitting actions, keeping them at a distance greater than 6mm.

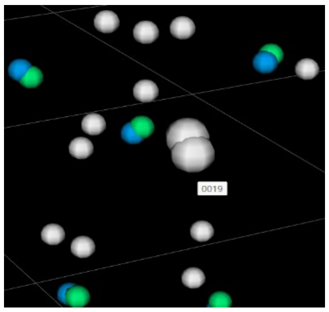


Figure 4-6: Overlapped Markers

4.3.2 Statistical Processing

Extraneous markers and far outlier removal are discussed in the following sections.

4.3.2.1 Extraneous Removal

Extraneous markers are defined as those belonging to stationary objects with a fill level of 100%. Our innovative approach for eliminating these markers utilized clustering of norm differences. We assumed the extraneous markers were far from the valid ones in all frames. If this was not the case, the markers would remain and be addressed in subsequent steps.

We calculated the Euclidean norm of each point $\boldsymbol{p} = (x, y, z)$ in every frame relative to the origin $O = (0,0,0)$ as $norm(p) = \|\boldsymbol{p}\| = \sqrt{\boldsymbol{p} \cdot \boldsymbol{p}} = \sqrt{x^2 + y^2 + z^2}$. Subsequently, we sorted these norms and their correspondence labels, then computed the difference between each norm value and the preceding norm value. To eliminate abnormalities, we applied a threshold of three standard deviations (SD) [25] from the mean (e.g., using "isoutlier()" function in MATLAB).

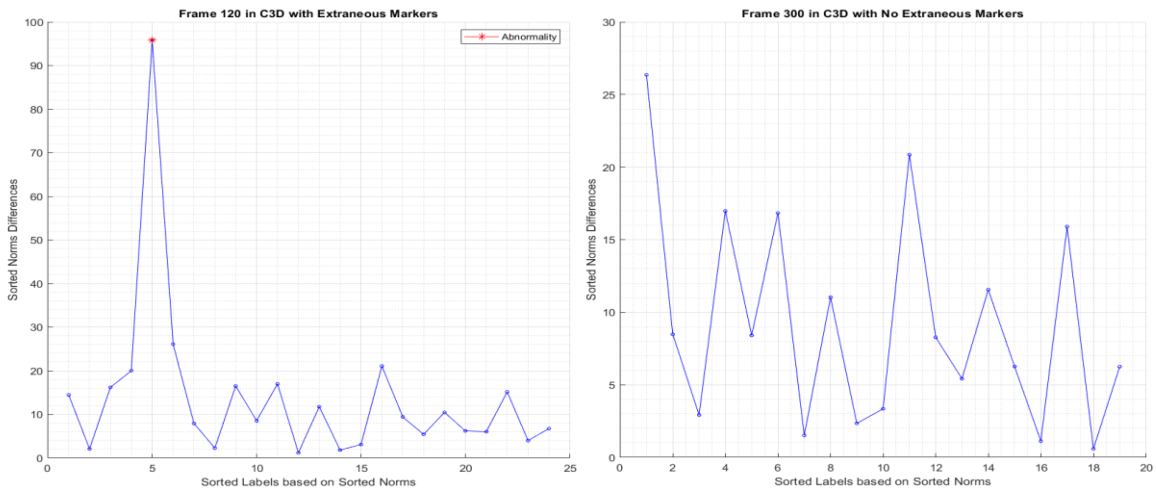


Figure 4-7: Sorted norms differences for two C3Ds with (left) and without (right) extraneous markers..

Figure 4-7 displays an example for two C3D files, with and without extraneous markers. A red star indicates an abnormality where the distance between sorted labels 5 and 6 is significant, leading to two clusters. Consequently, this C3D file contains 5 extraneous markers (sorted labels 1, 2, 3, 4, and 5).

It is essential to note that utilizing k-means clustering with our sparse dataset of 21 valid markers and significant space between markers on the hips and knees is ineffective.

4.3.2.2 Far Outlier Removal based on Anomaly in Norm Profile

Far outliers were eliminated based on abnormalities identified in the sorted norm plot for each frame, utilizing a threshold of three standard deviations (SD) from the mean. This function cannot be applied before removing extraneous markers due to the number and distribution of valid and invalid points within each frame. We used 21 sparse real markers, alongside 5 closely located extraneous markers. The presence of these extraneous markers leads to a lack of abnormal peaks in the sorted norm plot when assessed against three SD or other thresholds, as shown in the third image of the first row of Figure 4-8.

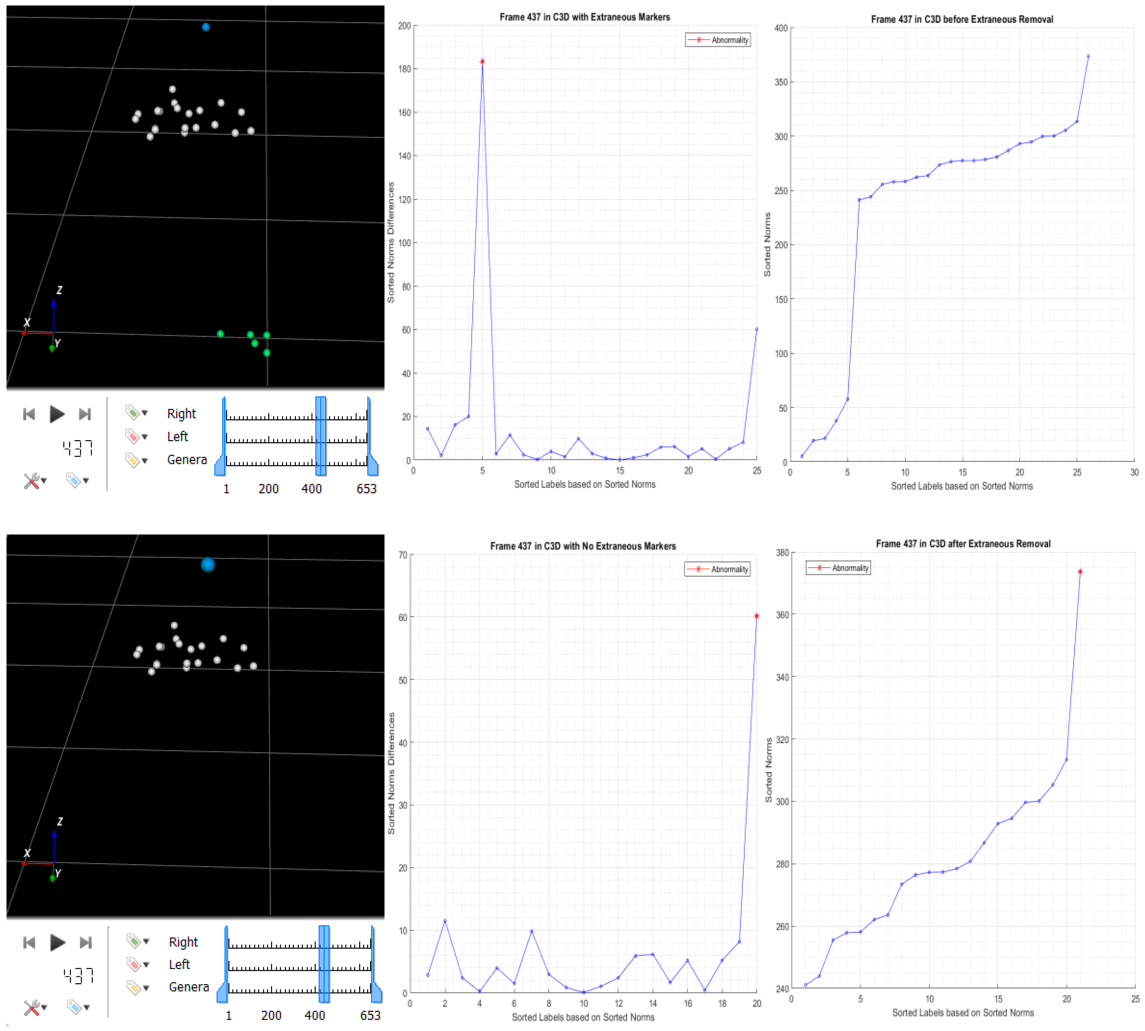


Figure 4-8: Far outlier removal; Before (top) and after (bottom) extraneous removal.

In this figure, the first row displays plots of the sorted norm difference (middle image) and sorted norm (right image) for a C3D file that includes extraneous markers, represented as green orbs in the left image. No abnormalities are detected in the norm plot; the abnormality identified in the norm differences plot is associated with the extraneous markers (i.e., the blue orb in the first image). In the second row, after removing the extraneous markers, an abnormality is observed in both the middle and right images, which correctly corresponds to a far outlier.

The inability to remove far outliers based on the distance of neighboring markers using Euclidean distance (e.g., using the "pdist()" function in MATLAB) within three SD (or other thresholds, such as median and quartiles) is primarily due to the sparsity of our dataset. Moreover, the low fill level of valid passive markers also impacts this process.

4.3.2.3 Anomaly detection in Velocity, Acceleration, and Jerk Profiles

Nearby outliers that were not removed in the previous step due to their proximity to valid markers (i.e., markers with norms within three SD) were detected based on their abnormal movement behavior. These anomalies exhibited irregularities in velocity (rate of change of position), acceleration (rate of change of velocity) (as seen in [26]), and jerk (rate of change of acceleration) profiles compared to the more consistent behaviours of valid markers, which are constrained by human biomechanical limits.

However, a major challenge may arise in removing noise from passive markers due to their low fill level, which can cause their behavior to mimic noise during their brief lifespan. This makes it difficult to distinguish between actual marker behavior and noise using above traditional methods. Moreover, valid markers can remain static throughout their short lifetime due to short segment trajectories or lack of movement in certain actions, such as the chest when a person is treading water and only moves their hands or upper body.

To resolve this issue, we eliminated movement anomalies with a fill level below 1%. These anomalies typically consist of only a few frames of detection, so even if false detections occur, their removal does not significantly affect the overall tracking of the marker's movement. Interpolation or other gap-filling methods can compensate for the removed data points. A visual inspection was performed to ensure the accuracy of this removal. Any remaining outliers with higher fill levels will be addressed in later stages of data processing.

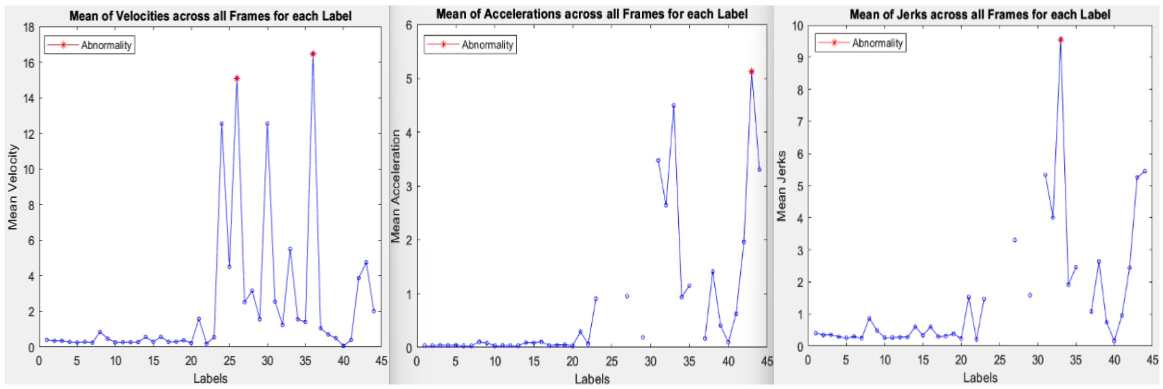


Figure 4-9: Abnormalities in mean velocity (left), acceleration (middle), and jerk (right) profiles representing outliers.

It is essential to note that currently, no single method can eliminate all outliers from MoCap data. Therefore, a combination of techniques was used to enhance the quality of data analysis. As illustrated in Figure 4-9, two anomalies, ID_{26} and ID_{36}, were identified using the mean velocity profile. Additionally, label 43 was pinpointed utilizing the mean acceleration profile, while label 33 was detected through the mean jerk profile. Notably, all anomalies were accurately identified. In this context, outliers are defined as data points that fall beyond three SD from the mean values of velocity, acceleration, and jerk.

Analysis of Figure 4-10 reveals a valid marker, label 7 highlighted in blue, that was identified as an abnormality in the mean velocity and jerk profiles but was not classified as an outlier due to fill level greater than 1%. Notably, this marker is valid, representing data collected from the left hand. Label 27, highlighted in green, was correctly identified as an anomaly and therefore an outlier in the acceleration profile.

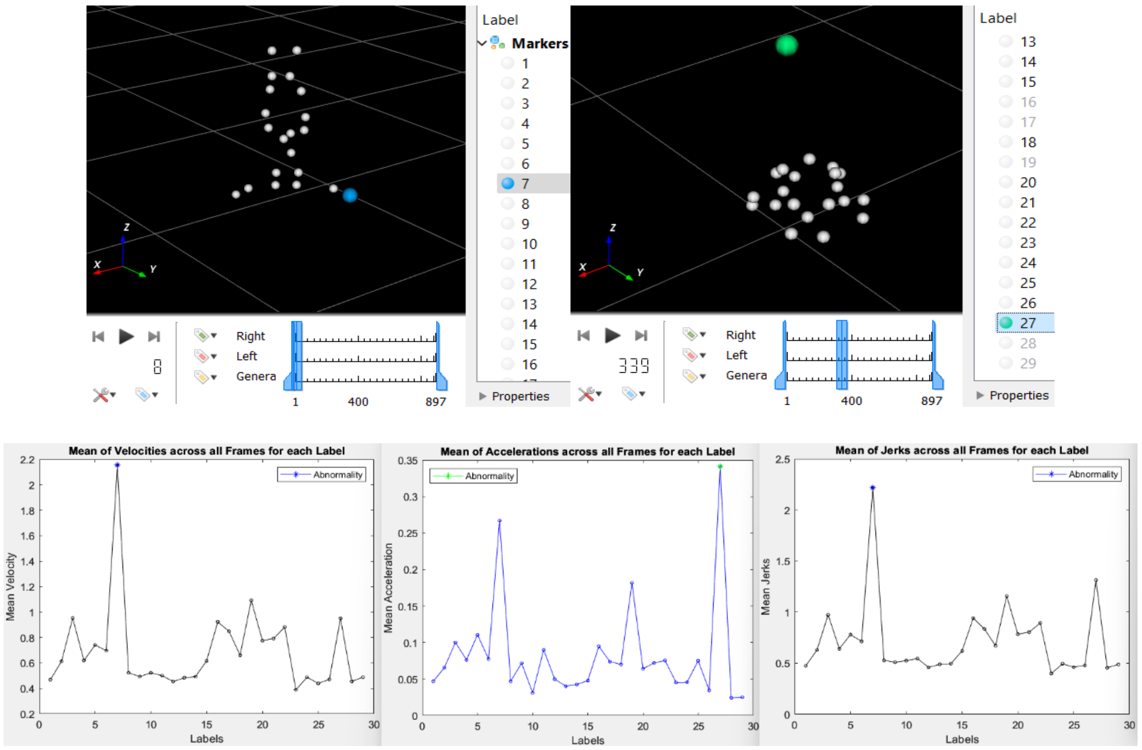


Figure 4-10: Label 7 (Hand) was identified as an abnormality in the mean acceleration profile but later excluded as an outlier due to a filling level exceeding 1%.

4.3.2.4 Repetitive Anomaly Detection

Noise significantly affects the mean values of norms, velocity, acceleration, and jerk. Thus, by applying the methods in steps 4.3.2.2 and 4.3.2.3, some noise was eliminated with a single application. Repeated applications of these methods, until no abnormalities were detected, identified most remaining outliers, drastically reducing data points (e.g., one dataset reduced from 41 to 28 points).

4.3.3 Geometry-based Inlier Detection

We implemented a geometry-based algorithm to identify valid markers amid remaining noise and ghost markers based on rigid body assumptions. Markers on a single bone form a rigid body part that can combine into rigid body triangle segments. We detected valid markers by comparing the bone lengths and joint angles of these segments in a MoCap file with predefined measurements from the marker set. However, the inherent flexibility of human anatomy—such as stretching and compressing—complicates this assumption. To address this challenge, we introduced tolerances and action constraints to approximate these segments as rigid bodies.

First, we identified pelvis points (spine, left and right waists) using a body side detection algorithm, which served as reference points for detecting other body markers. Next, torso points (left and right shoulders, chest, stomach) were identified. To reduce computational load, we eliminated near outliers by estimating a maximum region of interest (ROI) for valid markers based on the detected pelvis and torso markers. Subsequently, we detected lower limb points (hip, knee, calf, ankle), head points (left and right), and upper limb left and right points (elbow and hand). After identifying all valid markers, any remaining noise was removed. Finally, a triangulation method was used to locate a dropped ankle marker that was not captured in any frames. The evaluation was conducted visually.

Algorithm 2 presents the key steps for detecting inlier valid markers. Initially, we processed the C3D file from prior statistical analysis (Section 4.3.2). This process returned detected label (*assignedLabel*) for each pelvis points and a new C3D file (*newC3D*) that contained the visually confirmed cleaned pelvis segment. Subsequently, we inputted this new C3D file (*newC3D*) back into Algorithm 2 to detect and label the chest marker. This iterative process continued until all inlier valid markers were detected and labeled.

Algorithm 2 Inlier Detection

```
1:    $C3D \leftarrow$ Cleaned $newC3D$ file outputted from the previous iteration
2:    // point(trajectory) $\equiv$ point(frames of visibility)

3:  Function INLIERDETECTION ($C3D$)
4:    $points, num\_points, Labels, \sim, num\_frames \leftarrow$ READC3D($C3D$)
5:    $detectedpoints \leftarrow$ reappearing IDs for a passive marker
                                and their corresponding $trajectories$
6:    // "Shared Steps":
7:    $intersectFlag = 1$  if $\bigcap(detectedpoints(trajectories)) \neq \emptyset$
8:    $mergedTrajectory \leftarrow$ Merge short $trajectories$ of $detectedpoints$
9:    $gapfilledTrajectory \leftarrow$ linear_interpolate($mergedTrajectory$)
10:   $assignedLabel \leftarrow$ Assign a label (unique ID) to $detectedpoints$
11:   $removedLabels \leftarrow$ IDs that should be removed from $C3D$
12:   $newLabels \leftarrow Labels(removedLabels) = \emptyset$
13:   $newPoints \leftarrow points(:, newLabels, gapfilledTrajectory)$
14:   $newC3D \leftarrow$ Modify $C3D$ with $newPoints$
15:   return $newC3D, assignedLabel$
16: End
```

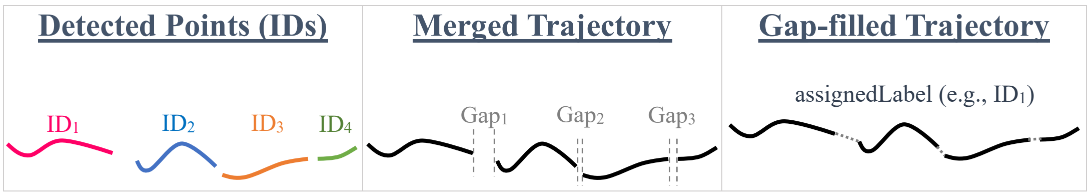


Figure 4-11: Inlier Detection; Detected Points (left), Shared Steps (middle and right).

A passive physical marker can be associated with multiple IDs (i.e., “reappearing” markers) due to occlusion. Consequently, the *detectedPoints* variable may include several IDs for each marker along with their trajectories, as illustrated in Figure 4-11 (first tile). The *detectedPoints* were identified according to specific geometric constraints and tolerances that are unique to each body segment.

After identifying the IDs (*detectedPoints*) for a specific physical marker, several steps were followed to create a new C3D file (*newC3D*) where this specific marker was cleaned and labeled. We refer to these steps as "shared steps" (Algorithm 2, steps 6 to 16) because their functionalities remain consistent for each specific physical marker.

In the initial phase of the shared steps, the short trajectories associated with a specific physical marker were analyzed to confirm that there were no intersections. This is essential because each physical marker should appear only once per frame; any overlap may indicate a false detection or a nearby ghost marker that meets the geometric criteria and is incorrectly identified as valid. If intersections were detected, users should be referred to conduct a visual inspection to keep a valid marker and remove any ghost or false detection. In the next phases of shared steps, as illustrated in Figure 4-11 (second and third tiles), the short trajectories of a specific physical marker were merged into a single trajectory (*mergedTrajectory*), gaps were filled using linear interpolation (*gapfilledTrajectory*), and each was assigned a unique identifier (*assignedLabel*) like ID_1. The remaining IDs (*removedLabels*) were eliminated from the original list of labels (*Labels*), creating a new list (*newLabels*). New data points (*newPoints*) were then created using *newLabels* and the *gapfilledTrajectory*. The original C3D file (*C3D*) was modified with these *newPoints* to produce an updated C3D file (*newC3D*). Ultimately, this inlier detection function returned a cleaned and labeled new C3D file (*newC3D*) along with the unique label (*assignedLabel*) for a specific physical marker, facilitating the detection of the next valid physical marker.

The following sections outline the process for identifying valid markers (*detectedPoints*) for various body segments: pelvis, torso, lower limb, head, and upper limb. The "shared steps" will not be repeated. Algorithm 3 outlines the essential functions for calculating distances and angles between markers utilized in all steps.

