ASHESI UNIVERSITY COLLEGE

BUILDING A LOW-COST MOTION CAPTURE SUIT FOR ANIMATION

CAPSTONE PROJECT

B.Sc. Computer Engineering

Kwabena Nyarko Dennis

2021

ASHESI UNIVERSITY COLLEGE

BUILDING A LOW-COST MOTION CAPTURE SUIT FOR ANIMATION

.

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University College in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Engineering.

Kwabena Nyarko Dennis

2021

DECLARATION

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Name: Kwabena Nyarko Dennis

Date: April 27, 2021

I hereby declare that preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University College.

Supervisor's Name: Dr. Siby Samuel

Date: April 27, 2021

Acknowledgements

To my supervisor, Dr Siby Samuel whose encouragement and academic advice helped me undertake this project.

Abstract

Motion capture has become a major player in the film and animation industry. Mimicking natural and subtle movements of humans and animals has made animation a lot more believable and engaging. This paper presents a low-cost design of a motion capture unit based on the ESP32 and a 9-DOF inertial sensor Bno055. A calibration method based on a standing posture is used. The sensors data are retrieved via the serial port of the Arduino IDE and sent to Blender, a 3D program, to display real-time motion. If a believable performance can be achieved with low-cost inertial sensors, then the overall cost of existing motion capture technology can be reduced.

Table of Content

Chapter 1: Introduction

Motion Capture is the process of tracking the movements of people or objects and processing that information. Motion capture is used in many fields such as medicine, military, and the entertainment industry. In the Game, films and animation industry, the attention to detail of the graphics of the characters and scenes have increased dramatically of the past few years. Hair strands, sweat, cloth simulation, and fluid simulation have become more photorealistic. With budget-friendly software like Houdini, Blender, Cinema 4D, Maya, Unreal, and Unity, individuals and small teams can produce films, games, and animations of similar quality to the top studios in the world. However, as graphics begin to get more photorealistic, the movements of the characters are expected to be more convincing. It will be a lot more challenging to move the characters manually by keyframing. Motion capture makes animating these characters easy, and the natural movement of the performer sells the illusion of realism.

The earliest form of motion capture was rotoscoping, where actual footage was traced over, frame-by-frame. Rotoscoping was used in most of the animated films in the Golden age of Disney, including “Snow White and the Seven Dwarfs” and “Cinderella” [1].

Today, motion capture systems can be grouped into optical and video/markerless and inertial systems. Optical systems use cameras to detect LEDs, paint, or reflectors that are placed as physical markers on the body being tracked. Video systems do not require markers but rely on software to track movement. Inertial systems use inertial sensors (IMUs) to track the motion of objects [2-3].

Despite being the most accurate system currently, optical systems require camera setups in a controlled space and need the object to be always in the cameras’ view. This

makes it difficult to record complex and continuous motions such as skiing, skydiving, and cycling. Inertial motion, however, can record every motion possible in every environment and lighting conditions. They are also relatively cheaper and easier to setup than optical systems, making it ideal in most situations.

This work presents the use of a low-cost motion capture system that takes advantage of a cheap microcontroller and Inertial Motion Units (IMUs) to track motion used for animation.

Chapter 2: Background and Related Work

2.1 Background

IMUs can measure a variety of factors, including speed, direction, acceleration, specific force, angular rate, and (in the presence of a magnetometer), magnetic fields surrounding the device. IMUs can measure acceleration, angular rate, speed, direction, and magnetic fields neighbouring the sensor. Modern day IMUs have 3 triaxial accelerometers, gyroscopes, and magnetometers. The accelerometers are responsible for measuring acceleration and velocity. Gyroscope measures rotation and rotational rate and the magnetometer identifies the cardinal direction [4]. Sensor fusion algorithms can be used to figure out the sensor's orientation, position, and heading.

IMU based motion tracking systems make use of miniaturized IMUs placed on different parts of the body to calculate the orientations of those body parts. They are non-obtrusive, comparably cost effective and easy to setup and use [5]. High performance motion capture hardware like the MVN Link from Xsens consists of 17 or more IMUs with an update rate of 240hz. They can operate withing a range of 150m with a battery life of 8 to 10 hours. They operate via Wi-Fi. They are highly accurate and produce clean production ready data. Price point for such hardware is about $13,836.96. These are normally used mainly by used by big studios such as Avalanche Studios in the making of their AAA games like Mad Max [6].

