An Inertial Sensor-Based Motion Capture Pipeline for Movement Analysis

By Qingyao Bian

A thesis submitted to the University of Birmingham

for the degree of MASTER OF SCIENCE (BY RESEARCH)

Y3, School of Engineering Lab

Department of Mechanical Engineering

College of Engineering and Physical Sciences

University of Birmingham

MAY 2022

UNIVERSITY OF
BIRMINGHAM

University of Birmingham Research Archive

e-theses repository

This unpublished thesis/dissertation is copyright of the author and/or third parties. The intellectual property rights of the author or third parties in respect of this work are as defined by The Copyright Designs and Patents Act 1988 or as modified by any successor legislation.

Any use made of information contained in this thesis/dissertation must be in accordance with that legislation and must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the permission of the copyright holder.

Abstract

Advanced human motion capture technologies have benefited both clinical diagnosis and rehabilitation in the past decades, which provided scientists and clinicians with a comprehensive understanding of human motion. So far, various motion capture methods based on different sensors, such as IMU sensors and high-speed cameras, have appeared and boosted the development of joint angle estimation methods. At present, the utilization of musculoskeletal models has enabled biomedical communities to calculate joint angles by applying a standard protocol. However, it costs great expense when we try to develop a human motion intention prediction method because few studies predict joint angles without EMG signals, which is a sort of neuro signal several ten microseconds ahead of the movement. Thus, this thesis presents works aiming to develop a lower limb human motion intention prediction method with pure IMU sensors. To improve the calculation efficiency of joint angle estimation, we developed an advanced algorithm based on Riemannian distance. A new comprehensive dataset, including both single-joint and multi-joint trials, was also collected in this part to find out suitable experimental parameters for IMU-based measurements and validate the R-distance-based joint angle estimation method. Moreover, we collected data from 6 healthy subjects to validate the motion intention prediction method proposed in the study. The subjects were asked to perform the 20s static calibration and the sit-stand-sit-walk task in the experiment. The human motion intention prediction method gives out a novel solution to low-cost motion intention prediction based on pure IMU data by fusing both musculoskeletal modeling

technologies and Long Short-term memory(LSTM) neural networks. This thesis reveals that: (1) Motion in the horizontal plane performs worse when compared with those in the other two planes. The Root Mean Squared Error(RMSE) increases when movement range increases and motion speed increases except for motion in the horizontal plane, where measurement performance gets better at either a high speed or a low speed. (2) The R-distance-based method outperforms the Euler method when it comes to calculation efficiency. (3) The comprehensive motion intention prediction method outperforms both the pure LSTM method and ANN fusing MSK model when it comes to prediction accuracy.

Acknowledgement

The author would like to thank anyone who has helped him.

Contents

List of Figures

List of Tables

Chapter1 Introduction

1.1 Background

In recent years, motion capture technology has played an important role in game design, 3D film production, and virtual reality.[1] It has attracted widespread attention in the biomechanics community to design interventional treatment: on the one hand, motion capture can help physicians better understand the functional movements of the patients and design more specific treatment solutions; on the other hand, motion capture can be used to provide biofeedback in wearable devices, which will enable advanced control algorithms to be deployed in rehabilitation.

Fig 1.1 XSENS IMU

Fig 1.2 Vicon Camera

Fig 1.3 EMG signal collection

Musculoskeletal lower limb disorders(LLDs) are one of the most popular topics in biomechanics analysis, mainly including hip and knee osteoarthritis, knee bursitis, meniscal lesions/tears, stress fracture/reaction injury, and lower limb varicose veins. LLDs may cause patients to suffer from movement disorders, chronic diseases and mental sub-health. Therefore, it is highly recommended to utilize movement tracking technologies to provide technical assistance for the treatment and rehabilitation of lower limb movement diseases. Movement could be measured by optical motion capture devices or inertia measurement units (IMUs), as shown in Fig 1.1. Optical

motion capture device mainly refers to a measurement system composed of several high-speed cameras (Fig 1.2) and a central signal processing device, which estimates the joint angle by tracking the trajectory of the reflective markers. An IMU sensor commonly contains a gyroscope, an accelerometer, and an optional magnetometer. The postures of different body segments are estimated through a sensor fusion algorithm like Kalman Filter, Complementary Filter, and so on to calculate the value of the joint angles between the tracked segments. Compared to the optical sensing signals, the signals measured by IMU sensors possess unique advantages in joint angle estimation: (1) IMU sensing equipment can be used in outdoor environments flexibly, and the equipment has higher portability. In contrast, optical motion capture devices raise high requirements of dedicated areas with limited capture volume, like a laboratory or large hospital, and tend to be bulky to transport. (2) The cost of an IMU motion capture device is much lower and therefore it is more economical to be deployed on different occasions. (4) IMU-based human motion tracking pipeline also has a promising accuracy like the optical method [2] . Therefore, IMU sensing technology is a promising lower limb motion tracking method.

IMU-based motion capture techniques could estimate joint angles between the tracking segments. However, it suffers from signal delay, mainly caused by signal transmission and the calculation of joint angles. It is difficult to meet the demand for real-time joint angle estimation and control of wearable devices Therefore, it is also necessary to develop a joint angle prediction technology to provide reliable real-time

biofeedback for rehabilitation devices. Most IMU joint angle prediction technologies fuse IMU signals and other sensor signals, such as optical motion capture signals, EMG (Fig 1.3) and other signals with sensor fusion algorithms, machine learning algorithms, and statistical methods to predict joints. For example, when it comes to joint angle prediction, electromyogram (EMG) data are often applied to provide extra bio information because neuromuscular currents are often several tens of milliseconds ahead of human movement. Using signals from multiple sorts of collection devices for data fusions to conduct joint angle prediction can provide relatively accurate predictions. However, this approach greatly increases the cost of the equipment, which in turn raises the financial burden.

1.2 Structure of the thesis

Chapter 2 is a literature review on IMU-based human motion tracking technologies. First, this thesis will give out a brief scan of the IMU-based Human Motion Capture pipeline and some necessary pre-processing technologies. Then a series of IMU data sensor fusion algorithms were introduced, from raw strapdown integration to Complementary Filter, Kalman Filter, Extended Kalman Filter, etc. After sensor fusion, we made a detailed review on the calibration and its function. Various valid designs of calibration protocols were introduced. It’s difficult to design a motion tracking pipeline from scratch, thus, we also reviewed different multi-body simulation software which provides proven technologies of MSK modelling and simulation. Last

but not least, to emphasize the significance of motion intention prediction technologies, we introduced some up-to-date research on human motion intention prediction.

In Chapter 3, we proposed and validated a novel optimization algorithm designed for joint angle estimation in the orientation-based inverse kinematics problems. Moreover, the experiment in this chapter aims to find out suitable measurement conditions for IMU data collection. First, different sorts of orientation representation methods and an Euler-angle-based optimization algorithm, which suffers from gimbal lock, were introduced. To get rid of the drawback of the Euler-angle-based method, we proposed a novel method by adopting a set of generalized coordinates to build the cost function. Then the validation experiment was conducted with the Delsys inertial data and the TM5-900 robot arm to validate the proposed algorithm.

Chapter 4 presented a human motion intention prediction method with the utilization of modern AI technologies. A series of lower limb motion trials were carried out on 6 healthy subjects, in which both IMU sensors and 8 high-speed cameras were used to capture human motion simultaneously. Then, further data processing was conducted, including an IMU-based human motion tracking pipeline and a marker-based human motion capture pipeline. The IMU-based method was adopted for further application of motion intention prediction while the marker-based method was utilized to work as a golden standard for further validation. Moreover, we proposed a novel lower limb

motion intention prediction method by fusing an LSTM neural network and MSK modelling technologies in this part.

Chapter 5 gives out conclusions of this study and future directions for further research.

Chapter2 Literature Review

2.1 IMU-based motion capture

The IMU-based motion capture pipeline often takes the orientation values over time calculated with 6 or 9 axes IMU data as the input. The orientation at each frame is estimated by fusing 6-axes data (3-axes accelerometer, 3-axes gyroscope) or 9-axes data (3-axes accelerometer, 3-axes gyroscope, and 3-axes magnetometer). So far, a lot of research work has been devoted to the topic of estimating the orientation of corresponding IMU based on the raw data. To increase the accuracy of raw data of IMU sensors, various methods, including gyroscope compensation, and accelerometer calibration has been developed. The raw output data tends to contain high-frequency noise, which might corrupt the result of orientation estimation. Thus, the Butterworth filter was applied to suppress the noise, whose order was chosen between 2 and 4. To reduce the effect of high-frequency noise, the cut-off frequency was normally set below 20Hz[3].

2.1.1 Sensor fusion algorithm

To estimate the orientation of IMUs, sensor fusion algorithms were adopted to integrate the raw data of IMUs, i.e., 3-axis accelerometer data, 3-axis gyroscope data, and 3-axis magnetometer data(optional). There are 3 commonly used sensor fusion methods: strap-down integration, complementary filter, and extended Kalman filter. (1)

strap-down integration: A strapdown algorithm integrates attitude changes in pitch, roll and yaw, as well as gross movements to estimate orientation. Rouhani, et al. measured foot joint angles with a combination of strap-down integration and low-acceleration instants detection[4]. Favre, et al. combined a former alignment method and a fusion algorithm based on strap-down integration to track the 3D motion of knee joint angle[5]. Tadano, et al. applied a quaternion-based strap-down integration method to conduct lower limb motion capture with 7 IMUs[6]. A strap-down method was also utilized by Fasel, et al. to reduce the orientation drift on the occasion where highly dynamic movements exist[7]. Strap-down is a classical framework for IMU sensor fusion. However, random noise and nonlinear error require more advanced sensor fusion algorithms to suppress. (2) Complementary filter: Seel, et al. applied a standard complementary filter to estimate the orientation of IMUs, which were in turn used to calculate flexion/extension joint angles[8]. Cockcroft, et al. proposed a novel nonlinear complementary filter, which possesses dynamic acceleration compensation[9]. The authors assumed that the skeletal model movement could be regarded as the thigh moving like a pendulum around the centre of the hip. (3) Extended Kalman filter: Jonathan, et al. applied an extended Kalman filter, which enables the proposed approach to extract joint angles from arbitrary human movements by fusing the raw data of the accelerometer and gyroscope [10]. A Rhythmic Extended Kalman Filter method was developed by Joukov, et al. to estimate the orientation of IMUs[11]. The algorithm improved the accuracy of pose estimation by learning individualized models after every period, and in turn utilize the learned

model to improve the performance of extended Kalman Filter(EKF). The proposed method could improve the accuracy of EKF estimation of acceleration by 40% and velocity by 37%. To track long-term walking, Slajpha, et al. proposed a method fusing angular velocity and linear acceleration of 7 segments with the prior knowledge of relationships between the segments[12]. Moreover, data from two instrumented shoe insoles were also fused with that from 7 IMUs. Similarly, Joukov, et al fused gyroscope data, accelerometer data, and trajectories of optical markers with an extended Kalman filter to perform the gait analysis [13] . To avoid singularity and improve computational efficiency, Yun, et al. tracked human motion with a novel Kalman filter whose rotation representation was quaternion rather than Euler angle or axis/angle[14]. Linear sensor fusion algorithms have a higher running speed. However, nonlinear algorithms were recommended to apply in tasks of higher velocity in order to have a satisfactory accuracy. In general, novel sensor fusion algorithms have been proposed to solve specific technical problems while ensuring a high accuracy.

The pre-processing and sensor fusion gives out a reliable estimation of IMU orientation. Thus, in this thesis, we directly take the fused orientation of IMUs output by the Delsys Trigno system as the input of the human motion tracking pipeline.

2.1.2 Sensor calibration

After the raw data preprocessing and sensor fusion, the IMU measurement system can output the orientation of IMUs over time. To calculate the joint angle by numerical

method, a calibration procedure is also demanded. The calibration process registers the IMU to the corresponding body segment and calculates the rotation transformation from the IMU coordinate system to the corresponding body segment coordinate system (usually given in the form of a rotation matrix, Euler angle, or quaternions).

Cheng, et al. surveyed a calibration method validated by a rigid robot arm [3] . The placement of IMU sensors was regarded to be well designed to make sure that the axes of the IMU reference frame coincided with those of the corresponding body segment reference frame. This kind of calibration method works based on the assumption that the tested subject is a system with multiple rigid bodies. However, in reality, artefacts exist in the measurement of the human motion process, which means that soft tissues could seriously challenge the placement of markers. Thus, the method can introduce extra errors to the measurement.

Predefined calibration postures are introduced to calculate the transformation rotation matrix between the IMU reference frame and body segment reference frame. Vries, et al. applied a static calibration method to upper limb IMU to body segment alignment [15] . To get the calibration matrices, the subjects were asked to stand in a so-called SAP pose for about 5 seconds. The SAP refers to standing straight with arms hanging along the torso naturally and hand pointing to the direction of the anterior. Palermo, et al. proposed a novel body-to-sensor calibration method for lower limb gait analysis [16] . Three static poses, including standing straight, sitting with the back

of the torso inclined and the legs stretched, and lying on a flat surface were applied in the experiment. Both two calibration methods can calculate the calibration matrices with relatively high accuracy (RMSE<4 degs).

Adopting functional calibration movement is also a popular method to improve the accuracy of the calibration process. Favre, et al. proposed a functional calibration procedure for 3D joint angle estimation with a moderate error between 4.0 degrees and 8.1 degrees[5]. The research used up to three different functional movements: (1) Hip abduction/adduction motion in the frontal plane. (2) Passive shank motion in the sagittal plane. (3) Passive shank motion in the frontal plane. Similarly, two functional movements were adopted in the research conducted by Karol, et al[17].: one with the subjects standing straight while rotating the lower limb about the body's longitudinal axis in the range of 0 degrees to 360 degrees and the other with the subject sitting at the neutral pose while the legs and feet perform a flexion/extension movement. Chardonnens, et al. combined the IMU reference with another motion capture system reference system by designing a calibration movement that the subjects simultaneously hold the IMU sensor and mobile part of the motion capture system with one hand and rotate around three orthogonal axes[18]. A combination of static calibration pose and functional calibration movements was also adopted by researchers to better perform IMU-to-segment alignment. In the study, all of the 3 above methods were utilized to estimate the axes of body segments. Replacing body segment axes with those of IMUs was denoted as TECH methods. Estimating body segment axes by performing predefined poses was denoted as STATIC. Estimating

body segment axes with functional movements were denoted as FUNC. Two combinations of various calibration approaches were raised in this paper. One is that combine the TECH method and two STATIC calibration procedures to achieve a full-segment calibration. The other is that combine both STATIC and FUNC methods. Haoyang, et al. developed a calibration process with 4 predefined postures(1 static and 3 functional) to track upper limb human motion[19]. (1) Stand straight with two arms hanging towards two sides of the body and palms pointing down in direction. (2) Stand at (1) pose and pronate/supinate the palms for about 180 degrees. (3) Stand in a neutral pose and raise the arms in the sagittal plane. (4) Stand in a neutral pose and raise the arms in the frontal plane. Both static poses and functional movements can provide a reliable calibration procedure for an IMU-based human motion capture pipeline. Thus, this thesis calibrates the skeletal model with a classical static pose recommended by the OpenSense official documents. The subjects were asked to stand straight in a neutral pose with their hands crossed on their chest.

2.1.3 A pipeline for automatic calibration and measurement

So far, numerous scientists and engineers track human motion by coding from scratch. However, it could be difficult to develop a comprehensive simulation environment for human movement analysis from nothing. Thus, there existed several multi-body simulation engines/software which allows developers to develop or validate their ideas about human motion tracking. As surveyed by Ivaldi, et al., there are many simulation software for multibody kinematics and dynamics calculation which can be

classified into two groups [20] : one for robotics analysis, and the other for biomechanics use. Commonly used multibody analysis tools are Gazebo, Webots, ODE, V-REP, and ROS. Quigley, et al. developed a robot operating system named ROS, which provides the developers with a platform characterized by peer-to-peer, tools-based, multi-lingual thin, free and open-source, to support the modular development of robotics technologies [21] . When it comes to ROS, ODE is also a popular topic for engineers and scientists. ODE is an extensive open dynamics engine in ROS to improve the accuracy and efficiency of the multibody simulation. Moreover, the robotics community usually discusses Gazebo, Webots, and V-REP. Webots is a professional mobile robotics simulation tool. The Gazebo is a 3D dynamic simulation tool that can accurately and effectively simulate robot groups in complex indoor and outdoor environments. Both three tools above were capable of multibody simulation. Some related work has been conducted in the above tools to track human movements. Du, et al. developed an upper limb virtual training rehabilitation scene with a combination of Gazebo and ROS[22]. A virtual scene was set up in the Gazebo to help the rehabilitation of patients with upper limb disorders. Qi, et al. proposed a human motion analysis system with inertial sensors in V-rep [23] . A quaternion-based orientation estimation algorithm was adopted in the research, after which the orientation data was imported to the V-REP to reconstruct the human motion. Although related research works reveal that robotics multibody simulation tools can be utilized to perform human motion tracking, it's not highly recommended to use these tools to solve biomechanical problems because most of them are based on

gaming engines. A gaming engine can provide stable multibody simulation, but it's based on simple gaming physics and doesn't incorporate most musculoskeletal models.

