Motion Capture Modeling of Dexterous Hand for Intelligent Sensing

Xiaoyan Zhao[1,2](✉), Siyi Cui[1], Zhaohui Zhang[1,3], Qi Cao[1], Yuan Yuan[1,2], Xianhao Wu[1,2], and Shaowen Zheng[1]

[1] School of Automation and Electrical Engineering, University of Science and Technology Beijing, 30# Xueyuan Road, Haidian District, Beijing 100083, China
18510424607@163.com, {zhaoxiaoyan,zhangzhaohui}@ustb.edu.cn

[2] Beijing Engineering Research Center of Industrial Spectrum Imaging, University of Science and Technology Beijing, 30# Xueyuan Road, Haidian District, Beijing 100083, China

[3] Shunde Innovation Institute, University of Science and Technology, Beijing, Fo Shan 528399, China

Abstract. Motion capture technology is an emerging comprehensive multidisciplinary field cross technology involving graphics, engineering and communication technology. In order to acquire the posture of dexterous hand, a motion capture model for intelligent perception of dexterous hand was proposed in this paper. Sensing system, which is composed of 16 six axis inertial sensors is used to obtain the acceleration and angular velocity of dexterous hand movement. The Kalman filter algorithm was used to calculate the angle after sensor calibration and fusion. The sensor data was sent and received through an acquisition board using a serial port and the collected sensor angles were transmitted through multiple threads with Socket. A buffer was written to store a large amount of data, which map the fused angle with the hand model on the Unity 3D platform, and transmit the angle information to the corresponding virtual hand joint according to the port number to achieve real-time transmission. This article mapped 15 knuckles one by one, resulting in more accurate mapping. Socket was used for data transfer, offering a faster speed compared to USB. Experimental results show that the proposed method could capture the movements of dexterous hands, and realize the interaction between virtual hands and dexterous hands in Unity 3D. Besides, the angle information is saved by the Json file.

Keywords: Inertial Sensor · Socket Communication · Unity 3D Modeling · Dexterous Hand

1 Introduction

In recent years, capturing hand movements with high precision for natural human-computer interaction [1] has been explored for many applications, such as Virtual Reality (VR), gaming, robotics, biomechanical analysis, and rehabilitation. The acquisition of position and attitude data is based on the premise of transforming reality information into virtual reality, which is the primary problem to be considered in the geometric modeling

B. Xin et al. (Eds.): IWACIII 2023, CCIS 1932, pp. 329–342, 2024.
https://doi.org/10.1007/978-981-99-7593-8_28

and motion reconstruction of virtual hand. There are many ways to collect the position information and posture information of the hand. According to whether there is contact with the hand, data collection methods can be divided into contact data collection and non-contact data collection.

The non-contact data acquisition mainly uses computer vision to obtain the gesture information of the human hand. Wang [2] et al. proposed a new method to reconstruct a high-fidelity personalized hand model from multi-view color images of multiple poses. For each color image of poses, the convolution neural network was used to estimate two-dimensional key points of the hand, and the estimated key points with estimation errors were collected by using multi-view information. This method takes less time and produces a personalized hand model that synthesizes high-quality hand-datasets. However, non-contact data collection may encounter occlusion issues and can-not achieve good interaction during the modeling process. Nevertheless, contact measurement can solve the problem of occlusion.

The traditional contact measurement methods are based on fiber optic and inertial sensors, using fiber optic sensors [3] for measurement, such as 5DT gloves [4], which have high accuracy, stability, and data repeatability. However, due to its high cost, the above-mentioned gloves are usually used for scientific research and purchased in small quantities. Fiber optic sensors require intimate contact with fingers, which also leads to low measurement accuracy during long-term using. The advantages of using inertial sensors [5] for measurement are low cost, non-sheltered, intimate contact with hand joints, and high accurate measurement. Therefore, they are widely used in scientific research. The method was composed of a nine-axis inertial sensor (including a three-axis accelerometer, gyroscope and geomagnetic sensor), a voltage adapter and a Raspberry PI to realize a manual motion capture system. The proposed system showed good accuracy in both static and dynamic conditions. Because the geomagnetic sensor is greatly affected by the magnetic field, the six-axis inertial sensor was selected in this research, which could still be measured without being affected by the magnetic field.

This paper focuses on the modeling of motion capture for dexterous hand. Based on the inertial sensors, we collected the angle information of each joint of the hand and transferred the angle information to the virtual hand in Unity 3D. There are two common communication methods: USB and Socket. Previous studies usually adopted USB communication method [6]. However, the speed of USB communication is obviously lower than that of Socket communication. Thus, this paper chooses Socket communication to build the local area network, and realize high-speed real-time data transmission. The mechanical dexterous hand used for experiments can be bent through the upper computer system to achieve free grasping function. The dexterous hand is pulled by the tendon rope to move the knuckles. The upper computer sends the instruction and the dexterous hand begins to move. Innovatively applying Socket communication to mechanical dexterous hands enables the capture of dexterous hand movements and the storage of angular information for angle analysis.