Aside the high-performance suits, there are relatively cheaper suits geared towards independent studios and induvial. One popular suit amongst independent creators is the Rokoko suit. The Rokoko suit has 19 IMUs with a 6-hour operation time with 5000 mAh. It has real time streaming of up to 100 frames per second. The suit has separate plugins for

3d software of the user's choice [7]. Its price point is at $ 2,986.19 which is still arguably expensive especially for upcoming animators and filmmakers who are finding their feet in the industry. Until they can afford the motion capture systems, these people will not be able to reach their potential in the art. The aim of my project is to work on a low-cost motion capture system that the average consumer can afford and use.

2.2 Related Work

The literature on inertial motion capture is huge so in this paper, I focus only on low-cost inertial motion capture designs in general and motion capture designs that have been conceived specifically for Blender3D.

In [8] they design and implement a low-cost motion tracker system using a network of wired IMUs. They used miniaturized IMUs and textile cables which fits into a suit. The IMUs were connecter to buffers which were then connected to the I^2C bus and finally to the CPU. The power source was a rechargeable 9V battery. Although it is a low-cost design, it still uses 21 sensors which is more than enough and could be reduced further to lower the costs even more.

In [9], they design a low-cost motion capture system with inertial motion capture and a Kinect sensor. This system combines optical and inertial systems to provide precise calculations of joint angles. This system as designed to improve the motion capture of the Kinect system for biomechanical applications, however, it still has the problem that all optical systems have. They need to be in a controlled environment and to be always in the line of sight of the camera during the motion capture. Therefore, such a system will not be able to record in the dark or record complex motions that are not localised.

In [10], they build a wireless inertial motion capture with MPU-9150 inertial sensor and the nrf-51822 Bluetooth module at the nodes and placed at the joints. To derive the absolute orientation of the IMU, they put the IMU through stages of calibration and finally a sensor data fusion algorithm. They calibrated the gyroscope to remove the zero offset. The accelerometer was passed through a low pass filter to remove jitter effects. They then pass the calibrated data through a series of algorithms to generate the initial quaternion information to solve the problem of gimbal lock. They then convert the quaternion data to the Euler angles to easily represent the rotation matrix. Taking advantage of the internal fusion algorithms in the bno055 inertial sensor through the process of abstraction, I will easily get the quaternions directly from the IMUs without having to task the ESP32 with extra calculations thus reducing runtime. However, the wireless system proposed in this paper is less intrusive and will allow the use to move more freely as compared with mine.

For [11], the IMU Node design consists of MPU9250 inertial sensor with STM32F103 as MCU and NRF24L01 as RF transceiver. The TP4056E and the PL3500(662K) were selected for the power management of the nodes. For the PC to receive the information from the IMU node, the sink node was designed to act as a receiver. The STM32F407ZET6 MCU was used as the microcontroller and the NRF24L01 was used as the RF transceiver. A USB Serial Converter FT232 was used to connect the PC to the sink node to connect the motion capture unit to Unity3D, a 3d software used mainly for video games. Like [10], they calibrate the accelerometer, magnetometer and gyroscopes and generate the quaternions from the three sensors. They then converted the quaternions to Euler Angles.

In [12], a low-cost motion capture system is designed using MPU6050 integrated with Arduino Uno board. The output is serially given to the computer via Arduino board, which is then visualised in Unity3D. Angles to estimate orientation were generating by

passing the accelerometers and the gyroscopes through the complementary filter. This paper was a guide to using Arduino and the MPU6050 to record data as opposed to it being a design of a full body suit. Although the Arduino and the MPU6050 are cheap, the ESP32 and the Bno055, as a set, are relatively cheaper, which just goes to show that the cost could be reduced further.