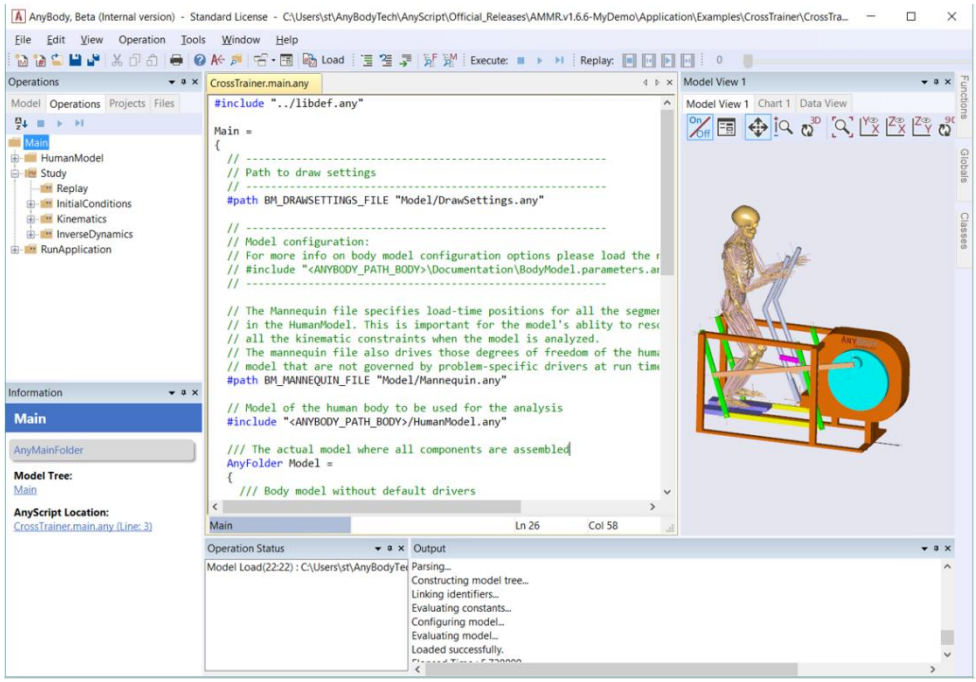


Fig 2.1 Anybody GUI

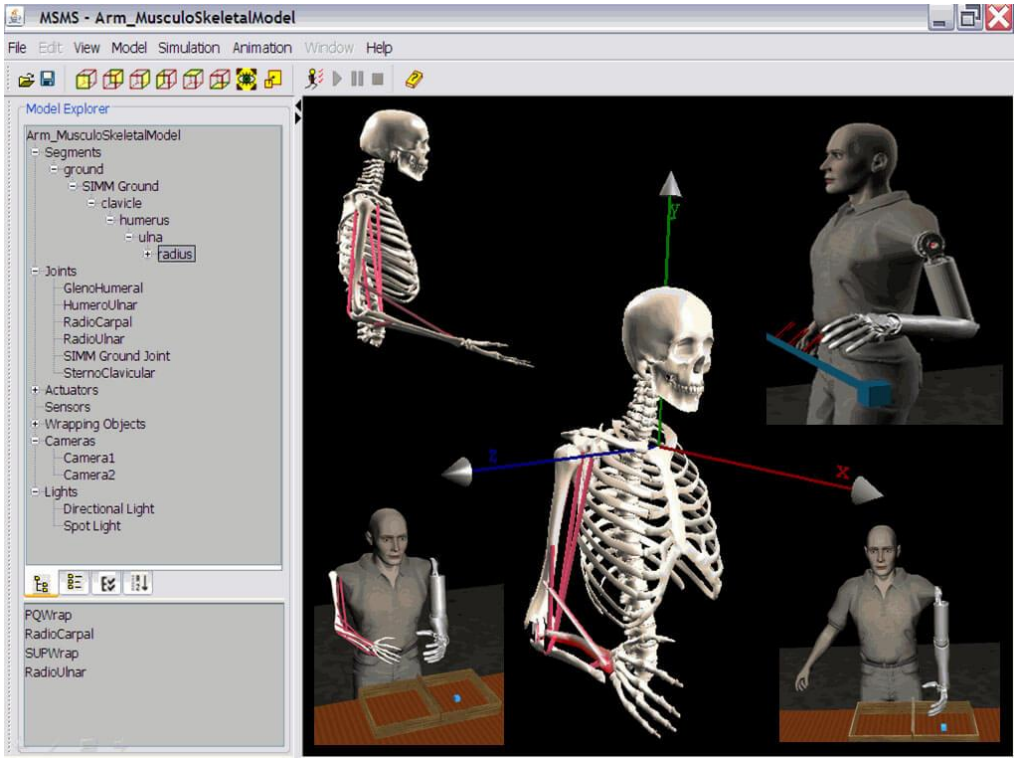


Fig 2.2 MSMS GUI

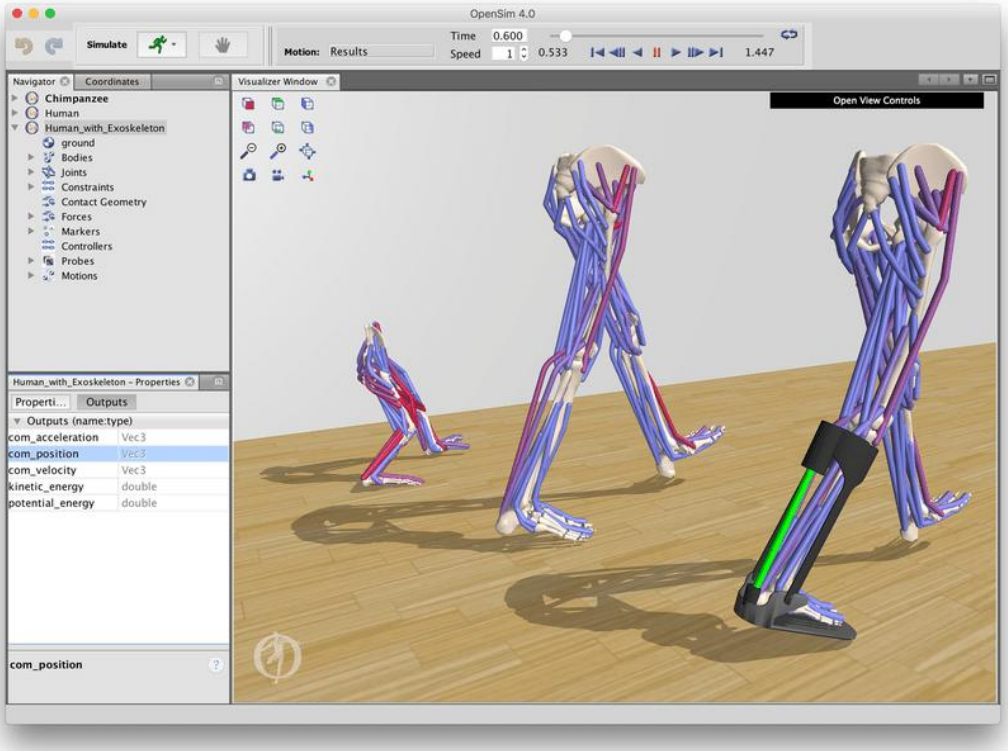


Fig 2.3 OpenSim GUI

Nunes, et al. surveyed 25 tools designed for human motion analysis and simulation [24] . After a thorough comparison, we could conclude that OpenSim is more suitable for this study. Among the surveyed tools, those which support the academic project, well-designed modelling module, 3D human motion analysis, and 3D human motion simulation are AnyBody modelling system(Fig 2.1), MSMS(Fig 2.2), OpenSim(Fig 2.3), and SIMM. The AnyBody Modeling System is a tool developed at the Aalborg University, Denmark which allows a 3D human-environment interaction simulation. MSMS is a free tool for modelling and neural prostheses simulation developed at the University of South California (USA). The software is designed to simulate the movements of human and prosthetic limbs. It allows MSK modelling and real-time 3D simulation like OpenSim.SIMM (Software

for Interactive Musculoskeletal Modeling) is developed by MusculoGraphics, Inc. (USA), which supports musculoskeletal model analysis.OpenSim is a software for modelling the human body and the environment, which could simulate human movements and interactions with the environment. OpenSim possesses a graphical user interface (GUI) capable of visualizing musculoskeletal (MSK) models and performing multibody simulations, as shown in Fig 2.4. OpenSim is open-source software with all the source code accessible to global developers. Engineers are free to develop the software by utilizing the OpenSim application programming interface (API). The simulation platform was firstly introduced by Scott Delp, Jennifer Hicks, Ajay Seth et al. at the American Society of Biomechanics Conference in 2007, which was known as version 1.0. An application programming interface (API) was added in V2.0 and the API was extended to MatLab and Python in the latest version. The extension to MatLab and Python greatly enhanced the data processing ability of the developers. The software OpenSim is also an interdisciplinary simulation platform that is compatible with multiple programming languages and enables engineers and scientists from different academic institutions to collaborate on biomechanics research. The core code of the software is developed in C++ while the graphical user interface (GUI) is developed based on Java. Developers could develop new MSK models or simulate human motion with data collected from sensors. Besides, there are numerous plug-ins on the OpenSim website for people to download and run without compiling, which makes it accessible to control OpenSim on other platforms. OpenSense is a novel workflow in OpenSim 4.0 which is developed for IMU-based human motion

tracking tasks. The workflow is designed to solve orientation-based inverse kinematics(OB-IK) problems based on IMU data (i.e. calculate joint angles between adjacent body segments). A multi-body model with a target joint is required before the human motion capture pipeline. Any model in .osim format could be utilized to conduct inverse kinematics research and no scaling is required in the procedure. Since the workflow provides user-friendly APIs for developers to communicate with MatLab and gives out an advanced solution to joint angle estimation, we chose to develop an IMU-based pipeline based on the workflow of OpenSense.

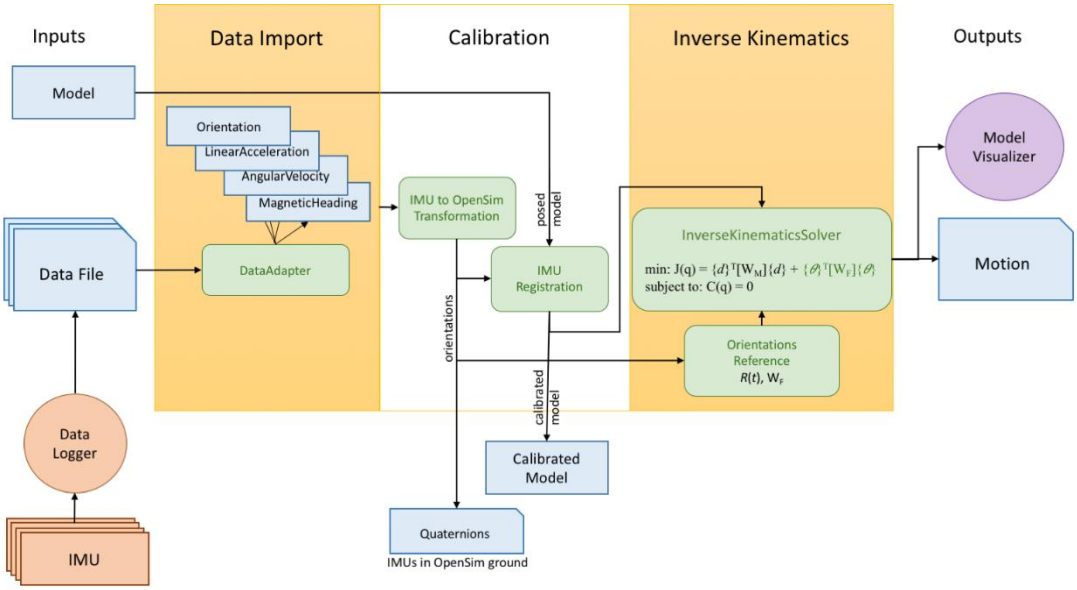


Fig 2.4 Standard OpenSense Workflow

2.2 IMU-based human motion prediction

As shown above, most contributions to human motion tracking in the past focused on joint angle estimation. However, with the delay caused by transmission and calculation, real-time tasks call for a predictive joint angle estimation method, as

shown in Fig 2.5. Thus, some research works shifted their attention to the motion intention prediction tasks.

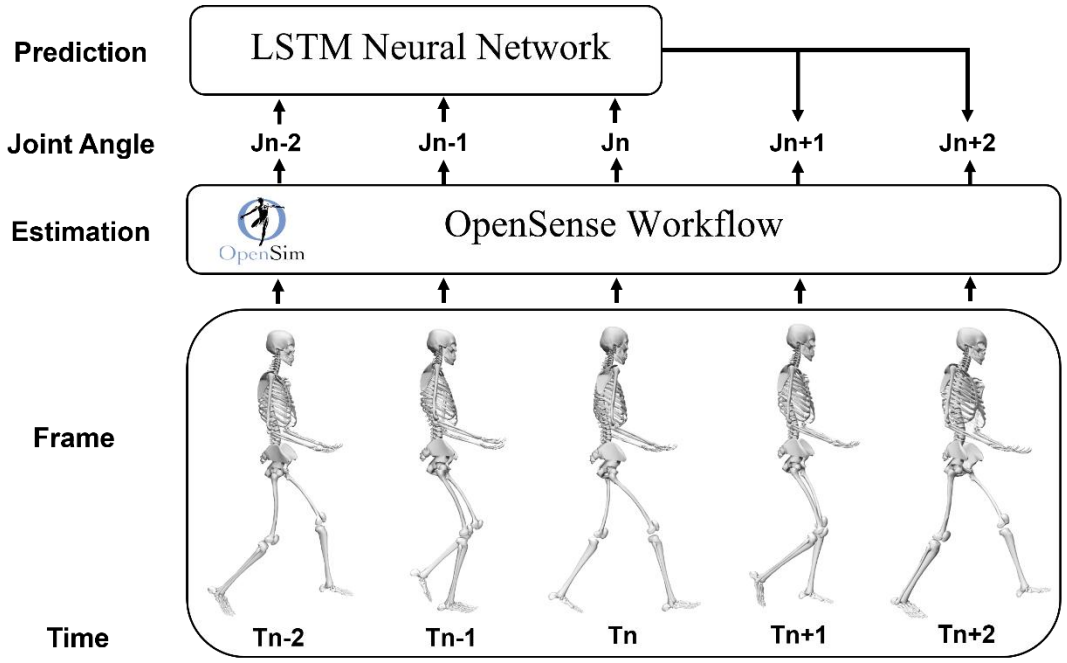


Fig 2.5 Flowchart of human motion intention Prediction

To predict both the upper limb motion and lower limb motion, Huang, et al. applied a structure of deep neural networks named recurrent neural network(RNN) to predict joint angles based on a combination of data from IMU sensors and EMG electrodes[25]. The method achieved a low error of 2.93 degrees within the prediction horizon of 50 ms. In this work, the fusion of IMU data and EMG data both contribute to the high accuracy of the prediction result: IMU is a promising way for accurate human motion capture. Meanwhile, the EMG data is normally several ten ms ahead of kinematic movements, thus it could provide information for motion intention prediction. Liu, et al. predicted the joint angles with pure sEMG data by integrated muscle synergy theory with generalized regression neural network(GRNN) [26] . Unlike RNNs, this work extracted the time-domain features of sEMG signals and

compressed them into a synergy matrix and an activation coefficient matrix. Then, a GRNN was applied to build the relationship between activation coefficients and joint angles. This work gets a result of a coefficient of determination of 0.933 on average. Information for prediction was provided only by sEMG data in this research. Gautam, et al. proposed an long-term recurrent convolutional network(LRCN) named MyoNet to predict joint angles [27]. This work performed the joint angle prediction by using end-to-end deep learning technology. The method was validated by three lower limb functional movements and got an error result of 8.1% in the healthy group and 9.2% on the group with knee pathology. Jiehe, et al. proposed an LSTM-based human motion intention prediction method with the use of Kinect visual data[28]. This study focused on predicting lower limb movements while walking on the treadmill. Both upper limb swing and lower limb swing data were collected. The hip and knee angles were predicted by the LSTM model with the input of visual data of the elbow and shoulder on the opposite side. The most obvious difference between this reference and others in this part is that applying visual data of one body area to predict joint angles in other areas of the human body. Xie, et al. proposed a limb joint angle prediction method by utilizing GS-GRNN[29]. The hip joint angles and lower limb EMG signal were used as the input of the neural network to predict lower limb joint angles. The result revealed that GRNN has better performance than BP Neural Networks in joint angle prediction.

There is no comprehensive study to predict human motion with pure IMU data. Thus, this paper presents a hybrid method of the skeletal model and LSTM to predict the

lower limb joint angles with pure IMU data, which has the potential to be a low-cost, easy-to-use alternative in motion prediction.