Algorithm 3 Fundamental Functions

```
 1: Function VectorizeTrio($points, ID_1, ID_2, ID_3$)
 2:     for $frame = 1, \ldots, num\_frames$ do
 3:         $[x_1 \quad y_1 \quad z_1] \leftarrow points(:, ID_1, frame)'$
 4:         $[x_2 \quad y_2 \quad z_2] \leftarrow points(:, ID_2, frame)'$
 5:         $\overrightarrow{P_1}[frame] \leftarrow [x_1 \quad y_1 \quad z_1] - [x_2 \quad y_2 \quad z_2]$
 6:         $P_{1_{mag}} \leftarrow ||\overrightarrow{P_1}[frame]|| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$
 7:     end for
 8:     $\vec{P}_2, P_{2_{mag}}, \vec{P}_3, P_{3_{mag}} \leftarrow$ Loop steps 2 to 7 for $(ID_1, ID_3), (ID_2, ID_3)$
 9:     return $\vec{P}_1, P_{1_{mag}}, \vec{P}_2, P_{2_{mag}}, \vec{P}_3, P_{3_{mag}}$
10: End

11: Function Dist($points, ID_1, ID_2, selectedFrame$)
12:     $\overrightarrow{P_1}, P_{1_{mag}}, \sim, \sim, \sim, \sim \leftarrow$ VectorizeTrio($points, ID_1, ID_2, \sim$)
13:     $d_{sel} \leftarrow P_{1_{mag}}[selectedFrame]$
14:     for $i = 1, \ldots, num\_frames$ do
15:         $d \leftarrow$ Append $P_{1_{mag}}[i]$to $d$    if $P_{1_{mag}}[i] \neq$ NaN
16:     end for
17:     $d_{mean} \leftarrow mean(d)$
18:     $d_{std} \leftarrow std(d) = \sqrt{\frac{\sum_{j=1}^{num\_frames}(d_i - d_{mean})^2}{num\_frames}}$
19:     return $d, d_{mean}, d_{std}, d_{sel}$
20: End

21: Function AngleTrio($points, ID_1, ID_2, ID_3$)
22:     $\vec{P}_1, P_{1_{mag}}, \vec{P}_2, P_{2_{mag}}, \vec{P}_3, P_{3_{mag}} \leftarrow$ VectorizeTrio($points, ID_1, ID_2, ID_3$)
23:     // MATLAB Functions: $atan2d, cross, dot, isnan, std, norm, mean$
24:     // Search all frames and retain not NaN $\theta_1, \theta_2, \theta_3$ values
25:     $\theta_1 \leftarrow \sim$isnan$(atan2d(norm(cross(\overrightarrow{P_1}, \overrightarrow{P_2})), dot(\overrightarrow{P_1}, \overrightarrow{P_2})))$
26:     $\theta_2, \theta_3 \leftarrow$ Loop step 25 for $(\overrightarrow{-P_1}, \overrightarrow{P_3}), (\overrightarrow{-P_2}, \overrightarrow{-P_3})$
27:     $\theta_{1_{mean}}, \theta_{2_{mean}}, \theta_{3_{mean}} \leftarrow mean(\theta_1), mean(\theta_2), mean(\theta_3)$
28:     $\theta_{1_{std}}, \theta_{2_{std}}, \theta_{3_{std}} \leftarrow std(\theta_1), std(\theta_2), std(\theta_3)$
29:     return $\theta_1, \theta_2, \theta_3, \theta_{1_{mean}}, \theta_{2_{mean}}, \theta_{3_{mean}}, \theta_{1_{std}}, \theta_{2_{std}}, \theta_{3_{std}}$
30: End
```

4.3.3.1 Pelvis Detection

The pelvis is the most rigid segment of the body within our tracking data. Therefore, we can identify the triangle formed by the spine, right waist, and left waist markers based on our marker set and action constraints such as distances and angles while considering user-defined tolerances. We assumed that pelvis points appear simultaneously in at least one frame.

First, we detected the probable pelvis points (i.e., reappearing passive marker IDs due to occlusion, see Figure 4-11 (first tile)) based on the geometric constraints of the marker set, without relying on precise values for distances and angles. Next, we refined our initial detection of potential candidates by applying orientation constraints. We employed PCA to determine the majority axis, attempting to identify the orientation of the human. Third, we accurately identified the correct pelvis points among the potential candidates by analyzing precise distances and angle values derived from our marker set. Fourth, we differentiated between the left and right sides of the body to allow detected pelvic triangles to be connected across broken trajectories (i.e., reappearing markers). Finally, we executed the "shared steps" (refer to Algorithm 2, lines 6 to 16) to output the detected pelvis point IDs (*Sp, RW, LW*) and generated a new C3D file containing the cleaned pelvis segment. This cleaned segment was utilized for the detection of the next valid markers (i.e., torso segment) in the subsequent section (section 4.3.3.2). The details of this process are outlined in Algorithm 4 and are further elaborated upon here. We inputted the points (*points*) retrieved from the C3D file using Algorithm 1 (i.e., the READC3D function) into this algorithm. It returned pelvis points (*pelvisPoints*) (i.e., *Sp, RW, LW*) and a new C3D file (*new C3D*).

Algorithm 4 Pelvis Detection

```
1:  Function PELVISDETECTION($points$)
2:      for $i = 1, \dots, num\_points$ & j = 1, ..., num_points & $i \neq j$ do
3:          $d, d_{mean}, d_{std}, \sim \leftarrow$ DIST($points, i, j, \sim$)
4:          if $d_{std} < \Delta d_{std}$ then
5:              $probablePelvisPoints \leftarrow$ Append $[i, j]$ to $probablePelvisPoints$
6:      end for
7:      $probTrio \leftarrow probablePelvisTriplets \leftarrow C(probablePelvisPoints, 3)$
8:      for $k = 1, \dots, num\_Trios$ do
9:          $frame_c \leftarrow intersect(trajectory(probTrio[k]))$
10:         //e.g., $frame_c[Trio_1] = \cap\left(p_1(traj), p_2(traj), p_3(traj)\right)$ if $Trio_1 = (p_1, p_2, p_3)$
11:         $probTrio \leftarrow probTrio$ if $frame_c \neq \emptyset$
12:         $A[k], B[k], C[k] \leftarrow probTrio[k][1], probTrio[k][2], probTrio[k][3]$
13:         $f_{cs} \leftarrow \min(frame_c[k])$ //first common frame in each $Trio$
14:     end for
15:     $AB, d_1, AC, d_2, BC, d_3 \leftarrow$ VECTORIZETRIO($points, A, B, C$)
16:     $AB_{cs}, d_{cs1} \leftarrow AB[f_{cs}], d_1[f_{cs}]$
17:     $AC_{cs}, d_{cs2}, BC_{cs}, d_{cs3} \leftarrow$ Loop step 15 for $AC, d_2, BC, d_3$
18:     $A, B, C \leftarrow$ Keep noncollinear Trios (i.e., if $cross(AB_{cs}, AC_{cs}) == 0$)
19:     if $|d_{cs1} - d_{cs2}| < \Delta d, |d_{cs1} - d_{cs3}| < \Delta d, |d_{cs2} - d_{cs3}| < \Delta d$ then
20:         $A, B, C \leftarrow A, B, C$ (Keep Isosceles $A, B, C$ Triangles)
21:     end if
22:     $\theta_1, \theta_2, \theta_3, \theta_{1_{mean}}, \theta_{2_{mean}}, \theta_{3_{mean}}, \theta_{1_{std}}, \theta_{2_{std}}, \theta_{3_{std}} \leftarrow$ ANGLETRIO($points, A, B, C$)
23:     $A, B, C \leftarrow$ Keep $A, B, C$ Trios if in each Trio: $\theta_{1_{std}}, \theta_{2_{std}}, \theta_{3_{std}} < \Delta\theta_{std}$
24:     $Sp, W_1, W_2 \leftarrow$ Sort $A, B, C$ Trios with $max(\theta_{1_{mean}}, \theta_{2_{mean}}, \theta_{3_{mean}})$ first
25:     $\theta_{sp}, \theta_{w1}, \theta_{w2} \leftarrow \theta_1[f_{cs}], \theta_2[f_{cs}], \theta_3[f_{cs}]$  //assume $\theta_1$ is max
26:     if $\theta_{W1} + \theta_{W2} < \theta_{Sp}$ & $|\theta_{W1} - \theta_{W2}| < \Delta\theta$ & $|d_{W1} - d_{W2}| < \Delta d$ then
27:         $Sp, W_1, W_2 \leftarrow$ Keep $Sp, W_1, W_2$ Trios
28:     end if
29:     $frame_{maj} \leftarrow$ first frame in $frame_c$ of each Trio with minimum $num\_points$
30:     $flag_{maj} \leftarrow$ PCAMAJORITYAXIS($points, frame_{maj}, Sp, W_1, W_2$)
31:     $Sp, W_1, W_2 \leftarrow$ Keep $Sp, W_1, W_2$ Trios  if $flag_{maj} == 1$
32:     // Compare with Marker set Measurments
33:     if $|\theta_{Sp} - \theta_{MSp}| < \Delta\theta_{MSp}$ & $|\theta_{W1} - \theta_{MW1}| < \Delta\theta_{MW}$ & $|\theta_{W2} - \theta_{MW2}| < \Delta\theta_{MW}$ &
            $|d_{W1} - d_{MW1}| < \Delta d$ & $|d_{W2} - d_{MW2}| < \Delta d$ & $|d_{WW} - d_{MWW}| < \Delta d$ then
34:         $Sp, W_1, W_2 \leftarrow$ Keep $Sp, W_1, W_2$ Trios
35:     end if
36:     $f_{ce} \leftarrow \max(frame_c)$ //last common frame in each $Trio$
37:     $frame_{side} \leftarrow [f_{cs}, f_{ce}]$ in each Trio
38:     $pelvisPoints = Sp, RW, LW \leftarrow$ BODYSIDEDETECTION($Sp, W_1, W_2, frame_{side}$)
39:     return $pelvisPoints, newC3D_{pelvis}$
```

4.3.3.1.1 Probable Pelvis using Rigid Body and Geometric Constraints

First, we identified pairs of points as potential pelvis points that formed a rigid bone by maintaining a constant distance within a specified threshold across all frames. To achieve this, we computed the distance between each point and all other points in each frame and repeated this process for all frames. Subsequently, we calculated the standard deviation (SD) of these distances (e.g., using the "std" function in MATLAB). The SD of the distance between two points is a measure of the variation in their distance across all frames about their mean distance. Pairs of points with a SD of less than a predetermined threshold (Δd_{std}) were retained as probable pelvis points (*probablePelvisPoints*). A lower threshold indicates reduced variance in the distance between two points, suggesting that they are static relative to one another throughout all frames. We applied a threshold of 0.5 cm ($\Delta d_{std} = 0.5\ cm$) to the calculated SDs for effectiveness across all our datasets (C3Ds). Next, we created all combinations of three probable pelvis points, refer to as "Trio" (*probTrio*). All predefined tolerances for pelvis detection are displayed in Table 4-1.

Table 4-1: Pelvis Detection Predetermined Tolerances

$\Delta d_{SWstd} = \Delta d_{Wstd} = \Delta d_{std}$	0.5 cm	Δd_{maj}	14.5 cm
$\Delta d_{SW} = \Delta d_{W} = \Delta d_{M}$	2.4 cm	Δsym_{maj}	8.9 cm
$\Delta\theta_{Spstd} = \Delta\theta_{Wstd} = \Delta\theta_{std}$	3.5°	$\Delta\theta_{SWW_{maj}}$	35.4°
$\Delta\theta_{MSp}$	6°	$\Delta\theta_{WW_{maj}}$	15.2°
$\Delta\theta_{W} = \Delta\theta_{MW}$	5°	* M: Marker Set Values	

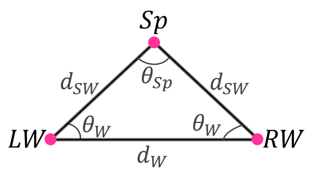


Figure 4-12: Pelvis Isosceles Triangle

To form a pelvis triangle (Figure 4-12), we assumed that the spine, right waist, and left waist must be visible in at least one frame. Therefore, we first identified the common frames for each Trio (*frame$_c$*) by calculating the intersection of the trajectories of the three points in each Trio. Subsequently, we filtered out Trios to retain only those that shared at least one common frame. We labeled three points in each Trio as A, B, and C. Subsequent steps focused on identifying the most probable pelvis Trios and determining which of these points correspond to the spine, right waist, and left waist. To reduce the computational load, we performed the remaining calculations in the initial frame of a common frame for each Trio (f_{cs}), as shown in Figure 4-13. Consequently, within this frame for each Trio, we verified whether the three points were non-collinear [27], thus forming a triangle.

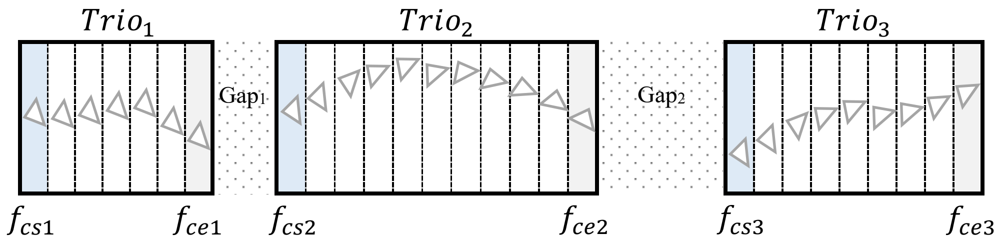


Figure 4-13: Probable Pelvis Trios; fcs and fce are the first and last frames of a Trio's Common frames

Based on our marker set geometry configuration (Figure 4-5), the pelvic segment is represented as an isosceles triangle. Accordingly, we retained the Trios that formed such a triangle by verifying whether two points of three points maintained equal distances to the third point, within an experimentally learned tolerance of 2.4 cm ($\Delta d = 2.4\ cm$). We further refined these Trios using the rigid body rule, retaining only those for which the variance of each of the three angles across all frames was less than a specified tolerance (i.e., $\Delta\theta_{std} = 3.5°$). In our marker set, the spine point has the largest angle, which exceeds the sum of the angles at the waist points. Thus, we organized the Trios (A, B, and C) by placing the point with the maximum angle first, renaming them as Sp, W1, and W2. We retained only those Trios where the spine angle (θ_{Sp}) was greater than the sum of the angles of the other two points (i.e., $\theta_{Sp} > \theta_{W1} + \theta_{W2}$). We then refined these Trios by verifying the isosceles triangle rules based on the detected spine point. We ensured that the two waist angles were equal within a tolerance of 5 degree ($\Delta\theta_W = 5°$) and that the side lengths (the distance between spine and waist) were equal within a tolerance of 2.4 cm ($\Delta d = 2.4\ cm$).

Figure 4-14 (a) illustrates a rejected isosceles triangle based on the SD of the distance criterion due to elbow movement. The distance between the elbow point (represented in blue) and the chest point (represented in green) varies significantly and is not static over time. This emphasizes the challenges associated with passive markers. An elbow passive marker may meet the SD criterion because it can remain stationary during its potentially short lifespan due to occlusion. Consequently, further criteria will be necessary in subsequent steps to address these situations effectively.

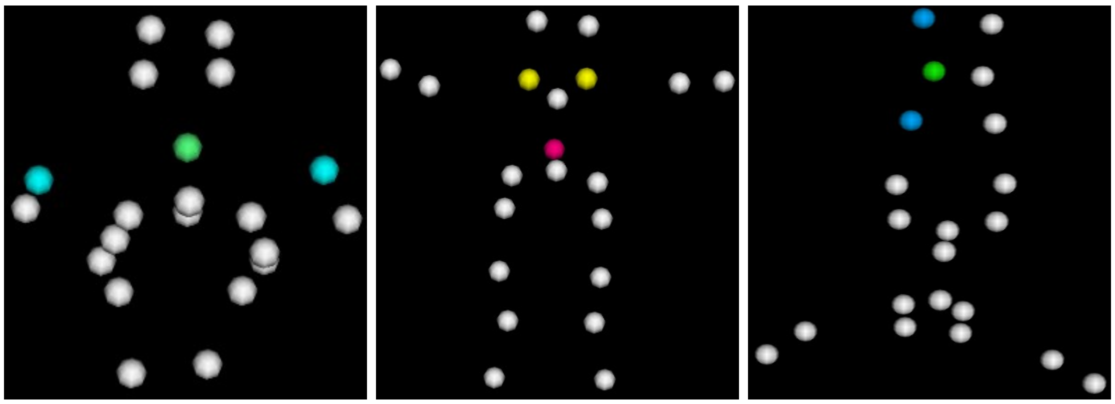


Figure 4-14: Probable Pelvis Triangles: (a) Not accepted due to SD criterion; (b) Not accepted due to a small θ_{Sp}; (c) Accepted due to the geometric and rigid body criteria, but this triangle does not represent a pelvis and requires additional criteria for rejection.

Figure 4-14 (b) shows an isosceles triangle that was not accepted due to a small θ_{Sp} (i.e., does not meet $\theta_{Sp} > \theta_{W1} + \theta_{W2}$). The third image (c) presents a triangle that met all criteria for being a probable pelvis segment based on isosceles triangle and rigid body requirements. To further refine our selection, we will introduce orientation criteria in the next section to eliminate this triangle from consideration as a probable pelvis Trio.

4.3.3.1.2 Probable Pelvis based on PCA-based Majority Axis Orientation

To eliminate incorrectly identified probable pelvis triangles, such as the one depicted in Figure 4-14 (c), we implemented an additional criterion based on human body orientation. Given the constraints of our marker set and dataset actions, we assumed that the pelvis triangle should align with the body's orientation, which corresponds to the majority axis of point distribution estimated using PCA.

Algorithm 5 illustrates this process. It was provided with probable pelvis Trios (*Sp, W1,W2*) identified in the previous section, along with the specific frame ($frame_{maj}$) for performing the calculations. It then returned a flag ($flag_{maj}$) indicating whether the Trio was accepted as a probable pelvis Trio, where a value of 1 signified an accepted Trio.

Algorithm 5 Majority Axis Criterion

```
 1: Function PCAMAJORITYAXIS$(points, frame_{maj}, Sp, W_1, W_2)$
 2:     $centroid \leftarrow nanmean(points(:,:,frame_{maj}), 2)$
 3:     $pcaCoefficients \leftarrow pca(points(:,:,frame_{maj})')$
 4:     $direction_{maj} \leftarrow pcaCoefficients(:,1)$
 5:     projection$SpDistanceToAxis_{maj} \leftarrow |dot(Sp - centroid, direction_{maj})|$
 6:     $spineFlag = 1$ if $distanceSpToAxis_{maj} < \Delta d_{maj}$
 7:     $distW_1ToAxis_{maj} \leftarrow |dot(W_1 - centroid, direction_{maj})|$ //Repeat for W$_2$
 8:     $W_1W_2Symmetry_{maj} \leftarrow distW_1ToAxis_{maj} - distW_2ToAxis_{maj}$
 9:     $symmetryFlag = 1$ if $|W_1W_2Symmetry_{maj}| < \Delta sym_{maj}$
10:     $//cross(WW_{maj}) \leftarrow cross(direction_{maj},$ waistl$ine)$
11:     $\theta_{WW_{maj}} \leftarrow \sim atan2d(norm(cross(WW_{maj})), dot(WW_{maj}))$
12:     $\theta_{WW_{maj}}Flag = 1$ if $\left|\theta_{WW_{maj}} - 90\right| < \Delta\theta_{WW_{maj}}$
13:     //SWWLine: Line connects Sp to $midpoint\ of$ waistl$ine$
14:     $//cross(SWW_{maj}) \leftarrow cross(direction_{maj}, SWWLine)$
15:     $\theta_{SWW_{maj}} \leftarrow \sim atan2d(norm(cross(SWW_{maj})), dot(SWW_{maj}))$
16:     $\theta_{SWW_{maj}}Flag = 1$ if $|\theta_{SWW_{maj}} - 90| < \Delta\theta_{SWW_{maj}}$
17:     $flag_{maj} \leftarrow spineFlag * symmetryFlag * \theta_{SWW_{maj}}Flag * \theta_{WW_{maj}}Flag$
18:     return $flag_{maj}$
```

4.3.3.1.2.1 Frame of Interest

First, as indicated in Algorithm 4, line 29, we specified a distinct frame (*frame*$_{maj}$) for the calculation of PCA-based majority axis. The rationale for not utilizing the same frame (*f*$_{cs}$) as in previous steps stems from the impact of noise. Given the sparsity of our dataset, noise can considerably affect the orientation of the major axis, as illustrated in Figure 4-15. In a noisy frame (left image), the majority axis is misaligned compared to the accurate majority axis of the valid points, as illustrated in the non-noisy frame (right image).

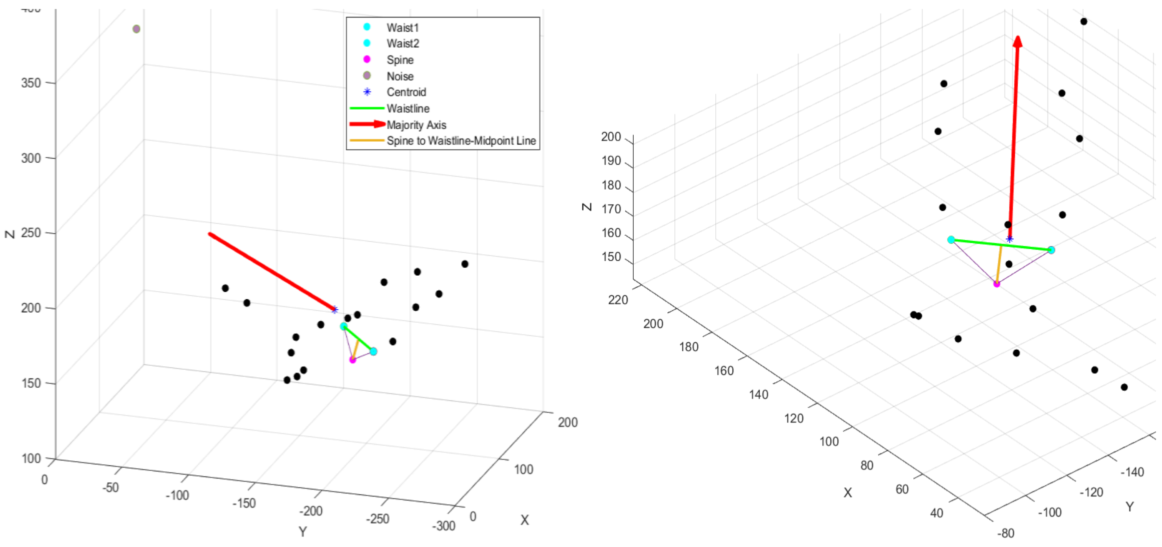


Figure 4-15: The Impact of Noise on Majority Axis Direction: (Left) Noisy Frame; (Right) Non-Noisy Frame

Consequently, we identified the first frame among the common frames (*frame*$_c$) of each Trio that contained a minimal number of points. We assumed that any frame with a minimal number of points would include only properly tracked markers. The criteria for minimal points were set to be greater than 19 and less than 21; if this condition was not met, a flag was raised for visual inspection.