The innovation and advantage of this paper are mainly reflected in the selection of sixteen angle sensors to capture the angular data, as opposed to using eight or eleven sensors for measuring each finger joint in the past, thereby providing a more accurate measurement for each joint, mapping every finger joint individually, compared to previous literature. Compared to the previous choice of USB communication, Socket communication is selected for faster transmission speed, which can realize real-time interaction between the client and the server.

2 System Composition

The system consisted of two parts: data acquisition and data processing [7]. Sixteen angle sensors were selected in the data acquisition part to obtain the original posture information of the dexterous hand. The angle information of the sensors was collected through the microcontroller board, and the communication between the microcontroller and the angle sensor was through serial communication. Each microcontroller could simultaneously collect data from eight angle sensors and transmit the data to the computer through ethernet network. The microcontroller was connected to the switch through a network cable, and the switch was then connected to the computer to achieve real-time data transmission. In the data processing section, the angle of each joint was calculated through the data processing module, and the angle information was transmitted to the corresponding joints in the Unity hand model to achieve the motion capture modeling of the dexterous hand.

2.1 Hardware Design

The hardware design of dexterous hand Motion capture system includes the following parts: (1) The inertial sensor IMU for hand Motion capture. (2) The acquisition board for collecting data from eight inertial sensors. (3) A switch that connects two acquisition boards. (4) A computer that processes sensor data. The system layout is shown in Fig. 1.

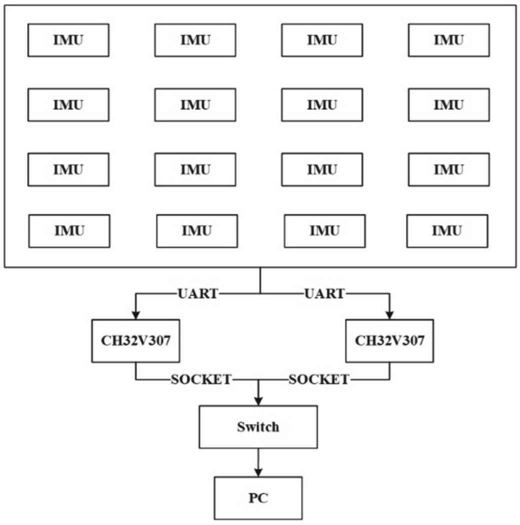
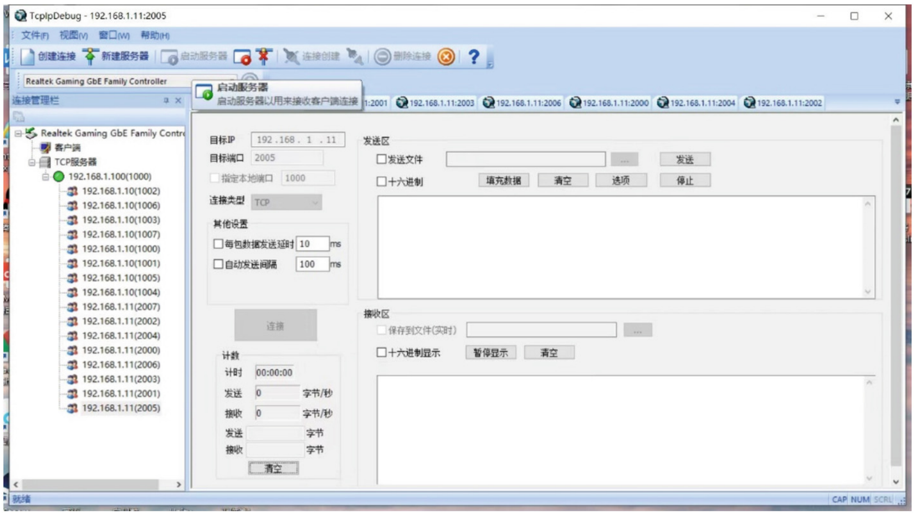


Fig. 1. System layout.

In order to obtain hand rotation angle with low cost and high precision, the six-axis inertial sensor MPU6050 is selected in this paper, which can export the rotation angle of the three-axis gyroscope and the three-axis accelerometer. The angular velocity range of the gyroscope is ±2000°/s, the acceleration range of the accelerometer is ±16 g, the frequency of the sensor is 100 Hz, and the baud rate is 9600. The accuracy of the angle sensor is 0.2°, which can capture hand movements in real-time and at high speed.