Chapter3 IMU-based joint angle estimation and validation

To estimate the lower limb joint angles, we start with the kinematic relationship between two rigid bodies. Given two rigid bodies A and B: while describing the motion of a rigid body B relative to another rigid body A, two reference frames are required to be built on the two rigid bodies. We assume that the reference frame on body A is still, which is defined as the global reference frame. The reference frame on rigid body B is moving relative to the global frame, which is defined as the local reference frame. The movement of the rigid body A is divided into translation and rotation. In this study, we focus on the joint angle calculation rather than tilt between body segments. Thus, only rotation between rigid bodies will be discussed in this thesis.

Usually, to calculate the joint angles from the rotation matrix output from the IMU, researchers process the rotation matrix corresponding to each frame. For simple tasks, reference frames rotate without any constraint. Problems of this sort are classified as unconstrained inverse kinematics problems. The solution to the unconstrained inverse kinematic problem is performed by multiplication of the rotation matrix and its inverse matrix. SO(3) space is a lie group whose elements are three-dimensional rotation matrices[30]. The tangent space of a SO(3) space is denoted as so(3), whose elements are skew-symmetric matrix. The multiplication of matrices in the rotation matrix space SO3 corresponds to the superposition of rotations. The inverse of a rotation matrix represents exchanging the local reference frame and the global

reference frame. Apart from unconstrained IK problems, there exist various constraints in the process of joint angle calculation. For example, when we calculate the knee joint with the rotation matrices of the thigh segment and the shank segment, the shank mainly acts in the sagittal plane and the other two directions possess limited movement range. Besides, the joint only act in a specific range.

The solution of the inverse kinematics with constraints is solved by transforming it into an optimization problem. The constrained inverse kinematics could also be classified into two groups: orientation-based inverse kinematics and marker-based inverse kinematics. The previous inverse kinematics problem takes the rotation matrices measured by IMUs as input while the following one takes marker trajectories in the global reference frame as input. The orientation-based inverse kinematic problem is similar to the marker-based inverse kinematic problem in that both express the loss function in terms of generalized coordinates and then search for the value of the generalized coordinate that minimizes the loss function ordered by the constraints through optimization methods.

IMU-based human motion tracking technologies play a significant role in joint angle estimation and motion reconstruction due to being free of test occasions and low cost, which makes them an excellent replacement for marker-based motion tracking when orientations and positions are collected outdoors[31]. Original data collected by IMU could contain nine-axes data, including three-axis acceleration, three-axis angular velocity, and three-axis magnetic field. The sensor fusion algorithm such as the Kalman filter and complementary filter will be applied to the original data to generate

estimated orientation data of every IMU[32]. By conducting some calibration poses, a transformation matrix could be used to calculate and transform the orientation from the IMU frame to the corresponding segment frame. After the calibration process, joint angles could be estimated by applying some optimization methods to the orientation data based on the musculoskeletal model [33] . Apart from model-based joint angle estimation, inverse kinematics problems could also be solved by applying machine learning methods either based on the raw data collected by IMUs or orientation data.

The study in this chapter aimed to propose a novel IMU-based joint angle estimation method. When compared to the marker-based motion capture system, it has the advantages of 1)It does not suffer from the gimbal lock. 2) It outperforms the Euler-based method when it comes to iteration required to converge to a fixed value. 3) To validate the method, we collected raw data using Trigno Avanti (Delsys, USA) and calculated the joint angle in the MatLab platform based on some optimization algorithm. The validation data was collected from the TM5-900 robot arm (Techman, Taiwan).

3.1 Representation of 3D Rotation

The orientation of a reference frame with respect to another reference frame can be represented by Euler angles, rotation matrix, and quaternions[34].

3.1.1 Euler Angle

Euler angles are a set of angles used to represent the rotation of a reference system, usually denoted as (α, β, γ) . These three angles represent the counterclockwise rotation of the reference frame around the corresponding coordinate axes. In general, Euler angles could be defined as either intrinsic rotations (around local coordinate axes) or extrinsic rotations (around global reference axes). A local coordinate axis stands for a coordinate axis in the local reference frame while a global coordinate axis stands for one in the global reference frame. Depending on the axis of rotation and the order of rotation, the intrinsic Euler angles are divided into 12 types, z-x-z, x-y-x, y-z-y, z- y-z, x-z-x, y-x-y, x-y-z, y-z -x, z-x-y, x-z-y, z-y-x, y-x-z. Similarly, the extrinsic Euler angles are classified into 12 types, Z-X-Z, X-Y-X, Y-Z-Y, Z-Y-Z, X-Z-X, Y-X-Y, X-Y-Z, Y-Z-X, Z-X-Y, X-Z-Y, Z-Y-X, Y-X-Z.

The advantage of Euler angles is that there are fewer parameters and only three angle values are required to represent a rotation. In addition, Euler angles are more intuitive than the other two rotational representations. The Euler angle representation has four drawbacks: (1) the Euler angle suffers from gimbal deadlock, which means there might be missing degrees of freedom in specific positions. (2) Euler angles could not be numerically calculated and calculations must be made with the assistance of a rotation matrix. (3) there is a larger error in the interpolation calculation, as the rotation is performed on three axes. (4) there are 24 representations of Euler angles, which need to be converted for different applications. Therefore, the Euler angle representation is more suitable for the rotation visualization while the numerical

calculation is usually performed with a rotation matrix or quaternions.

3.1.2 Rotation Matrix

The rotation matrix is a 3*3 matrix, which is a common rotation representation. The rotation matrix is often denoted as *R*. The rotation matrix *R* has the following properties: (1) The row and column vectors are unit vectors that are orthogonal to each other. (2) The determinant of the matrix is 1. (3) $RR^T = I$ i.e., the transpose matrix of a rotation matrix is its inverse matrix. The order of rotation needs to be specified, a three-dimensional rotation corresponds to a unique rotation matrix. Besides, the calculation of rotation matrices can be performed directly by multiplication and transposition. A rotation matrix contains nine elements, which makes it more time-consuming and storage-consuming during the calculation process.

3.1.3 Quaternion

A quaternion consists of a real part and three imaginary parts, which is often denoted as $q = q_1 + q_2 i + q_3 j + q_4 k$, $(q_1, q_2, q_3, q_4 \in R)$. Rotation is usually expressed in terms of unit quaternions. The quaternion operation has the following property: $i^2 = j^2 = k^2 = ijk = -1$. In comparison with the other two rotational representations, the quaternion representation could avoid the lack of degrees of freedom caused by the gimbal locking. At the same time, the quaternion representation requires only a 4-dimensional unit vector to represent arbitrary rotations, which in some cases is more computationally efficient than a rotation matrix. In addition, quaternion rotations can

provide smooth interpolation. The disadvantage of the quaternion representation is that it is more complex than the Euler angle representation, as it is more difficult to understand with an extra dimension.

3.2 Optimization Theory on Joint Angle Estimation

The output of an IMU device is in the form of a rotation matrix. By means of calibration, the movement over time of the anatomical reference frame of the body segment to which the IMU is attached can be obtained.

1 3.2.1 Unconstrained Joint Model

For an unconstrained joint model with three rotational degrees of freedom, its rotation matrix can be denoted as R_B^A , where A and B are the reference frames of A and B for two adjacent segments, and R is the rotation matrix of B with respect to A. Then this matrix can be calculated by the following equation:

$$R_B^A = R_{A_G}^A R_G^{A_G} R_{B_G}^G R_B^{B_G} \tag{3.1}$$

Where A_G denotes the IMU reference frame of segment A, B_G denotes the IMU reference frame of segment B_G denotes the global reference frame.

In order to calculate the joint angles, further conversion of the rotation matrix into Euler angles is required.

3.2.2 Constrained Joint Model

When it comes to human motion tracking tasks, human synovial joints are often modelled with limited rotational degrees of freedom, representative of their functional

movement and anatomy. The knee, ankle, and hip are three representative lower limb joints modelled with a different number of degrees of freedom (DOF) according to anatomy. The knee joint can be modelled (Fig 3.1) as a 1DOF with flexion/extension movement. The ankle joint can be modelled (Fig 3.2) as a 2DOF joint with both flexion/ extension and lateral rotation. The hip joint model (Fig 3.3) is a 3DOF joint, including flexion/ extension, lateral rotation, and adduction/ abduction.

Fig 3.1 Knee Joint Model

Fig 3.2 Ankle Joint Model

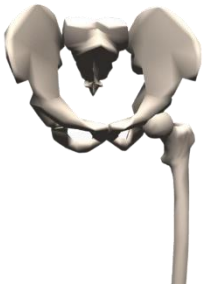


Fig 3.3 Hip Joint Model

The constrained joint models make the joint angle calculations difficult to be obtained directly by a rotation matrix. Thus, optimization methods are utilized to calculate joint angles.

In the orientation-based inverse kinematics method, the norm-2 square of the Euler angular error is often taken as the cost function. Thereby, the problem is transformed into a least-squared problem in order to minimize the mismatch of the segmentation orientation[32]. The cost function is defined as below:

$$J = \frac{1}{2}\sum_{k} w_k \left\| \theta_k - \theta_g \right\|^2 \tag{3.2}$$

Where *wk* denotes the weight of the *kth* body segment, $\theta_k = \begin{bmatrix} \alpha_k & \beta_k & \gamma_k \end{bmatrix}^T$ denotes the Euler angle error of the *kth* body segment.

The gradient descent (GD) method (Boyd, S., Vandenberghe, L., 2004) is usually applied to find the optimal solution to the problem. First, the gradient is calculated as:

$$\nabla = \begin{bmatrix} \frac{\partial J}{\partial \alpha} & \frac{\partial J}{\partial \beta} & \frac{\partial J}{\partial \gamma} \end{bmatrix}^T \tag{3.3}$$

Where: $\frac{\partial J}{\partial \alpha} = \alpha - \alpha_{given}$, $\frac{\partial J}{\partial \beta} = \beta - \beta_{given}$, $\frac{\partial J}{\partial \gamma} = \gamma - \gamma_{given}$.

In each calculation epoch, the algorithm can be updated with the following formula until a satisfactory result is obtained:

$$\theta = \theta - r * \nabla \tag{3.4}$$

Where r is the learning rate of the algorithm which should be specified before the loop.

3.2.3 Proposed Joint Angle Estimation Model

As mentioned above, the Euler angle-based representation has the drawback of gimbal lock, which might disable the optimization method when it deals with some special angle values. Thus, it is of great significance to find a method free of gimbal lock to reconstruct the cost function.

From the perspective of mathematics, Riemannian based representation is the intrinsic representation for 3D rotation[35]. A Lie group is a group that is also a differentiable manifold. The SO(3) group is a Lie group with three degrees of freedom, used to describe the rotation of a rigid body in three dimensions. the elements of SO(3) are often represented by a three-dimensional matrix, which is also known as the rotation matrix. the characteristics of SO(3) are twofold: on the one hand, the determinant of

the elements of SO(3) is equal to 1; on the other hand, the transpose of the elements of SO(3) is equal to its inverse matrix. In this thesis, we denote the SO(3) group as:

$$SO(3):\{R\in\mathbb{R}^{3\times3}\mid RR^{T}=I,\det(R)=1\} \tag{3.5}$$

Where R is the element of the assembly.

Correspondingly, the Lie group SO(3) corresponds to the Lie algebra so(3) for inducing exponential maps on SO(3) space. so(3) space is essentially the tangent space of SO(3) space. elements on so(3) are also three-dimensional matrices with the following properties: the transpose of a matrix is the opposite matrix of that matrix (i.e., the corresponding w elements sum to 0). The so(3) space is described as below.

$$so(3):\{\hat{\omega}\in\mathbb{R}^{3\times3}\mid\hat{\omega}^{T}=-\hat{\omega}\} \tag{3.6}$$

Where $\hat{\omega}$ denotes the element of the so(3) assembly, which has the below structure.

$$\hat{\omega}=\begin{bmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{bmatrix} \tag{3.7}$$

$\omega_1\ \omega_2\ \omega_3$ are three elements of the matrix .

The elements on the so(3) space, the exponential operation can be defined as follows: theta is the angle of rotation under the axial angle method representation. When the angle theta is equal to 0, the corresponding result is the unit matrix I. And when theta is not equal to 0, the corresponding result is calculated by the formula in the second row below.

$$e^{\hat{\omega}}=\begin{cases}I & \theta=0\\ I+\dfrac{\sin\theta}{\theta}\hat{\omega}+\dfrac{1-\cos\theta}{\theta^{2}}\hat{\omega}^{2} & \theta\neq0\end{cases} \tag{3.8}$$

Where $\theta=\sqrt{\omega_1^2+\omega_2^2+\omega_3^2}\quad\theta\in[0,\pi)$, which means $\hat{\omega}$ is equal to the rotation axis

multiplies the rotation angle in axis-angle representation.

Similarly, the logarithmic operation can be expressed as follows: when theta is equal to 0, the result is a zero matrix; when theta is not equal to 0, the result is calculated by the second row of the formula.

$$\log(R) = \begin{cases} 0 & \theta = 0 \\ \dfrac{\theta}{2\sin\theta}(R - R^T) & \theta \neq 0 \end{cases} \tag{3.9}$$

In the optimization method with Euler's angle as the rotational representation, the loss function is the square of the Euclidean distance between two elements in the three-dimensional linear space where Euler's angle is located. This distance is taken as the Euclidean distance between the set of Euler angles to be optimized and the target Euler angle in the process of calculating the optimal solution to characterize the error, which is then minimized to approximate the optimal solution.

However, this approach has two potential shortcomings: on the one hand, the use of Euler angles to represent rotation is subject to gimbal self-locking at some specific positions: for some joints with a limited range of motion, this shortcoming is not affected. However, when dealing with joints with multiple degrees of freedom and a large range of motion such as the shoulder joint, such algorithmic limitations may manifest themselves. On the other hand, designing the loss function on the 3D linear space where the Euler angles are located, the corresponding step size may be inhomogeneous with respect to the rotation space.

Therefore, it is necessary to develop optimization algorithms using representations that are more closely aligned with the nature of rotation. In this study, we use the

Riemann distance of the two rotation matrices on the SO(3) space to characterize the difference between the two rotations. The Riemann distance on the SO(3) space is calculated as follows.

$$d_R(R_1, R_2) = \frac{1}{\sqrt{2}} \left\| \log(R_1^T R_2) \right\|_F \tag{3.10}$$

Where R1, R2 respectively represent two rotation matrices in the SO(3) space, and *Log()* is the logarithmic operation defined in the formula 3.10. $\|\bullet\|_F$ stands for the Frobenius norm of a matrix.

Besides, the Riemannian distance could also be approximated as below according to the B-C-H formula. We need to perform the approximation in this thesis as analytic solution of the raw formula is difficult to calculate when it comes to gradient matrix calculation.

$$d_R(R_1, R_2) \approx \frac{1}{\sqrt{2}} \left\| \log(R_2) - \log(R_1) \right\|_F \tag{3.11}$$

In this study, we define the cost function F as half of the square of the Riemannian distance defined in formula 3.11.

$$F = \frac{1}{2} d^2 \tag{3.12}$$

i.e.:

$$F = \frac{1}{2} \left(\frac{1}{\sqrt{2}} \left\| \log(R_m^T R) \right\|_F \right)^2 \tag{3.13}$$

R represents the variable to be optimized in the minimization algorithm and Rm stands for the target rotation matrix calculated from the reference data. The reference data could be either in the format of normal rotation representation, including Euler

angle, rotation matrix, quaternion, and axis-angle representation, or generalized coordinates.

If we directly calculate the F-parametrization of the loss function, we find that we cannot separate the two variables Rm and R, which in turn makes the calculation much more difficult. Therefore, in this paper, we consider the BCH formula to simplify the original loss function formula, i.e.:

$$F = \frac{1}{2}\left(\frac{1}{\sqrt{2}}\left\|\log(R) - \log(R_m)\right\|_F\right)^2 \tag{3.14}$$

The logarithm of a rotation matrix yields the antisymmetric matrix corresponding to that rotation matrix, expressed as follows.

$$\hat{\omega} = \log(R) \tag{3.16}$$

Thus, the formula 3.14 could be simplified as:

$$F = \frac{1}{2}\left(\frac{1}{\sqrt{2}}\left\|\hat{\omega} - \hat{\omega}_m\right\|_F\right)^2 \tag{3.17}$$

For a matrix R, its F-parametrization is equal to the trace of the product of it and the transpose matrix. Then, the above formula could be written as:

$$F = \frac{1}{2}\frac{1}{2}tr\left((\hat{\omega} - \hat{\omega}_m)(\hat{\omega} - \hat{\omega}_m)^T\right) \tag{3.18}$$

Then, we could represent the cost function F with three parameters of the antisymmetric matrix $(\omega_1, \omega_2, \omega_3)$, which are also the three elements of the product of the rotation axis and the rotation angle in the axis-angle representation. And the cost function will be:

$$F = \frac{1}{2}(\omega_1 - \omega_{1m})^2 + \frac{1}{2}(\omega_2 - \omega_{2m})^2 + \frac{1}{2}(\omega_3 - \omega_{3m})^2 \tag{3.19}$$

However, if we directly use the coordinate set $(\omega_1, \omega_2, \omega_3)$ to minimize the cost function, the algorithm will also suffer some restrictions. First, w is the product of the rotation angle and the rotation axis, and the rotation angle varies from -180 degrees to 180 degrees, implying that the set of coordinates is little affected by the perturbation when the rotation angle is near 0 degrees, and much affected by the perturbation near 180 degrees. Second, at an angle of 0, the loss function decreases to 0, so that the same Euler-like self-locking phenomenon occurs. Finally, the product of axes and angles makes a coupling of two independent variables, which is not recommended in the mathematics.