In [13], they propose a wireless system. Each IMU node consists of GY-80 inertial sensor, an ESP8622-12E and a lithium polymer battery. The data is sent to a UDP server running in the Blender game engine to obtain a real-time animation of the avatar. The attitude estimation algorithm runs at 600Hz and the animation is displayed at 20FPS. Each node runs a quaternion based complementary filter to compute the orientation with respect to the inertial frame. Again, having wireless nodes is less intrusive, making wired systems less desirable. However, it is more expensive since all the nodes have their own microchip and battery. Perhaps a hybrid of wired and wireless could bring down the cost of this system whilst making it less intrusive than a fully wired system. The nodes on each limb could share one battery and microchip. But if the goal is to produce the cheapest system possible, then the wired approach is the way to go.

In [14], an arm tracker design is proposed to record the motion of the arm. The design utilises a single UM7-LT orientation sensor combined with an Unscented Kalman filter for the upper arm orientation quaternion. A potentiometer sensor is added to measure elbow joint angle estimations. The system has two wixel microcontrollers; one for transmitting the data from the sensors and the other microcontroller is connected to the PC for receiving data. The system transmits data to the PC wirelessly and real-time data is visualised using Blender3D. The system was verified with an Xsens MVN motion capture system. Having the potentiometer to calculate the rotation of the elbow is genius given that the elbow is a hinge joint and will only rotate in one direction. The only problem I see with this setup

(which might not be a big deal) is that you would require some sort of solid contraption strapped to the arms to help rotate the potentiometer. The design could potentially be weighty or intrusive which could prevent the user from moving freely.

Chapter 3: Requirements and Design

The motion capture system should be lightweight and unintrusive as possible for the free movement of the wearer. To record long hours of continuous movement, the suit should have a good battery life. The system should fit most body types so that more than one person could use the same system without it being too loose or too tight. The system should also have a good range of connectivity to allow for free movement whilst recording live data into the computer. The system should have low latency. The system should also be relatively inexpensive when compared with the ones in the market.

The main devices needed for the project were the IMUs and the microcontroller. For the IMUs I needed to choose from the low-cost IMUs in the market. The low-cost IMUs available were the MPU 9150, MPU 9255, MPU 6050, MPU 9250, Bno055 and the Bno080. They all had their pros and cons, but I chose the BNo055 purely because it possesses an in-built processor with fusion algorithm that will make quaternion calculation a lot easier. The BNO055 is a System in Package (SiP) solution that combines an accurate close-loop triaxial 16-bit gyroscope, a triaxial 14-bit accelerometer, a triaxial geomagnetic sensor and a 32-bit microcontroller running the BSX3.0 FusionLib software [15]. By adding sensors and sensors fusion algorithms in one device, the BNO055 makes integration easy, avoids multifaceted multivendor solutions and consequently streamlines innovations [16].

The microcontroller used in this project is the ESP32. The ESP32 is manufactured using Taiwan Semiconductor Manufacturing Company (TSMC)'s ultra-low-power 40 nm technology making it suitable for designing battery operated devices like wearables, audio equipment and smart watches [17]. The ESP32 is also of low-cost and equipped with ESP-WROOM-32 Module (consists of Wi-Fi + Bluetooth) and I2C serial communication protocol that will help send data from the IMUs to the PC.

The IMUs need a medium through which data is sent to the microcontroller. The question was that was it going to be through wires or wireless. With wireless, it was going to less intrusive and allow the wearer to move more freely so it seemed like the ideal choice. However, choosing wireless meant that each IMU would have to be accompanied by a WI-FI or Bluetooth module and its own power source which was going to come with its own charging station which was going to increase the overall cost and complexity of the design. Also, charging each IMU node after use will be tedious. Maintenance will also be a problem if more parts are included in the design. Due to all these reasons, opting for wired communication system between the IMUs and the ESP32 was the preferred choice. Additionally, the preferred communication protocol used by the Bno055 will be the I2C which the ESP32 only has two dedicated pins for. And with the Bno055s having an I2C address of 0x29 or 0x28, it would be impossible use the same dedicated pins for them. To solve this problem, the TCA9548A 1-to-8 I2C Multiplexer Breakout will be needed. Since the IMUs will be 15 and one I2C multiplexer has only 8 channels, an additional I2C multiplexer would be needed.