4.3.3.1.2.2 PCA

Principal component analysis (PCA) [18] is a statistical technique that analyzes and reduces data dimensionality while preserving as much variance as possible. In the context of 3D space, PCA helps identifying the directions (principal components) where variation is maximized (i.e., the data points are most spread out), effectively finding the majority axis of point distribution. The process begins by centering data around feature means, involving subtracting mean values from each coordinate across all points. Then, the covariance matrix of the centered data is computed to capture relationships between variables. Eigenvalue decomposition is then performed on the covariance matrix, identifying eigenvalues indicating variance along each principal component and eigenvectors representing directions of variance. The eigenvalues are sorted in descending order; corresponding eigenvectors are also rearranged accordingly. The top eigenvectors correspond to directions with maximum variance and thus represent principal components.

4.3.3.1.2.3 Finding Majority Axis using PCA

After identifying the specified frame ($frame_{maj}$), we calculated the centroid of the 3D point distribution in this frame (i.e., mean of the non-null points) in Algorithm 5, line2 ($centroid$). We then performed PCA on the transposed points to capture PCA coefficients ($pcaCoefficients$) (i.e., using “pca” function in MATLAB). Transposing is crucial because PCA typically requires data to be organized such that each row corresponds to an observation (in this case, a point in 3D space), and each column corresponds to a variable (the x, y, z coordinates). This arrangement allows PCA to analyze how the points vary across their features. We then determined the first coefficient, which represents the direction of maximum variance, identified as the majority axis direction ($direction_{maj}$).

4.3.3.1.2.4 Probable Pelvis Detection based on Orientation

We established an additional criterion for identifying a probable pelvis triangle based on the orientation of the majority axis. The first criterion in Algorithm 5, line 5, stipulates that the spine point must be near the distribution's centroid and aligned with the majority axis. To verify this, we first computed a vector that indicated the distance from the spine point to the centroid. Next, we calculated the dot product of this vector with the direction of the majority axis. The result ($\text{projection}SpDistanceToAxis_{maj}$) represented how far away the spine point was from the centroid when projected along this direction of maximum variance. This can be interpreted as a measure of distance or deviation along that principal component. If this deviation was less than an experimentally determined threshold of 14.5 cm ($\Delta d_{maj} = 14.5\ cm$), a flag (*spineFlag)* to 1, indicating the first criterion was met.

The second criterion requires that the right and left waists be symmetrically positioned relative to the major axis and close to it. We assessed this by calculating the projection distances from W1 and W2 to the major axis. If the absolute difference between these distances was less than a threshold of 8.9 cm ($\Delta sym_{maj} = 8.9\ cm$), we set a flag ($symmetryFlag$) to 1, indicating that we satisfied the second criterion.

As a third criterion, we assumed that the waistline should be perpendicular to the major axis based on our dataset actions. Consequently, we evaluated whether the angle ($\theta_{WW_{maj}}$) deviation between the waistline and the major axis from 90 degrees was less than a specified threshold of 15.2 degrees ($\Delta\theta_{WW_{maj}} = 15.2°$). If this condition was met, a flag ($\theta_{WW_{maj}}Flag$) was set to 1, indicating that the third criterion had been met.

In the fourth criterion, we assumed that the line connecting the spine to midpoint of waistline (*SWWLine*) should be perpendicular to the major axis based on our dataset actions. Therefore, if the angle ($\theta_{SWW_{maj}}$) deviation between this line and the major axis from 90 degrees was less than a threshold of 35.4 degrees ($\Delta\theta_{SWW_{maj}} = 35.4°$), a flag ($\theta_{SWW_{maj}}Flag$) was set to 1.

Finally, if all four criteria were met, we set a flag ($flag_{maj}$) to 1 to accept the probable pelvis Trio. Figure 4-16 shows a rejected triangle (spine (magenta), elbows(cyan)) due to the $\theta_{SWW_{maj}}Flag$, with $\theta_{SWW_{maj}} = 6.11°$. All criteria were satisfied. Note that the passive elbow markers (blue) met the SD criteria, and $\theta_{WW_{maj}} = 92.8°$.

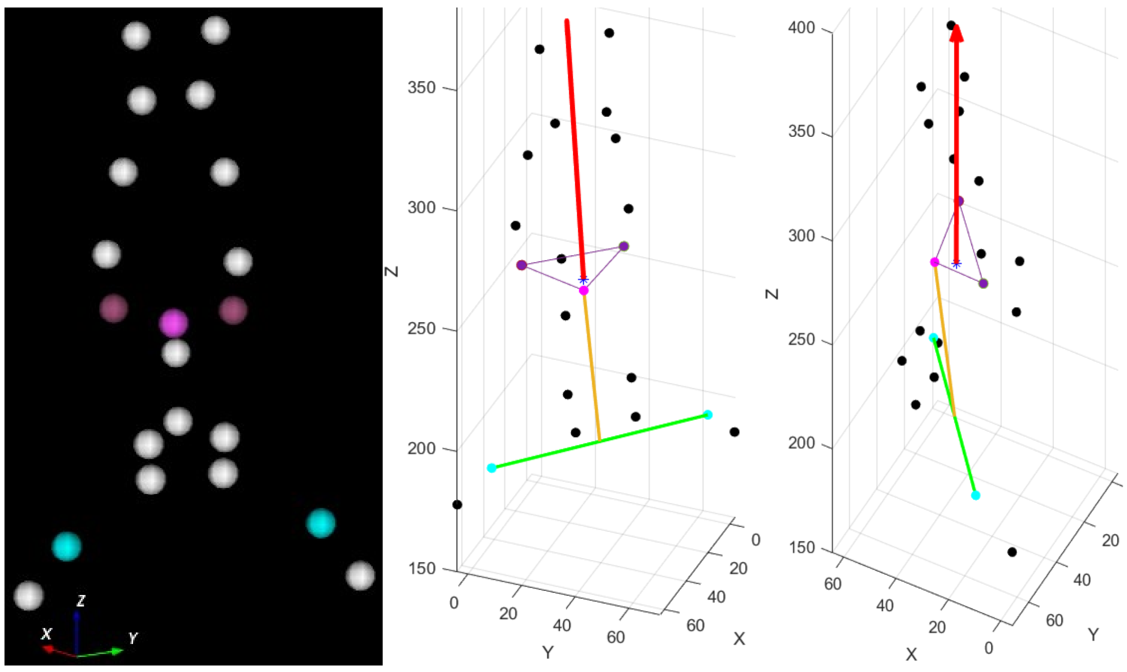


Figure 4-16: Rejected probable pelvis Trio (spine (magenta), elbows(cyan)) due to angle between SWW line (yellow) and majority axis (red);the right tile provides an additional view for clarity. The correct pelvis is shown in a purple triangular shape.

These orientation criteria, particularly the third and fourth ones, are influenced by the marker set and action constraints. This influence necessitates a high threshold to satisfy the criteria due to the inherent flexibility of the human body. Therefore, while these criteria effectively eliminate triangles that significantly deviate from resembling a pelvis segment —such as those depicted in Figure 4-16 and the image (c) in Figure 4-14— additional criteria are necessary for the accurate detection of pelvis points. These further criteria will be discussed in the next section.

4.3.3.1.3 Pelvis Detection based on Marker set Values

In the final stage of pelvis detection, we accurately identified these points by comparing the distances and joint angles of each probable pelvis Trio with those measurements derived from our marker set, taking tolerances into account as shown in Figure 4-17 and Table 4-2 (*M* represents marker set values). We then retained the Trios that met these constraints.

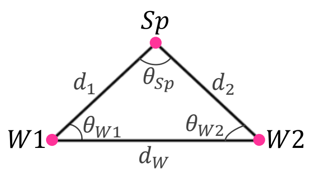


Figure 4-17: Pelvis Isosceles Triangle

Table 4-2: Pelvis Detection Marker set Constraints (M: Marker set)

Constraints	$\Delta\, cm$	**Constraints**	$\Delta\,°$
$\lvert d_1 - Md_1 \rvert \ < \Delta d_M$	2.4	$\lvert \theta_{Sp} - M\theta_{Sp} \rvert \ < \Delta\theta_{MSp}$	6
$\lvert d_2 - Md_2 \rvert \ < \Delta d_M$	2.4	$\lvert \theta_{W1} - M\theta_{W1} \rvert < \Delta\theta_{MW}$	5
$\lvert d_W - Md_W \rvert < \Delta d_M$	2.4	$\lvert \theta_{W2} - M\theta_{W2} \rvert < \Delta\theta_{MW}$	5

This stage significantly reduces the probability of false detection to nearly zero. However, due to user-defined tolerances, there is a possibility that ghost markers—those very close to valid markers—may satisfy all criteria and be mistakenly classified as valid markers. To address this issue, we checked an *intersectFlag* as the first step in the "Shared Steps" of Algorithm 2, line 8. This intersect step was executed after detecting the body side detection, which will be detailed in the next section.

4.3.3.1.4 Body Side Detection

After identifying pelvis Trios that met all criteria, we differentiated between the left and right waist. Since these triangles are isosceles, only the spine point was accurately detected. Algorithm 6 illustrates our body side detection function (BODYSIDEDETECTION). It used the detected Trios ($Sp, W1, W2$) identified in the previous section, along with specific frames ($frame_{side}$) as inputs and returned new Trios (Sp, RW, LW), which determined the spine, right waist, and left waist points for each Trio.

Algorithm 6 Body Detection

```
1:     $frame_{side} \leftarrow [f_{cs}, f_{ce}]$ in each Trio

2:  **Function** BODYSIDEDETECTION($Sp, W_1, W_2, frame_{side}$)
3:     **for** $i = 1, \dots, num\_\text{Trios}$ **do**
4:        $Sp_{cs}[i] \leftarrow points(:, Sp[i], f_{cs}[\text{i}])$   //Repeat for $W1_{cs}[i], W2_{cs}[i]$
5:        $Sp_{ce}[i] \leftarrow points(:, Sp[i], f_{ce}[\text{i}])$   //Repeat for $\text{W1}_{\text{ce}}[\text{i}], \text{W2}_{\text{ce}}[\text{i}]$
6:     **end for**
7:     **for** $j = 1, \dots, (num\_\text{Trios} - 1)$ **do**
8:        $translationVector[j] \leftarrow Sp_{cs}[j+1] - Sp_{ce}[j]$;
9:        $translationMatrix[j] \leftarrow eye(4)$; // Initialize identity matrix
10:       $translationMatrix[j](1:3, 4) \leftarrow translationVector[j]'$
11:       $homogeneousW1_{ce}[j] \leftarrow [W1_{ce}[j], 1]$ //homogeneous coordinate
12:       $translatedW1_{ce}[j] \leftarrow (translationMatrix[j] * homogeneousW1_{ce}[j]')'$
13:       $translatedW1_{ce}[\text{j}] = translatedW1_{ce}[j](1:3)$
14:       //Repeat Line 11 to 13 for W2
15:    **end for**
16:    **for** $k = 2, \dots, num\_Trios$ **do**
17:       $\text{distW1}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}] \leftarrow \text{norm}(translatedW1_{ce}[\text{k}-1] - W1_{cs}[\text{k}]))$
18:       $\text{distW2}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}] \leftarrow \text{norm}(translatedW1_{ce}[\text{k}-1] - W2_{cs}[\text{k}]))$
19:       **if** $\text{distW1}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}] < \text{distW2}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}]$ **then**
20:          $[Sp, RW, LW][\text{k}] \leftarrow [Sp, W1, W2][\text{k}]$      // No Change
21:       **else if** $\text{distW2}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}] > \text{distW1}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}]$
22:          $[Sp, RW, LW][\text{k}] \leftarrow [Sp, W2, W1][\text{k}]$      // Change
23:       **else if** $\text{distW1}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}] == \text{distW2}_{\text{cs}}\text{transW1}_{\text{ce}}[\text{k}]$
24:          $[Sp, RW, LW][\text{k}] \leftarrow$ user visual decision
25:    **end for**
26:    **return** $Sp, RW, LW$
```

4.3.3.1.4.1 Statement of the Problem

We need to distinguish between left waist and right waist (W1 and W2) in each Trio to identify corresponding reappearing markers on each side of the body. This involves recognizing the corresponding points of each Trio, as detailed in Algorithm 2, line 6, under "Shared Steps." By doing so, we can merge the short trajectories of each physical marker into a single complete trajectory with potential gaps, as illustrated in Figure 4-11. Since we have detected spine point (Sp) for each Trio, we can combine the trajectories of all spine points (i.e., $\cup\,(Sp_{trio1}(Trajectory), Sp_{trio2}(Trajectory), \ldots, Sp_{trioN}(Trajectory))$. Therefore, it is essential to determine whether to merge the trajectory of $W1_{trio1}$ with $W1_{trio2}$ or $W2_{trio2}$. This process must be repeated for all Trios (e.g., merging $W1_{trio2}$ with $W1_{trio3}$ or $W2_{trio3}$).

Our objective was to facilitate manual cleaning rather than labeling. Therefore, we only need to identify which points are on the same side of the body, rather than precisely distinguishing between the true left and right sides. The actual sides will be visually confirmed during the validation and labeling process of the cleaned C3D file.

4.3.3.1.4.2 Statement of the Solution

To solve this problem, we aimed to translate one Trio (e.g., $Trio_n$) from its initial frame (f_{csn}) of its common frame ($frame_{cn}$) to its subsequent Trio ($Trio_{n+1}$) initial frame (f_{csn+1}), based on the translation between two spine points of these two successive Trios. Then, we found the Euclidean distance between translated $W1_{trion}$ and $W1_{trion+1}$ (e.g., d_1), and translated $W1_{trion}$ and $W2_{trion+1}$ (d_2). Then, $d1<d2$ means $W1_{trion}$ corresponds to $W1_{trion+1}$, and $d2<d1$ means $W1_{trion}$ corresponds to $W2_{trion+1}$, as shown in Figure 4-18 and Figure 4-19.

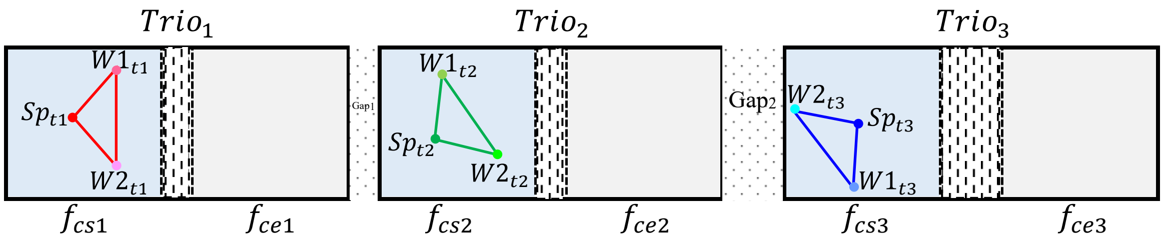


Figure 4-18: Probable pelvis triangles (Spine, Waist 1, Waist 2) in the first frame (fcs) of each Trio's common frame.

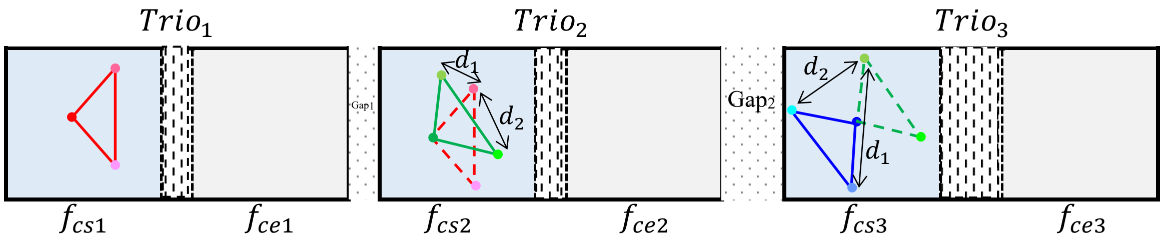


Figure 4-19: Solution 1 for left and right waist points detection: Translating the Trio$_n$ triangle from its fcs$_n$ to the next Trio$_{n+1}$ first frame (fcs$_{n+1}$) and calculating the distances between waist 1 in the Trio$_{n+1}$ and the translated waist 1 and waist 2 of the Trio$_n$.

However, two problems may arise: 1) if the distances are equal; or 2) if the body rotates during gaps between pelvis Trios, as shown in Figure 4-20.

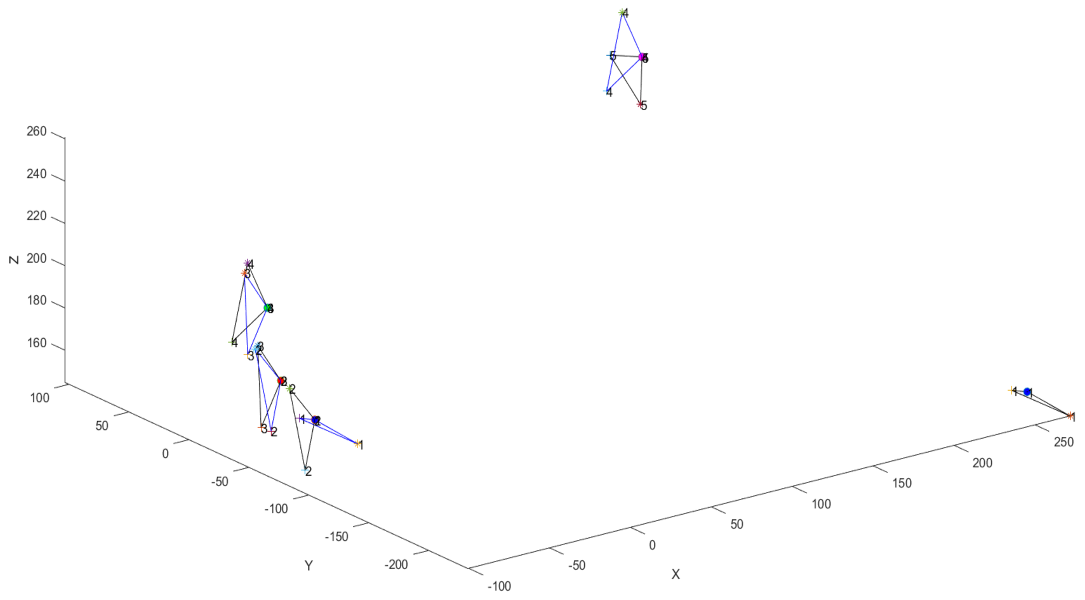


Figure 4-20: Potential problem of equal distances in Solution 1 for detecting left and right waist points, illustrated by the second blue triangle labeled "2," where the distance between points "2" (in the translated blue triangle) and points "1" (in the original black triangle) is identical.

Figure 4-20 illustrates five pelvis Trios from a C3D file positioned in 3D space, depicted as black triangles labeled 1 to 5. These Trios are pelvis reappearing markers due to occlusion (i.e., gaps between the black triangles). The first black triangle, labeled 1, corresponds to the initial detected pelvis Trio (Trio_1) presented in its f_{cs1} frame. The accompanying blue circle represents the spine (Sp_1), while the star and plus markers denote the waist markers. The second black triangle (Trio_2), labeled 2, is depicted in its f_{cs2} frame, with this pattern continuing for the remaining black triangles.

The blue triangles represent translated triangles. For instance, Trio_1 was translated to f_{cs2}, appearing as a blue triangle labeled 1, aligning the spine point (Sp_1) of Trio_1 with spine point (Sp_2) of Trio_2. This translation process was applied repeatedly, resulting in all blue triangles being classified as translated Trios.

The first issue occurred in the second Trio (Trio_2), with a large gap between Trio_1 and Trio_2, where the distance between the two waist points of triangle 1 are equal to the closest two waist points in triangle 2. The second problem may occur because of body rotation during transitional movements during gaps, such as when an individual shifts from a supine position (facing up) to a prone position (facing down). Solutions for addressing these two issues will be outlined in the following section.

4.3.3.1.4.3 Frame of Interest

To address the issue of potential body rotation during gaps between pelvis Trios, we hypothesized that performing translations between closer frames would decrease the likelihood of such rotation. This is based on the understanding that the human body has a maximum speed at which it can move, particularly in underwater environments.

First, as shown in Algorithm 6, line 1, we received a distinct vector (*$frame_{maj}$*) from Algorithm 4, line 37. This vector includes two frames for each Trio: f_{cs} and f_{ce}, which represent the first and the last frames of the common frame (*$frame_{cs}$*) for each Trio.

Then, as shown in Figure 4-21, we translated the first Trio (red triangle) from f_{ce1} to f_{cs2} and compared the distances (d_1 and d_2) between one waist point of Trio$_2$ (green triangle) and two waist points of the translated triangle (dashed-line red triangle), as illustrated in Algorithm 6, line 16 to 24. If the distances were equal (d1 = d2), the automatic processing of the dataset was paused until human intervention resolved the condition, as a decision was necessary for the subsequent translation. We continued this process for all Trios.

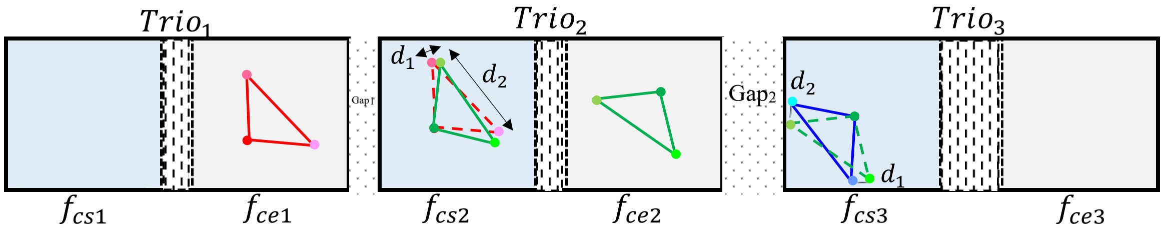


Figure 4-21: Solution 2 for the specific rotation problem arises with Solution 1 in detecting left and right waist points. Instead of translating between the first frames (fcs) of Trios, translation is done between the last frames (fce) of the current Trio$_n$ and the first frame (fcs) of the next Trio$_{n+1}$ to reduce gap size and, hence, the potential rotation problem.

This solution has consistently produced accurate results across all our datasets by effectively minimizing the gap size between two Trios. As an example, shown in Figure 4-21, the green triangle in f_{cs2} exhibits a significantly different orientation compared to the blue triangle in f_{cs3}. In contrast, the green triangle in f_{ce2} shows less variation than the blue triangle in f_{cs3}. However, there is still a potential for body rotation to occur during these small gaps between Trios. To ensure the reliability of the results, a final visual check was conducted to validate the outcomes after all data had been processed.