To prevent the proposed algorithm from the drawbacks mentioned above, the coordinate set should be highly independent from each other. The coordinate set $(\omega_1, \omega_2, \omega_3)$ is characterized by twofold: one is that its vector modulus length is equal to a specific value; the other is that after normalization the three elements of it lie on a sphere in the 3D space whose radius is 1. Therefore, we intended to utilize the spherical polar coordinates as a set of generalized coordinates to work as the variable to be optimized. The conversion relationships are shown below.

$$\omega_1 = \theta \cos\theta_1 \cos\theta_2 \tag{3.20}$$

$$\omega_2 = \theta \cos\theta_1 \sin\theta_2 \tag{3.21}$$

$$\omega_3 = \theta \sin\theta_1 \tag{3.22}$$

The gradient matrix of the loss function to the spherical polar coordinates is found by the matrix derivative chain rule.

$$\nabla = \begin{bmatrix} \frac{\partial F}{\partial \omega_1} \\ \frac{\partial F}{\partial \omega_2} \\ \frac{\partial F}{\partial \omega_3} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \omega_1}{\partial \theta_1} & \frac{\partial \omega_1}{\partial \theta_2} & \frac{\partial \omega_1}{\partial \theta} \\ \frac{\partial \omega_2}{\partial \theta_1} & \frac{\partial \omega_2}{\partial \theta_2} & \frac{\partial \omega_2}{\partial \theta} \\ \frac{\partial \omega_3}{\partial \theta_1} & \frac{\partial \omega_3}{\partial \theta_2} & \frac{\partial \omega_3}{\partial \theta} \end{bmatrix} \tag{3.23}$$

Replace the corresponding parts of the above equation with analytical solutions of the corresponding partial derivative.

$$\begin{bmatrix} \frac{\partial \omega_1}{\partial \theta_1} & \frac{\partial \omega_1}{\partial \theta_2} & \frac{\partial \omega_1}{\partial \theta} \\ \frac{\partial \omega_2}{\partial \theta_1} & \frac{\partial \omega_2}{\partial \theta_2} & \frac{\partial \omega_2}{\partial \theta} \\ \frac{\partial \omega_3}{\partial \theta_1} & \frac{\partial \omega_3}{\partial \theta_2} & \frac{\partial \omega_3}{\partial \theta} \end{bmatrix} = \begin{bmatrix} -\theta \sin\theta_1 \cos\theta_2 & -\theta \cos\theta_1 \sin\theta_2 & \cos\theta_1 \cos\theta_2 \\ -\theta \sin\theta_1 \sin\theta_2 & \theta \cos\theta_1 \cos\theta_2 & \cos\theta_1 \sin\theta_2 \\ \theta \cos\theta_1 & 0 & \sin\theta_1 \end{bmatrix} \tag{3.24}$$

$$\begin{bmatrix} \frac{\partial F}{\partial \omega_1} \\ \frac{\partial F}{\partial \omega_2} \\ \frac{\partial F}{\partial \omega_3} \end{bmatrix} = \begin{bmatrix} \omega_1 - \omega_{1m} \\ \omega_2 - \omega_{2m} \\ \omega_3 - \omega_{3m} \end{bmatrix} \tag{3.25}$$

Compared with the Euler angles-based method, the method proposed in this study avoids gimbal lock and variables being affected differently by perturbations at different positions. However, the introduction of generalized coordinates leads to a more complicated calculation of the gradient matrix, i.e., more computational resources are required. This problem is against our original intention of proposing the algorithm, so we consider further improvement of the algorithm. The bottleneck we encounter in this algorithm is the computational overload due to the fact that we use both the Riemann distance in the SO(3) space to construct the loss function and the

generalized coordinates guided by the parameters on this space as the variables to be optimized. It is not difficult to find that the key to avoid gimbal lock and uneven scaling of variable steps is to use the bootstrapped coordinates. Therefore, we imitate the Euler angle-based optimization algorithm and construct the loss function directly using the generalized coordinates.

In this study, we proposed a novel method with the representation of a set of generalized coordinates to search for the optimal solution in the range.

Firstly, we introduced a set of generalized coordinates based on the axis-angle representation. When dealing with three-dimensional space problems, it is often necessary to characterize spatial rotations with a 3*3 rotation matrix. This representation is usually the most convenient because multiplying a vector by that matrix is equivalent to rotating it in a corresponding way. The inconvenience is that it does not visualize the meaning of the rotation of a 3*3 matrix.

Another easy way to visualize the representation is to represent the rotation in vector form, and that rotation operates with a single angle at a time. In this case, the most standard way to illustrate the rotation around the coordinate axes is to use only one vector, the direction of the vector is determined by the right-hand rule, and the length of the vector indicates the angle of counterclockwise rotation around the axis, i.e., the axis angle method.

The axis-angle method and the rotation matrix method can be translated using the Rodriguez transformation. When we want to represent a rotation operation, we first define an axis which the local reference frame rotates around.

$$X = [x_1 \quad x_2 \quad x_3]^T \tag{3.26}$$

The vector X represents the rotation axis of the corresponding rotation operation, whose components respectively stand for its *x*,*y*,*z* coordinate in the global reference frame.

Similarly, the rotation angle of the rotation operation is represented as θ, which forms the axis-angle representation of a rotation operation together with the vector *X*.

To calculate the rotation matrix with the axis-angle representation, the Rodriguez formulation is often used.

$$R = I + \hat{X}\sin\theta + \hat{X}^2(1-\cos\theta) \tag{3.27}$$

Where *R*denotes the rotation matrix, $\hat{X}$ stands for the antisymmetric matrix of *X*. For a matrix *A*, if it satisfies this property: $A = -A^T$, it's denoted as a antisymmetric matrix. The antisymmetric matrix of a 3-dimensional vector could be calculated by the below formula:

$$\hat{X} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{3.28}$$

Considering that the axial angle method can represent rotation intuitively, one can try to use this representation to construct the loss function. But the three components of the vector *X* are not completely independent of each other. We know that in the axis-angle representation, the modulus of the rotation axis *X* is equal to 1, which makes the direct use of the axis-angle method to construct the loss function impose additional constraints and make the solution of the optimization problem more difficult. Therefore, we consider introducing generalized coordinates to represent the vector *X*.

The vector X is known to have modulus 1, so it can be viewed as a vector pointing from the origin to a point on the sphere of radius 1. Therefore, it can be replaced by a spherical polar coordinate system.

We introduce two spherical polar coordinates $\phi \in [0, 2\pi]$, $\varphi \in [0, \pi]$.

The conversion relationship between the spherical polar coordinate system and the Cartesian coordinate system is as follows:

$$x_1 = \sin\varphi \cos\phi \tag{3.29}$$

$$x_2 = \sin\varphi \sin\phi \tag{3.30}$$

$$x_3 = \cos\varphi \tag{3.31}$$

Similarly, when we calculate the spherical polar coordinates with Cartesian coordinates, we normally use the below formulations:

$$\phi = \arctan(\frac{x_2}{x_1}) \tag{3.32}$$

$$\varphi = \arccos(x_3) \tag{3.33}$$

The two spherical polar coordinates obtained by the right-angle coordinate-sphere polar coordinate transformation of the rotation axis X have no additional constraint constraints in the definition domain and can form a new set of generalized coordinates together with the rotation angle. Denote the generalized coordinate set as G. Imitating Equation 2.1, we use half of the 2-parametric square of the generalized coordinate error as the cost function.

$$f = \frac{1}{2}\left\| G - G_t \right\|^2 \tag{3.34}$$

Where f stands for the cost function, G_t denotes the reference value calculated from

the measurement data.

The partial derivatives are found for the loss function and gradient descent is taken to find the optimal solution. The derivatives of f with respect to the three generalized coordinates are found separately to obtain the gradient matrix.

$$\nabla=\left[\begin{array}{ccc}\frac{\partial f}{\partial \varphi} & \frac{\partial f}{\partial \phi} & \frac{\partial f}{\partial \theta}\end{array}\right]^{T} \tag{3.35}$$

Where φ , ϕ , θ respectively denotes three components of the generalized coordinate set G.

To continually search the optimal solution for the IK problem, we need to update the vector G with a update formula. The update formula for the gradient descent method is:

$$G = G - \alpha * \nabla \tag{3.36}$$

Where α is the learning rate of the optimization method.

Constraint:

(1) In Euler-based method, for a set of Euler angles $\theta = \left[\alpha \quad \beta \quad \gamma\right]^{T}$, if in any degree of freedom, the joint is constrained, then the corresponding theta will be set as 0.

(2) In R-distance-based method and the proposed method, for a set of generated coordinates $\psi = \left[\varphi_1 \quad \varphi_2 \quad \varphi_3\right]^{T}$, if :

(a) move around X axis: $\varphi_1=0$, $\varphi_2=0$.

(b) move around Y axis: $\varphi_1=\frac{\pi}{2}$, $\varphi_2=0$.

(c) move around Z axis: $\varphi_1=0$, $\varphi_2=\frac{\pi}{2}$.

3.3 Introduction to Techman Robot Arm

The OB-IK optimization algorithm could estimate the joint angle between two adjacent body segments with the input of rotation matrices of corresponding body segments. However, it's difficult to evaluate a method without reference. In order to effectively validate the human motion tracking workflow proposed above, a validation protocol is required to act as a golden standard. So far, validation protocols based on X-ray or high-speed cameras have been applied in the past decades as golden standards for motion tracking, which are characterized by high accuracy, high reliability, and low risk. However, to act as validation methods, they normally suffer from errors caused by body artefacts, which might be different among different groups. Thus, it is necessary to develop a validation protocol free of artefact error to work as a golden standard for joint angle estimation. The artefact error caused by subjects could be reduced by applying extra experimental operations or optimization algorithms. The artefact error is caused by soft tissues like skin, muscle, and fat and people with lower BMI/Body Fat rates tend to have less muscle and fat. To avoid the artefact error, we introduced a novel validation protocol where the robot arm was applied as the research subject. Robot arms are rigid bodies that are free of translation between the IMU sensor and the segment it attaches to during the experiment process. When we firmly attach an IMU on a link of the robot arm, it will move with the link with almost no translation. Moreover, a robot arm could perform functional actions like a human with high precision thanks to the employment of servo technologies, which makes the experiment highly repeatable.

TM Robot Arm is a 6-axis collaborative robot with force and power limiting capabilities, user-friendly programming, innovative vision integration and the latest safety measures to operate at full speed within barriers and in collaborative areas (Fig3.4). In this study, a TM5-900 robot arm was selected to perform basic operations to simulate human movements. The TM5 has a workspace of 900mm and a max payload of 4 kg, which ensures it perform some complicated tasks. The robot arm contains eight segments, including the base, the 1st segment, the 2nd segment, the 3rd segment, the 4th segment, the 5th segment, the 6th segment, and the end effector. Denote the base as a proximal end and the end as the distal end. From the proximal end to the distal end the joints was named the 1st joint, the 2nd joint, the 3rd joint, the 4th joint, the 5th joint, and the 6th joint (Fig 3.5). The TM robot consists of a robot arm, a control box, and a control panel. The robot arm takes responsibility for performing operations delivered by the control box and sending signals from the force sensor/ the camera to the control box. A flowchart-based operation is developed for the TM robot arm to make it accessible to more researchers. The control box receives signals from the operating system and sends control commands to the robot arm. The start/pause/end state and speed of the robot arm could be controlled by the control panel during the experiment.

Fig 3.4 TM5-900

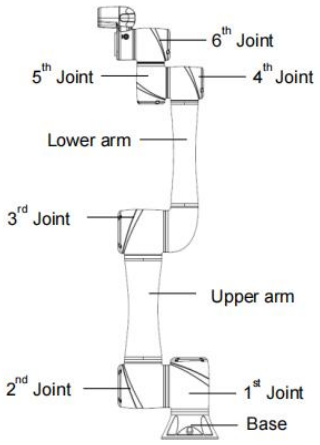


Fig 3.5 TM5-900 Structure

3.4 TM5 Robot Modelling

In order to simulate the robot arm movements in the OpenSim, a TM5 robot model in .osim format is required to be created from scratch. The OpenSim platform provides a workflow for multibody IK solution, with the input of MSK model, parameters configuration, and There are various ways to create a model in OpenSim, among which Simbody based programming and XML based programming. Simbody is a multi-body physics API in OpenSim which is developed for modelling. MSK models or any multi-body model could be generated by calling the API in C++. The rigid boy segments generated by Simbody can be connected with joints to make up a model. Force and constraints could also be defined in the interface. Like Simbody, a.osim model file could also be created by programming in XML. The advantage of Somebody is that all the processes in the API are highly repeatable as only parameters of the multi-body are required during the modelling process and there are various templates online. In this study, the TM5 robot arm model was generated by the XML

based methods due to its convenience and access to every modelling detail.

No dynamic parameters like mass or inertial moments were set during the modelling process as this model was designed for validation of a human motion capture pipeline (i.e., an inverse kinematics problem). The model in OpenSim is expected to possess the same parameters as the TM5 robot arm, including mechanical structures, size of segments, joint ranges, and joint types.

An entire TM5 model in .stp format was downloaded and then converted into .stl format. in Solidworks as the OpenSim is only compatible with .stl, .vtp, or .obj files. After the model of the segments were created, a series of parameters were set in the OpenSim model file model TM5.osim according to the Techman5 robot arm configurations. The size of segments in OpenSim model file depends on the size of model imported from Solidworks. Thus, no extra configuration of body size was required. Every joint in a TM5-900 contains a single degree of freedom, so joints in the model file were set as the sphere joint with two degrees of freedom locked. Body segments of the robot arm model were connected by the six joints in the sequence of segment number and the joint number. The base was attached to the origin of the ground with six degrees of free, i.e. three dimensional translation and three dimensional rotation. Then every segment was attached to its parent segment with the corresponding joint. The modelling process is conducted with reference to the reference frame of the segments, which were specified in the Solidworks. The end effector was attached to the last segment with the 6th joint. The assembling was followed by parameters configuration, where the max joint speed and the max/ min

joint position were set. The max speed of the first three joints is 190 degrees per second while that of the other three is 235 degrees of freedom. The six joints have the max joint position of +/-270 degs, +/- 180 degs, +/- 155 degs, +/- 180 degs, +/- 180 degs, +/-270 degs (Table 3.1). Moreover, there was no translation movement enabled for any joints except the base-ground joint due to the experiment refers to rotation motion only.

Table 3.1 Joint Parameters

	Joint Speed/dps	Min/Max Joint Position/deg
1st Joint	190	-270/ 270
2nd Joint	190	- 180/ 180
3rd Joint	190	- 155/ 155
4th Joint	235	- 180/ 180
5th Joint	235	- 180/ 180
6th Joint	235	-270/ 270

3.5 Experiment Design

To provide a dataset for proposed algorithm validation, we designed an experiment with the assistance of advanced robotic technologies. The experiment was conducted with a Delsys Trigno Avanti system and a Techman5-900 robot system. The Delsys Trigno Avanti system is a device capable of bio-signal capture, including EMG signals and IMU signals. When attached on a body segment, the device could detect the EMG signal under the skin in a non-invasive way. In this experiment, we only take IMU data as input, so the EMG mode was disabled. When it comes to inertial data measurement, the Delsys Trigno has two classic modes: one is orientation mode, which directly gives out the rotation matrices or quaternions over time estimated by

the embedded CPU; the other is inertial data mode, which output gyroscope data and accelerometer data. In this study, we directly outputted the quaternions matrices as the input. The TM5-900 robot arm is an advanced robot arm capable to undertake comprehensive tasks. It contains various motion modes and could be controlled by the command panel/ flow-chart based command programming. No human subjects were recruited in this experiment.

The target of the experiment is to measure the joint angle estimation accuracy of the human motion capture pipeline. In order to ensure that the motion tasks designed for the robotic arm can as much as possible represent the characteristics of the motion of the human joints, single joint trials are needed to simulate joint movement in one degree of freedom. In contrast, to simulate comprehensive joint motion tasks of specific joint like the hip, a multi-joint trial is required where movement tasks are performed by several joints simultaneously. Thus, single joint trials along with multi-joint trials were conducted with the robot arm work in different state to simulate human movements. In single joint trials, we chose the robot arm joint between base link and link 1 to perform the movement task as the axis along the joint is vertical to the ground. Similarly, other two sort of trials where joint moves around global Y axis and global Z axis were designed.

Four IMU sensors were used in this study, which were respectively attached on the base link, link 1, link 4, and link 5 with 3M medical tapes. Thus, only movements of joint 1, joint 4, and joint 5 were tracked while the motion of other unused joints was disabled during the process of data collection.