The model selected for the acquisition board is CH32V307, which is a universal microcontroller designed based on the 32-bit RISC-V of highland barley. This is a very characteristic resource structure, with eight sets of USART/UART serial ports, two I2C interfaces, and three SPI interfaces. This article aims to collect a large amount of sensor data, so eight sets of USART/UART serial ports were selected for sensor data collection, which are three sets of universal synchronous/asynchronous transceivers (USART1, USART2, and USART3), and five sets of universal asynchronous transceivers (UART4, UART5, UART6, UART7, UART8). This acquisition board can achieve the function of a serial server, converting the serial port into a TCP/IP protocol network interface, and achieving bidirectional transparent data transmission between the serial port and the network interface.

The acquisition board and sensors were connected, with eight sensors connected to each acquisition board. The IP address of the sensor connected to the first acquisition board was 192.168.1.10, with port numbers ranging from 1000 to 1007. The IP address of the sensor connected to the second acquisition board was 192.168.1.11, with port numbers ranging from 2000 to 2007. The two acquisition boards were connected to the upper computer software through a switch. The server IP address was set to 192.168.1.100, and the port number was set to 1000. As shown in Fig. 2, after starting the server, 16 sensors successfully connected as clients.


TcpIpDebug - 192.168.1.11:2005

启动服务器
启动服务器以用来接收客户端连接

Realtek Gaming GbE Family Controller
- 客户端
- TCP服务器
  - 192.168.1.100(1000)
    - 192.168.1.10(1002)
    - 192.168.1.10(1006)
    - 192.168.1.10(1003)
    - 192.168.1.10(1007)
    - 192.168.1.10(1000)
    - 192.168.1.10(1001)
    - 192.168.1.10(1005)
    - 192.168.1.10(1004)
    - 192.168.1.11(2007)
    - 192.168.1.11(2002)
    - 192.168.1.11(2004)
    - 192.168.1.11(2000)
    - 192.168.1.11(2006)
    - 192.168.1.11(2003)
    - 192.168.1.11(2001)
    - 192.168.1.11(2005)

1:2001 | 192.168.1.11:2003 | 192.168.1.11:2006 | 192.168.1.11:2000 | 192.168.1.11:2004 | 192.168.1.11:2002

目标IP 192.168.1.11
目标端口 2005
指定本地端口 1000
连接类型 TCP
其他设置
每包数据发送延时 10 ms
自动发送间隔 100 ms
计数
计时 00:00:00
发送 0 字节/秒
接收 0 字节/秒
发送 字节
接收 字节

文件(F) 视图(V) 窗口(W) 帮助(H)
创建连接 新建服务器 启动服务器 连接创建 删除连接
连接管理栏
发送区
发送文件
十六进制
发送 填充数据 清空 选项 停止
连接
接收区
保存到文件(实时)
十六进制显示
暂停显示 清空
清空
就绪
CAP NUM SCRL

Fig. 2. Server connection diagram.

2.2 Hand Joint Analysis and Sensor Distribution

According to the physiological anatomy model, the human hand contains bones, tissues and muscles, the movement of the smart hand can be represented by the rotation of the joint of the hand, so the sensor can be arranged according to the joint of the hand. The human hand has four fingers, namely the index finger, middle finger, ring finger, and little finger. Each finger is comprised of three knuckle joints, specifically referred to as the metacarpophalangeal joint (MP), proximal interphalangeal joint (PIP), and distal interphalangeal joint (DIP). As for the thumb, it only has two joints: the interphalangeal joint (IP) and the metacarpophalangeal joint (MP), and the MP is connected to the carpometacarpal joint (CM). According to reference [8], the length of adult finger knuckles is shown in Table 1.

Table 1. The length of adult finger knuckles (unit: cm).

	Proximal knuckles	Middle knuckles	Distal knuckles
Thumb	45–55	20–35	28–33
Index finger	43–50	24–30	23–26
Middle finger	44–51	25–31	24–27
Ring finger	43–50	24–30	23–26
Little finger	27–42	23–26	21–24

A dexterous hand is designed based on the joints of the human hand, with three joints in each of the five fingers. 15 sensors need to be deployed, and sensors need to be deployed on the palm of the hand to determine the state of the hand. Therefore, a total of 16 sensors are needed to predict the bending state of the fingers. The size of the hand joint was set according to the size of the human hand [9]. In order not to interfere with the bending of the dexterous hand, the sensor on the palm should be installed at the back

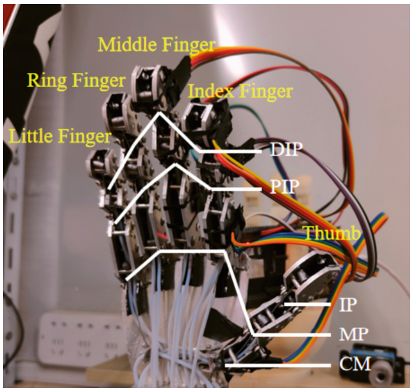


Fig. 3. Schematic diagram of a dexterous hand.

of the hand near the forearm. Fifteen IMU sensors were placed on five fingers and one IMU was placed on the back of the hand. Joint angles and hand posture can be estimated through 16 sensors. The dexterous hand is shown in Fig. 3.