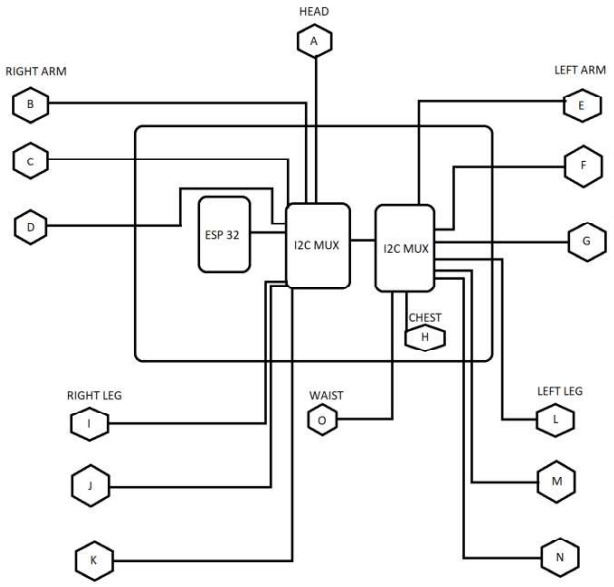

Figure 1.1: I2C Map of the Motion Capture Unit

Four pins from the Bno055, namely the VIN, GND, SCL, and SDA, will be connected to the power source, ground and I2C multiplexer, respectively. To reduce complexity in the wiring and to make it possible for the nodes to be detachable for easy storage after use, the RJ45 8P Pin Connectors with Breakout Adapter in conjunction with Cat 6 UTP Cables will be used. To avoid communication problems, there will be no master IMUs. All the IMUs will be connected directly to the I2C multiplexer.

The microcontroller and the two I2C multiplexer will be placed on a solid PCB and fitted into a vest in the chest area. 6 RJ45 8 Pin Connectors will be connected to the I2C multiplexers to receive data from the nodes from the limbs and the head. One IMU will be placed on the PCB along with the microcontroller and the I2C multiplexer and will be connected directly to the I2C multiplexer. For the power supply, a power bank is proposed. The power bank will be placed in a casing at the waist level in a utility belt structure. Also, there will be LEDs and a buzzer to indicate the on, calibration and recording status of the

system.