4.3.3.2 Detection of Remaining Inliers

After identifying the pelvis points (spine, right waist, left waist), we detected the remaining inliers using Algorithm 2. We will only discuss the detected points (Algorithm 2, line 5) and will not cover the "Shared Steps" (Algorithm 2, lines 6 to 15).

We first identified torso points (shoulders, chest, stomach) and then reduced computational load by eliminating near outliers. Next, we detected lower limb points (hip, knee, calf, ankle), head points (left and right), and upper limb points (elbow and hand). Then, we removed any remaining noise and reconstructed a missing ankle marker. The evaluation was conducted visually.

The detection of remaining valid markers involved forming a triangle with one or two previously detected points and applying geometric constraints by comparing angles and lengths to those in the marker set. We will detail the shoulder point detection process, while for other inliers, we will simply present the triangle and constraint table, as their detection is similar to that of shoulder points.

4.3.3.2.1 Torso Markers Detection

After identifying pelvis points (spine, right waist, left waist), we proceeded to detect torso points (left and right shoulders, chest, stomach). The subsequent sections describe the process.

4.3.3.2.1.1 Shoulder Markers Detection

We first located the shoulder points using potential triangles formed by the left and right shoulders and the detected spine point, as shown in Figure 4-22. This process applied constraints detailed in Table 4-3. Algorithm 7 outlines this method, utilizing the previously identified pelvis points (*Sp, RW, LW*) to return the right shoulder (*Sh_R*), left shoulder (*Sh_L*), and a new C3d (*newC3D_{shoulder}*). Additionally, Algorithm 8 illustrates the process for distinguishing between left and right shoulder points.

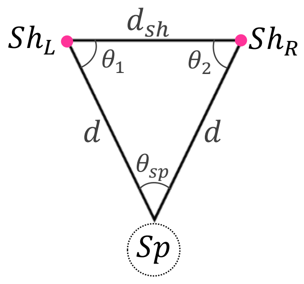


Figure 4-22: Shoulder Detection Isosceles Triangle

Table 4-3: Shoulder Detection Marker set Constraints (M: Marker set)

Constraints	**Δ *cm***	**Constraints**	**Δ °**
$\lvert d_{mean} - Md \rvert < \Delta d$	10	$\lvert \theta_{1_{mean}} - M\theta_1 \rvert < \Delta\theta_{sh}$	6
$\lvert d_{sh_{mean}} - Md_{sh} \rvert < \Delta d_{sh}$	5	$\lvert \theta_{2_{mean}} - M\theta_2 \rvert < \Delta\theta_{sh}$	6

Algorithm 7 Shoulder Detection

```
 1: $pelvisPoints, newC3D_{pelvis} \leftarrow$ PELVISDETECTION($points$)
 2: $Sp, RW, LW \leftarrow pelvisPoints$
 3: $newC3D, assignedLabel \leftarrow$ INLIERDETECTION ($newC3D_{pelvis}$)
 4: $points, num\_points, Labels, \sim, num\_frames \leftarrow$ READC3D($newC3D$)

 4: Function SHOULDERDETECTION($points, Sp, RW, LW, \Delta d, \Delta d_{sh}, \Delta\theta_{sh}$)
 5:     for $i = 1, \dots, num\_points$ do
 6:         $d, d_{mean}, d_{std}, \sim \leftarrow dist(points, Sp, i, \sim)$
 7:         if $|d_{mean} - Md| < \Delta d$ then
 8:             $probableShoulderPoints \leftarrow$ Append $i$ to $probableShoulderPoints$
 9:     end for
10:     $probPair \leftarrow probShoulderPairs \leftarrow C(probableShoulderPoints, 2)$
11:     for j $= 1, \dots, num\_Pairs$ do
12:         $d_{sh}, d_{sh_{mean}}, d_{sh_{std}}, \sim \leftarrow dist(points, probPair(j,1), probPair(j,2), \sim)$
13:         $probPair \leftarrow$ Keep $probPair(j)$ if $|d_{sh_{mean}} - Md_{sh}| < \Delta d_{sh}$
14:     end for
15:     for $k = 1, \dots, num\_Pairs$ do
16:         $\sim, \sim, \sim, \theta_{1_{mean}}, \theta_{2_{mean}}, \theta_{3_{mean}}, \sim, \sim, \sim$
                $\leftarrow$ ANGLETRIO($points, Sp, probPair(k,1), probPair(k,2)$)
17:         if $|\theta_{1_{mean}} - M\theta_1| < \Delta\theta_{sh}$ & $|\theta_{2_{mean}} - M\theta_2| < \Delta\theta_{sh}$ then
18:             $shoulderPair \leftarrow$ Keep $probPair(j)$
19:     end for
20:     $Sh_R, Sh_L \leftarrow$ SHOULDERSIDEDETECTION($shoulderPair, RW, LW$)
21:     return $Sh_R, Sh_L, newC3D_{shoulder}$
```

Algorithm 7 identified points that were within a distance of 10 cm ($\Delta d = 10\ cm$) from the detected spine point, matching the marker set length. It then formed pairs of these points ($probPair$) and retained those where the distance between each pair was consistent with the marker set, allowing for a threshold of 5 cm ($\Delta d_{sh} = 5\ cm$). The shoulder angles of these pairs were compared to the marker set values within a threshold of 6 degrees ($\Delta\theta_{sh} = 6°$). Pairs meeting these criteria were classified as shoulder points. It is important to note that we took into account the mean distances and angles; therefore, we did not need to perform the calculations within a specific frame.

The function (SHOULDERSIDEDETECTION), illustrated in Algorithm 8, differentiated between right and left shoulder points by comparing the Euclidean distances from detected shoulder points to the right and left waists. For each pair of shoulder points ($shoulderPair$), the point closer to the right waist was identified as the right shoulder, while the other was designated as the left shoulder.

Algorithm 8 Shoulder Side Detection

```
 1: Function ShoulderSideDetection($shoulderPair, RW, LW$)
 2:     for $i = 1, \dots, num\_pairs$ do
 3:         $\sim, d_{1mean}, \sim, \sim \leftarrow dist(points, RW, shoulderPair(i,1), \sim)$
 4:         $\sim, d_{2mean}, \sim, \sim \leftarrow dist(points, RW, shoulderPair(i,2), \sim)$
 5:         if $d_{1mean} < d_{2mean}$ then
 6:             $Sh_R, Sh_L \leftarrow shoulderPair(i,1), shoulderPair(i,2)$
 7:         else
 8:             $Sh_R, Sh_L \leftarrow shoulderPair(i,2), shoulderPair(i,1)$
 9:         end if
10:     end for
11:     return $Sh_R, Sh_L$
```

4.3.3.2.1.2 Chest Marker Detection

The chest point was determined by constructing potential triangles using the detected left and right shoulder points along with the chest, as illustrated in Figure 4-23, and adhering to the constraints outlined in Table 4-4.

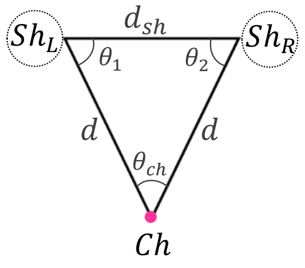


Figure 4-23: Chest Detection Isosceles Triangle

Table 4-4: Chest Detection Marker set Constraints (M: Marker set)

Constraints	$\Delta\ cm$	Constraints	Δ °
$\lvert d_{mean} - Md \rvert < \Delta d$	5	$\lvert \theta_{1_{mean}} - M\theta_1 \rvert < \Delta\theta_{sh}$	7
		$\lvert \theta_{2_{mean}} - M\theta_2 \rvert < \Delta\theta_{sh}$	7

4.3.3.2.1.3 Stomach Marker Detection

The stomach point was identified by creating potential triangles with the spine and chest points, as shown in Figure 4-24, while following the constraints in Table 4-5.

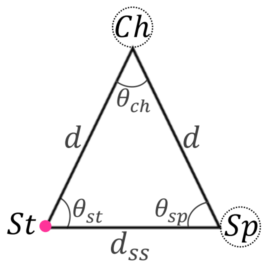


Figure 4-24: Stomach Detection Isosceles Triangle

Table 4-5: Stomach Detection Marker set Constraints (M: Marker set)

Constraints	$\Delta\ cm$	Constraints	Δ °
$\lvert d_{mean} - Md \rvert < \Delta d$	5	$\lvert \theta_{st_{mean}} - M\theta_{st} \rvert < \Delta\theta_{sh}$	7
$\lvert d_{ss_{mean}} - Md_{ss} \rvert < \Delta d_{ss}$	5	$\lvert \theta_{ch_{mean}} - M\theta_{ch} \rvert < \Delta\theta_{ch}$	7

4.3.3.2.2 Near Outlier Removal

Pelvis markers (left waist, spine, right waist) and the torso markers (left shoulder, right shoulder, chest, stomach) have been identified. This allowed us to identify noisy points based on their distance from the torso and pelvis segments, leveraging our knowledge of human anatomy. We defined the maximum arm length as the distance from the shoulder marker to the hand marker within the “T-shape” marker set (Figure 4-5), as this configuration represents the maximum extension of the arm:

$$length_{arm_{max}} = dist(Shoulder, Hand)$$

Similarly, the maximum leg lengths were simply the distance from the waist to the ankle:

$$length_{leg_{max}} = dist(Waist, Ankle)$$

We then created two spheres of radii equal to $length_{arm_{max}}$ centred at each shoulder (highlighted blue and green), and two spheres of radii $length_{leg_{max}}$ centred at each waist marker (highlighted red and purple). The region of interest (ROI) was defined by the union of the four spheres as shown in Figure 4-25. Any point outside this ROI was considered noise and was subsequently removed, since anatomically the human body cannot contain points further away than the boundaries of the spheres. This step significantly reduced the computational load for subsequent processes.

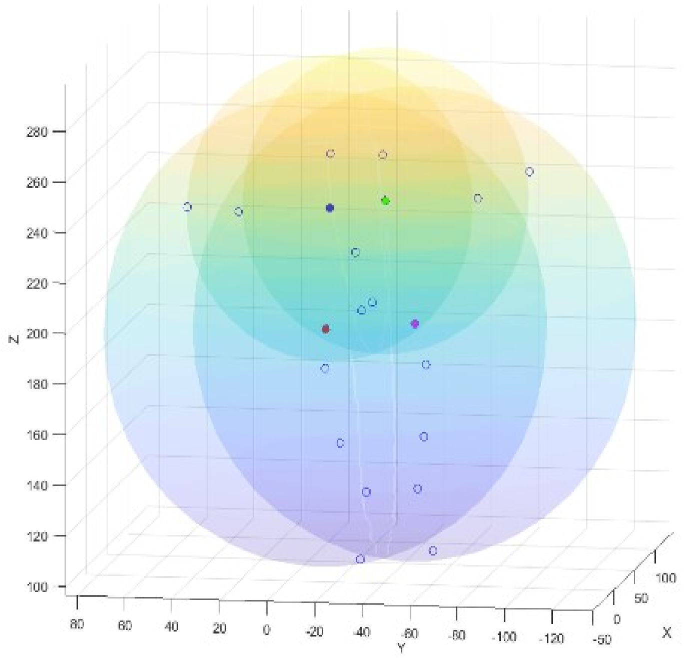


Figure 4-25: Valid Points Maximum ROI

4.3.3.2.3 Lower Limb Markers Detection

After identifying torso points (left shoulder, right shoulder, chest, stomach), we detected lower limb points for the left and right side of the body (hip, knee, calf, ankle). The subsequent sections describe the process.

4.3.3.2.3.1 Hip Markers Detection

The hip points were identified by forming potential triangles using the detected spine and waist points for each side of the body, as shown in Figure 4-26, and based on the constraints in Table 4-6.

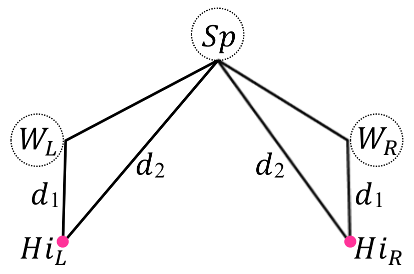


Figure 4-26: Hip Detection Triangle

Table 4-6: Hip Detection Marker set Constraints (M: Marker set)

Constraints	Δ **cm**
$\lvert d_{1mean} - Md_1 \rvert < \Delta d$	3.03
$\lvert d_{2mean} - Md_2 \rvert < \Delta d$	3.03

4.3.3.2.3.2 Knee Markers Detection

The knee points were identified by forming potential triangles using the detected waist and hip points for each side of the body, as shown in Figure 4-27, and according to the constraints in Table 4-7.

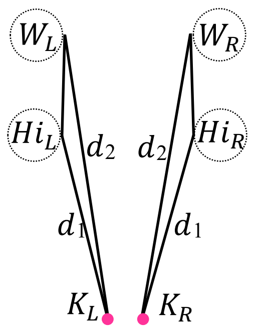


Figure 4-27: Knee Detection Triangle

Table 4-7: Knee Detection Marker set Constraints (M: Marker set)

Constraints	Δ **cm**
$\lvert d_{1mean} - Md_1 \rvert < \Delta d$	5.5
$\lvert d_{2mean} - Md_2 \rvert < \Delta d$	5.5

4.3.3.2.3.3 Calf and Ankle Markers detection

The calf and ankle detection process resembled shoulder point detection. We created potential triangles using two unknown points (calf and ankle) and one detected point (knee), as shown in Figure 4-28. We first identified points with a static length based on the standard deviation (SD) criteria (SD<1), indicating that the SD of distances between all pairs of points was less than 1 (i.e., std(d_1)<1, std(d_2)<1, and std(d_3)<1). Subsequently, we applied the constraints listed in Table 4-8 to the points that satisfied the SD<1 condition.

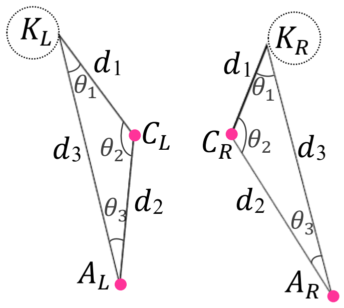


Figure 4-28: Calf and Ankle Detection Triangle

Table 4-8: Calf and Ankle Detection Marker set Constraints (M: Marker set)

Constraints	**Δ *cm*** **L,R**	**Constraints**	**Δ °** **L,R**
$\lvert d_{1mean} - Md_1 \rvert < \Delta d_1$	7.3, 5	$\lvert \theta_{1_{mean}} - M\theta_1 \rvert < \Delta\theta_1$	8, 16
$\lvert d_{2mean} - Md_2 \rvert < \Delta d_2$	7.3, 5	$\lvert \theta_{2_{mean}} - M\theta_2 \rvert < \Delta\theta_2$	13, 35
$\lvert d_{3mean} - Md_3 \rvert < \Delta d_3$	7.3, 5	$\lvert \theta_{3_{mean}} - M\theta_3 \rvert < \Delta\theta_3$	20, 20

To check these constraints, we formed combinations of the points with SD<1 into groups of two probable calf and ankle points (*probPair*). For each side, we generated two sets of Trios by adding the left knee to all double groups for the first set (*K_L, probPair(1), probPair(2)*) and adding the right knee for the second set (*K_R, probPair(1), probPair(2)*). In each Trio, the unknown point with the larger angle was the calf, while the other unknown point was the ankle.

In previous steps, we required three markers to identify triangles. However, we faced an issue where one ankle marker was missing, from some of our datasets, when a swimmer jumped into the water, and it was physically removed, leading to its absence in all frames. To resolve this, we developed a new method for locating calf and ankle points in such cases.

Although we could have removed this assumption earlier, it would have complicated previous steps unnecessarily. Our main goal is to simplify the manual cleaning process.

Hence, we applied constraints on bone lengths and angles from Table 4-8 to identify calf and ankle points in each Trio. If three markers were not present in at least one frame, the corresponding ankle point was considered missing, resulting in an empty set of Trios for that side (left or right). We retained only valid non-empty Trios for either side. Since we had separate groups for the left and right knee, our side detection process was not needed. This process yielded two sets of Trios—one for the right side and one for the left, with the possibility of one being empty. For any empty Trios, we determined calf points based solely on their Euclidean distance from the knee point on the same side.

4.3.3.2.4 Head Markers Detection

The head point was determined by creating potential triangles with the detected left and right shoulder points and the head point for each side of the body, as depicted in Figure 4-29, while following the constraints specified in Table 4-9.

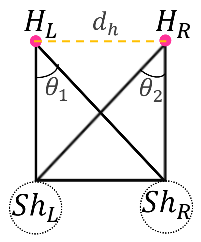


Figure 4-29: Head Detection Triangle

Table 4-9: Head Detection Marker set Constraints (M: Marker set)

Constraints	$\Delta\, cm$	Constraints	$\Delta°$
$\text{std}(d_h) < \Delta d_{h_{std}}$	1	$M\theta_{h_{max}} = \max(M\theta_1, M\theta_2)$ $M\theta_{h_{min}} = \min(M\theta_1, M\theta_2)$ $M\theta_{h_{min}} - \Delta\theta < \theta_h < M\theta_{h_{max}} + \Delta\theta$	19
$\lvert d_{hmean} - Md_h \rvert < \Delta d$	1.3		

We initially identified points that maintained a consistent distance from one another based on the SD criteria (std(d_h)<1). This was due to the limited stretching and compressing of the head's anatomy, which resulted in minimal variability between the left and right head markers. Among these pairs of points, those with distances that fell within a 1.3 cm threshold of the specified marker set value were retained as potential head pairs.

Subsequently, we applied anatomical and action constraints to the triangles since they were not rigid body segments. We calculated the head angle relative to both shoulders using our marker set values. The smaller angle defined the lower limit, while the larger angle defined the upper limit, extending the variation duration. A learned tolerance of 19 degrees was established, which all datasets met. Finally, the side detection was performed based on Euclidean distances of the detected head points with the left and right shoulders, with the shortest distance being on the same side.

Even with high tolerance due to head movement, false detections are rare because other body parts such as the torso, pelvis, and lower limb were identified previously, and the potential elbows and hands have more significant variations in their distances from the shoulders (i.e., they do not meet SD<1 criteria except for passive markers with short lifetimes). Additionally, using precise distance values between two head markers further minimized potential false detections.

4.3.3.2.5 Upper Limb Markers Detection

The remaining points are the elbow and hand points, amid the remaining noise.

4.3.3.2.5.1 Elbow Markers Detection

Elbow points were reliably identified using the bone length between the elbow and shoulder markers on each side as shown in Figure 4-30, by considering the constraints in Table 4-10. Side detection utilized Euclidean distances from shoulder points, with the shortest distance indicating the corresponding side. By applying mean distances and a

SD<1 cm criterion, the likelihood of false detections was minimized, particularly when the hand points were located at the same distance as the length between the shoulder and elbow. However, it is important to acknowledge that passive hand markers with short lifespans and near ghost markers could still satisfy these criteria.

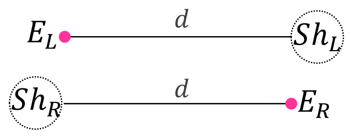


Figure 4-30: Elbow Detection Bone

Table 4-10: Elbow Detection Marker set Constraints

Constraints	$\Delta\, cm$ L, R
$\mathrm{std}(d) < \Delta d_{std}$	1,1
$\lvert d_{mean} - Md \rvert < \Delta d$	11.8, 6.4

4.3.3.2.5.2 Hand Markers Detection

The remaining points were the hand points identified amidst noise, based on the distance between potential hand points and detected elbows as shown in Figure 4-31, following the constraints in Table 4-11. Side detection was applied, ensuring that shorter distances from elbows were maintained on the same side.

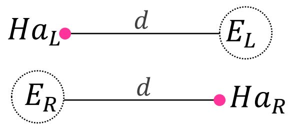


Figure 4-31: Hand Detection Bone

Table 4-11: Hand Detection Marker set Constraints

Constraints	$\Delta\, cm$ L, R
$\mathrm{std}(d) < \Delta d_{std}$	1
$\lvert d_{mean} - Md \rvert < \Delta d$	5, 6.5

4.3.3.2.6 Remaining Noise Removal

After identifying all valid points and labeling them correctly, we removed any remaining unlabeled points, which were considered noise. The final output of this step was a cleaned and labeled C3D file. The evaluation was conducted visually.

4.3.3.2.7 Dropped Ankle Marker Reconstruction

During the analysis of specific datasets, it was noted that there was a missing ankle marker for all frames. In order to resolve this issue, a trajectory for the absent ankle was reconstructed using a semi-supervised method. It was concluded that the missing ankle marker should be situated at the intersection of two spheres based on our understanding of which side may be lacking the marker. The first sphere had its center at the calf, with a radius equal to the distance between the ankle and calf, while the second sphere had its center at the knee, with a radius equal to the distance between the knee and ankle, as shown in Figure 4-32.

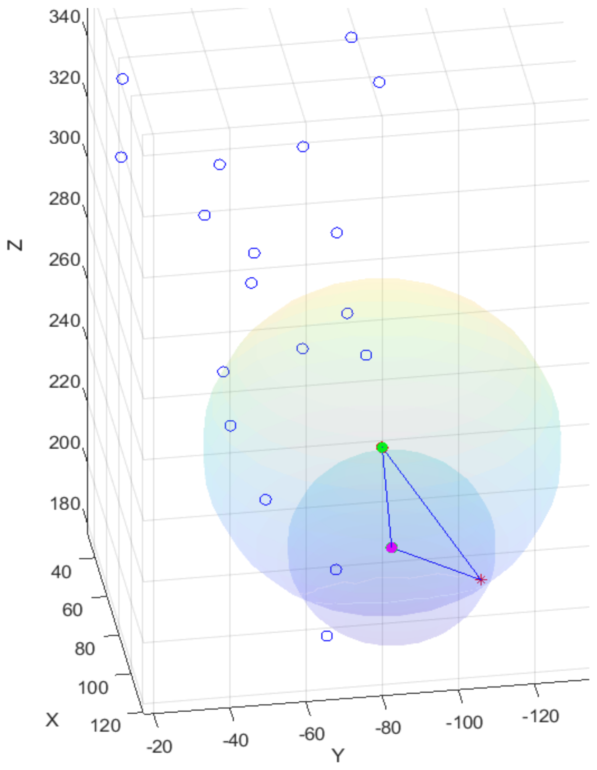


Figure 4-32: Dropped ankle marker reconstruction triangle; Knee (green circle), Calf (violet circle), Ankle (red star: intersection of two spheres).