2.3 Software Design

The software design included data processing module and interactive module. The initial step involved creating the TcpClient class to manage the connection with the client, where the client represents the sensor and Unity 3D acts as the server. A thread was opened for each client based on the Socket communication mode, and multi-threading was employed to receive sensor data concurrently. In the data processing module, binary data sent by the microcontroller was received and processed according to the frame structure. The data packet contained gyroscope, acceleration, and angle data from the sensor. The module unpacked the data packet and stored the data into a ring buffer. Take the data from the buffer, judge the position of the angle packet head, convert the data behind the packet head into Decimal angle data, and calculate the attitude of the data to obtain the angle data.

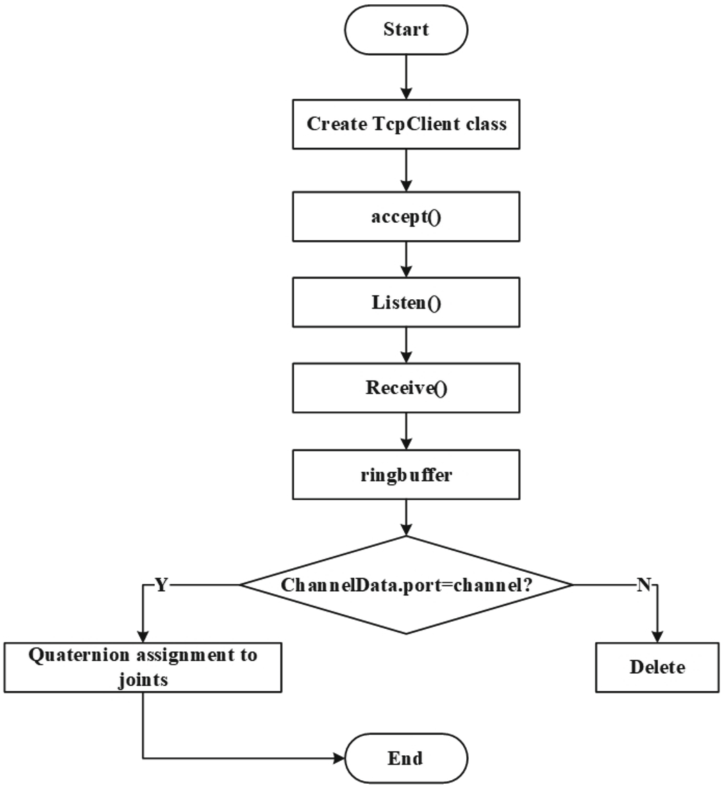


Fig. 4. Socket communication flowchart.

The interaction module included hand modeling, setting parent-child hierarchical relationships between fingers, and adding a bone skin renderer. By determining whether the port number of the sensor matches the channel in the program, the interaction between the dexterous hand and the hand model is achieved. If the port numbers are consistent, the sensor data will be transmitted to the corresponding joints to achieve the motion capture function of the dexterous hand. The Socket communication flowchart is shown in Fig. 4.

3 Implementation of Motion Capture System

3.1 Sensor Calibration and Filtering

After prolonged use, sensors may develop certain inaccuracies, making calibration of the sensor necessary. In a horizontal stationary state, the output of the gyroscope should be 0, the output of the accelerometer's X and Y axes should be 0g, and the output of the Z axis should be 1g.

The calibration of a gyroscope required taking the average of multiple samples as the bias value while stationary, and subtracting the bias value after measurement to obtain the true value. The possible errors in an accelerometer include zero bias error, scale factor error, and non-orthogonality error. Accelerometer calibration was used to remove the zero bias of the accelerometer. The accelerometer could be calibrated during motion through algorithms [10]. The sensor may have zero bias errors at different angles when leaving the factory, and measurement could only be used normally after manual calibration. After the accelerometer was calibrated, as shown in Fig. 5, X-axis and Y-axis accelerations are 0, and Z-axis accelerations are 1.

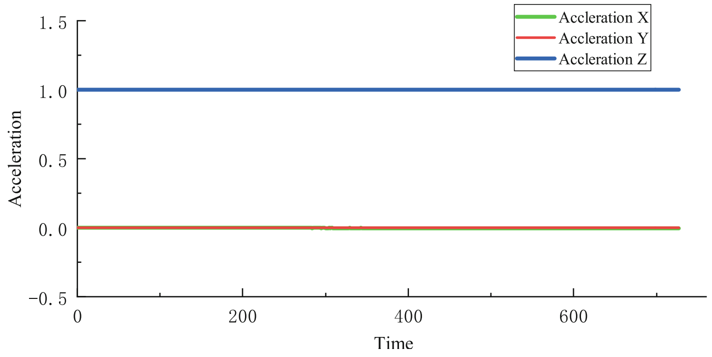


Fig. 5. X-axis, Y-axis, and Z-axis parameters after accelerometer calibration.