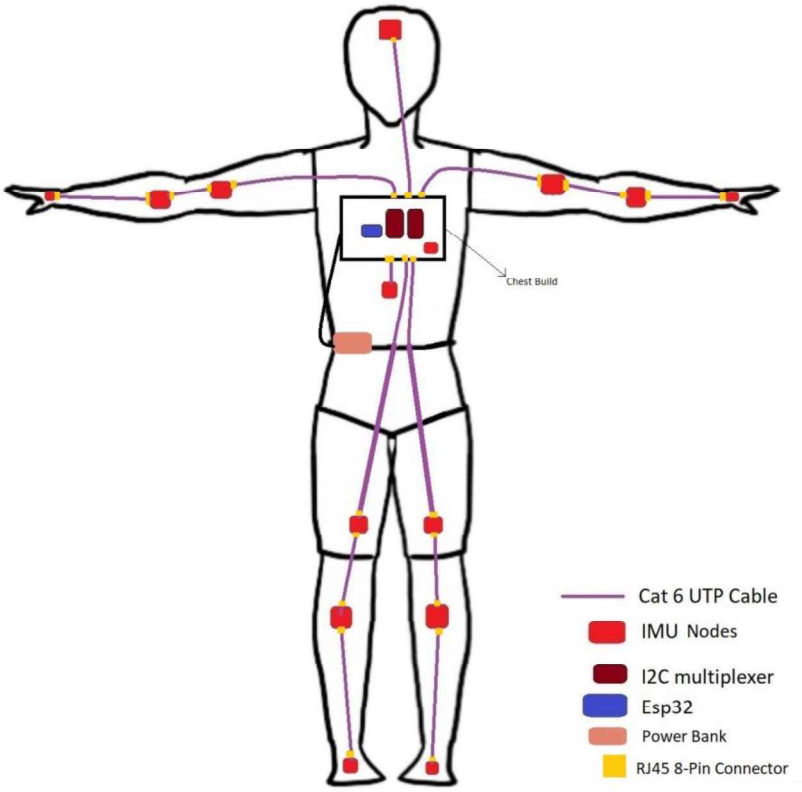

Figure 1.2: Overview of the Motion Capture System

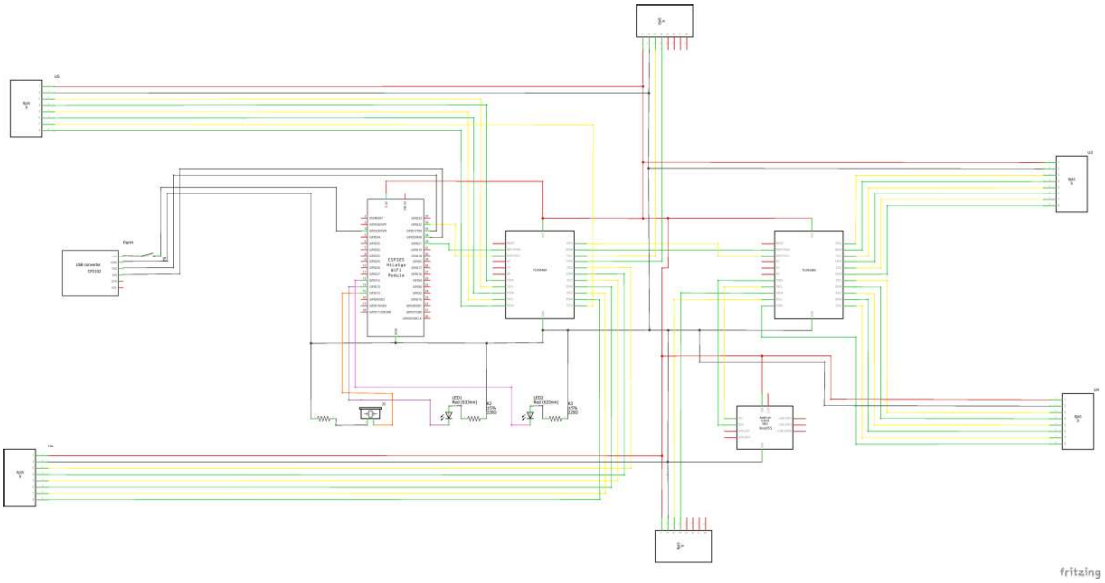

Figure 1.3: Schematic of the Chest Build

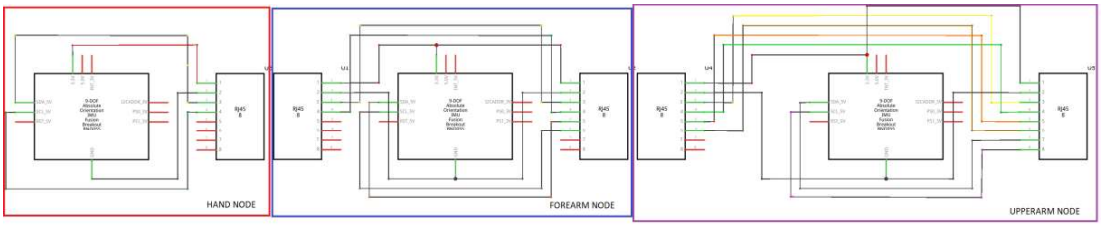

Figure 1.4: Forelimb Connection to Avoid Having Master IMUs

Item	Source	Quantity	Cost
ESP-WROOM-32	Amazon: HiLetgo Store	1	$10.99
GY-Bno055 Absolute Orientation Sensor	Amazon: Reland Sung	15	$9.18
RJ45 8-P Pin Connectors Breakout Board Adapter Set	Amazon: Milageto	27	$7.99
CP2102 USB 2.0 to TTL Module Serial Converter Adapter Module USB to TTL	Amazon: HiLetgo Store	1	$5.19
TCA9548A I2C IIC Multiplexer Breakout Board 8 Channel Expansion Board	Amazon: HiLetgo Store	2	$6.99
Cat 6 UTP Cable	Amazon: Ultra Clarity Cable Store	14	$11.04
5V Active Alarm	Amazon: QMseller	1	$0.75
LED	Amazon: DiCUNO	3	$0.03
Portable Charger 25800mAh	Amazon: HHETP	1	$18.95
30 AWG Flexible Silicone Wire	Amazon: striveday	1	$12.99
Total Cost			$570.93