The tracked robot arm joint moved around different axes with reference to the base frame, in different ranges, and at different speeds. Take joint 1 as example, the base frame was attached on the base link, i.e. the parent rigid body of the joint.

Table 3.2 Movement Around Global Z Axis

			Joint Range		
5% speed(9.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
10% speed(19dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
15% speed(28.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
20% speed(38dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
25% speed(47.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°

Table 3.3 Movement Around Global Y Axis

			Joint Range		
5% speed(9.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
10% speed(19dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
15% speed(28.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
20% speed(38dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
25% speed(47.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°

Table 3.4 Movement Around Global X Axis

			Joint Range		
5% speed(11.75dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
10% speed(23.5dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
15% speed(35.25dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
20% speed(47dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
25% speed(58.75dps)	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°

As every joint of the robot arm only possesses one single degree of freedom, it is difficult to measure three-axis joint motion with one joint. To measure joint movements around different axis, the robot arm was set at a neutral position with the 6 joints at 0 deg, 0 deg, 0 deg, 90 deg, 0 deg, and 0 deg. The movement joint one,

three, and five separately corresponds to the motion around X, Y, and Z-axis. Therefore, the IMU sensors were attached to the base, the first segment, the fourth segment, and the fifth segment. To investigate the relationship between measurement accuracy of IMU-based methods and the movement speed of the rigid body they attached to, the movement speed should be taken into consideration as an experimental parameter. Moreover, it is helpful to find out an optimal speed range for IMU-based joint angle estimation. The joints moved at a constant speed and the speed of the joint motion varied from 5% to 25% in the experiment. The speed was directly controlled by the control panel, thus, we set the movement command in the PC software and change the speed with the control panel. Apart from the movement speed, the motion range is also significant for research IMU-based measurement methods as it reveals the effects of the motion range of tracked rigid body on the measurement accuracy. The joint motion range moves from +/- 30 deg to +/- 90 deg, which was configured in the flow-chart based operation system (Table 3.2, Table 3.3, Table 3.4).

The experiment protocol was designed with a robot arm operation system called TMflow, which aims to provide researchers or engineers with a user-friendly and reliable interface for robot arm control. Unlike most programming environments, the TMflow is a flowchart-based logistic programming environment, ensuring researchers with limited coding experience have access to the robot arm control. In this study, the robot arm needs to perform a calibration pose and then the tracking movements. The calibration pose was designed to calculate the rotation matrices between IMUs and the rigid bodies they attached to. In a calibration pose, all joints of the robot arm were

controlled to keep still in a neutral position while the IMUs recorded the orientation information and in turned fed back it to the PC. The tracking movements were motion tasks in which robot arm moves by following specific control commands. In single joint trials, the tracked joint moved in a specific motion range at a constant speed. Similarly, in multi-joint trials, the tracked joints move simultaneously with the inputted parameters like above.

Take the 'X-axis, +/-45°, 5% speed' trial as an example: Four modules were imported in the trial, i.e., Point, Move, Set, and If (Fig 3.6). The position module could make the robot arm move into a specific position. In this module, there are mainly two modes to set the state of the TechMan 5-900: one is to input the position of the end effector with reference to the base frame, while the other is to set the values of the six joint angles. In this study, we took the joint angles as input to set a Position module. A Position flowchart was first imported to make sure the robot arm was in a neutral position at the start of the trial. The Move module takes the Position module as input, which controls the robot arm move from one position to another at a constant speed. When the robot arm system is conducting(the Move command, it always chooses the shortest route to move from one position to another, which means the motion is uniform with respect to the time scale. And then three Move modules were imported into the trial to make joint 1 of the robot arm move from -45 deg to 45 deg. The SET module could either set a value to an available variable generated before the module. The If module could discriminate whether the conditions are met to determine which branch the process will enter. After that, a Set module and an If module were imported

to discriminate when to stop the movement. A variable ‘flag’ with the initial value of 0 was added by one in this step every loop. At last, the robot arm was set to the initial pose by a Position module in the first step. In the calibration protocol, the end termination condition was set as ‘flag>=1’, which made the robot arm joint 1 move from -45 deg to 45 deg, and then the neutral pose. Movements before the calibration step aimed at IMU sensor activation, which was helpful for the sensors to perform better. The procedure controlled the TM5 to move in the specific range to activate the inertial sensors, which might suffer from drift errors when keep still for a long time. After that, the robot arm stay at the neutral pose for calibration data collection. The calibration data will be sent to the PC in the Delsys software. In the motion tracking protocol, the end termination condition was set as ‘flag>=9999’, making the robot arm keep moving in the range of -45 deg to 45 deg at a constant speed. When set ‘flag>=9999’, we keep the robot arm moving in the given range without termination. After data collection it could be stopped by the control panel.

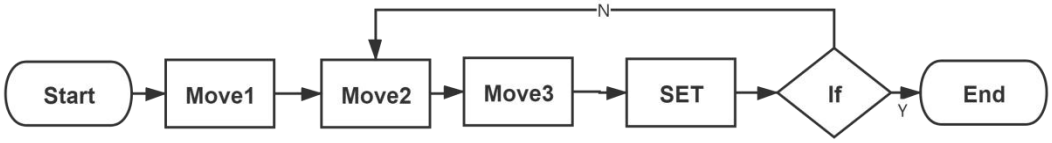


Fig 3.6 Programming Protocol

3.6 Data Processing

In order to quantitatively compare the optimal estimation effects of different methods and find out the relationship between estimation accuracy and measurement parameters, this part was divided into three parts: unconstrained optimization

estimation performance, measurement parameters effects on estimation accuracy, and constrained optimization estimation performance.

To validate the performance of the three optimization methods, we need to test them both under unconstrained conditions and constrained conditions. In this study, we validated the estimation performance under unconstrained condition by adopting a fixed value of orientation. It was represented both in the form of Euler angle and generalized coordinates and then taken as the input of corresponding methods. Considering that we have deduced related formulas or equations based on Euler angle, rotation matrix, and generalized coordinates, we took quaternions as the representation for orientation when it comes to performance comparison. In this part, we calculated RMSD between estimation results and reference data at every iteration to visualize the convergence performance of different methods. After that, the minimum required iterations to decrease the RMSD to 0.1 were calculated to quantify the convergence efficiency. Besides, the required time to run 1000 standard iterations were recorded to compare their running speed.

In the second part, we conducted a series of trials of different rotation axes, motion speeds, and motion ranges to get a comprehensive understanding of what experimental condition is suitable for the motion capture process. We evaluated accuracy of each trial by calculated the RMSD between reference data and the estimated joint angle. In every single-joint trial, there was only one rotation axis which could move. Thus, we assume that the data is reliable and estimate the joint angle at every frame by treating it as an unconstrained joint angle estimation. The

joint angle was calculated by rotation inverse and multiplication.

In the first part of the study, we could compare the computational efficiency between different algorithms and the proposed algorithm. However, to evaluate the performance of an algorithm, it is necessary to verify its accuracy. In the third part, we use multi-joint experiments to act as validation data to compare accuracy of the Euler-based method with the generalized-coordinates-based method. To represent the accuracy of different optimization algorithms, we compare the estimated joint angles with the reference data and calculate the RMSE between them.

3.7 Results and Discussion

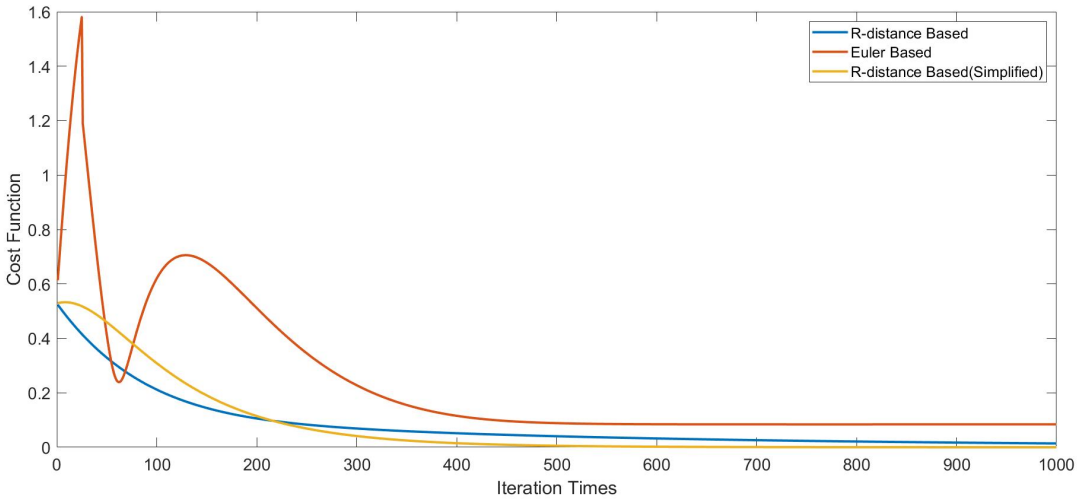


Fig 3.7 Error represented by quaternion: the relationship between iteration and RMSD.

Table 3.5 Iteration Required (RMSD < 0.1)

Method	R-distance based	Euler based	Proposed
Iteration	211	438	214

Table 3.6 Running Time for 1000 Iterations

Method	R-distance based	Euler based	Proposed
Time/s	0.0242	0.0036	0.0035

As shown in Fig 3.7, all three algorithms allow the RMSE to be continuously reduced

and eventually allow the results to converge to the reference value.

The calculation efficiency of a algorithm can be characterized by the number of iterations required to converge to a fixed value and the average time required for a single iteration.

Calculating the RMSE with quaternions representation and counting the number of iterations required by different algorithms to get the RMSE down to 0.1, it turns out that: R-distance based method requires 211 iterations while Euler based method and proposed method respectively require 438 and 214 iterations, as shown in Table 3.5. The number of required iterations are about the same when it comes to the R-distance based method and the proposed method while the Euler based method requires about twice as many iterations as them.

The running time required for different algorithms to complete 1000 iterations could reflect the running time required for different algorithms to complete a single iteration. It reveals that: R-distance based method takes 0.0242 seconds to finish 1000 iterations while Euler based method and proposed method respectively take 0.0036 seconds and 0.0035 seconds, as shown in Table 3.6. The running time of Euler based method and the method are similar, which are far less than that of the R-distance based method.

Thus, the proposed method was assumed to have a better performance than the other two methods when it comes to calculation efficiency.

Table 3.7 RMSE around global Z axis

	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
5% speed(9.5dps)	0.1317	0.2248	0.2883	0.3155	0.3307
10% speed(19dps)	0.4618	0.0696	0.4255	0.3835	0.4644
15% speed(28.5dps)	0.3210	0.9086	1.0430	0.2188	0.4076
20% speed(38dps)	0.2394	0.7045	0.9042	1.7055	2.0223
25% speed(47.5dps)	0.2223	0.5835	0.7359	1.3885	1.6666

Table 3.8 RMSE around global Y axis

	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
5% speed(9.5dps)	0.0793	0.0375	0.0557	0.1835	0.1966
10% speed(19dps)	0.0803	0.1149	0.1001	0.1321	0.3879
15% speed(28.5dps)	0.0899	0.1132	0.1471	0.1964	0.2680
20% speed(38dps)	0.0822	0.0510	0.1800	0.1217	0.2521
25% speed(47.5dps)	0.0777	0.0927	0.2111	0.1833	0.3553

Table 3.9 RMSE around global X axis

	+/-30°	+/-45°	+/-60°	+/-75°	+/-90°
5% speed(11.75dps)	0.0747	0.0731	0.0677	0.1510	0.1764
10% speed(23.5dps)	0.0287	0.1192	0.0986	0.1498	0.2884
15% speed(35.25dps)	0.0435	0.0524	0.1326	0.1827	0.1223
20% speed(47dps)	0.0464	0.0680	0.1446	0.2004	0.6123
25% speed(58.75dps)	0.0964	0.0720	0.1262	0.3617	0.5510

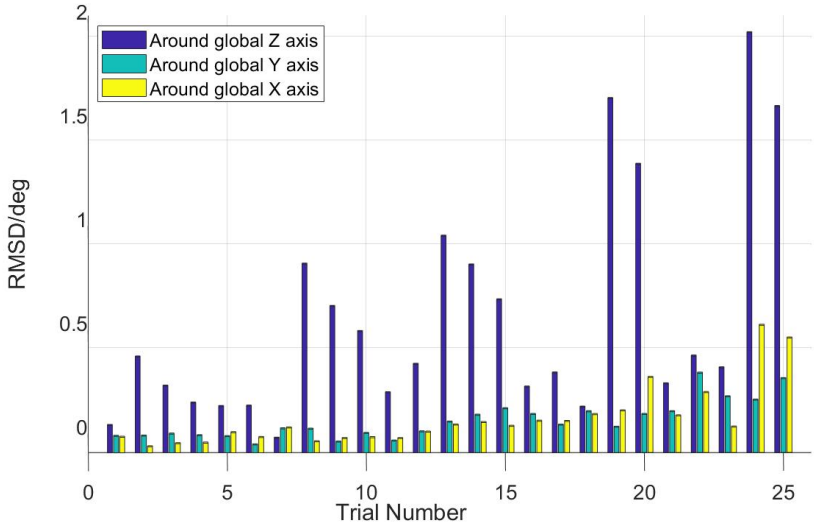


Fig 3.8 RMSE of different single-joint trials

The blue bar is for trials around global Z axis, while the green bar and the yellow bar

are respectively for those around Y and X axis. The 25 trials were divided into five adjacent groups, with the motion range of each group increasing, i. e. +/-30°, +/-45°, +/-60°, +/-75°, +/-90°. The first 5 trials stand for trials with a speed of 5%, 10%, 15%, 20%, 25%. Similarly, in other groups the speed increases as number of a trial increases.

Moreover, an N-way ANOVA test in Matlab were also performed to investigate the relationship between Joint Number, Motion Range, Speed, and measurement accuracy. The result is shown in Table 3.10.

Table 3.10 ANOVA Test result

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
Joint_Num	4.0066	2	2.0033	38.12	0
Motion Range	1.4504	4	0.3626	6.9	0.0004
Speed	1.2341	4	0.30825	5.87	0.0012
Joint_Num* Motion Range	0.463	8	0.05787	1.1	0.3882
Joint_Num* Speed	1.5142	8	0.18927	3.6	0.0044
Motion Range* Speed	1.2805	16	0.08003	1.52	0.1519
Error	1.6819	32	0..05256	N/A	N/A
Total	11.6305	74	N/A	N/A	N/A

The RMSE of different single-joint trials are recorded in the Table 3.7, Table 3.8, Table 3.9, Table 3.10, and Table 3.11Fig 3.8. It reveals that:

(1) The RMSEs of trials which are around the global Z axis are obviously larger than those of trials around Y and X axis. It might be caused by magnetic distortion, as the motion in the horizonal plane is always effected by the magnetic fields more than motions in the other two planes.

(2) In trials around global Z axis, either a high motion speed or a low motion speed leads to a lower RMSE, which means a better estimation performance. However, in trials around the other two set of trials, there was no obvious evidence to support this

phenomenon. We inferred that it's caused by the AHRS algorithm itself, in which the heading axis are estimated both by the accelerometer, gyroscope and magnetometer. On low-speed occasions, magnetic distortion caused by movement is relatively little and the magnetometer could provide accurate information. On high-speed occasions, the drift of the accelerometer and gyroscope is small, which means a better estimation of the heading axis.

(3) In all groups of trials, the RMSE increases with the motion range, which means a larger motion range leads to a worse estimation performance. This result indicates that a larger motion range leads to a larger RMSE.

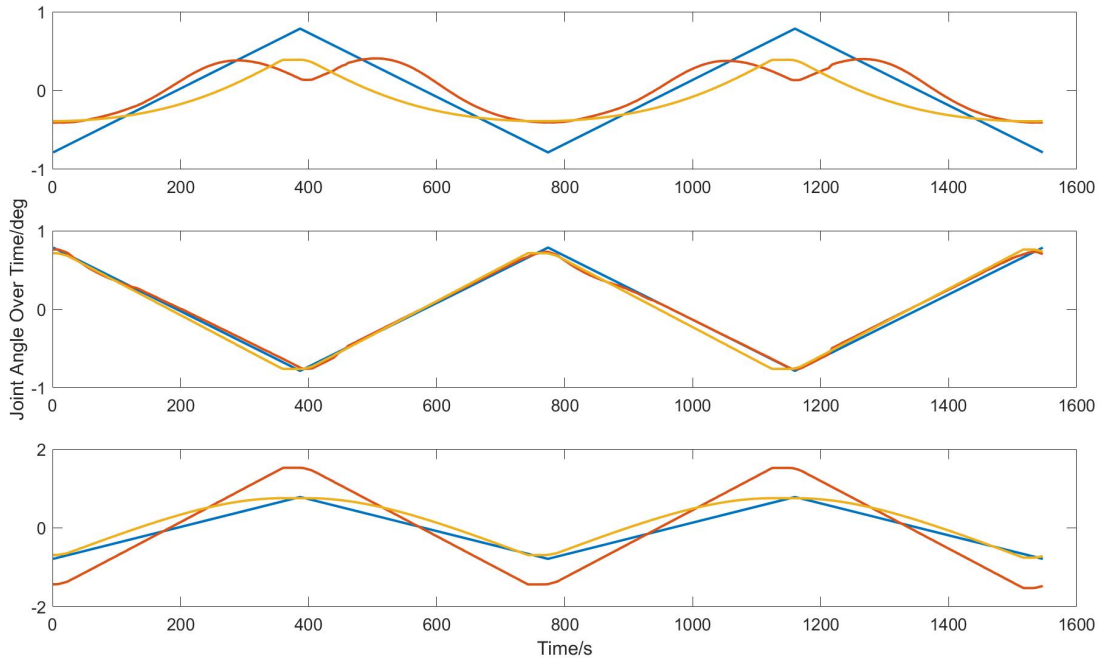


Fig 3.9 result of multi-joints trial: The blue line stands for the reference data, the red line stands for the Euler-based method, while the yellow line stand for the proposed method.