Kalman filter is an algorithm that uses the state equation of a linear system to estimate the optimal state of the system through the input and output of the observed data. Because of the noise interference in the observed data, the optimal estimation can also be regarded as a process of filtering, which is suitable for linear Gaussian systems. The principle of Kalman filtering is to use Kalman gain to correct the predicted state value, making it approximate the true value. By calculating the covariance of the optimal state estimation, the Kalman gain matrix is calculated, which is used to correct the state prediction value to achieve the optimal estimation of the state.

In this paper, Kalman filter algorithm was used to realize multi-sensor fusion, and the attitude was calculated by combining the information of accelerometer and gyroscope. The advantages of both were combined to obtain a method that can accurately measure the attitude in dynamic environment. The Kalman filter is very suitable for changing systems. Through the attitude fusion algorithm of the Kalman filter, the X, Y, and Z axis angles of the fused gyroscope and accelerometer can be calculated. The angle accuracy can reach 0.2° for the X and Y axis, and 0.5° for the Z axis due to the cumulative error.

3.2 Data Transmission and Processing

The acquisition board was connected with the sensor through USART/UART to realize the data sending and receiving function. The acquisition board was connected to the computer via Ethernet. The interaction between the acquisition board and Unity 3D can be carried out in the form of serial communication or Ethernet. Ethernet communication has a transmission rate of 100 Mbps, while serial communication has a transmission speed of 9600 bps (bit/s). Since Ethernet transmission speed is faster, this paper chose Ethernet for transmission. Each sensor has an IP address and a separate port number. The IP addresses of the eight sensors are 192,168,1,11, and the port numbers range from 1000 to 1007. The server-side program was written on Unity 3D. Eight sensors were used as clients to conduct Socket communication with Unity 3D.

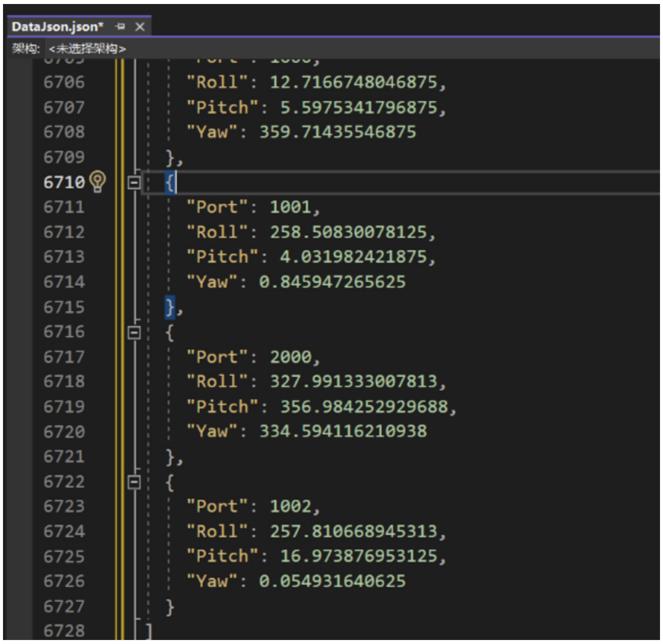
The original sensor data collected by the acquisition board is binary data. In order to ensure the reliability of the transmitted data and no packet loss, frame check and buffer Settings need to be carried out, and ring buffer Settings need to be set for data storage. When a data element is read out, other data elements do not need to move their storage positions, so the buffer length does not need to be set too long. The setting of the ring buffer allows large amounts of data to be stored, ensuring that the data is up-to-date every time it is read. When solving the attitude, the angle data packet needs to be solved. First, identify the position of the angle data packet, and then use the formula to convert the binary data into Decimal data.

$$Roll = ((RollH << 8)|RollL)/32768 * 180^\circ \tag{1}$$

$$Pitch = ((PitchH << 8)|PitchL)/32768 * 180^\circ \tag{2}$$

$$Yaw = ((YawH << 8)|YawL)/32768 * 180^\circ \tag{3}$$

A server program was compiled in Unity 3D to create a queue. The port number of the sensor, as well as the Roll, Pitch, and Yaw values obtained after attitude calculation, were written to the queue in real-time. All data in the queue was stored in a circular buffer for data retrieval and processing. A program was implemented to store the data in Json format and generate a Json file. While the program was running, the values in the queue were stored in real-time in Json. This facilitated the storage of data from various sensors during dexterous hand movements. The stored data was then transferred to different software for calculations, enabling the computation of flexion angles for each knuckle. These angles could be used for subsequent angle analysis. Json data storage is shown in Fig. 6.