Table 1.1: Bill of Electrical Components for The Full Body Motion Capture System

Chapter 4: Implementation

4.1 Use Case of the Motion Capture Unit

The Individual nodes are fitted onto adjustable straps that the user wears at the various regions indicated in figure 2. After setting up the nodes and the chest build, the user connects the power bank, which is also put into a casing attached to a utility belt, to the chest build. The user turns on the switch to power the system. The first LED lights up to show that the system is powered. The IMUs are initialized and the second LED lights up. If the IMUs are not detected, the second LEDs flashes to signal the user. At this point, the user will not be able to do much except reboot the system and hope that it works. Calibration of the IMUs begin immediately the IMUs have been initialised. The user must move about for a few minutes to calibrate all the IMUs properly. After, the second calibration of the IMUs with respect to the bones in Blender3D starts. The ESP32 connects wirelessly to the PC via Wi-Fi. A python script connects Blender to the ESP32 and the IMU data is applied to a 3D model in Realtime. During the second calibration, the user must stand in the pose similar to the standing pose of the 3D model in Blender3D. To indicate the end of the calibration, the buzzer at the chest beeps since the user will be in a pose that will prevent him or her from look downwards to the chest build where the LEDs are located. The third LED lights up after the beep to indicate the system is operating smoothly. Once the user is done, he or she turns off the switch off, unplugs the power bank, loosens the straps, and stores the motion capture unit in a safe place.

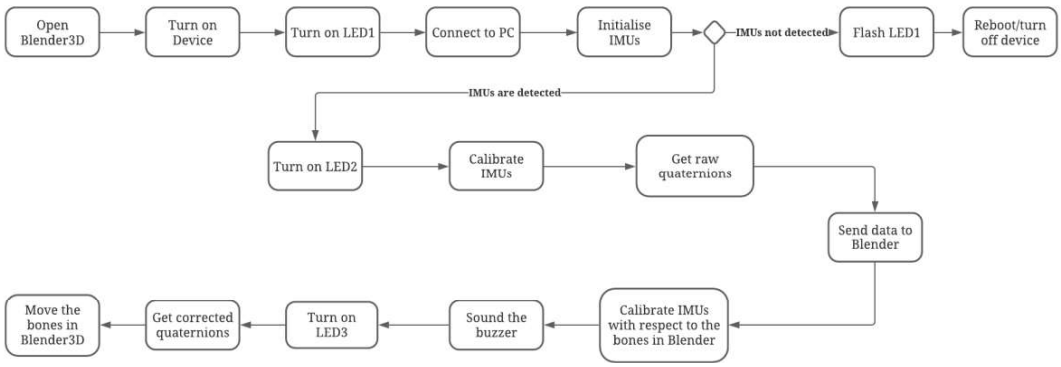

Figure 2.1: Activity Model of the Motion Capture Operation

4.2 Data Acquisition

Determining the orientation of the IMUs is a one of the ways to record motion. This is because the individual rotations of the various joints of the body work in harmony to move the entire body. Euler Angles and Quaternions are the conventional mathematical notations to represent rotation in the 3d space. Euler angles are simpler and intuitive since the roll, pitch and yaw can easily be interpreted and manipulated accordingly. However, a major disadvantage with working with Euler angles is the possibility of encountering gimbal lock. Gimbal lock is when a body in 3D space loses one degree of freedom because of two of the three gimbals being steered into a parallel alignment [18]. Quaternions are much more accurate and do not succumb to gimbal lock. Quaternions were invented by William Hamilton in 1843 to allow him to multiply and divide vectors to rotate and stretch them. Since there was no way to multiply or divide two sets of three numbers that resulted in three numbers (vectors that might represent coordinates), he introduced a four-number system which consisted of a scalar and a vector.

$$Q = (w, r)$$

$$Q = (w, xi, yj, zk)$$

$$Q = w + xi + yj + zk$$

Where w, x, y, z are real numbers and i, j, k are quaternion units [19].

Fortunately, the Bno055 has sensor fusion algorithm that combines accelerometer, gyroscope and magnetometer data and converts them into the quaternions. Using the Adafruit Bno055 library in the Arduino IDE, the quaternions for the individual IMUs are generated. The IMUs send to the ESP32 one after the other so quickly that it is seemingly simultaneous. The quaternions are printed on the serial port for Blender to read them.