Table 3.11 RMSE of Multi-joints Trials

	Joint 1	Joint 2	Joint 3
Euler based	0.1509	0.0242	0.3168
Proposed	0.1620	0.0383	0.1299

In third part, we validated the RMSEs of both Euler-based method and our proposed method. The result is recorded in the Fig 3.9 and table 3.5. It reveals that:

(1) Both two methods performed best when it came to joint 2. The 1st joint suffered from magnetic distortion most, which restricted its accuracy. The 3rd joint was located at the distal end of the robot arm, meaning that errors from the first two joints might accumulated in its result.

(2) Our proposed method had a similar performance with the Euler-based method when it came to joint 1 and joint 2. However, it revealed that for joint 3, the RMSE of the proposed method was about half of that of Euler-based method.

Thus, the proposed method was assumed to have a better performance than the Euler-based method when it comes to calculation accuracy.

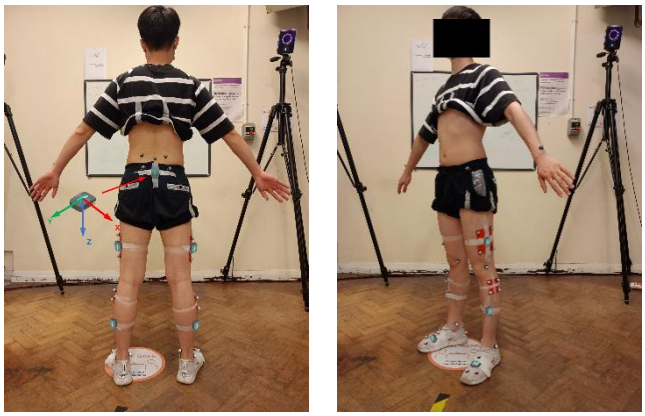
Chapter4 Human Motion Prediction

In the past decades, it reveals that there's a growing interest in the biomechanics community in wearable sensors, which enables the clinical diagnosis of motion disorders and design of the rehabilitation devices. To provide reliable feedback in the human-machine interface for advanced rehabilitation devices, methods to predict motion intention are developed. An inertial measurement unit (IMU) is a promising device for motion tracking, with the advantages of low cost and high convenience in sensor placement to measure motion in almost every environment. However, it reveals that there is no comprehensive study to predict human motion with pure IMU data. Thus, this paper presents a hybrid method of the skeletal model and LSTM to predict the lower limb joint angles with pure IMU data, which has the potential to be a low-cost, easy-to-use alternative in motion prediction. The LSTM is a recurrent neural network where the input of a node is the output of another one, making it robust for solving time-series problems. In comparison with RNN, nodes in the input layer of an Artificial Neural Network (ANN) are independent. A standard RNN commonly suffers from gradient explosion or gradient dispersion. Thus, we chose LSTM to undertake the task of human motion intention prediction. Our research purpose are: (1) to estimate the motion using an IMU driven MSK model while the marker-based model output was considered as the gold standard; (2) to train machine learning models (RNN and LSTM) to predict motion intention; (3) to compare the errors from the different machine learning models with the hypothesis that: LSTM outperforms traditional artificial neural networks.

4.1 Experiments

To get access to human motion data for validation, a functional movement data experiment was designed and carried out in the motion capture lab.

This study focuses on algorithm validation rather than specific lower limb diseases research, thus trials on healthy groups were enough for following research.Six healthy subjects (4 males, mean ± SD; age 22.8±0.4 years; height 168.7±5.6 cm; body mass 55.5±7.7 kg) with no self-reported lower limb musculoskeletal (MSK) pain or impairments were recruited. Before the data collection process, institutional ethics approval and informed consent had been obtained.



(a) (b)

Fig 4.1 Experimental Setup and Marker Placement

Considering that we carried out such trials to compare performance of IMU-based method and a golden standard, an extra optical motion capture system was also introduced in this part. IMU data and optical motion capture data were collected by using an integrated Inertial Measurement Units system(DelsysTrigno Avanti, USA, 2000 Hz) and an 8-camera optical motion capture system(Vicon, UK, 100 Hz), respectively. The DelsysTrigno Avanti system was composed of IMU module and

EMG module, which enabled it to output both inertial motion data and functional muscle activity simultaneously. The following data processing didn't deal with EMG data, thus this module was disabled. The Vicon system consisted of a PC for data processing, a data exchanger, 8 advanced high-speed cameras, and a wand for calibration. It is worth mentioning that the data exchanger of the Vicon motion capture system was able to collect motion capture data either from the high-speed cameras and the Trigno Avanti inertial sensors.The motion data and IMU data were synchronized during motion tracking.

With the advanced motion tracking devices mentioned in the last paragraph, we could carry out motion capture trials with high reliability. However, there would not be any possibility to get high accuracy data without dedicated maker placement. In this study, the marker placement process was much complicated than those of experiments with single sensor type. There were 7 inertial sensors for lower motion tracking and 39 reflective markers for cameras to track.For each subject, seven IMUs were placed on the lower limb body segments for the pelvis, thighs, shanks, and feet. The principle of how to place an IMU was to put it on a body area with as less artefact as possible. The pelvis IMU was placed on the lower back where pelvis was located as there might be more soft tissues on the belly of most subjects. The IMU sensors on the shanks and thighs were located at the centre of marker plates, which were solid and tightly taped to the corresponding body segments. Moreover, the IMUs on the feet were located at the front area of the shoes.The sensor reference frame of a Delsys IMU is shown in Fig 4.1(a). The global reference frame was defined as ENU(X-axis to the East, Y-axis

to the North, Z-axis to the Up). At the same time, 39 reflective markers were also attached to body segments (Fig 4.1). They were placed on the anatomical bony landmarks of anterior/posterior superior iliac spine (APSIS), medial/lateral femoral epicondyles, medial/lateral malleoli, second/fifth metatarsal head, and posterior calcaneus. Additionally, two clusters of four markers were placed onto the lateral aspects of the thighs and shanks. The bony markers were used for both scaling and motion tracking while other markers such as those on the marker plates were attached on different body areas to provide extra movement information.

Apart from functional movement tasks, calibration was also necessary for us to align IMU reference frame with corresponding body segment reference frame (i.e. anatomical reference frame). Thus, three tasks were assigned to each subject, including calibration, sit-to-stand, and walking. For IMU calibration, both functional calibration and static calibration were adopted by former researches. However, for scaling data collection, it was highly recommended that we adopted a static pose as the calibration data could be calculated as average value in a specific time period. In the calibration task, the subjects were asked to stand upright in a neutral pose for about 20 seconds. They were also asked to stand with their hands crossed on their chests to avoid blocking bony marker from camera sights. After that we carried out formal functional movements. In the sit-to-stand task, the subjects were asked to sit on a height-adjustable seat. The height of the seatwas adjusted so their knees were at 90° flexion in the sitting position.Subjects were askedto firstly stand in the calibration pose, and then they were informed to sit down and stand up slowly with a relatively

stable speed themselves. In the walking task, the subjects were asked to walk forward naturallyat a comfortable speed, which was like how they walked on normal occasions.

4.2 IMU-based human motion tracking protocol in OpenSense

Joint angle calculation of data collected by inertial sensor was carried out in OpenSense, which was a standard solution protocol provided by OpenSim.

4.2.1 Introduction to OpenSim/OpenSense

OpenSim is a software for modelling the human body and the environment, which could simulate human movements and interactions with the environment. OpenSim possesses a graphical user interface (GUI) which could visualize musculoskeletal (MSK) models and perform simulations (Fig 4.2). OpenSim is an open-source software with all the source code accessible to global developers. Engineers are free to develop the software by utilizing the OpenSim application programming interface (API). OpenSim was firstly introduced by Scott Delp, Jennifer Hicks, Ajay Seth et al. at the American Society of Biomechanics Conference in 2007, which was known as version 1.0. An application programming interface (API) was added in V2.0 and the API was extended to Matlab and Python in the latest version. The software OpenSim is an interdisciplinary simulation platform that is compatible with multiple programming languages and enables engineers and scientists from different academic

institutions to collaborate on biomedical research. The core code of the software is developed in C++ while the graphical user interface (GUI) is developed based on Java. Developers could develop new MSK models or simulate human motion with data collected from sensors. Besides, there are numerous plug-ins on the OpenSim website for people to download and run without compiling, which makes it accessible to control OpenSim in other platforms.

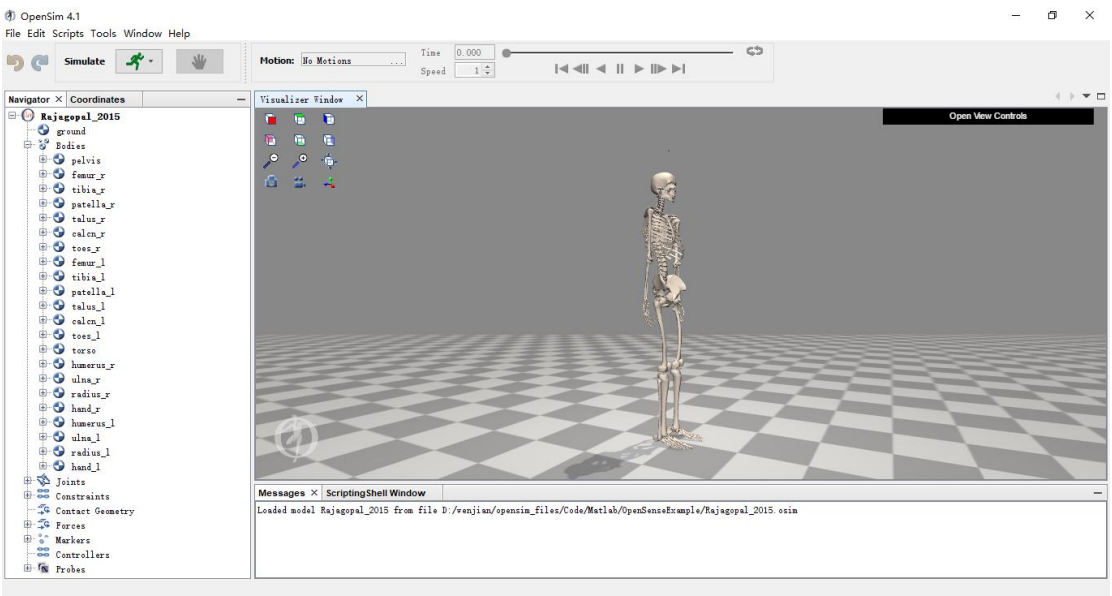


Fig 4.2 OpenSim Graphic User Interface

OpenSense is a novel workflow in OpenSim 4.0 which is developed for IMU data based human motion tracking tasks. The workflow is designed to solve inverse kinematics problems based on IMU data (i.e. calculate joint angles between adjacent body segments). A multi-body model with target joint is required before the human motion capture pipeline. Any model in .osim format could be utilized to conduct inverse kinematics research and no scaling are required in the procedure. A standard OpenSense workflow includes five steps: data collection & preprocessing, data format conversion, calibration, human motion tracking, and results visualization.The output

of the Delsys IMU system is the orientation between the IMU reference frame and the global reference frame over time. Normally, one IMU sensor is attached to one body segment, and the placement information should be recorded before the experiment. OpenSense currently only supports data collected from the Xsens/ADPM IMU system. Calibration data were collected before motion capture data to make sure the data could be converted from the IMU reference frame to the anatomical reference frame. There are two types of calibration: static calibration and functional calibration, and in this study, only static calibration pose was adopted. The OpenSense workflow also provides developers with a heading correction option. To correct IMUs' heading, a base IMU whose heading information will be utilized to conduct correction is required. Calibration data was set in the first line of the time-series orientation file as OpenSense assumes the calibration pose corresponds to the zero-timestamp pose. Another work in the calibration process is to calculate the sensor to OpenSim rotations because the OpenSim world reference frame is Y to the up, Z to the right. After calibration, joint angle estimation could be performed in OpenSense to track human motion. The inverse kinematics minimize the angular error between Euler angles to seek the optimized solution in every epoch.

4.2.2 Data Collection and Pre-processing

Data collection plays a significant role in the human motion tracking pipeline. Normally, one IMU sensor is attached to one body segment, and the placement information should be recorded before the experiment. Calibration data was collected

before motion capture data to make sure the data could be converted from IMU reference frame to anatomical reference frame. here are two types of calibration: static calibration and functional calibration. The previous one performs a pose where the tracked joint angle is assumed as 0 deg or any other specific values and keep still. The other one performs a function action to optimize the joint angle estimation. In this study, a basic static calibration method was adopted. The OpenSense workflow also provides developers with a heading correction option. To correct IMUs' heading, a base IMU whose heading information will be utilized to conduct correction is required. There was no heading IMU in this study in order to reduce the number of IMUs. Raw data is collected in the format of rotation matrix or quaternions from IMU sensors. The official workflow is only compatible with Xsens IMU and ADPM IMU. Sensor fusion, synchronizing, and interpolation for missing entries are assumed to be performed before the pipeline.

4.2.3 Data Format Conversion

After the first step, the data is expected to be converted into a format which is compatible with OpenSim and then attached to the model in OpenSim. At this stage, OpenSense only support data from Xsens/APDM IMU sensors. After import, a time-series quaternions data will be generated in .sto format by converting the rotation matrices into quaternions.

In this study, we used DelsysTrigno Avanti IMU sensors to perform human motion measurement, which possesses a different output format from that of Xsens/ADPM.

The collected data was first collected in the EMGworks Acquisition software and then transferred into .mat format by the Trigno File Utility for further processing. Two methods were adopted in this step to import IMU data: one is generate the same .txt files from Trigno Avanti data in MatLab, and then follow the standard human tracking pipeline provided by OpenSense developers; the other one is directly generate time-series quaternions data which is in .sto format in MatLab without callback the import functions provided byOpenSense. The rotation matrices provided by IMU sensors should by direct cosine matrices (DCM).

In a standard data conversion workflow, an XML file is expected to inform the platform which sensor corresponds to which body segment in the OpenSim model. The XML file specifies trial_prefix, Experimental Sensor name, and name_in_model. The naming convention of the study should go along with OpenSense where an IMU will be named as <body segment>_imu. The OpenSense will generate a .sto file with orientation data and IMU information of each sensor.

In this study, another MatLab function was developed to directly generate the .sto file from time-series orientation data.

4.2.4 Calibration in OpenSense

Whe it comes to calibration, an OpenSim model and orientation data are required to calculate the offset, i.e. the rotation of IMU reference frame relative to the anatomical reference frame. The model applied in the trial was the Rajagopal (2015) model. Virtual markers are attached to the the corresponding body segments due to the

information provided by the .sto file. Calibration data was set in the first line of the time-series orientation file as OpenSense assumes the calibration pose is corresponded to the zero time stamp pose. Another work in calibration process is to calculate the sensor to OpenSim rotations because the OpenSim world reference frame is Y to the up, Z to the right, which is different from the IMU world reference frame(Z to the up, Y to the north). Moreover, the base IMU and its heading axis could be specified in OpenSense to make adjustments to the initial orientation of the model. Normally, the pelvis IMU will be chosen as the heading IMU toperform correction to initial orientation. The heading axis is specified related to the experiment, which could be x, -x, y, -y, z, or -z. In the trial, no heading IMU was applied as the number of IMUs were expected to be minimized.

4.2.5 Human Motion Tracking in Opensense

After calibration, joint angle estimation could be performed in OpenSense to track human motion. The invers kinematics minimize the angular error between Euler angles to seek for the optimized solution in every epoch. A calibrated OpenSim model and a time-series orientation file was needed for motion capture. Several parameters could be specified before motion capture, including time range, sensor to opensim rotations, model file name, orientation file name. The time range specifies the start time and end time of the simulation procedure. The sensor to opensim rotations specifies the rotation (Euler angles in global XYZ) from IMU world reference frame(Z to the up, Y to the north) to OpenSim world reference frame(Z to the right, Y

to the up). The model file name specifies the calibrated model utilized to perform motion tracking while the orientations file name specifies the orientation data. In order to ensure the motion tracking is accurate, all the joint angles in the model except the tracked one were locked in this step.