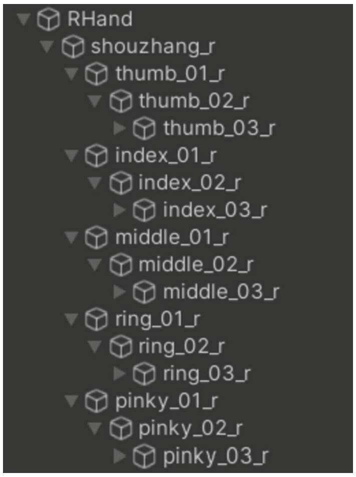
```
DataJson.json*
架构: <未选择架构>
            "Roll": 12.7166748046875,
            "Pitch": 5.5975341796875,
            "Yaw": 359.71435546875
        },
        {
            "Port": 1001,
            "Roll": 258.50830078125,
            "Pitch": 4.031982421875,
            "Yaw": 0.845947265625
        },
        {
            "Port": 2000,
            "Roll": 327.991333007813,
            "Pitch": 356.984252929688,
            "Yaw": 334.594116210938
        },
        {
            "Port": 1002,
            "Roll": 257.810668945313,
            "Pitch": 16.973876953125,
            "Yaw": 0.054931640625
        }
    ]
```

Fig. 6. Json storage data.

Since the Euler angle will appear gimbal lock when rotating around the object coordinate system, it is necessary to convert the Euler angle into quaternion and then assign to the joint in Unity. Quaternion can not only avoid the problem of gimbal lock, but also carry out the angle of rotation from one direction to the other direction greater than 180°. The Roll, Pitch, and Yaw values of each sensor were converted to quaternions and mapped onto the corresponding finger joints of the virtual hand in Unity 3D, enabling motion capture.

3.3 Model Architecture

A scene containing virtual hands was built on Unity 3D. The appearance of virtual hand is usually based on grid model [11] and ball-and-stick model. The grid model could be modeled by polygon to better fit the real hands, so the grid model was chosen in this paper. Add a parent-child relationship to the finger, forming a parent-child relationship from bottom to top. The whole hand was the parent object, the palm was the child object of the hand, each knuckle was the child object of the palm, the metacarpophalangeal joint of each finger was the parent object, the proximal interphalangeal joint was the child object of the metacarpophalangeal joint, and the distal interphalangeal joint was the child object of the proximal interphalangeal joint. The hierarchy diagram of the hand joint created in Unity Hierarchy view is shown in Fig. 7.



```
RHand
  shouzhang_r
    thumb_01_r
      thumb_02_r
        thumb_03_r
    index_01_r
      index_02_r
        index_03_r
    middle_01_r
      middle_02_r
        middle_03_r
    ring_01_r
      ring_02_r
        ring_03_r
    pinky_01_r
      pinky_02_r
        pinky_03_r
```

Fig. 7. Hierarchy diagram of the hand.

The virtual hand was designed after the human hand. In the process of modeling the virtual hand, it is not only necessary to establish a father-son hierarchical relationship, but also necessary to add a skinned mesh render to the virtual hand and import the skinned mesh into the grid.

The skin depended on the bone to make the hand move, and the addition of the skin made the position of the bone still have stretching effect after the change. In the materials element, we selected a material that was close to the human hand skin and added it to the virtual hand to make it closer to the human hand skin. A collider was added to the virtual hand. In order to eliminate the penetration phenomenon that would occur when two objects without colliders collided, it was necessary to add a physical collider to the virtual hand model to give it a physical collision. The virtual hand is shown in Fig. 8.

A one-to-one mapping structure was established between the sensor and each joint of the hand. Scripts were written in Unity to identify different sensors by port numbers, convert the fused Euler angles into quaternions, and map the Euler angles obtained through sensor conversion to the corresponding finger joints. Unity 3D has object coordinate

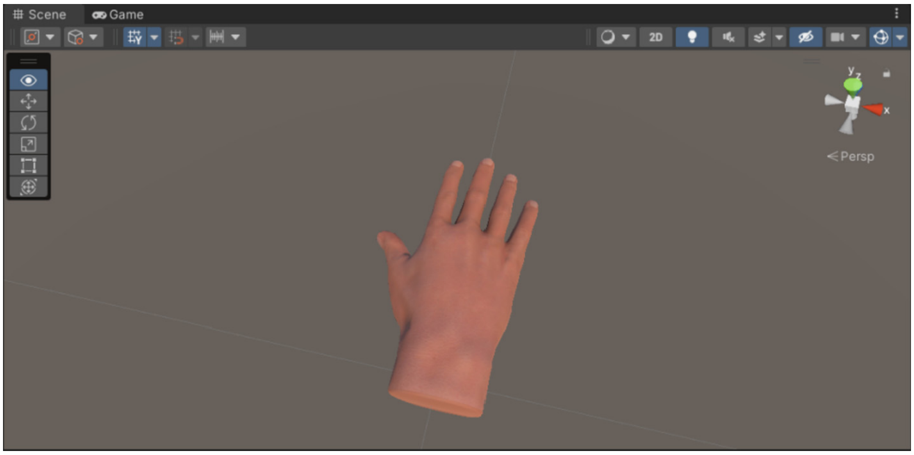


Fig. 8. The virtual hand.

system and world coordinate system, and sensor also has its own coordinate system, so coordinate system needs to be transformed. The transformation relationship was shown in Table 2.