However, the intersection of these spheres resulted in a circle. As a result, a point on this circle was manually identified as an ankle point (A_1) for the initial frame to create a triangle with the knee and calf that matched the angles and lengths of the marker set. Subsequently, a transformation matrix was computed between the calf point in the first frame and the calf point in the second frame. Ankle (A_1) was then translated from the first frame to the second frame based on this transformation matrix, resulting in the location of A_2 as the ankle point in the second frame. This process was iterated for all frames, with adjustments made manually for variations in body rotation and orientation. Refinements were implemented through manual inspection of the constructed ankle trajectory to identify frames requiring adjustments. Frames that exhibited changes in orientation were identified, and the position of the translated ankle in those frames was refined. This process continued by translating with a new refined ankle point in a specific frame until reaching the next frame requiring refinement. The procedure was repeated for all frames to develop a comprehensive trajectory for the missing ankle marker.

4.4 Experimental Results

Underwater MoCap data were collected using seven Qualisys Miqus M5U underwater MoCap cameras positioned at various points around a four-meter-deep pool at the Memorial University Marine Institute. Reflective passive markers were placed on 21 anatomical locations directly on the swimmer's skin or suit (Figure 4-5). Once calibrated [28], the data were recorded at 100Hz using Qualisys Track Manager (QTM) [4] and then exported to C3D files. Ten C3D files, each containing specific actions, have been exported from a larger, noisy C3D file to serve as our datasets. These files correspond to ten different activities performed by a single swimmer.

4.4.1 Results

In an interactive experimental setting, manual parameters were estimated for each step of the proposed algorithm. With these adjusted tolerances, all ten datasets successfully passed each step's objectives with a 100% success rate. Table 4-12 presents the parameters and tolerances utilized during the pelvis detection step. Table 4-13 showcases the tolerances related to the remaining inlier detection. NA stands for "not applicable."

Table 4-12: Pelvis detection parameters and tolerances

Parameters	Tolerance	Unit
SD (Distances between Markers)	< 0.5	cm
Sides Lengths compared to Marker set	± 2.4	cm
SD (Angles)	< 3.5	º
Spine Angle compared to Marker set	± 6	º
Waists Angle compared to Marker set	± 5	º
Spine Projection Distance on Majority Axis (How Far Spine is from Majority Axis)	± 14.5	cm
Symmetry of Waist points against the Majority Axis	± 8.9	cm
Angle between the Majority Axis and Spine to Waistline-midpoint Line	± 35.4	º
Angle between the Majority Axis and Waistline	90 ± 15.2	º

Table 4-13: Remaining inlier detection tolerances

Table	Algorithms	Parameters	Distance (cm)	Parameters	Angle (º)
4-3	**Shoulder Points Detection**	Shoulder - Spine	±10	Shoulder	± 6
		Shoulder - Shoulder	± 5	Spine	NA
4-4	**Chest Point Detection**	Shoulder - Chest	± 5	Shoulder	± 7
4-5	**Stomach Point Detection**	Stomach - Spine	± 5	Stomach	± 7
		Stomach - Chest	± 5	Chest	± 7
4-6	**Hip Points Detection**	Spine – Hip, Waist - Hip	± 3.03	Hip	NA
4-7	**Knee Points Detection**	Waist – Knee, Hip - Knee	± 5.5	Knee	NA
4-8	**Calf and Ankle Points Detection**	SD (distances)	< 1		NA
		Knee - Calf (L/R)	± 7.3, ± 5	Knee (R/L)	± 8, 16
		Calf - Ankle (L/R)	± 7.3, ± 5	Calf (R/L)	± 13, 35
		Knee - Ankle (L/R)	± 7.3, ± 5	Ankle (R/L)	± 20, 20
4-9	**Head Points Detection**	Shoulder - Head	NA		± 19
		Head - Head STD	<1		NA
4-10	**Elbow Points Detection**	SD (distance)	< 1		NA
		Shoulder - Elbow (L/R)	± 11.8, 6.4		NA
4-11	**Hand Points Detection**	SD (distance)	< 1		NA
		Elbow - Hand (L/R)	± 5, 6.5		NA

Table 4-14 presents the outcomes of each stage of the proposed algorithm across ten datasets. The remaining points and reduction percentage are shown after certain steps to illustrate the algorithm's impact on reducing the number of invalid or reappearing markers, thereby reducing computational load. The significant reduction percentage in *dataset #7* and *#10* after removing NaN points was due to their improperly exported, untrimmed C3D files.

Table 4-14: The results of each step of the algorithm on 10 datasets

Data / **Datasets**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
# Frames	921	1241	536	686	803	897	460	792	803	653
# Original Points	65	62	34	65	51	35	245	30	48	1719
# Original Labels	65	62	34	65	51	35	245	30	48	255
# NaN Points	5	15	0	5	5	4	221	4	10	1667
# Remaining points	60	47	34	60	46	31	24	26	38	52
# Reduction percentage (%)	7.7	75.8	0.0	7.7	9.8	11.4	90.2	13.3	20.8	97.0
# Extraneous markers	0	0	0	0	0	0	4	0	0	5
# Far Outliers	2	3	4	0	10	2	0	0	2	2
# Velocity, Acceleration, Jerk Anomalies	4	0	2	4	3	1	0	0	0	4
# Norm, Vel., Acc., and Jerk Anomalies	12	0	1	3	2	0	0	0	0	13
# Overlapped points	0	1	0	1	0	0	0	0	1	0
# Remaining points After Noise Reduction	42	43	27	52	31	28	20	26	35	28
# Reduction percentage (%)	30.0	8.5	20.6	13.3	32.6	9.7	16.7	0.0	7.9	46.2
# Pelvic Triplets	2	5	2	3	1	3	1	2	4	1
# Shoulder points (L,R)	1,1	1,1	2,2	1,1	2,2	1,1	1,1	1,1	1,1	1,1
# Chest points	1	3	1	1	1	1	1	1	1	1
# Stomach points	1	1	1	1	2	2	1	1	1	1
# Remaining points	40	37	25	50	29	24	20	24	32	28
# Reduction percentage (%)	4.8	13.9	7.4	3.8	6.5	14.3	0.0	7.7	8.6	0.0
# Near Outliers	9	0	1	8	1	0	0	0	0	4
# Remaining points	31	37	24	42	28	24	20	24	32	24
# Reduction percentage (%)	22.5	0.0	4	16	3.4	0.0	0.0	0.0	0.0	14.3
# Hip points (L,R)	1,1	1,2	1,2	2,2	1,1	1,1	1,1	2,2	1,1	1,1
# Knee points (L,R)	1,1	1,2	1,1	1,2	1,1	1,2	1,1	1,1	1,1	1,1
# Calf points (L,R)	1,1	1,1	1,1	2,3	1,2	2,1	2,1	1,1	1,1	1,1
# Ankle points (L,R)	2,0	2,0	1,0	0,4	1,0	1,1	1,1	1,1	1,0	1,0
# Head points (L,R)	1,4	1,3	1,2	4,3	1,2	1,1	1,1	1,1	1,3	1,2
# Elbow points (L,R)	1,1	3,4	2,1	1,2	4,1	1,1	1,1	1,2	3,4	2,1
# Hand points (L,R)	5,1	6,3	1,1	1,7	4,1	1,2	**0,0**	1,1	4,3	4,1
# Remaining points	20	20	20	20	20	21	**19**	21	20	20

Table cells with multiple values separated by a comma (e.g., 4,3 for # Head points in *dataset 4*) represent the number of detected reappearing markers on the left side (4) and right side (3). In *dataset #7*, no hand points were detected due to dropped markers in all frames, resulting in a total of 19 valid markers. For all datasets with a total of 20 remaining points, the right or left ankle marker was not captured across all frames in the dataset, prompting the proposal of a recovery algorithm to reconstruct them (see subsection 4.3.3.2.7). Notably, there were no instances of false labeling or unlabeled markers.

4.4.2 Discussion

We developed an interactive algorithm to streamline the tedious manual cleaning of MoCap Data. The algorithm removes outliers, extraneous, overlapping and ghost markers, focusing on detecting inliers based on the geometric criteria like joint angles and bone lengths using a marker set of 21 markers. Visual evaluation demonstrated a 100% accuracy rate across ten underwater MoCap datasets.

First, we provide an analysis of interesting phenomenon that occurred during the workflow development, specifically considering the pelvis triplets, issues with no hand points detected, and the necessity of the proposed algorithm to overcome challenges with manual cleaning. Subsequently, we will delve into specific considerations of assumptions made during the process and propose potential solutions for enhancing future iterations.

4.4.2.1 Analysis of Pelvis Triplets and Side Detection

In Table 4-14 the number displayed in front of the pelvis detection row for each dataset represents the count of detected triplets for pelvis points. For example, in *dataset #2*, five triplets were detected: [8 9 14], [8 34 9], [8 9 36], [8 38 9], [8 42 38]. In each triplet, the first number denotes a spine marker, which in this case was reliably labeled with ID_8, while the other two numbers represent waist IDs with unknown sides. The proposed algorithm places the spine point as the first index in the triplet due to its greater angle. However, since the pelvis forms an isosceles triangle, initially distinguishing between the other two waist points as left or right was challenging (see subsection 4.3.3.1.4.1).

Firstly, the triplets (we referred them to Trios) were sorted by the proposed algorithm if they contained identical numbers. This means that if the value in index 2 of one Trio is the same as the value in index 3 of another Trio, then it is evident that both should be placed in the same index and side. Therefore, the corrected Trios are as follows: [8 9 14], [8 9 34], [8 9 36], [8 9 38], [8 42 38]. These numbers can vary across different Trios. If a label has the same number in multiple instances, it signifies that the marker was visible throughout, while the other label may have disappeared and reappeared with a new ID.

For example, in *dataset #2*, the spine marker with ID_8 remained continuously visible across all Trios. One of the waist markers with ID_9 maintained its original ID in the first four Trios but disappeared and reappeared with a new ID_{42}, in the last Trio. The other waist marker transitioned through ID_{14}, ID_{34}, ID_{36}, and finally settled on ID_{38}. This sequence suggests that initially, it had an ID_{14} which then disappeared. Upon reappearing, it was assigned ID_{34}, then ID_{36}, and finally settled on ID_{38}. At the final stage of processing, the first pelvis Trio [8 9 14] was retained. The trajectories of ID_{34}, ID_{36}, and ID_{38} are combined with the trajectory of ID_{14}. The same merging process was applied to ID_{42} and ID_9. Any gaps present in all trajectories were linearly interpolated. Consequently, the resulting C3D file contained one spine point with ID_8 and its corresponding complete trajectory, one waist point with ID_9 and its corresponding complete trajectory, and another waist point with ID_{14}

and its corresponding complete trajectory. It is important to note that in the proposed algorithm, distinguishing between left or right is not crucial; only the same side matters. Other IDs such as ID_{34}, ID_{36}, ID_{38}, and ID_{42} are eliminated in the final C3D file, resulting in a reduction of four points from the total number of points.

In this example, the side detection was unnecessary due to identical numbers in Trios. To demonstrate side detection, another example is presented from *dataset #1*, which includes two Trios: [13 16 21] and [13 40 41]. Just like before, index 1 corresponds to the spine point (ID_{13}). Now, we need to conduct side detection to determine if ID_{16} belongs to the same side as ID_{40} or ID_{41}. If ID_{16} is on the same side as ID_{40}, these Trios are confirmed as correct, and the trajectory of ID_{40} should be combined with that of ID_{16}. Similarly, the trajectories of ID_{21} and ID_{41} should be merged. If not, then IDs 16 and 41 will be merged while ID_{21} and ID_{41} would be combined. The correct order of indices for this dataset after side detection confirmed that the first scenario was accurate.

4.4.2.2 Analysis of Dataset 7 with no Hand Points Detected

In *dataset #7*, it was observed that no hand markers were detected. This absence of hand markers signifies that throughout the data capture process across all frames, there were no hands present as the markers were physically dropped. The proposed algorithm did not incorporate a recovery algorithm for this dataset due to the distinct nature of hand movements compared to other body parts. Since specific hand data was not available in the dataset, implementing a recovery algorithm would not have provided meaningful results. This is because without a reference point for hand movements within the dataset itself, any attempt at recovery would lack accuracy and relevance. While it may be possible to manually simulate hand movements based on swimming actions in some instances, such as breaststroke or butterfly stroke, it is not feasible for freestyle movement. Freestyle swimming involves complicated arm motions that are challenging to replicate accurately without actual hand marker data, and any estimation techniques would simply provide synthetic data that likely would not reflect the actual movements.

4.4.2.3 Evaluating the Importance of Implementing This Algorithm

Manual cleaning is a time-consuming and tedious process, especially in complex tasks. Specifically, the reappearing passive markers further complicate the situation, as their movement behavior can resemble noise if their lifespan is short.

Additionally, during manual cleaning, changing perspectives is crucial for accurately detecting points, as distances can be misleading. For instance, in Figure 4-33, the red points in image (a) appear close to valid markers, but in image (b), they are shown to be far away. Similarly, the yellow points in image (c) seem to overlap, yet a different angle in image (d) reveals they are also distant from valid markers.

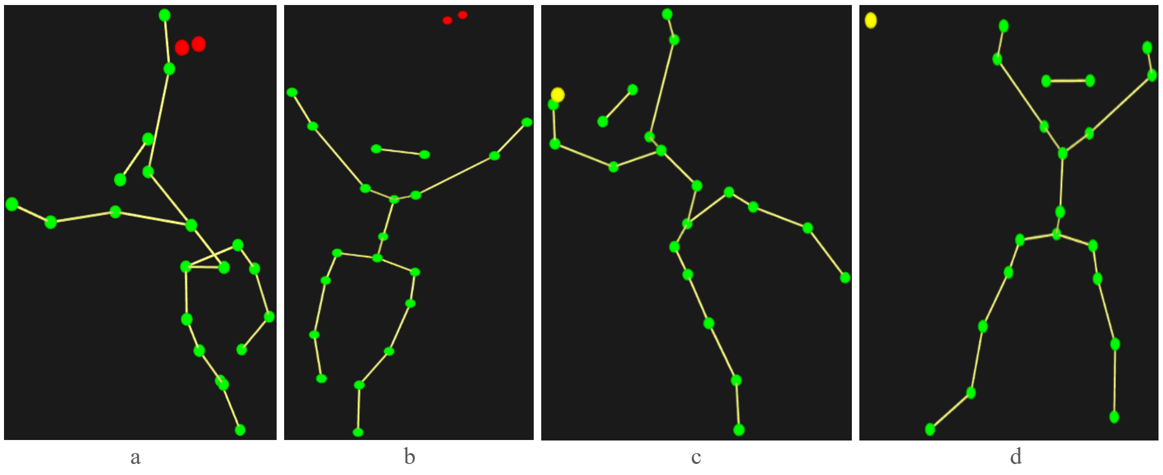


Figure 4-33: The effect of changing the view in Manual Cleaning: Real far distances (b, d) seem near (a, c) from a different perspective.

In manual cleaning processes, it is necessary to meticulously examine hundreds of potential points over thousands of frames. Particularly in cases of complex movements, proximity to outliers, and data gaps, detecting anomalies especially ghost markers that are very near to valid markers can be challenging, even for a skilled human.

Taking tolerances into account increases the likelihood of misclassifying nearby ghost markers as valid markers. However, this method has the advantage of accurately identifying the frame and label number associated with any abnormalities, facilitating quick visual verification. Moreover, the intersection check process (Algorithm 2, line 7) triggered an alarm for visual verification when two points were identified as one physical marker in a single frame, consisting of one ghost marker and one valid marker.

Depending on the specific marker set used, this process can be automated. Nonetheless, due to the inherent challenges presented by our sparse underwater passive marker dataset and the presence of ghost markers, a semi-automatic approach that combines automatic identification with manual verification proved to be more reliable.

4.4.2.4 Consideration of Assumptions and Future Solutions

We had varying assumptions throughout our algorithm. In the extraneous marker removal phase, valid markers should not be near extraneous ones. For example, in our datasets, swimmers did not swim at the pool's bottom where extraneous markers were located (i.e., the system's calibration fixture). If extraneous markers were not removed initially, they would be addressed later, increasing computational load.

In far outlier detection we removed anomalies with fill levels less than 1% due to potential passive markers meeting the criteria. This step was used to reduce the computational load, as these outliers could be detected during geometry-based inlier detection due to their far distance from valid markers. In the future, we can calculate the mean velocity, acceleration, and jerk over the lifespan of the identified anomalies.

In pelvis detection using the majority axis, we utilized a frame with a minimum point and made assumptions about the alignment of the pelvis triangle relative to the majority axis. While our datasets aligned with these assumptions, further testing with various actions is necessary. Additionally, for pelvis side detection, we assumed there would be no significant variation in small gaps, which may not hold true. As a future endeavor, we should focus on developing an automatic robust solution and explore whether it can be based on the direction of movement, the cross vector, and other criteria. The initial attempt to detect sides based on these criteria has not yet yielded results.

In the inlier detection algorithm, a rigid body assumption was made by incorporating user-defined tolerances to demonstrate deviations from rigidity concerning the angles and distances within the marker set. This assumption was correct as long as there was no stretching or compressing of the body. However, using mean distances and angles helps mitigate this issue even in those situations. Moreover, certain segments exhibit considerable deviations from the rigid body assumption (e.g., triangle formed by head and shoulders). We addressed this using action constraints from our datasets..

To identify most of our body parts, we assumed that any three points could form a triangle, necessitating the presence of all three points simultaneously in at least one frame. To address this assumption in future studies, we plan to pinpoint potential body part locations by considering marker set distances. If these points fail to create a triangle, we will then proceed to handle these potential points after identifying all body parts, similar to our approach for dropped ankle reconstruction. We could remove this assumption in our current algorithm but because all our datasets met this assumption, we did not do that because such actions would have unnecessarily complicated these procedures, which was not our intention. Our primary objective is to streamline the manual cleaning process.

Using interpolation for gap filling can affect angles, distances, and other aspects. Exploring alternative recovery approaches is crucial to achieve more natural gap filling, especially with large gaps. However, considering the mean distances reduced the impact.

We visually evaluated the C3D of each step to ensure accuracy. It was essential to maintain consistent tolerances across all datasets. Therefore, if we encountered a situation where, for example, certain body points reappeared in the third dataset but were not detected, we adjusted the tolerances and reviewed the previous dataset to prevent false detections or missed points. While starting with high initial tolerances may mitigate this issue due to varying criteria, it is challenging in conditions with near ghost markers.

Overall, assumptions based on domain knowledge, such as specific action constraints, are fundamental in feature-based approaches. Many MoCap solving methods did not consider ghost markers, and deep learning approaches may struggle with unseen complex actions that differ significantly from the training set. Since there are no underwater MoCap data available, and freestyle actions in this environment are unique and unpredictable, manual cleaning is necessary in our dataset. The purpose of this algorithm was facilitating laborious manual cleaning, especially for complex actions with numerous reappearing markers, where assumptions and interactive approaches prove to be effective in noisy and complicated scenarios where manual cleaning is not feasible.

4.5 Conclusion

We have tackled the issue of labeling raw MoCap data, particularly using underwater sparse marker sets and freestyle underwater motion. These data are susceptible to various types of noise, including outliers, extraneous and ghost markers, as well as missing markers caused by occlusions, reduced visibility under water, and dropped markers due to water resistance. Moreover, handling the reappearing valid markers caused by using passive markers results in a high number of points with small trajectory segments instead of a certain number of valid markers with corresponding trajectories and possible gaps.

This interactive approach aims to facilitate the tedious and error-prone manual cleaning process of creating training and ground truth datasets for machine learning and deep learning algorithms. Additionally, it can be utilized in the alignment, evaluation, or correction phase of these systems. Furthermore, it serves as a standalone solution for cleaning and labeling smaller datasets that do not warrant the use of advanced algorithms.

We have addressed this issue by implementing a range of innovations, such as extraneous removal based on anomaly detection in the difference of sorted norms profile. Additionally, we have developed methods for detecting abnormalities in norm, velocity, acceleration, and jerk profiles. Furthermore, we have established a process for identifying inliers using a geometry-based approach that considers marker set values like lengths and angles. The algorithm’s central feature is PCA based pelvis detection, which can also be utilized in the alignment step of the other MoCap solving systems. The evaluation was conducted visually, showcasing a 100% accurate detection of valid markers for our ten captured underwater MoCap datasets containing 21 valid markers.

References

[1] G. B. Guerra-filho, “Optical motion capture: Theory and implementation,” *J. Theor. Appl. Informatics*, vol. 12, pp. 61--89, 2005, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.7248

[2] M. Menolotto, D. S. Komaris, S. Tedesco, B. O’flynn, and M. Walsh, “Motion capture technology in industrial applications: A systematic review,” *Sensors (Switzerland)*, vol. 20, no. 19, pp. 1–25, 2020, doi: 10.3390/s20195687.

[3] Qualisys, “Cameras for underwater motion capture,” *Qualisys*. https://www.qualisys.com/cameras/underwater/ (accessed Jul. 15, 2024).

[4] Qualisys, “Qualisys Track Manager,” *Qualisys*, 2011. https://www.qualisys.com/software/qualisys-track-manager/ (accessed Dec. 10, 2023).

[5] M. Kitagawa and B. Windsor, *MoCap for Artists Workflow and Techniques for Motion Capture*, no. 0. Elsevier Inc, 2008.

[6] J. Yang, T. Li, Z. Chen, and X. Li, “Research on the Method of Underwater Swimming Motion Capture,” *J. Phys. Conf. Ser.*, vol. 1982, no. 1, pp. 1–4, 2021, doi: 10.1088/1742-6596/1982/1/012075.

[7] K. Chen, Y. Wang, S. H. Zhang, S. Z. Xu, W. Zhang, and S. M. Hu, “MoCap-solver: A neural solver for optical motion capture data,” *ACM Trans. Graph.*, vol. 40, no. 4, 2021, doi: 10.1145/3450626.3459681.

[8] N. Ghorbani and M. J. Black, “SOMA: Solving Optical Marker-Based MoCap Automatically,” *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 11097–11106, 2021, doi:

10.1109/ICCV48922.2021.01093.

[9] G. Ascenso, “Development of a non-invasive motion capture system for swimming biomechanics,” 2021.

[10] M. A. Hidayat Yani, S. Bayu Aji, I. F. Ariyanti, S. Sukaridhoto, M. A. Zainuddin, and A. Basuki, “Implementation of Motion Capture System for Swimmer Athlete Monitoring,” *IES 2019 - Int. Electron. Symp. Role Techno-Intelligence Creat. an Open Energy Syst. Towar. Energy Democr. Proc.*, pp. 400–405, 2019, doi: 10.1109/ELECSYM.2019.8901554.