4.2.6 Results Visualization

The result of the trial could be visualized by OpenSim GUI by loading the motion file(.mot) on the calibrated model and the click run in the OpenSim interface. When it comes to data visualization, the OpenSim provides a embedded plottin function. The result could also be output to another software for postprocessing.

4.3 Marker-based human motion tracking in OpenSim

It's also of great significance to collect accurate data to work as a golden standard for following validation steps. Thus, we chose a marker-based based protocol to generate reference data due to its high accuracy and satisfactory repeatability in former research. Marker labeling and gap-filling were performed in Vicon Nexus (Vicon, UK). Smoothed marker trajectories were the input of a generic, full-body MSK model (Rajagopal, 2015) in OpenSim (Version 4.3, USA), i.e. the same as that in the standard OpenSense pipeline. The model contains 22 rigid bodies and each lower limb is modelled with 7 degrees of freedom including hip rotation, hip flexion, hip adduction, knee flexion, ankle inversion, ankle dorsiflexion, and toe flexion. Our

simulation workflow began with scaling the geometry ofthe generic model to match the anthropometry ofeach subject, using the OpenSim Scale Tool. The marker trajectories from the calibration tasks were used in scaling. A tracking weight of 1000 was assigned to the anatomical bony landmarks while a small weight of 1 was assigned to other markers such as markers on the thigh and shank clusters. Lower limb joint angles during tasks of sit-to-stand and walking were generated by usingOpenSim's Inverse Kinematics (IK) Tool.

4.3.1 Data collection and preprocessing in the Vicon Nexus

The eight high-speed cameras directly captured raw trajectories of markers attached to the subject and sent them to the PC software via the central data exchanger. However, it's difficult for us to analyze human motion without preprocessing because it's hard for a machine to discriminate which marker corresponded to which segment. Thus, we designed a 39-marker template for the experiment in Vicon Nexus. It included two parts: one was for marker labelling and the other was for segment generation. We labelled every marker in the first frame, and then the software automatically labelled the remained frames. Segments were also created in this part: we picked out four markers to specify a segment they were attached to, after which we assigned all of the markers to the body segment they belonged to.

The created template enabled us to label the trajectories with high efficiency, which means that we didn't need to label the markers and assign them to the corresponding body segment frame by frame. It's revealed that there also existed a problem that the

automatically labelled trajectories suffered from trajectory gaps and unlabeled markers in some frames where optical information provided by the cameras was not sufficient. To solve the problem, we could utilize the standard protocols in the Vicon Nexus to carry out the following marker labelling and gap-filling manually. We could conveniently select markers from the created template and assigned the specification to the unlabeled marker to finish marker labelling. However, when it came to gap-filling, we needed to utilize suitable gap-filling methods. The software provided us with fivemodes: spine fill, pattern fill, rigid body fill, kinematic fill, and cyclic fill. This study only referred to the previous 3 modes. The spine fill conducted simple interpolation for the missing points in a trajectory, which was suitable for the situation where the number of missing points between two frame was not too large. We defined the max value as 20, which meant that we would not consider this method when missing gaps between two frames were more than 20. The pattern fill generated part of the trajectory of a marker with reference to another marker. Thus, the reference marker should possess a similar movement pattern to the marker to be gap-filled. The rigid body fill generated the position of a marker in a specific frame with 3D position values of another 3 markers on the same rigid body.

4.3.2 Scalingand marker-based inverse kinematics in OpenSim

To calculate joint angles accurately, we adopted a standard inverse kinematics protocol in OpenSim. Two steps were included: scaling and inverse kinematics. The

protocol took .trc file which recorded marker trajectories, a generic OpenSim model, and a series of parameter setting as input.

In case that the difference of body segment geometries could introduce extra errors, we scaled the generic model with the scale tool. A marker set in the OpenSim was required before scaling, in which each maker was located with reference to marker placement in the experiment and assigned to the corresponding body segment. The maker set and a generic skeletal model Rajagopal in OpenSim were both inputted into the scale tool. For scaling, we also loaded a static trial to provide true measurement information and adopted the average data between start time and end time. For each body segment, to scale its geometry, we needed to assign corresponding scale factors either by manual measurement or using measurements. In this study, we used measurements to scale different body segments by choosing suitable marker pairs. Apart from marker pairs, we also assigned different weights to different markers, depending on its contribution to the scaling.The scale tool would automatically generate a scaled model in which each body segment had been tailored into the same size of the subject. Then the scaled model, along with the marker set mentioned above were inputted into the inverse kinematic tool to calculate lower limb joint angles. The weightsof bony landmarks were set as 1 while weights of other markers were set as 0.1.

4.4 Development of LSTM Neural Network

4.4.1 Introduction to ANN &LSTM

BP neural network is a research hotspot in the field of artificial intelligence, it information processing perspective to abstract the human brain neurons, to build some kind of simple model, according to different ways to form a network, each node of the network is called a neuron. For a single neuron, the model is as follows, i.e., Fig 4.3.

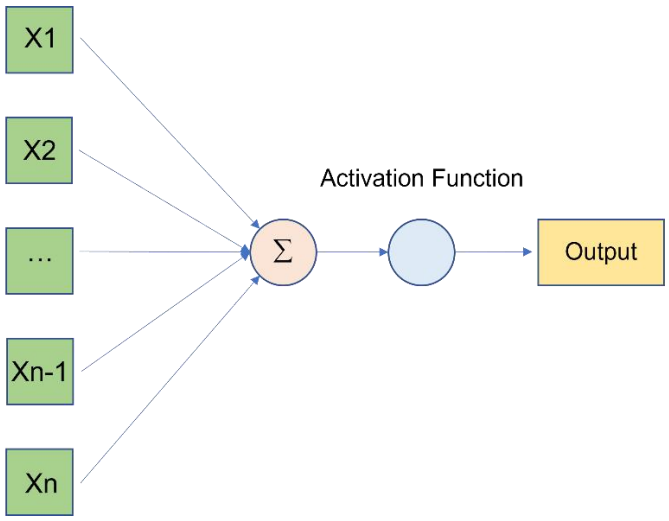


Fig 4.3 Structure of a node

The model contains input data, output data and neuron nodes, which are generally composed of a summation function and an activation function. The activation function can be chosen by oneself and is commonly used as an S function or hyperbolic tangent function.

The data processing effect of a single neuron is often limited, and combining a large number of such neurons into a neural network can greatly improve the effectiveness of the model. The model is as follows, i.e., Fig 4.4.

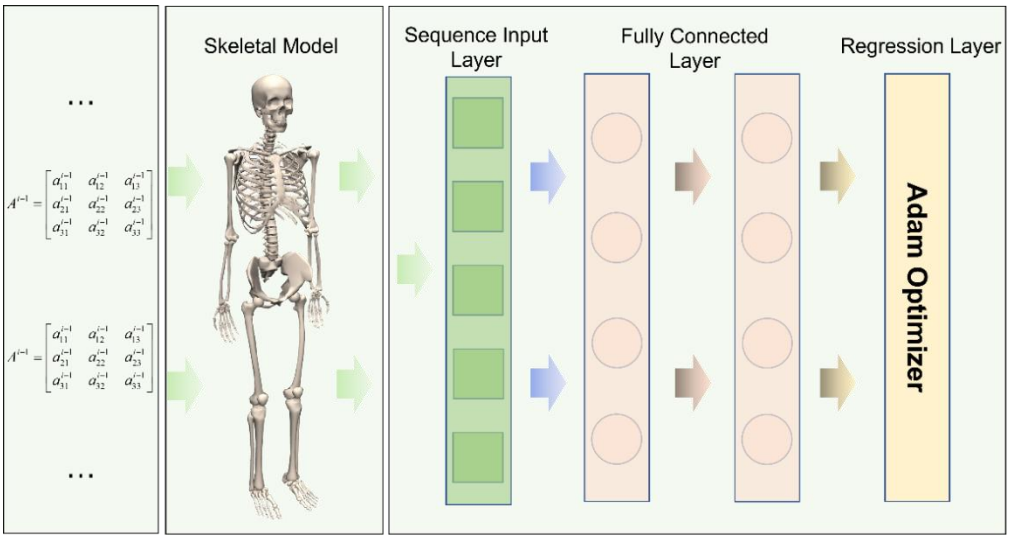


Fig 4.4 Structure of ANN

The model mainly consists of an input layer, a hidden layer and an output layer, which are responsible for input, operation and output tasks respectively. The mathematical model of the S-function is as follows.

$$y = \frac{1}{1+e^{-z}} \tag{4.1}$$

$$z = w^T x + b \tag{4.2}$$

where y is the neuron output, z is the summation function output, x is the input, w is the weight matrix, and b is the bias.

In the feed-forward process, a series of operations are required to make predictions and calculate the loss function, which is calculated as follows.

$$J = -\frac{1}{m}\sum_{i=1}^{m}(y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)) \tag{4.3}$$

where y is the observed value and is the predicted value.

To ensure that the loss function converges to a minimum, the gradient needs to be calculated during the feedback process and the weight matrix and bias matrix need to

be updated.

$$w = w - \alpha \frac{\partial J(w,b)}{\partial w} \tag{4.4}$$

$$b = b - \alpha \frac{\partial J(w,b)}{\partial b} \tag{4.5}$$

where is the model learning rate.

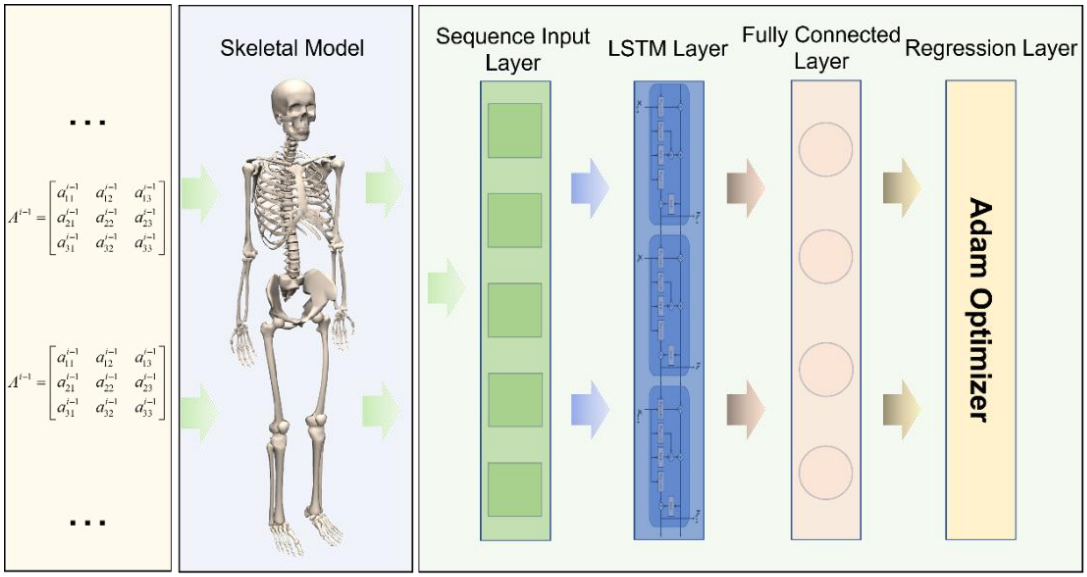


Fig 4.5 Structure of the Predictive Model

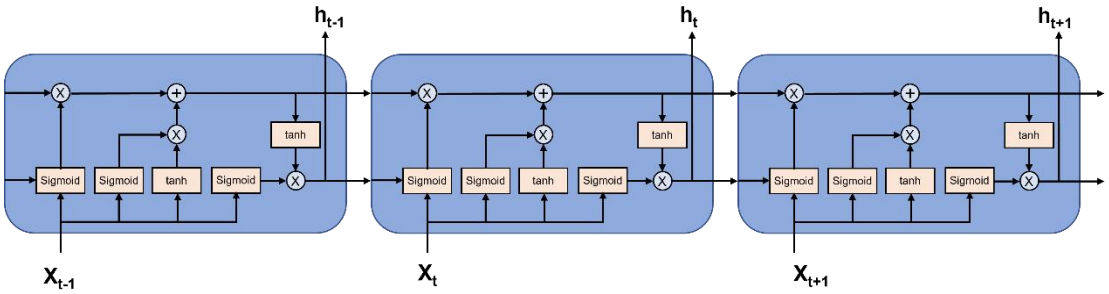


Fig 4.6 Structure of an LSTM layer

LSTM (Fig 4.5) is one of the variants of recurrent neural networks, which includes SequenceInputLayer, LSTM Layer,FullyConnectedLayer, and RegressionLayer. Similar to basic artificial neural networks, LSTM networks are composed of multiple layers, each of which in turn consists of multiple nodes. the LSTM layer (Fig 4.6) is the core layer of the LSTM network, where the input of each node is the output of the previous node and the output is the input of the previous node.The structure of the LSTM layer dictates that the relationships of neighboring nodes are coupled, however, in an ANN, the nodes in the same layer are independent of each other.

4.4.2 Human motion prediction with different methods

The LSTM neural network was trained and validated in MatLab with MatLabDeepLearning Toolbox. Before training the neural networks, the data was normalized to gain better performance by eliminating the influence of magnitudes.

The parameters of different neural networks are shown in Table 4.1. The NumFeatures, NumResponse, and NumHiddenUnits respectively represent the number of units in the corresponding layer. The MaxEpochs denotes the training times while LearnRate stands for the convergence speed in every epoch.

Table 4.1 Parameters of Neural Networks

Parameter	LSTM+MSK Model	ANN+MSK Model	LSTM only
NumFeatures	1	1	8
NumResponses	1	1	8
NumHiddenUnits	200	200	200
MaxEpochs	1000	1000	1000
LearnRate	0.005	0.005	0.005

Another two methods were tested in this study to compare with the developed motion intention prediction method. One is a model-free motion intention prediction method with LSTM. This method inputs the output of IMUs on adjacent segments to predict the corresponding joint angle. The result of the neural networks is the joint angle reference data several periods ahead of the input. In another trial, we replace the LSTM neural networks with traditional ANN to predict the joint angles. Similarly, the output of the IMUs will firstly be processed by the OpenSense human motion tracking pipeline and then trained by ANN to gain the capability to predict joint angles. The input of the neural network is the time-series joint angle estimated by the MSK model in OpenSense while the output is the joint angle data several periods ahead of the input.

4.5 Data analysis

In this study, we applied one sit-to-stand-to-walk trial to train and test the prediction model. We stitched together ten trials of time-series data for data analysis. The first nine trials are the training dataset to train the models, and the last trial is the test dataset to analyze the accuracy of the different methods. We predict joint angle in the next frame with the input of the previous few frames. Our proposed model could predict the data at any time after the current moment in theory. The prediction accuracy decreases as the time gap increases. We calculated RMSD between marker-based estimation result and IMU-based estimation result to validate the reliability of

IMU-based human motion tracking pipeline in OpenSim/OpenSense, and then, calculated the RMSD between IMU-based estimation result and prediction result to evaluate the accuracy of the proposed method. We plotted the result of prediction, IMU-based estimation, and marker-based estimation in the same figure for reference. Moreover, we calculated prediction results of the pure LSTM and ANN/MSK model in comparison with our proposed method. Last but not least, K-fold validation was also performed on the six-subject dataset. For data of each subject, data every trial took turns to work as test dataset, and data in the rest trials work as training dataset. Both RMSE and decision coefficient R2_score were calculated.