Table 2. Unity object coordinate system and sensor coordinate system transformation.

Object coordinate system in Unity	Coordinate system of attitude angle
X	-X
Y	-Z
Z	-Y

The sensor was fixed on the smart hand, and the upper computer software LabView was used to control the movement of the smart hand. The smart hand pulled the tendon rope through the steering machine to drive the movement of each finger joint, and the length of the tendon rope movement corresponds to the movement angle of the smart hand finger joints. Each finger joint was driven by a steering machine to achieve accurate control of the finger joint, and angle data was recorded during the movement. The saved real-time angle data was stored in a Json file for angle analysis.

3.4 Output of Experimental Results

The sensor was placed flat on a table for calibration, and then the sensor was placed on the three knuckles of the index finger, middle finger and the carpometacarpal joint. The dexterous hand is not perpendicular to the ground in its initial state, but has a certain degree of bending. The program was run and the initial state of the dexterous hand was recorded after a three-second pause, as shown in Fig. 9.

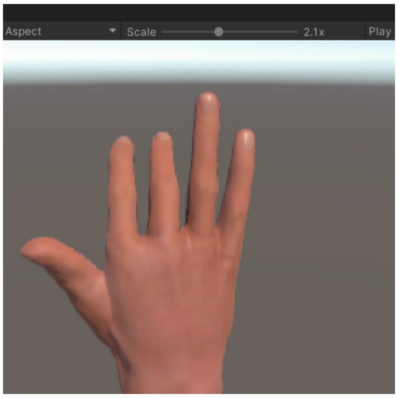


Fig. 9. Initial state.

The thumb, index finger, and middle finger were bent separately, and ultimately, they were bent simultaneously as shown in Fig. 10. In order to directly observe the bending of the finger joints of the dexterous hand, a front view of the dexterous hand was chosen. It was observed from the experimental results that the device described in this paper was able to display the real-time movement of the dexterous hand through Unity 3D and achieved motion capture of the dexterous hand.

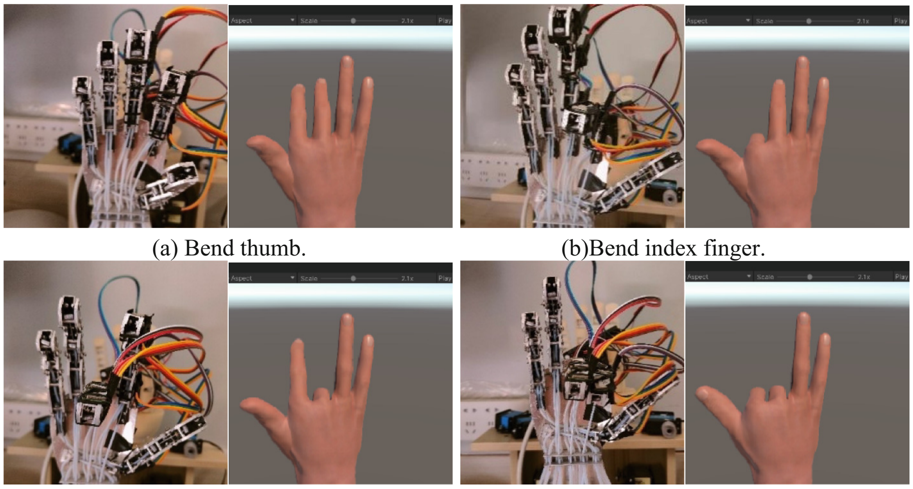
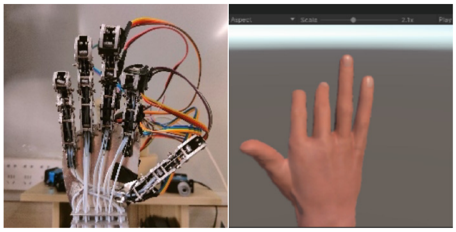
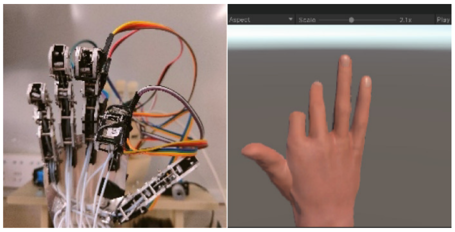
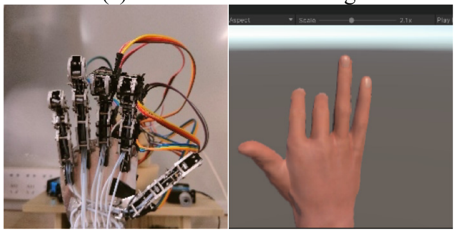


Fig. 10. Bend gesture diagram.