[11] E. Martini, S. Member, A. Calanca, and N. Bombieri, “Denoising and Completion Filters for Human Motion Software : a Survey with Code,” pp. 0–14, 2023, doi: 10.36227/techrxiv.22956482.v1.

[12] A. L. Clouthier, G. B. Ross, M. P. Mavor, I. Coll, A. Boyle, and R. B. Graham, “Development and Validation of a Deep Learning Algorithm and Open-Source Platform for the Automatic Labelling of Motion Capture Markers,” *IEEE Access*, vol. 9, pp. 36444–36454, 2021, doi: 10.1109/ACCESS.2021.3062748.

[13] Qualisys, “Using AIM models.” https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/using_aim_models/using_aim_models.htm (accessed Aug. 01, 2024).

[14] Biomechanical-toolkit.github.io, “Mokka,” *Biomechanical-toolkit.github.io*. https://biomechanical-toolkit.github.io/mokka/ (accessed Jan. 06, 2024).

[15] Qualisys, “Swimming Technique: dual media motion capture,” *Qualisys*. https://qfl.qualisys.com/#!/project/swimming-techniques (accessed Dec. 02, 2023).

[16] Qualisys, “Qualisys Sports Marker Set,” *Qualisys*. https://cdn-

content.qualisys.com/2022/07/Sports-Marker-Set.pdf (accessed Dec. 05, 2023).

[17] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. Black, "AMASS: Archive of motion capture as surface shapes," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-Octob, pp. 5441–5450, 2019, doi: 10.1109/ICCV.2019.00554.

[18] Wikipedia, "Principal component analysis." https://en.wikipedia.org/wiki/Principal_component_analysis

[19] B. Motion, "The C3D File Format A Technical User Guide," p. 134, 2021.

[20] Qualisys, "Exporting files to C3D." https://docs.qualisys.com/getting-started/content/13_how_to_visualize_data_in_visual3d/exporting_files_to_c3d.htm?Highlight=export (accessed Aug. 01, 2024).

[21] Qualisys, "What is a trajectory." https://www.qualisys.com/my/qacademy/#!/tutorials/what-is-a-trajectory (accessed Jun. 30, 2024).

[22] Qualisys, *Qualisys Track Manager User Manual*, 2022.1. 2022. [Online]. Available: https://cdn-content.qualisys.com/2022/07/QTM-user-manual.pdf

[23] Github, "ezc3d." https://github.com/pyomeca/ezc3d

[24] Qualisys, "Super-spherical mocap markers." https://www.qualisys.com/accessories/markers/super-spherical-markers/ (accessed Jul. 28, 2024).

[25] Wikipedia, "Standard deviation." https://en.wikipedia.org/wiki/Standard_deviation (accessed Aug. 17, 2024).

[26] X. Pan *et al.*, "A Locality-based Neural Solver for Optical Motion Capture," 2023, doi: 10.1145/3610548.3618148.

[27] Wikipedia, “Collinearity.” https://en.wikipedia.org/wiki/Collinearity (accessed Sep. 05, 2024).

[28] Qualisys, “Calibrating your system,” *Qualisys*. https://docs.qualisys.com/getting-started/content/getting_started/running_your_qualisys_system/calibrating_your_system/calibrating_your_system.htm (accessed Jan. 06, 2024).

5. Deep Learning based Auto-Labelling for Underwater Sparse Freestyle MoCap Data

Abstract

This paper discusses auto-labeling for sparse freestyle underwater optical motion capture (MoCap) data using 21 passive markers, recorded by Qualisys Miqus M5U Mocap Cameras. MoCap data contains noise such as outliers, ghost markers, and occlusion, which are exacerbated by water's unique properties. The algorithm aims to train a Long Short-Term Memory (LSTM) network with various inputs to assess the impact of feature selection, training set size, and noise on accuracy. The process involves augmenting the training set with random noise and gaps, training an LSTM model with 3D position inputs, then with positions, velocity, and acceleration as five inputs, and using transfer learning with simulated trajectories to expand the training set. Labels are assigned using the Hungarian algorithm, Procrustes analysis locates unlabeled markers, and an OpenSim marker set post-processing corrects mislabeled markers. A semi-supervised geometry-based labeling method establishes ground truth and training sets. PCA-based pelvis detection aids in data alignment, and an extraneous marker removal algorithm boosts LSTM performance from 66% to 98%. The semi-supervised algorithm achieved 100% on our 10 datasets. Overall, the auto-labelling algorithm streamlines the MoCap manual cleaning.

5.1 Introduction

Marker-based optical motion capture (OMC) [1] is a technique used in various industries [2], such as computer vision, biomechanics, entertainment, sports analysis, medical research, and robotics, to accurately track and record the three-dimensional (3D) motion of humans. This method involves attaching markers to the subject's body while cameras capture their positions in real-time. Specialized software (e.g., Qualisys [3]) processes this MoCap data to determine the 3D coordinates of each marker by triangulating their locations from multiple camera views.

However, the raw Mocap data contain errors due to calibration issues, noisy environments (e.g., reflective surfaces), and occlusion, resulting in noise, outliers, ghost markers, mislabeling, and gaps. Extraneous markers, which are real markers from other objects, may also be present. The underwater environment poses greater challenges due to surface reflections and water's unique properties, which amplify noise [4]. For example, reduced visibility underwater increases the possibility of occlusion. This poses a greater challenge for passive systems, as occluded passive markers typically receive new random IDs upon reappearance, resulting in multiple short trajectories for a single marker. In contrast, active markers retain their ID during occlusion, allowing for continuous trajectories with potential gaps. Therefore, passive markers can be resembled as noise throughout their lifespan. We refer to instances of a passive marker as "reappearing" markers.

The mentioned challenges complicates manual cleaning and labeling MoCap data. Automatic labeling functions in commercial software (e.g., Qualisys AIM model;

Automatic Identification of Markers [5]) can expedite this process but is still time-consuming, expensive, and requires manual intervention and cleaned training data. Consequently, researchers have sought automatic cleaning methods. Early techniques utilized rigid body and action constraints based on user-defined tolerances, limiting their generalizability across different actions [6]. Recently, there has been a shift towards deep learning methods to improve this process [7]. However, the lack of labeled MoCap data for training these models is a major hurdle; for example, the largest dataset, AMASS [8], is much smaller than video datasets in other fields. Despite these challenges, deep learning-based approaches promise better generalization with reduced user intervention.

The challenge of using deep learning-based auto-labeling methods for underwater swimming actions in the absence of a Mocap dataset, particularly those involving intricate and freestyle movements, is significant. A limited number of existing datasets are only partially underwater, not comprehensive, and sparse [9]. Therefore, we captured underwater MoCap data using Qualisys system as our primary objective and contribution.

The auto-labeling algorithm in this study employs a Long Short-Term Memory (LSTM) [10] network to generate a vector of probabilities, which are then assigned labels using the Hungarian algorithm [11]. A semi-supervised geometry-based algorithm, as detailed in Chapter 4 of this thesis, is utilized to establish ground truth and serves as a standalone labeling algorithm for comparison with the LSTM method. A pelvis detection algorithm utilizing Principal Component Analysis (PCA) [12] is used to align data. Additionally, an extraneous marker removal algorithm is proposed to enhance the results of LSTM auto-labeling. Post-processing identifies and correct mislabeled data using an OpenSim [13] marker set, and the Procrustes algorithm [14] assigns labels to unlabeled data. Transfer learning [15] is used to expand the training set using simulated trajectories, and due to the small dataset size, it is augmented using random noise and gaps.

The paper is organized as follows: Section 5.2 discusses the former works on auto-labeling in optical Mocap systems. Section 5.3 describes the proposed method with its rationale and design considerations. Section 5.4 presents experimental results. Section 5.5 summarizes the article.

5.2 Related Work

Early auto-labeling methods (e.g., moving average filters and low-rank matrix [16], unscented Kalman filter and inverse kinematics [17], automatic kinematic model building based on Markov random field [18], and skeleton-based body models [6], [19]–[21]), mainly relied on empirical parameters and hand-crafted features. Although these approaches could produce acceptable outcomes for specific patterns and noise under assumptions and constraints, they consistently faced difficulties adapting to real-world data with intricate situations.

Data-driven methods have been employed to address the limitations above by learning from a large database, such as kd-tree [22], local PCA [23], self-similarity [24], sparse encoding [25], [26], graph matching [27], [28], and deep learning-based approaches [29]–[32]. We provided a thorough literature review on addressing MoCap data solving includes denoising, recovery, alignment, and auto-labelling in Chapter 2 of this thesis.

We utilize the source Python code of the deep learning-based auto-labeling approach presented in [33]. However, we enhance their algorithms in several ways. One significant enhancement is proposing an automatic pelvis detection method for data alignment instead of their manual approach. Additionally, they reported low accuracy due to extraneous markers, which we address by proposing an extraneous markers removal algorithm. This improvement significantly enhances the accuracy from 66% to 98%. Furthermore, we input three data (X, Y and Z position) into LSTM instead of their five inputs (X, Y, Z, velocity and acceleration) and conduct comparisons. A semi-supervised geometry-based algorithm is proposed to establish ground truth and functions as an independent labeling method for comparison with the LSTM approach.

5.3 Methodology

The overview of our system is shown in Figure 5-1. It consists of 11 main steps: capturing C3D files, creating an OpenSim marker set, ground truth creation using semi-supervised geometry-based labeling algorithm, preprocessing, PCA-based pelvis detection for alignment, training set creation using augmentation and simulated trajectories, training LSTM using three and five inputs [33], labelling test data using the Hungarian algorithm, post-processing using Procrustes, and accuracy calculation. They will be described in the following sections.

The semi-supervised geometry-based labeling algorithm, detailed in Chapter 4 of this thesis, was proposed to clean and label datasets. This process generates ground truth and training datasets for the LSTM-based auto-labeling method presented in this article. This algorithm relies on the geometric characteristics of the human body considering user defined tolerances based on rigid body assumptions and actions' constraints. This algorithm as shown in Figure 5-1, read C3D file using MATLAB. The steps of this algorithm are preprocessing including invalid label removal, extraneous marker removal, far outlier removal, PCA-based pelvis detection, torso detection (shoulders, chest, and stomach points), near outlier removal, limb (hips, knees, calves, ankles, elbows, and hands) and head detection, remaining noise removal, and dropped marker reconstruction. We will not cover the details of this algorithm in this article.

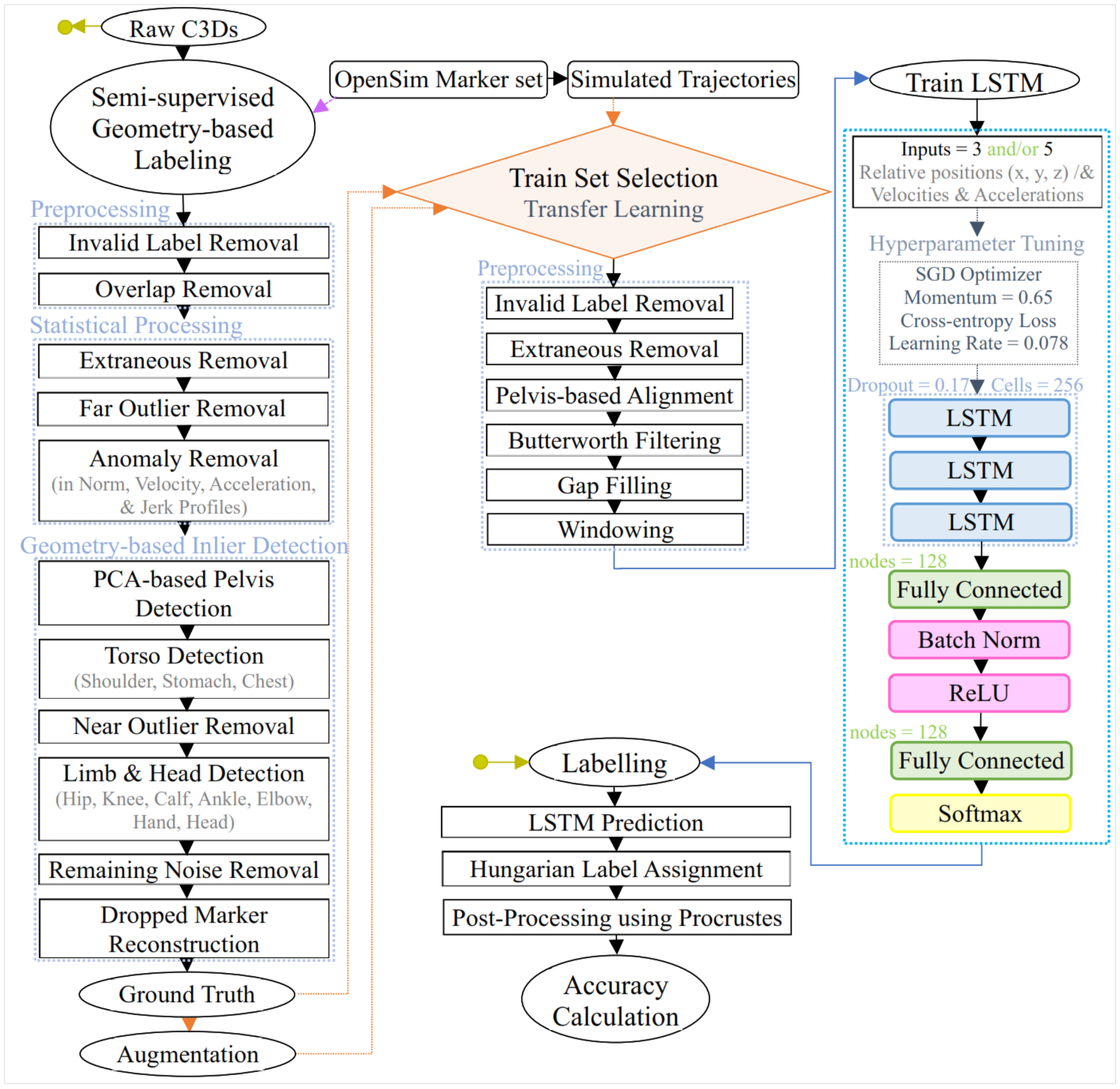


Figure 5-1: System Overview

5.3.1 Capturing MoCap C3D Data

Underwater MoCap data were captured using seven Qualisys Miqus M5U underwater MoCap cameras [34] installed at different locations around a four-meter-deep pool at the Memorial University Marine Institute. Passive markers were placed on 21 anatomical locations directly on the swimmer's skin or suit. After calibration [35], data were recorded at 100Hz using Qualisys Track Manager (QTM) [36] and exported to C3D [37], [38] files.

A MoCap C3D file consists of a defined number of frames and points, which can be categorized into valid markers and noise. These markers are tracked across all frames, and their trajectories are represented by their 3D coordinates *x, y*, and *z* over time. To process the data, we utilized the EZC3D library [39] in MATLAB and Python to read C3D files.

5.3.2 Open Sim Model Marker Set

OpenSim 4.4 software [13] was used to create the marker set based on the musculoskeletal model [40]. Our marker set consisted of 21 passive markers attached to our swimmer’s suit and body, as shown in Figure 5-2. These markers were placed on an OpenSim Simbody based on our marker set. The markers’ local coordinates are defined in a “MarkerSet.xml” file.

Furthermore, triangle-based calculations, which include angles and bone lengths, were derived from a “T-pose” skeleton. The markers were attached at the same locations as those on a Simbody. These measurements served as inputs for a semi-supervised geometry-based algorithm, along with the post-processing steps of the LSTM-based auto-labeling approach described in this article.

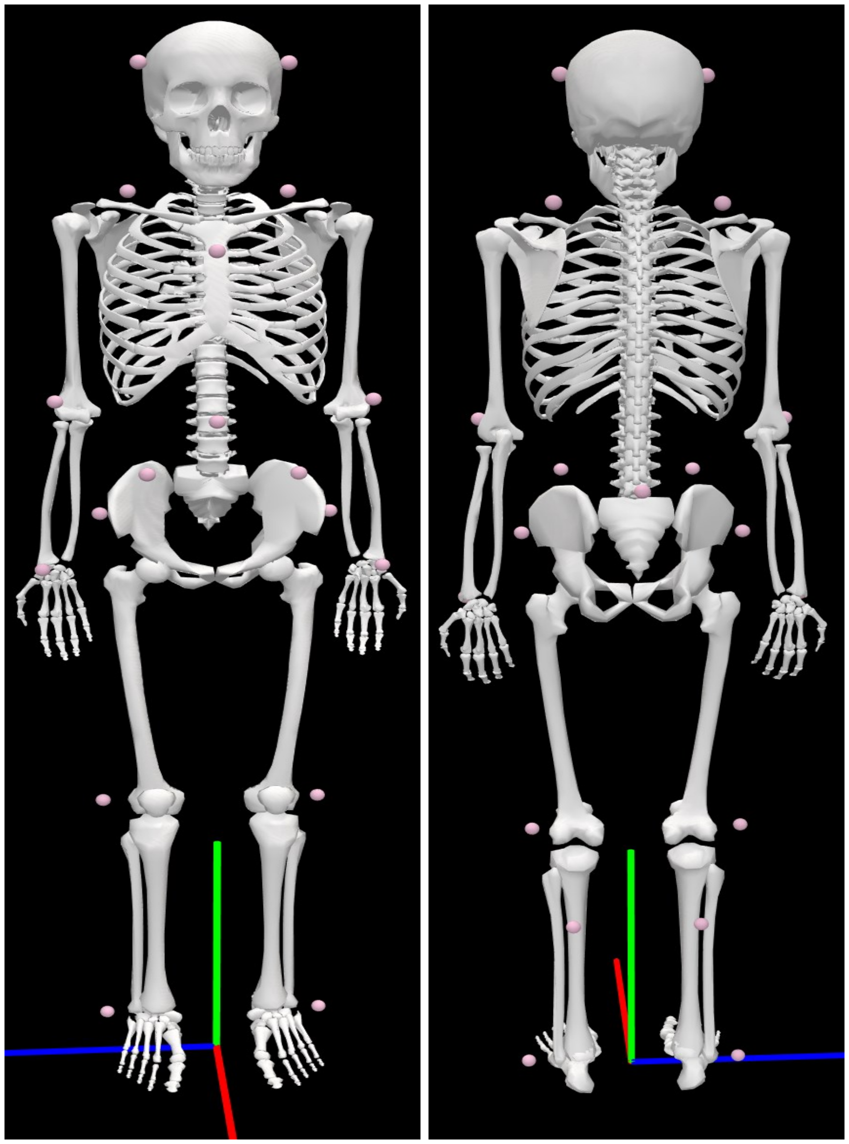


Figure 5-2: OpenSim Marker Set; locations of 21 passive markers (pink orbs)

5.3.3 Simulated Trajectories

The simulated trajectories were generated based on our marker set using the kinematics of 100 participants from “bodykinematics.hdf5” [33]. Right waist and left waist markers were used to align the subject to face in the positive *x*-direction (+*x*). The sampling frequency of data was 100 Hz. The generation of simulated data addresses the issue of insufficient data for training deep learning models. This body kinematics contained actions such as running, walking, and others described in [33], which are completely different from our underwater freestyle movements.

5.3.4 Training Data Sets Preparation

The raw C3D files captured contained outliers, extraneous markers, ghost markers, and gaps or reappearing markers due to occlusion. Denoising, recovering missing markers, and labeling these MoCap data could be manually performed using free software such as Mokka [41] or commercial software such as QTM. To avoid the tedious manual cleaning process, we employed our semi-supervised geometry-based labeling algorithm to clean and label the datasets, creating the ground truth or training sets.

5.3.4.1 Transfer Learning

Transfer learning was utilized to augment the training set of the network by incorporating the accurately labeled outputs from the auto-labeling algorithm. This allowed us to enhance the training set by adding the trained model using simulated trajectories or by adding new labeled C3D files to a previously trained model. This approach improved the network's performance and accuracy.

5.3.4.2 Augmentation

Because our underwater datasets were small and the created simulated trajectories contained actions that were very different from our freestyle underwater movements, we augmented our datasets by adding random noise and gaps. However, we did not incorporate the simulation of reappearing markers.

5.3.5 Pre-processing Data

There were several steps to prepare input data for a neural network, including invalid label removal, extraneous marker removal, pelvis detection for alignment, Butterworth filtering, gap filling, and data windowing, which are described in the following sections. We compared the results with and without this enhancement to demonstrate the significant improvement in accuracy achieved through our contribution.

5.3.5.1 Invalid Label Removal

Removing labels that hold no information in all frames reduced the computational load. This happens when a C3D file is a subset exported from a larger C3D file without trimming.

5.3.5.2 Extraneous Removal

This algorithm used clustering of norm differences to identify and remove extraneous markers that were visible in all frames and located far from valid markers. This method involved calculating the Euclidean norm of each point $p = (x, y, z)$ in every frame relative to the origin $O = (0,0,0)$ as $\|p\| = \sqrt{(p \cdot p)} = \sqrt{(x^2 + y^2 + z^2)}$. Then, the norms and their corresponding labels were sorted, and the difference between each norm value and the preceding norm value was computed. To remove anomalies, we set a threshold of three standard deviations (SD) from the mean (e.g., using "isoutlier()" function in MATLAB). This method improved the low accuracy, due to extraneous markers, noted in [33].

5.3.5.3 Pelvis-based Alignment

The marker coordinates were rotated around the vertical axis so that the swimmer faced the positive x-direction at the beginning of the trial, to prepare the data for input into the neural network. We used PCA-based pelvis detection, as detailed in Chapter 4 of this thesis and will be summarized here, to automatically align data in the dataset based on the right and left waist points. This proposed method resolved the issue of manual alignment of test data described in [33].

5.3.5.3.1 PCA-based Pelvis Detection

First, we identified probable pelvis points (i.e., reappearing passive marker IDs) by assuming that the spine, right waist, and left waist markers appeared simultaneously in at least one frame to form a triangle, as illustrated in Figure 5-3.