4.3 Blender Software

Blender is a free open-source 3D software. It supports modelling rigging, simulation, composition, video editing, 2D, animation and motion tracking. Blender can be used to make high detail Hollywood quality films and animations. It can also be used in architecture, medicine, and sports.

3D characters in Blender are animated using armatures and bones which are like skeletons in the human body. The bones can rotate individually. There is a hierarchy of bones in the armature that allow parent bones to move children bones using forward kinematics. These bones can be rotated by interacting with the GUI of the bones in the 3D environment or by editing the Euler angles or quaternions. The motion capture unit will be rotating the assigned bones by real-time editing of the quaternions.

The real-time manipulation of the quaternions will be done using an extended feature of Blender which is the Python Script. Python script will allow Blender to connect to the serial port that the ESP32 is connected to with a baud rate of 9600. The python script will read the quaternions as strings and separate them into the various sensors whilst converting them to floats. The data is then applied to the bones assigned to the respective sensors.

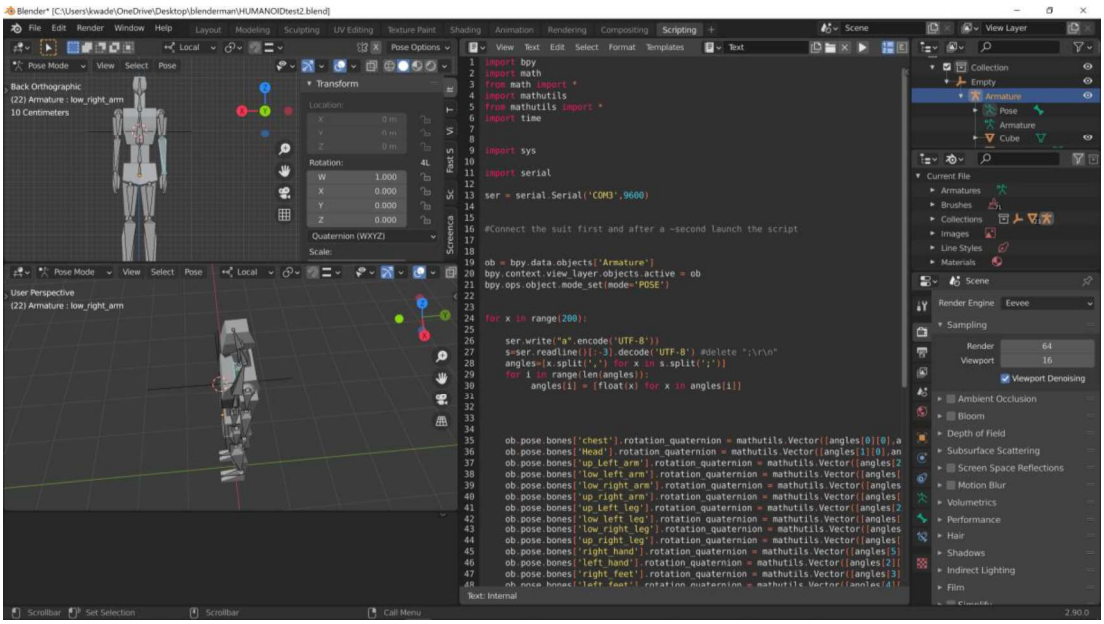

Figure 3.1: Blender Workspace with Python Script

The line “ser. Write (“a”. encode (‘UFT-8))” sends the “a” command to the serial port from Blender asking for the quaternions to be printed onto the serial monitor. This means that a major part of the upload speed of the system would rely on how fast Blender3D keeps asking for the data.