In order to observe the movement of the proximal interphalangeal joint and the distal interphalangeal joint, the distal interphalangeal joint of the index finger, the proximal interphalangeal joint of the index finger, the distal interphalangeal joint of the middle finger, and the proximal interphalangeal joint of the middle finger were bent successively, as shown in Fig. 11. The experimental results demonstrate that this method exhibits excellent real-time performance and enables real-time interaction with dexterous hands.


(a) Bend DIP of index finger.


(b) Bend PIP of index finger.


(a) Bend DIP of middle finger.

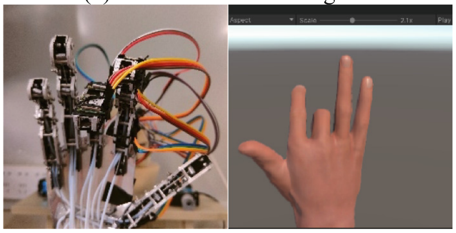

(b) Bend PIP of middle finger.

Fig. 11. Bend each joint.

4 Conclusion

This article focused on the modeling of motion capture for dexterous hand perception and designed a multi-sensor data acquisition system, attitude solving module, and interaction module. This article collected real-time data from multiple angle sensors, processed a large amount of data, calibrated the sensors, fused the sensor data, and stored the attitude angles of the fused sensors in a Json file. During the communication process, Socket communication is a high-speed data transmission method, the sensors were used as clients, while Unity 3D served as the server, facilitating parallel transmission of client data to the server through multiple threads. The sensor angle is correspondingly mapped to the virtual hand by determining whether the port numbers are consistent.

We propose a motion capture modeling method for dexterous hand perception that requires multiple IMU as inputs and runs in real time, avoiding occlusion issues in camera-based systems. It can be used for capturing dexterous hand motion and plays an important role in real-time acquisition by multiple sensors.

Acknowledgements. This work was supported by the National Key Research and Development Project (2020YFF0304704) and Scientific and Technological Innovation Foundation of Shunde Innovation School (BK20BE019).

References

1. Yang, M., Lei, Y., Fuhai, Z., Yili, F.: Design of human-computer interaction system for hand rehabilitation robot. Mech. Electron. **35**(07), 69–72 (2017)
2. Yangang, W., Ruting, R., Cangqing, Z.: Personalized hand modeling from multiple postures with multi-view color images. Comput. Graph. Forum **39**(7), 339–350 (2020)
3. Haijiang, G., Fei, P., Ning, J., Shiyang, L.: Research on virtual hand motion modeling system based on data glove. Electron. Des. Eng. **28**(5), 167–170 (2020)
4. Zhichun, Z., Xiaowen, Y., Liqun, Q., Xie, H.: Research on gesture simulation based on virtools and 5DT data glove. Sci. Technol. Eng. **15**(04), 140–144 (2015)
5. Chenghong, L., Jiangkun, W., Lei, J.: Hand motion capture system based on multiple inertial sensors: demo abstract. In: Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys'20), pp. 619–620 (2020)
6. Fei, F., et al.: Development of a wearable glove system with multiple sensors for hand kinematics assessment. Micromachines **12**(4), 362 (2021)
7. Christopher, H., Katrin, W., Mathias, W.: Whole hand modeling using 8 wearable sensors: biomechanics for hand pose prediction. In: Proceedings of the 4th Augmented Human International Conference (AH '13), pp. 21–28. Association for Computing Machinery, New York, NY, USA (2013)
8. Doyle, J. R., Botte, M. J.: Surgical Anatomy of the Hand and Upper Extremity, 1st edn. Lippincott Williams & Wilkins (2002)
9. Ming, A., Shanguang, C., Yuqing, L.: Research and implementation of virtual hand accurate modeling based on Data glove. Comput. Simul. **27**(01), 241–244 (2010)
10. Chenghong, L., Zeyang, D., Jing, L.: Measurement of Hand Joint Angle Using Inertial-Based Motion Capture System. In: IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1–11(2023)
11. Zhu, Z., Gao, S., Wan, H., Yang, W.: Trajectory-based grasp interaction for virtual environments. In: Nishita, T., Peng, Q., Seidel, H.-P. (eds.) CGI 2006. LNCS, vol. 4035, pp. 300–311. Springer, Heidelberg (2006). https://doi.org/10.1007/11784203_26