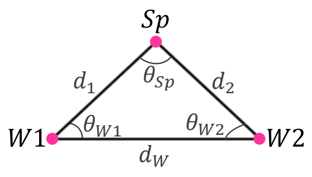


Figure 5-3: Pelvis Triangle

Table 5-1: Pelvis Detection Marker set Constraints (M: Marker set)

Constraints	**Constraints**
$\lvert d_1 - Md_1 \rvert < \Delta d_M$	$\lvert \theta_{Sp} - M\theta_{Sp} \rvert < \Delta\theta_{MSp}$
$\lvert d_2 - Md_2 \rvert < \Delta d_M$	$\lvert \theta_{W1} - M\theta_{W1} \rvert < \Delta\theta_{MW}$
$\lvert d_W - Md_W \rvert < \Delta d_M$	$\lvert \theta_{W2} - M\theta_{W2} \rvert < \Delta\theta_{MW}$

We detected probable pelvis points in each frame by applying geometric constraints based on rigid body assumptions. First, we selected pairs of points with a static distance (i.e., SD(distance) < 0.5 cm). Next, we formed all combinations of three points to create triangles and evaluated their distances and angles against specific criteria. Then, we refined the triangles that met these criteria by assessing their orientation. We determined body orientation using the first PCA component as the majority axis. We assumed that the right and left waist points should be symmetrically aligned with this axis, while the spine point should be close to both the centroid of the points and this axis. Finally, we accurately identified the correct pelvis points by comparing distances and angles with precise values from our marker set, as shown in Table 5-1. We used a side detection algorithm to differentiate between the right and left waist points. Next, we combined the short trajectories for each pelvis point into a single trajectory and applied linear interpolation to fill in any gaps.

5.3.5.4 Butterworth Smoothing

Trajectories were smoothed using a zero-phase second-order 6Hz Butterworth filter, akin to [33]. This smoothing process occurred after our pelvis detection step since it could impact angle values crucial for the pelvis detection algorithm. However, using mean angles helped reduce the influence of smoothing.

5.3.5.5 Gap Filling

The small gaps, defined as gaps 10 frames or less, were filled using linear interpolation. As is common in passive marker systems, we observed short trajectories for each marker, rather than complete trajectories with potential gaps as seen in active systems. However, accommodating a general solution for datasets with active or passive markers and partially processed data, we filled any small gaps.

5.3.5.6 Data Windowing

Windowing marker trajectories before inputting them into an LSTM network enhances the network's ability to capture temporal dynamics, extract relevant features, handle variable-length sequences, improve training efficiency, and enhance prediction accuracy.

For the test and validation data, windows of 100 frames were chosen, if the total number of frames was divisible by 100. The last window would be smaller for fewer frames. If there were 10 or fewer frames left, these frames were added to the previous window. For example, for 202 total frames, we created two windows with a size of 100 each. The remaining 2 frames, being less than 10, were added to the second window. In the training data, a random length between 10 and 100 frames was considered so that the neural network could encounter windows of various sizes to prepare for the different-sized final window of each marker. The detail of the windowing procedure was described in [33].

5.3.5.7 Neural Network Input Data

We used three inputs (x, y, and z relative location of each marker) and suggested five inputs [33] (x, y, z, velocity and acceleration of each marker) separately as inputs into the neural network to compare them. The x, y, and z coordinates of the retained markers in relation to the current marker were computed by iterating through all markers. Subsequently, the trajectories were arranged based on their mean distance from the current marker. The Euclidean norm of the velocity and acceleration of the retained markers concerning the current marker were computed. The relative positions, velocities, and accelerations were normalized by the mean values observed across all markers in the training dataset. A matrix of dimensions $(window\ size) \times (5\ (num\_labels - 1))$ was created, which included the positions (coordinates x, y, z), velocities, and accelerations of the retained markers relative to the current marker. This matrix served as input to the neural network. A matrix of dimensions $(window\ size) \times (3\ (num\_labels - 1))$ was used for 3-input.

5.3.6 Long short-term memory (LSTM)

An LSTM network was used to calculate label probabilities for each window. It is a type of recurrent neural network (RNN) specifically designed to overcome the vanishing gradient problem found in traditional RNNs. One of its main advantages over other RNN architectures and sequence learning techniques is its effectiveness in handling varying lengths of temporal gaps in the data. This feature allows LSTM networks to significantly improve the capture of long-sequence dependencies by utilizing a short-term memory mechanism that retains information over thousands of time steps.

The suggested network [33] comprised a recurrent layer (LSTM) with a 10% dropout rate and 128 cells, a fully connected layer with 128 nodes, one-dimensional batch normalization, a rectified linear unit (ReLU), another fully connected layer, and a softmax function. The network was implemented using a stochastic gradient descent (SGD) optimizer with momentum to train the neural network using a cross-entropy loss criterion. The algorithm underwent training for 10 epochs on the training sets and was subsequently tested on the data set. The training and testing processes were executed on an 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz processor with 16.0 GB of RAM.

Hyperparameter tuning was conducted in [33] to determine the optimal settings for various parameters in a neural network model. These parameters included the number of LSTM layers, the number of LSTM cells, the dropout percentage for LSTM layers, the number of nodes in fully connected layers, and the momentum and learning rate of the optimizer. The process of finding the best combination of these hyperparameters was achieved through Bayesian hyperparameter optimization using the Ax Platform. In this optimization process, the micro-averaged F_1 score was utilized as the metric for evaluation.

The micro-averaged F_1 score combines precision and recall into a single metric and serves as an indicator of classification accuracy. The best results were obtained with three LSTM layers, 256 LSTM cells, a dropout rate of 0.17, 128 nodes in fully connected layers, a learning rate set at 0.078, and a momentum value of 0.65.

The neural network produced a $1 \times num\_labels$ vector for each window of a marker, indicating the probabilities of each label being correct. New windows were created by dividing the trial marker data at frames where any marker appeared or disappeared. This ensures that each marker label appeared only once in each window, which was essential because a single marker label may be spread across multiple trajectories, and this guarantee is not provided with randomly segmented windows.

5.3.7 Hungarian based Label Assignment

Formulating the assignment of labels involved solving an unbalanced assignment problem on a weighted bipartite graph. Consequently, the optimal marker labels for each window were determined using the Hungarian algorithm. Using the weighted mode involved assigning a single label to the entire trajectory. The predicted labels for each frame were weighted based on the probability of the prediction. The issue of assigning two markers to one label was solved by keeping the marker with the highest probability [33].

5.3.8 Post-processing

The OpenSim marker set was utilized to determine the precise local coordinates of each marker in relation to its corresponding body segment. This process helped identify inaccurately labeled markers by assuming a rigid body model. If the distances between a marker and other markers within the same segment deviated more than three SD from the distances in the training dataset specific to that marker, the assigned label was removed. Assigning labels to unlabeled markers was determined by calculating the mean probabilities and comparing the distance to other markers in a segment with three SD from those observed in the training set. If the distances fell outside this range for all available labels, the marker remained unlabeled [33].

5.3.8.1 Procrustes Analysis

Procrustes Analysis [14] is a statistical method for analyzing the distribution of a set of shapes. It was used [33] to align the local marker coordinates in the marker set with the measured markers based on scaling, rotation, and translation to assign labels for body segments with at least three markers, but with one or more remaining unlabeled. Locating unlabeled markers was achieved by ensuring that all distances between the aligned marker set and the measured markers were below a specified threshold. If an unlabeled marker fell within a second threshold of the expected position based on the aligned marker set coordinates, it was then assigned the missing label. The manual adjustment of thresholds was based on the spacing of markers within the respective marker set.

5.3.9 Evaluation Metrics

We compared labelled data with ground truth with different metrics. We reported average of per-frame accuracy, precision, recall, and F_1 score in percentages. We used a confusion matrix which is a table that is often used to describe the performance of a classification model on a set of data for which the true values are known. It is called a confusion matrix because it can show what types of errors are being made by the model. Below are the definitions of confusion matrix values and the evaluation metrics:

A confusion matrix is a 2×2 matrix that contains the following four values:

- ✓ **True Positive (TP):** These are the points that are actual markers and correctly labelled by the algorithm.
- ✓ **False Positive (FP):** These points are incorrectly identified as markers by the algorithm, either because they are actually noise or because the labels are wrong.
- ✓ **False Negative (FN):** Actual markers that are falsely unlabelled by the algorithm.
- ✓ **True Negatives (TN):** Instances where the model correctly identifies negative instances (i.e., the model correctly classifies data points as noise or when the marker is null in a frame and is labeled correctly as null).

Accuracy: The proportion of correctly predicted labels over all labels. This is calculated by dividing the number of correct predictions by the total number of labels:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: The proportion of actual correct labels over predicted labels. This is calculated by dividing the number of true positives by the total number of positive labels predicted:

$$Precision = \frac{TP}{TP + FP}$$

Recall: The proportion of correct predicted labels over actual labels. This is calculated by dividing the number of true positives by the total number of actual positive labels:

$$Recall = \frac{TP}{TP + FN}$$

F1 Score: The harmonic-average of precision and recall:

$$F_1 = \frac{Precision \times Recall}{Precision + Recall} = \frac{2 \times TP}{2 \times TP + FP + FN}$$

5.4 Experimental Results and Analysis

We captured 10 underwater MoCap C3D data sets at 100 Hz, with 21 passive markers, as described in Table 5-2. We cleaned and labeled these files using our proposed semi-supervised geometry-based labeling algorithm to create a ground truth dataset and training data. Then, these datasets were augmented through transfer learning by using simulated trajectories. An additional augmentation was performed by introducing gaps and noise into these datasets. We evaluated the deep-learning-based approach on these 10 raw C3D files because with passive markers, the raw data contained many reappearing markers, which were not like the cleaned data with gaps where each physical marker had a unique, complete trajectory with potential gaps.

We conducted a series of tests to evaluate the impact of different factors on network accuracy including feature selection as an input to the network, size of the training set, and the effects of our proposed extraneous removal technique. The tests were as follows:

Test 1: 5-inputs LSTM, and train set of 10 C3D files

Test 2: 5-inputs LSTM, and train set of 10 C3D files and simulated trajectories

Test 3: 5-inputs LSTM, and train set of 100 C3D files

Test 4: 5-inputs LSTM, and train set of 100 C3D files and simulated trajectories

Test 5: 3-inputs LSTM, and train set of 100 C3D files

All tests mentioned above were conducted once with the application of the extraneous removal algorithm on the datasets and then again without applying the extraneous removal to investigate the effectiveness of this algorithm.

5.4.1 Raw C3D Data Sets

The actions' description of 10 captured underwater MoCap data sets is displayed in Table 5-2. Data were captured at 100 Hz, with 21 markers attached to the swimmer's body. The major movements are described, while they can be quite intricate. The axis directions of 3D coordinates are described in Figure 5-4.

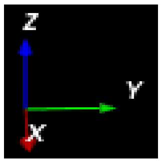


Figure 5-4: Axes directions of 3D coordinates

Table 5-2: Actions Description of Raw C3D datasets

C3D	Underwater Action Description	Head Position	Transition Posture	Face
1	- Going down (Head UP) to -Z (Upright)	+Z	(+Z) Vertical	-X
	- Overhead clapping (still feet)	+Z	(+Z) Vertical	-X
	- Transition to a horizontal position (+X)	+X	(+X) Horizontal	+Z
	- Float (Supine) : Face +Z	+X	(+X) Horizontal	+Z
2	- Front Crawl to -X	-X	(-X) Horizontal	-Z
	- Turn & Front Crawl to +Y	+Y	(+Y) Horizontal	-Z
	- Turn & Front Crawl to +X	+X	(+X) Horizontal	-Z
3	- Going downward to -Z (Upside-down)	-Z	(-Z) Vertical	+X
	- Bending Transition to Float (-XY) (Supine)	-XY	(-XY) Horizontal	-Z
4	- Going downward to -Z (Upside-down)	-Z	(-Z) Vertical	+XY
	- Downward Butterfly	-Z	(-Z) Vertical	+XY
	- Rotate Body around XY (few times)	+XY	(+XY) Horizontal	+Z,-Z
	- Transition to Float (Supine) to +Y	+Y	(+Y) Horizontal	+Z
5	- Crawl downward to -Z (Upside-down)	-Z	(-Z) Vertical	+X
	- Rotate Body around Z	-Z	(-Z) Vertical	+X,-X
	- Transition to Horizontal +Y	+Y	(+Y) Horizontal	-Z
	- Jumping position & going up (Upright) (+Z)	+Z	(+Z) Upright Vertical	-X
	- Rotate around Z to Face -Y	+Z	(+Z) Upright Vertical	-Y
6	- Breaststroke downward (Upside-down)	-Z	(-Z) Vertical	-XY
	- Bending Turn to (+Z)	+Z	(+Z) Vertical	+XY
7	- Walk horizontally to +Y	+Z	(+Z) Vertical	+Y
	- Bend Knee & Jump up (+Z) (Upright)	+Z	(+Z) Vertical	+Y
8	- Breaststroke downward (Upside-down)	-Z	(-Z) Vertical	-X
	- Bending turning to horizontal +XY	+XY	(+XY) Horizontal	-Z
	- Going up (Upright)	+Z	(+Z) Vertical	+X
9	- Front Crawl to -X	-X	(-X) Horizontal	-Z
	- Turn & Front Crawl to +Y	+Y	(+Y) Horizontal	-Z
	- Turn & Front Crawl to +X	+X	(+X) Horizontal	-Z
10	- Moving hands	+Z	(+Z) Vertical	-X
	- Bend Backward & Float horizontal +X	+X	(+X) Horizontal	+Z

The properties of these 10 datasets are in Table 5-3. These datasets contain null markers due to being exported from another large C3D file, noise which includes ghost markers and outliers, extraneous markers which belong to another object, and dropped markers across the entire frames. Two datasets, numbers 7 and 10, have extraneous markers. All datasets except numbers 6 and 8 have dropped markers. In all, only one side ankle (left or right) was dropped, except for dataset number 7, which had two hand markers (left and right) dropped. Additionally, this table provides information about passive marker characteristics, specifically those that reappear after being occluded. Passive markers can have multiple short trajectories, called tracklets [42], with different IDs, meaning a single physical marker can be associated with more than one point. The maximum number of tracklets per marker and the number of tracklets exceeding one, highlighting the complexity of cleaning and labeling data. The minimum, maximum, and mean filling levels are also presented, further emphasizing the labeling challenges as markers with low filling levels can resemble noise.

Table 5-3: Datasets Properties

Data \ C3D		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
#Frames		921	1241	536	686	803	897	460	792	803	653
#Points		65	62	34	65	51	35	245	30	48	1719
#NaN		5	15	0	5	5	4	221	4	10	1667
#Valid Points		30	44	25	43	30	28	20	26	36	24
#Extranous		0	0	0	0	0	0	4	0	0	5
#Noise		30	3	9	17	16	3	0	0	2	28
#Dropped		1	1	1	1	1	0	2	0	1	1
Max #Tracklets		5	6	2	7	4	3	2	2	4	4
#Tracklets > 1		5	12	5	13	6	6	1	5	7	2
Valid Markers Filling Level	Min	0.65	1.13	2.24	0.15	2.86	4.68	33.26	7.45	1.74	3.22
	Max	100	100	100	100	100	100	100	100	100	100
	Mean	65.25	43.75	79.13	44.21	65.47	74.09	94.41	80.05	54.16	82.52

5.4.2 Labeling Performance Results

The results of test numbers 1, 3, 4, and 5 are shown in Table 5-4.

Table 5-4: Results for Test 1,3,4, and 5 (E = Extraneous)

Dataset	Rotation Angle (º)	Test	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
1	177.2	1	81.19	89.35	89.33	89.34
		3	98.59	99.68	98.84	99.24
		4	99.47	99.61	99.84	99.72
		5	87.32	96.41	89.30	92.69
2	177.2	1	92.26	99.77	91.86	95.63
		3	99.86	99.99	99.85	99.92
		4	98.64	99.74	98.63	99.17
		5	99.18	99.74	99.19	99.46
3	176.9	1	79.65	100	78.26	87.76
		3	98.79	99.13	99.57	99.35
		4	99.68	99.67	100	99.83
		5	99.59	99.58	100	99.79
4	110	1	65.58	89.25	67.53	76.70
		3	96.92	98.71	97.57	98.12
		4	89.59	94.82	93.46	94.11
		5	95.12	97.01	97.21	97.10
5	177	1	94.41	99.54	94.51	96.94
		3	99.41	99.66	99.69	99.67
		4	99.98	99.98	100	99.99
		5	99.98	99.98	100	99.99
6	-40	1	100	100	100	100
		3	99.98	99.98	100	99.99
		4	99.31	99.70	99.55	99.62
		5	99.65	99.82	99.81	99.81
7	-20	1 E	62.88	76.28	68.25	72.04
		1	100	100	100	100
		3 E	70.87	72.58	94.14	81.96
		3	100	100	100	100
		4 E	59.30	62.33	88.22	73.05
		4	100	100	100	100
		5 E	58.66	62.33	83.29	71.30
		5	100	100	100	100
8	0	1	100	100	100	100
		3	100	100	100	100
		4	100	100	100	100
		5	100	100	100	100
9	177	1	87.18	94.53	91.63	93.04
		3	99.92	100	99.91	99.95
		4	99.88	99.96	99.91	99.94
		5	99.85	99.96	99.87	99.92
10	-177.5	1 E	73.71	83.03	76.99	79.87
		1	84.54	90.17	92.76	91.43
		3 E	65.71	71.42	81.13	75.94
		3	98.69	99.43	99.26	99.33
		4 E	76.89	80.53	89.57	84.80
		4	99.35	99.44	99.90	99.66
		5 E	59.93	65.23	80.84	72.19
		5	99.50	99.65	99.84	99.74

Table 5-4, presents the accuracy, precision, recall, and F_1-score evaluation metrics for each dataset across Test 1, 3, 4, and 5. Figure 5-5 and Figure 5-6 present bar charts for accuracy and F_1-Score, that facilitate a better understanding and analysis of these results which will be provided in further pages. However, the most significant result is the improved performance on all four tests using datasets without extraneous markers (i.e., in datasets 7 and 10). These demonstrate the effectiveness of one of our contributions, which is the extraneous marker removal algorithm. These tests were also conducted on cleaned, unlabeled versions of these 10 datasets, and all tests resulted in 100% accuracy.

The results for Test 2, which utilizes a "5-input LSTM" model with a training set of 10 C3D files and simulated trajectories, are presented in Table 5-5 and Table 5-6. These tables are separated due to the lower performance of this test. Table 5-5 displays the results for the cleaned, unlabeled version of these 10 datasets, while Table 5-6 shows the results for the raw, noisy datasets. The mean of each metric is calculated once for the dataset without extraneous markers and once with extraneous markers, considering that the cleaned version contains no such markers.

Table 5-5: Results for Cleaned Datasets (Free of Extraneous) for Test 2

Data \ C3D	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Average**
Rotation Angle (º)	177.2	177.2	176.9	110	177	-40	-20	0	177	-177.5	
Accuracy (%)	26.09	19.05	19.05	19.05	19.05	19.05	19.05	19.05	19.05	26.09	20.458
Precision (%)	75	100	100	100	100	100	100	100	100	75	95.0
Recall (%)	28.57	19.05	19.05	19.05	19.05	19.05	10.62	19.05	19.05	28.57	20.111
F1-Score (%)	41.38	32.00	32	32	32	32	19.19	32.00	32	41.38	32.595

Table 5-6: Results for Raw Datasets for Test 2 (E = Extraneous) (Av= Average)

Data \ C3D	**1**	**2**	**3**	**4**	**5**	**6**	**7 E**	**7**	**8**	**9**	**10 E**	**10**	**Av. (E)**	**Av.**
Rotation Angle (º)	177.2	177.2	176.9	110	177	-40	-20	-20	0	177	-177.5	-177.5		
Accuracy (%)	61.66	54.78	57.61	54.30	46.30	55.12	46.64	37.36	100	52.55,	39.89	50.12	51.63	51.725
Precision (%)	69.55	82.81	71.75	75.54	61.01	72.72	51.95	63.27	100	90.94	53.01	60.55	72.928	74.814
Recall (%)	83.14	58.18	74.04	64.39	64.70	69.44	70.52	38.53	100	52	46.25	70.56	68.266	67.498
F1-Score (%)	75.72	67.79	72.83	69.34	62.69	70.77	59.83	47.89	100	66.11	49.39	65.16	69.447	69.83

5.4.3 Analysis of Low Performance of Test 2:

We expected to see improved results by using transfer learning to augment the training set with simulated trajectories. However, when comparing the results of Test 2 (Table 5-5 and Table 5-6: a 5-input LSTM trained with 10 C3D files and simulated trajectories) to Test 1 (Table 5-4: a 5-input LSTM trained only with 10 C3D files), the results were significantly decreased rather than improved. This could be due to several factors:

1- Mismatch in Data Distributions:

The primary issue at hand is the mismatch between the data distributions of the pretrained model and the new dataset. This discrepancy arises from the fact that the pre-trained model was generated based on a specific marker set distribution, while the kinematics of the new data are different, with distinct actions. Consequently, characteristics like velocity and acceleration in the new data vary significantly from those in the pre-trained model. This mismatch can lead to the pre-trained model's initial weights being poorly suited for the new data, causing the model to struggle to learn effectively.

2- Pre-trained Model Bias due to small datasets (10 C3D files):

Table 5-4 shows that transfer learning yielded either enhanced or nearly similar results when comparing Test 3 (5-input LSTM, trained on 100 C3D files without simulated trajectories) and Test 4 (trained on 100 C3D files with simulated trajectories). Initially, this might seem contradictory to the first factor, which is the mismatch problem. However, this discrepancy arises because Test 2 uses a dataset of 10 C3D files, while Test 3 and 4 utilize datasets of 100 C3D files. In essence, transfer learning augmented the larger datasets in

Test 3 and 4 (100 C3D files), whereas in Test 2, it augmented a smaller dataset (10 C3D files). Consequently, the smaller dataset in Test 2 introduced a bias.

5.4.4 Analysis of Results: Test 1, 3, 4, and 5

Figure 5-5 and Figure 5-6 present bar charts, for accuracy and F1-score of Table 5-4 that facilitate a better understanding and analysis of the results of Test 1, 3, 4, and 5.