4.6 Results

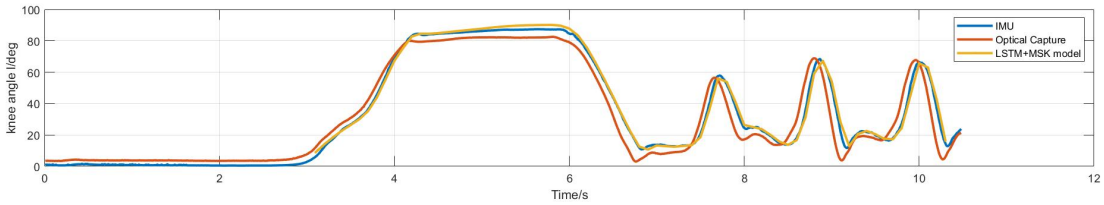
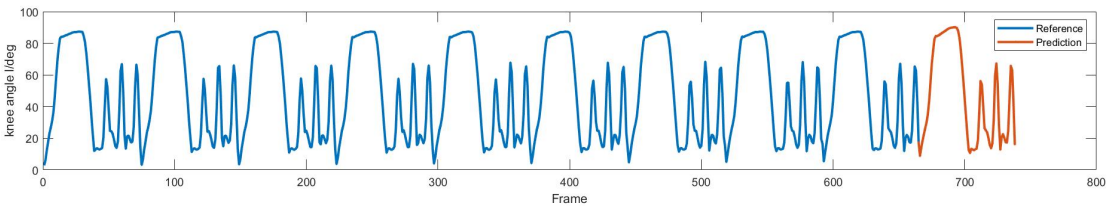
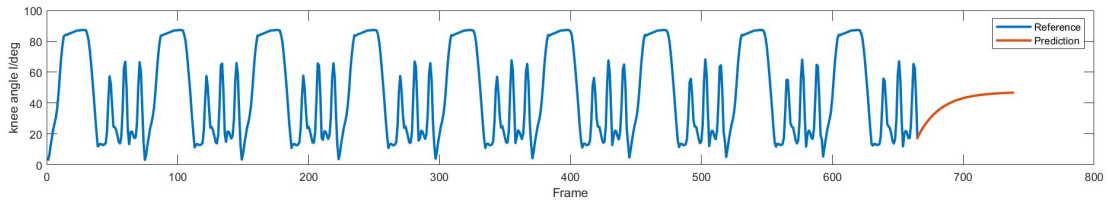


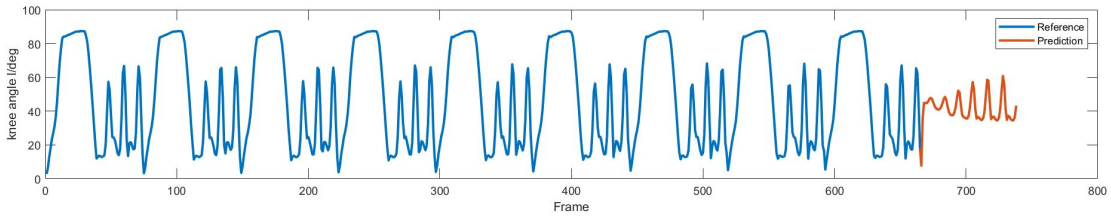
Fig 4.7 A comparison of knee joint angles during tasks of stand-to-sit-to-stand and walking from one representative subject. The knee joint angles are calculated from the marker-based MSK model (in red), from the IMU-based MSK model (in blue) and predicted from the method of LSTM integrating with MSK modeling (in yellow).



(a)LSTM+MSK Model



(b)ANN+MSK Model



(c)LSTM only

Fig 4.8 Prediction performance of knee joint angles from one representative subject between the (a) LSTM with MSK modeling, (b) ANN with MSK modelling and (c) LSTM without MSK modelling.

Table 4.2 Performance of Different Methods

	Pure LSTM	ANN+MSK	LSTM+MSK
RMSE/deg	31.15	31.66	2.93

Table 4.3 Performance of Proposed Method

Subject Number	RMSE/deg	R2_score
1	26.82	0.41
2	8.38	0.76
3	32.13	0.64
4	6.72	0.88
5	5.74	0.89
6	15.66	0.49

Ten trials (including stand-to-sit-to-stand and walking of the subject) were stitched together. The data was first downsampled from 100Hz to 10Hz to achieve a better performance in the model training. The first nine trials were set as the training dataset and the last trial was the prediction. The reference data was the joint angle overtime

calculated from the MSK model in OpenSim with IMU data as inputs.

As shown in Fig 4.7 and Table 4.2, the result of the proposed method follows the estimated joint angle well. The RMSE between IMU-based estimation joint angle and optical-marker-based estimation joint angle is 7.26 deg. As revealed in Fig 4.8(a), the result of the proposed hybrid method has an excellent performance in predicting human intention. In comparison, the ANN-based hybrid method and the pure LSTM neural network turn out to possess a limited capability to predict human intention (Fig 4.8(b), Fig 4.8(c)). The RMSE of the proposed method is 2.93 deg while the RSME of the other two is respectively 31.15 deg and 31.66 deg.

As shown in Table 4.3, the RMSEs of some subjects are lower than 10 degrees and decision coefficients are relatively high. However, for other subjects, RMSE could be higher than 20 degrees, which indicates a bad performance. This is because the repeatability of picked trials varies from each other, for those with a higher repeatability, the LSTM has a better prediction performance.

4.7 Discussion

The proposed method which fuses the skeletal model and LSTM greatly outperforms pure LSTM and the method fuses the skeletal model and ANN. The result reveals that the hybrid method possesses a high accuracy of 2.93 degrees in RMSE.

The LSTM layer enables the hybrid method to predict human motion intention with satisfactory accuracy. In an LSTM layer, the output of a node is the input of the next node, which makes it suitable for time-series data processing, including

prediction.Applying pure LSTM neural networks to perform the motion intention prediction tends to result in limited accuracy. Compared with Neural Networks, model-based methods perform better in joint angle estimation.If the LSTM layer is replaced with an ANN layer, it reveals that the hybrid model will lack the capability to predict human motion. Thus, EMG data was customarily adopted to provide preliminary information for an ANN-based deep learning model, significantly increasing cost.

Most studies applied human motion intention prediction methods by measuring both kinematics information (optical data, IMU data, etc.) and biomechanics information (EMG). Our proposed hybrid method has three advantages: (1) Prediction model with pure IMU data has less limitation when it comes to outdoor human motion measurement. (2) Accuracy of marker-based or EMG-based motion intention method is more likely to be affected by maker placement. (3) Pure IMU devices are cheaper than those designed for EMG or marker trajectory measurement. (4) By applying LSTM rather than ANN, the model could predict the motion intention flexibly.

Chapter5 Conclusion and Future Work

5.1 Conclusion

This thesis aims to develop a lower limb motion capture protocol with high accuracy and reliability. To fulfill the basic requirement of this project, both robot arm research and human motion research were carried out.

In robot arm validation experiment, data of a series of trials with different parameters, such as rotation axis, motion speed, and motion range were collected to give a comprehensive understanding of the suitable experimental conditions for IMU-based motion capture trials. Moreover, we developed a novel optimal joint angle estimation method to improve the running efficiency of the motion tracking pipeline. In the lower limb human motion capture experiment, we collected data from six healthy subjects with both IMU sensors and high-speed cameras. We proposed a novel method to predict motion intention of subjects by fusing both MSK modelling technologies and LSTM neural networks. The results reveal that:

For IMU-based human motion tracking, the motion in the horizontal plane has a larger measurement error in comparison with motion in the other two planes. The measurement error increases when motion range increases.Either a high speed or low speed causes lower measurement error for motion in the horizontal plane. However, when it comes to the other two plane, the error increases when the motion speed increases.

The proposed method to calculate joint angle with high calculation efficiency

outperforms the Euler method when it comes to iteration required to converge to a fixed value. It means the method has better efficiency. It differs from traditional joint angle estimation algorithms in that it's proposed based on Riemannian distance between rotation matrices rather than Euclid distance between some specified parameters.

The developed method in chapter 4 outperforms pure LSTM and ANN fusing MSK model when it comes to prediction accuracy. It differs from with traditional methods in: (1) Focusing on solving motion intention prediction with pure IMU data, which means for lower cost and higher adaptability to outdoor occasions. (2) Integrating both MSK models and LSTM neural networks to take advantage of two models, i.e. MSK model for estimation and LSTM for prediction.

5.2 Future Work

There are also some future works which deserves further research:

The result in the chapter 3 reveals that experimental parameters, i.e. rotation axis, motion range, motion speed, and so on, requires to be carefully picked up. Thus, it's highly recommended that further research focuses on choosing suitable parameters with reference experimental conditions before starting an IMU-based motion capture experiment. Future work related to joint angle estimation might focus on the intrinsic representation of 3D orientation to improve efficiency of further methods.

Motion in the horizontal plane, i.e., around the global Z axis has a larger measurement than the other two. Three further solutions are recommended: (1) To develop an more

advanced sensor fusion algorithm to suppress the measurement error caused by magnetic distortion. (2) To design a better experimental protocol to reduce the negative effect of magnetic field. (3) To improve the performance of hardware, for example, utilize magnetometer with higher robustness.

The motion intention prediction method can be improved by developing an adaptive module for more flexible prediction, which enables it to predict motion after specific time period. Moreover, it reveals that fusing MSK models with AI technologies might give out a novel perspective to improve performance of related methods.

Academic Output

1. “an Automatic Inertial MeasurementUnit Alignment Pipeline in HumanMotion Measurement”. Qingyao Bian, Duncan Shepherd, and Ziyun Ding. XXVIII CONGRESS OF THE INTERNATIONAL SOCIETY OF BIOMECHANICS (ISB). (Poster Presentation)
2. “A Hybrid Method Integrating a Musculoskeletal Model with Long Short-Term Memory (LSTM) for Human Motion Prediction”. Qingyao Bian, Duncan Shepherd, and Ziyun Ding. the 44th IEEE Annual International Conference of the IEEE Engineering in Medicine and Biology Society. (full contributed paper accepted)

Reference

[1] 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. 2020. doi: 10.3390/s20195687.

[2] I. Arun Faisal, T. Waluyo Purboyo, and A. Siswo Raharjo Ansori, "A Review of Accelerometer Sensor and Gyroscope Sensor in IMU Sensors on Motion Capture," *Journal of Engineering and Applied Sciences*, vol. 15, no. 3, 2019, doi: 10.36478/jeasci.2020.826.829.

[3] P. Cheng and B. Oelmann, "Joint-angle measurement using accelerometers and gyroscopes - A survey," *IEEE Trans Instrum Meas*, vol. 59, no. 2, 2010, doi: 10.1109/TIM.2009.2024367.

[4] H. Rouhani, J. Favre, X. Crevoisier, and K. Aminian, "Measurement of multi-segment foot joint angles during gait using a wearable system," *J Biomech Eng*, vol. 134, no. 6, 2012, doi: 10.1115/1.4006674.

[5] J. Favre, R. Aissaoui, B. M. Jolles, J. A. de Guise, and K. Aminian, "Functional calibration procedure for 3D knee joint angle description using inertial sensors," *J Biomech*, vol. 42, no. 14, 2009, doi: 10.1016/j.jbiomech.2009.06.025.

[6] S. Tadano, R. Takeda, and H. Miyagawa, "Three dimensional gait analysis using wearable acceleration and gyro sensors based on quaternion calculations," *Sensors (Switzerland)*, vol. 13, no. 7, 2013, doi: 10.3390/s130709321.

[7] B. Fasel, J. Sporri, J. Chardonnens, J. Kroll, E. Muller, and K. Aminian, "Joint Inertial Sensor Orientation Drift Reduction for Highly Dynamic Movements," *IEEE J Biomed*

Health Inform, vol. 22, no. 1, 2018, doi: 10.1109/JBHI.2017.2659758.

[8] T. Seel, J. Raisch, and T. Schauer, "IMU-based joint angle measurement for gait analysis," *Sensors (Switzerland)*, vol. 14, no. 4, 2014, doi: 10.3390/s140406891.

[9] J. Cockcroft, J. H. Muller, and C. Scheffer, "A novel complimentary filter for tracking hip angles during cycling using wireless inertial sensors and dynamic acceleration estimation," *IEEE Sens J*, vol. 14, no. 8, 2014, doi: 10.1109/JSEN.2014.2318897.

[10] J. F. S. Lin and D. Kulić, "Human pose recovery using wireless inertial measurement units," *Physiol Meas*, vol. 33, no. 12, 2012, doi: 10.1088/0967-3334/33/12/2099.

[11] V. Joukov, V. Bonnet, M. Karg, G. Venture, and D. Kulić, "Rhythmic Extended Kalman Filter for Gait Rehabilitation Motion Estimation and Segmentation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 2, 2018, doi: 10.1109/TNSRE.2017.2659730.

[12] S. Šlajpah, R. Kamnik, and M. Munih, "Kinematics based sensory fusion for wearable motion assessment in human walking," *Comput Methods Programs Biomed*, vol. 116, no. 2, 2014, doi: 10.1016/j.cmpb.2013.11.012.

[13] V. Joukov, M. Karg, and D. Kulic, "Online tracking of the lower body joint angles using IMUs for gait rehabilitation," 2014. doi: 10.1109/EMBC.2014.6944082.

[14] X. Yun and Y. Yamamoto, "On feedback linearization of mobile robots," 1992.

[15] W. H. K. de Vries, H. E. J. Veeger, A. G. Cutti, C. Baten, and F. C. T. van der Helm, "Functionally interpretable local coordinate systems for the upper extremity using inertial & magnetic measurement systems," *J Biomech*, vol. 43, no. 10, 2010, doi: 10.1016/j.jbiomech.2010.03.007.

[16] E. Palermo, S. Rossi, F. Marini, F. Patanè, and P. Cappa, “Experimental evaluation of accuracy and repeatability of a novel body-to-sensor calibration procedure for inertial sensor-based gait analysis,” *Measurement (Lond)*, vol. 52, no. 1, 2014, doi: 10.1016/j.measurement.2014.03.004.

[17] K. J. O’Donovan, R. Kamnik, D. T. O’Keeffe, and G. M. Lyons, “An inertial and magnetic sensor based technique for joint angle measurement,” *J Biomech*, vol. 40, no. 12, 2007, doi: 10.1016/j.jbiomech.2006.12.010.

[18] J. Chardonnens, J. Favre, and K. Aminian, “An effortless procedure to align the local frame of an inertial measurement unit to the local frame of another motion capture system,” *J Biomech*, vol. 45, no. 13, 2012, doi: 10.1016/j.jbiomech.2012.06.009.

[19] H. Yang and J. Ye, “A calibration process for tracking upper limb motion with inertial sensors,” 2011. doi: 10.1109/ICMA.2011.5985732.

[20] S. Ivaldi, J. Peters, V. Padois, and F. Nori, “Tools for simulating humanoid robot dynamics: A survey based on user feedback,” in *IEEE-RAS International Conference on Humanoid Robots*, 2015, vol. 2015-February. doi: 10.1109/HUMANOIDS.2014.7041462.

[21] M. Quigley *et al.*, “ROS: an open-source Robot Operating System,” in *ICRA workshop on open source software*, 2009, vol. 3, no. 3.2.

[22] Z. Du, Y. Sun, Y. Su, and W. Dong, “A ROS/Gazebo based method in developing virtual training scene for upper limb rehabilitation,” 2014. doi: 10.1109/PIC.2014.6972347.

[23] M. Qi, Y. Li, K. Xiang, and Y. Ge, “A wireless inertial measuring system for human

motion analysis," 2017. doi: 10.1109/ICInfA.2016.7831969.

[24] J. F. Nunes, P. M. Moreira, and J. M. R. S. Tavares, "Human motion analysis and simulation tools: A survey," in *Handbook of Research on Computational Simulation and Modeling in Engineering*, 2015. doi: 10.4018/978-1-4666-8823-0.ch012.

[25] Y. Huang *et al.*, "Real-Time Intended Knee Joint Motion Prediction by Deep-Recurrent Neural Networks," *IEEE Sens J*, vol. 19, no. 23, 2019, doi: 10.1109/JSEN.2019.2933603.

[26] Q. Liu, L. Ma, Q. Ai, K. Chen, and W. Meng, "Knee joint angle prediction based on muscle synergy theory and generalized regression neural network," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, 2018, vol. 2018-July. doi: 10.1109/AIM.2018.8452230.

[27] A. Gautam, M. Panwar, D. Biswas, and A. Acharyya, "MyoNet: A Transfer-Learning-Based LRCN for Lower Limb Movement Recognition and Knee Joint Angle Prediction for Remote Monitoring of Rehabilitation Progress from sEMG," *IEEE J Transl Eng Health Med*, vol. 8, 2020, doi: 10.1109/JTEHM.2020.2972523.

[28] J. He, Z. Guo, Z. Shao, J. Zhao, and G. Dan, "An LSTM-Based Prediction Method for Lower Limb Intention Perception by Integrative Analysis of Kinect Visual Signal," *J Healthc Eng*, vol. 2020, 2020, doi: 10.1155/2020/8024789.

[29] H. Xie, G. Li, X. Zhao, and F. Li, "Prediction of limb joint angles based on multi-source signals by GS-GRNN for exoskeleton wearer," *Sensors (Switzerland)*, vol. 20, no. 4, 2020, doi: 10.3390/s20041104.

[30] B. C. Hall, "Lie Groups, Lie Algebras, and Representations," 2013. doi: 10.1007/978-

1-4614-7116-5_16.

[31] T. Seel, J. Raisch, and T. Schauer, “IMU-based joint angle measurement for gait analysis,” *Sensors (Switzerland)*, vol. 14, no. 4, 2014, doi: 10.3390/s140406891.

[32] L. Tagliapietra, L. Modenese, E. Ceseracciu, C. Mazzà, and M. Reggiani, “Validation of a model-based inverse kinematics approach based on wearable inertial sensors,” *Comput Methods Biomech Biomed Engin*, vol. 21, no. 16, 2018, doi: 10.1080/10255842.2018.1522532.

[33] R. Mallat, V. Bonnet, M. A. Khalil, and S. Mohammed, “Upper Limbs Kinematics Estimation Using Affordable Visual-Inertial Sensors,” *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, 2022, doi: 10.1109/TASE.2020.3024033.

[34] N. Lord, W. Fulton, and J. Harris, *Representation Theory: A First Course*, vol. 79, no. 486. 1995.

[35] Y. T. Liu, Y. A. Zhang, and M. Zeng, “Sensor to segment calibration for magnetic and inertial sensor based motion capture systems,” *Measurement (Lond)*, vol. 142, 2019, doi: 10.1016/j.measurement.2019.03.048.