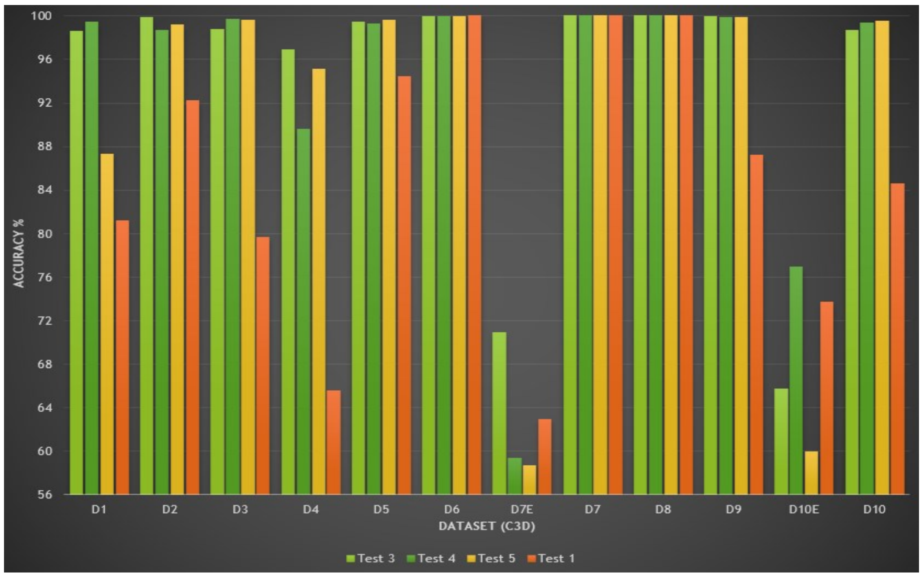


Figure 5-5: Accuracy for 10 datasets in Table 5-4: Tests 1, 3, 4, and 5

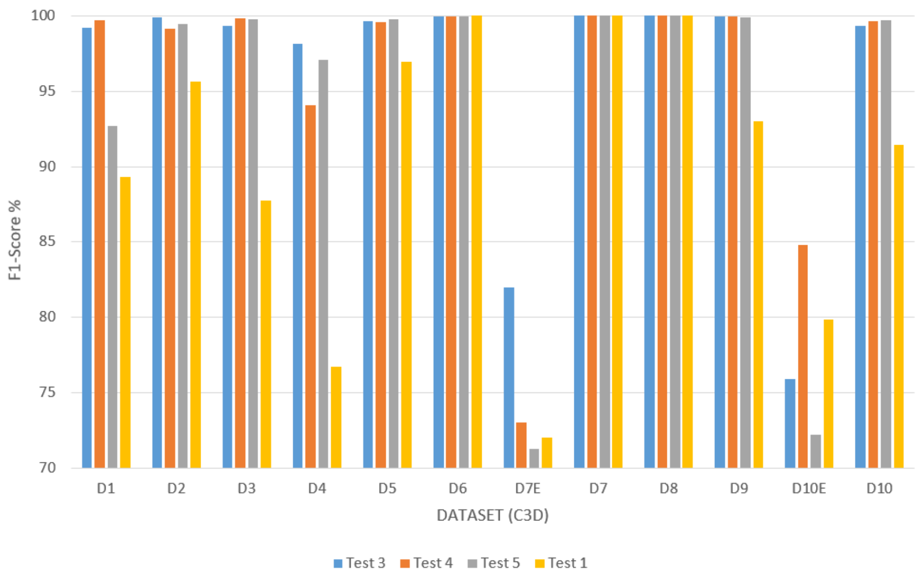


Figure 5-6: F1-score for 10 datasets in Table 5-4: Tests 1, 3, 4, and 5

Summary of the results (Table 5-7)**:**

Table 5-7: Summary of Results for Comparison of different Factors (V: Vertical), (H: Horizontal)

Datasets	**Main Postures**		**Accuracy (%)**		**Ranked Results**			
C3D	**V**	**H**	**Min**	**Max**	**Highest**	**Rank 2**	**Rank 3**	**Lowest**
1	+	+	81	99.4	Test 4	Test 3	Test 5	Test 1
2	-	+	92	99.9	Test 3	Test 5	Test 4	Test 1
3	+	+	79	99.7	Test 4	Test 5	Test 3	Test 1
4	+	+	65	96.9	Test 3	Test 5	Test 4	Test 1
5	+	+	94	99.9	Test 5	Test 4	Test 3	Test 1
6	+	-	99	100	Test 1	Test 3	Test 5	Test 4
7 E	+	-	58	70.9	Test 3	Test 1	Test 4	Test 5
7	+	-	100	100	Same Rank Across Tests			
8	+	-	100	100	Same Rank Across Tests			
9	-	+	87	99.9	Test 3	Test 4	Test 5	Test 1
10 E	+	+	60	76.9	Test 4	Test 1	Test 3	Test 5
10	+	+	84	99.5	Test 3	Test 4	Test 3	Test 1

The conclusions from these results based on different factors:

1- **3-Inputs vs 5-Inputs:** 5 input variables are more effective; however, for some datasets, 3 inputs yield better results than 5 inputs. In most cases, their results are quite similar. The poorest performances were in datasets 7E and 10E.

2- **Augmented training set with simulated trajectories:** overall, similar to 3 inputs, except that the poorest results were obtained for dataset 6.

3- **Augmented training set from 10 to 100:** the small training set of 10 nearly consistently achieved the lowest results across all datasets.

4- **Extraneous Removal:** Removing extraneous markers significantly enhanced the results, increasing accuracy from 62.9% to 100% for dataset 7 and from 69% to 95.5% for dataset 10. This demonstrates the substantial impact of our contribution which is our extraneous removal algorithm.

We expected the exclusive horizontal movements in datasets 2 and 9 to produce lower results, as the simulated trajectories mainly involve vertical actions like running and walking. Contrary to our expectation, this did not happen, possibly due to the larger underwater training sets (100 C3Ds) compared to 20 participants-simulated trajectories.

5.4.5 Overall Conclusion: The effectiveness of Extraneous Removal

Table 5-8 and Figure 5-7 show the average accuracy and F_1-score across all datasets for Tests 1, 3, 4, and 5, both with and without extraneous markers. This table demonstrates the effectiveness of the extraneous marker removal algorithm. Datasets 7 and 10 are the only ones containing extraneous markers. This table also compares these four tests across all datasets, showing that Test 3 achieved the highest results, followed by Test 4 and Test 5. The lowest result was for the non-augmented training set with 5 inputs.

Table 5-8: The Average Accuracy and F1-score across all Datasets

Tests		**1**	**3**	**4**	**5**
Accuracy (%)	(E) With Extraneous	83.69	93.01	92.27	89.93
	Without Extraneous	88.48	99.22	98.59	98.02
F1-Score (%)	(E) With Extraneous	89.13	95.41	95.02	93.23
	Without Extraneous	93.08	99.56	99.20	98.85

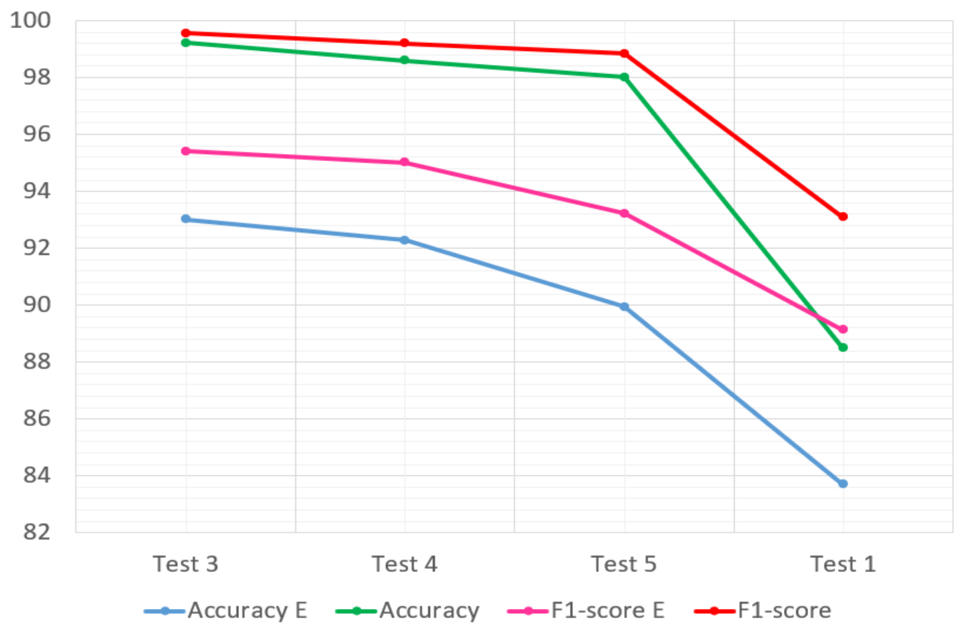


Figure 5-7: The Average Accuracy and F1-Score across all Datasets

5.5 Conclusion

We addressed the challenge of automatically labeling raw MoCap data for underwater motion using sparse marker sets. These data are prone to noise such as outliers, ghost markers, extraneous markers, and missing markers due to occlusions and reduced visibility underwater. Additionally, dealing with reappearing valid markers from passive markers leads to numerous small trajectory segments instead of a consistent number of valid markers with trajectories and potential gaps.

We tackled this issue by training an LSTM network using various inputs to evaluate the impact of feature selection, training set size, and noise on accuracy. The training set was augmented with random noise and gaps. Our LSTM network was trained with three 3D relative position inputs, then with relative positions, velocity, and acceleration as five inputs, and utilized transfer learning with simulated trajectories to expand the training data. The Hungarian algorithm was used for label assignment. Procrustes analysis was used to locate unlabeled markers. Post-processing was used to correct mislabeled markers using an OpenSim marker set. A semi-supervised geometry-based labeling method established the ground truth and training sets and allowing results to be compared with LSTM results. PCA-based pelvis detection was designed for data alignment, and an extraneous marker removal algorithm was proposed to enhance our LSTM model performance. The extraneous markers removal algorithm increased the accuracy of our approach for datasets with extraneous markers from 66% to 98%. The final semi-supervised algorithm achieved 100% on our 10 datasets. The auto-labelling algorithm reduced the time and tedious efforts of the manual labelling of MoCap data.

References

[1] G. B. Guerra-filho, "Optical motion capture: Theory and implementation," *J. Theor. Appl. Informatics*, vol. 12, pp. 61--89, 2005, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.7248

[2] M. Menolotto, D. S. Komaris, S. Tedesco, B. O'flynn, and M. Walsh, "Motion capture technology in industrial applications: A systematic review," *Sensors (Switzerland)*, vol. 20, no. 19, pp. 1–25, 2020, doi: 10.3390/s20195687.

[3] Qualisys, "Qualisys," *Qualisys*. https://www.qualisys.com/ (accessed Dec. 10, 2023).

[4] I. M. Thompson, J. Banks, D. Hudson, and M. Warner, "Assessment of error levels across the domain of a three dimensional underwater motion capture methodology," *40th Int. Soc. Biomech. Sport. Conf.*, pp. 703–706, 2022.

[5] Qualisys, "Using AIM models." https://docs.qualisys.com/getting-started/content/getting_started/processing_your_data/using_aim_models/using_aim_models.htm (accessed Aug. 01, 2024).

[6] J. Meyer, M. Kuderer, J. Muller, and W. Burgard, "Online marker labeling for fully automatic skeleton Tracking in optical motion capture," *Proc. - IEEE Int. Conf. Robot. Autom.*, no. May 2014, pp. 5652–5657, 2014, doi: 10.1109/ICRA.2014.6907690.

[7] K. Chen, Y. Wang, S. H. Zhang, S. Z. Xu, W. Zhang, and S. M. Hu, "MoCap-solver: A neural solver for optical motion capture data," *ACM Trans. Graph.*, vol. 40, no. 4, 2021, doi: 10.1145/3450626.3459681.

[8] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. Black, “AMASS: Archive of motion capture as surface shapes,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-Octob, pp. 5441–5450, 2019, doi: 10.1109/ICCV.2019.00554.

[9] Qualisys, “Swimming Technique: dual media motion capture,” *Qualisys*. https://qfl.qualisys.com/#!/project/swimming-techniques (accessed Dec. 02, 2023).

[10] Wikipedia, “Long short-term memory.” https://en.wikipedia.org/wiki/Long_short-term_memory (accessed Jul. 29, 2024).

[11] H. W. Kuhn, “The Hungarian method for the assignment problem,” *50 Years Integer Program. 1958-2008 From Early Years to State-of-the-Art*, vol. 2, no. 1, pp. 29–47, 2010, doi: 10.1007/978-3-540-68279-0_2.

[12] Wikipedia, “Principal component analysis.” https://en.wikipedia.org/wiki/Principal_component_analysis

[13] SimTK, “OpenSim,” *SimTK*. https://simtk.org/frs/index.php?group_id=91 (accessed Jan. 05, 2024).

[14] S. Chatterjee, “Procrustes Problems,” *Technometrics*, vol. 47, no. 3, pp. 376–376, 2005, doi: 10.1198/tech.2005.s296.

[15] Wikipedia, “Transfer Learning.” https://en.wikipedia.org/wiki/Transfer_learning

[16] X. Liu, Y. M. Cheung, S. J. Peng, Z. Cui, B. Zhong, and J. X. Du, “Automatic motion capture data denoising via filtered subspace clustering and low rank matrix approximation,” *Signal Processing*, vol. 105, pp. 350–362, 2014, doi: 10.1016/j.sigpro.2014.06.009.

[17] A. Aristidou and J. Lasenby, “Real-time marker prediction and CoR estimation in optical motion capture,” *Vis. Comput.*, vol. 29, no. 1, pp. 7–26, 2013, doi:

10.1007/s00371-011-0671-y.

[18] S. Rajko and G. Qian, "Real-time automatic kinematic model building for optical motion capture using a markov random field," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4796 LNCS, pp. 69–78, 2007, doi: 10.1007/978-3-540-75773-3_8.

[19] M. Ringer and J. Lasenby, "A procedure for automatically estimating model parameters in optical motion capture," *Image Vis. Comput.*, vol. 22, no. 10 SPEC. ISS., pp. 843–850, 2004, doi: 10.1016/j.imavis.2004.02.011.

[20] A. Cuevas, J. Rodriguez-Navarro, and A. Susín, "Auto-labeling as a minimization problem with virtual occlusions," *18th Spanish Comput. Graph. Conf. CEIG 2008*, vol. 5, pp. 133–139, 2008.

[21] C. Schönauer, T. Pintaric, and H. Kaufmann, "Full Body Motion Capture A Flexible Marker-Based Solution," 2011.

[22] J. Baumann, B. Krüger, A. Zinke, and A. Weber, "Data-driven completion of motion capture data," *VRIPHYS 2011 - 8th Work. Virtual Real. Interact. Phys. Simulations*, no. January, pp. 111–118, 2011, doi: 10.2312/PE/vriphys/vriphys11/111-118.

[23] G. Liu and L. McMillan, "Estimation of missing markers in human motion capture," *Vis. Comput.*, vol. 22, no. 9–11, pp. 721–728, 2006, doi: 10.1007/s00371-006-0080-9.

[24] A. Aristidou, D. Cohen-Or, J. K. Hodgins, and A. Shamir, "Self-similarity analysis for motion capture cleaning," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 297–309, 2018, doi: 10.1111/cgf.13362.

[25] J. Xiao, Y. Feng, M. Ji, X. Yang, J. J. Zhang, and Y. Zhuang, "Sparse motion bases

selection for human motion denoising," *Signal Processing*, vol. 110, pp. 108–122, 2015, doi: 10.1016/j.sigpro.2014.08.017.

[26] S. Alexanderson, C. O'Sullivan, and J. Beskow, "Real-time labeling of non-rigid motion capture marker sets," *Comput. Graph.*, vol. 69, pp. 59–67, 2017, doi: 10.1016/j.cag.2017.10.001.

[27] S. Xia, L. Su, X. Fei, and H. Wang, "Toward accurate real-time marker labeling for live optical motion capture," *Vis. Comput.*, vol. 33, no. 6–8, pp. 993–1003, 2017, doi: 10.1007/s00371-017-1400-y.

[28] J. Li, D. Xiao, K. Li, and J. Li, "Graph matching for marker labeling and missing marker reconstruction with bone constraint by LSTM in optical motion capture," *IEEE Access*, vol. 9, pp. 34868–34881, 2021, doi: 10.1109/ACCESS.2021.3060385.

[29] S. Han, B. Liu, R. Wang, Y. Ye, C. D. Twigg, and K. Kin, "Online optical marker-based hand tracking with deep labels," *ACM Trans. Graph.*, vol. 37, no. 4, 2018, doi: 10.1145/3197517.3201399.

[30] Y. Zhu, "Denoising method of motion capture data based on neural network," *J. Phys. Conf. Ser.*, vol. 1650, no. 3, 2020, doi: 10.1088/1742-6596/1650/3/032068.

[31] J. Bütepage, M. J. Black, D. Kragic, and H. Kjellström, "Deep representation learning for human motion prediction and classification," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1591–1599, 2017, doi: 10.1109/CVPR.2017.173.

[32] S. Graßhof, M. Bastholm, and S. S. Brandt, "Neural Network-Based Human Motion Predictor and Smoother," *SN Comput. Sci.*, vol. 4, no. 6, 2023, doi: 10.1007/s42979-023-02195-0.

[33] A. L. Clouthier, G. B. Ross, M. P. Mavor, I. Coll, A. Boyle, and R. B. Graham, "Development and Validation of a Deep Learning Algorithm and Open-Source Platform for the Automatic Labelling of Motion Capture Markers," *IEEE Access*, vol. 9, pp. 36444–36454, 2021, doi: 10.1109/ACCESS.2021.3062748.

[34] Qualisys, "Cameras for underwater motion capture," *Qualisys*. https://www.qualisys.com/cameras/underwater/ (accessed Jul. 15, 2024).

[35] Qualisys, "Calibrating your system," *Qualisys*. https://docs.qualisys.com/getting-started/content/getting_started/running_your_qualisys_system/calibrating_your_system/calibrating_your_system.htm (accessed Jan. 06, 2024).

[36] Qualisys, "Qualisys Track Manager," *Qualisys*, 2011. https://www.qualisys.com/software/qualisys-track-manager/ (accessed Dec. 10, 2023).

[37] Qualisys, "Exporting files to C3D." https://docs.qualisys.com/getting-started/content/13_how_to_visualize_data_in_visual3d/exporting_files_to_c3d.htm?Highlight=export (accessed Aug. 01, 2024).

[38] B. Motion, "The C3D File Format A Technical User Guide," p. 134, 2021.

[39] Github, "ezc3d." https://github.com/pyomeca/ezc3d

[40] S. A *et al.*, "OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement Ajay," *PLOS Comput. Biol.*, vol. 14, no. 7, 2018, doi: 10.3758/BF03326891.

[41] Biomechanical-toolkit.github.io, "Mokka," *Biomechanical-toolkit.github.io*. https://biomechanical-toolkit.github.io/mokka/ (accessed Jan. 06, 2024).

[42] N. Ghorbani and M. J. Black, "SOMA: Solving Optical Marker-Based MoCap

Automatically," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 11097–11106, 2021, doi: 10.1109/ICCV48922.2021.01093.

6. Conclusion

6.1 Summary

As outlined in Table 1-1, this thesis consists of six chapters that address the challenges related to cleaning and labeling MoCap data. Chapter 1 introduces the background and problem statement concerning underwater OMC systems. Chapter 2 presents a through literature review of MoCap solving methods. Chapter 3 illustrates manual cleaning of noisy MoCap data, highlighting the challenges involved in this process.

Chapter 4 proposes a semi-supervised geometry-based labeling and cleaning algorithm to streamline the time-consuming manual cleaning process described in Chapter 3. While this method has limitations, such as requiring significant user input and making certain assumptions, it provides high accuracy without needing training data.

Chapter 5 introduces a deep learning-based auto-labeling method designed to address the challenges presented by the semi-supervised algorithm discussed in Chapter 4. While the algorithm in Chapter 5 effectively mitigates issues related to high user intervention and assumptions inherent in the Chapter 4 algorithm, achieving good accuracy, it remains dependent on a large set of cleaned and labeled training data. This dataset is generated by the semi-supervised algorithm outlined in Chapter 4.

Overall, combining these two algorithms (Chapter 4 and Chapter 5) in the future could address all the issues of both algorithms while benefiting from their respective advantages.

6.2 Proportional Contributions of Each Chapter

As stated in Section 1.4, this thesis makes seven main contributions:

1. *Creation of the first underwater freestyle swimming MoCap dataset.*

   Ten MoCap C3D files were captured using the Qualisys system and subsequently underwent manual cleaning and labeling, as detailed in Chapter 3. Furthermore, Chapters 4 and 5 introduced innovative algorithms designed to streamline the time-consuming manual cleaning process.

2. *Addressing the challenges of using sparse MoCap datasets.*

   An underexplored topic covered in Chapters 3, 4, and 5.

3. *A comprehensive literature review on MoCap solving approaches.*

   This is presented in Chapter 2.

4. *A semi-supervised geometry-based labeling algorithm that includes an anomaly detection method utilizing norm, velocity, acceleration, and jerk profiles.*

   This is discussed in Chapter 4 as a standalone labeling algorithm and used in Chapter 5 to generate the training set for the deep learning network.

5. *An innovative extraneous removal algorithm based on the difference of the norms.*

   This method, part of the semi-supervised algorithm from Chapter 4, serves as a preprocessing step in Chapter 5 to eliminate extraneous markers, improving the algorithm's accuracy from 66% to 98%.

6. *A novel pelvis detection technique using PCA, implemented along with a method to recover dropped markers.*

   These methods are part of the semi-supervised algorithm from Chapter 4, with pelvis detection serving as an alignment step in the Chapter 5 algorithm.

7. *An auto-labeling algorithm based on LSTM, the Hungarian label assignment, and Procrustes alignment, incorporating a geometry-based method for initial*

alignment, ground truth and training set creation, and an enhancement of the accuracy of the results.

This algorithm in Chapter 5 incorporates steps from the Chapter 4 algorithm, including pelvis detection for alignment and the use of extraneous markers to enhance accuracy. Additionally, part of the training set is generated using the algorithm from Chapter 4.

6.3 Future Works

As concluded in Section 6.1, we recommend integrating our cleaning and labeling algorithms from Chapters 4 and 5 to leverage their strengths and resolve their respective issues.

To enhance our future work, we should collect more underwater data with diverse actions and additional noise and ghost markers. This will allow us to evaluate our algorithms more comprehensively. Furthermore, having a larger dataset will improve the training of our deep learning algorithm discussed in Chapter 5, resulting in higher accuracy and better generalization for handling unseen data.

Moreover, key areas for enhancement in this work to establish a standalone approach for any MoCap auto-labeling methods include pelvis detection and body side differentiation. Developing a more robust solution in these areas can be applied as the alignment method across various MoCap auto-labeling techniques.