4.4 Calibration of the IMUs

First, the calibration of the IMU is done to improve its functionality. The following block of code, which is in the 'void setup' section of the code, performs the calibration [20].

```
//Calibrating the IMU
uint8_t system, gyro, accel, mag = 0;
while(system != 3)
{
bno.getCalibration(&system, &gyro, &accel, &mag);
Serial.print("CALIBRATION: Sys=");
Serial.print(system, DEC);
Serial.print(" Gyro=");
Serial.print(gyro, DEC);
Serial.print(" Accel=");
Serial.print(accel, DEC);
Serial.print(" Mag=");
Serial.println(mag, DEC);
delay(100);
}
Serial.println(""); Serial.println("Calibrated");
```

Figure 3.2: Calibration of the Bno055

The program waits for the system to be fully calibrated and sends a signal to the user using the LEDs. After, the calibration of the IMUs, the calibration of the motion capture unit commences.

4.5 Calibration of the Motion Capture Unit

Before I calibrated, I had to see how accurate the original quaternion values were at representing the orientation of the IMUs. Starting with the IMU at the chest, I run the Arduino code and the blender script and analyzed the results. I noticed that there was a significant error in the heading or compass direction of the IMU. But I did notice that when I rotated about the X, Y and Z axes, the IMU was able to pick up on the rotations and the pace at which those rotations were changing. So, the problem at hand was to figure out a way to way to derive quaternions that more accurately represent the desired orientation. The goal was to identify a reference quaternion and figure out how to apply a bias to the IMU quaternion to arrive at that reference quaternion. The bias was going to be an extra rotation

after the IMU has done its rotation. Essentially, there will be two rotations in the background and the resulting quaternion will be sent to the bones in blender. Quaternion multiplication combines two rotations in that manner [21]. The IMUs' internal fusion algorithm will handle precision, whilst the quaternion multiplication algorithm redirects the data to the desired position.

Consider a reference quaternion, Q_R, IMU quaternion, Q_{IMU}, and a bias quaternion, Q_B the equation to generate the bias quaternion be:

$$Q_{IMU} \cdot Q_B = Q_R$$

$${Q_{IMU}}^{-1} \cdot Q_{IMU} \cdot Q_B = {Q_{IMU}}^{-1} \cdot Q_R$$

$$Q_B = {Q_{IMU}}^{-1} \cdot Q_R$$

After generating the bias quaternion, the IMU quaternion will be multiplied to it to generate the corrected quaternion.

$$Q_{IMU} \cdot Q_B = Q_{Corrected}$$

```
void correctQuat(double IMUqW,double IMUqX,double IMUqY,double IMUqZ){
  //setting the reference quaternion
  double  QrW=1;
  double  QrX=0;
  double  QrY=0;
  double  QrZ=0;


  //getting the inverse of the imu quaternion
  double  inv_imucW=IMUqW;
  double  inv_imucX=-IMUqX;
  double  inv_imucY=-IMUqY;
  double  inv_imucZ=-IMUqZ;

  /*multiplying the inverse of imu with the reference
  quaternion to arrive at the bias quaternion which
  will eventaully be multiplied to the original IMU quaternion after calibration*/
  biasW=-inv_imucX * QrX - inv_imucY * QrY - inv_imucZ * QrZ + inv_imucW * QrW;
  biasX= inv_imucX * QrW + inv_imucY * QrZ - inv_imucZ * QrY + inv_imucW * QrX;
  biasY=-inv_imucX * QrZ + inv_imucY * QrW + inv_imucZ * QrX + inv_imucW * QrY;
  biasZ= inv_imucX * QrY - inv_imucY * QrX + inv_imucZ * QrW + inv_imucW * QrZ;
```

Figure 3.3: Algorithm to generate the Bias Quaternion.

```
double correctedcW=-biasX * chestqX - biasY * chestqY - biasZ * chestqZ + biasW * chestqW;
double correctedcX= biasX * chestqW + biasY * chestqZ - biasZ * chestqY + biasW * chestqX;
double correctedcY=-biasX * chestqZ + biasY * chestqW + biasZ * chestqX + biasW * chestqY;
double correctedcZ= biasX * chestqY - biasY * chestqX + biasZ * chestqW + biasW * chestqZ;

Serial.print(correctedcW,4);//w chestqW
Serial.print(",");
Serial.print(correctedcX, 4);//x chestqX
Serial.print(",");
Serial.print(-correctedcY, 4);//y chestqY
Serial.print(",");
Serial.print(-correctedcZ, 4);//z chestqZ
Serial.print(";");
```

Figure 3.4: Corrected Quaternions for the Chest IMU after calibration

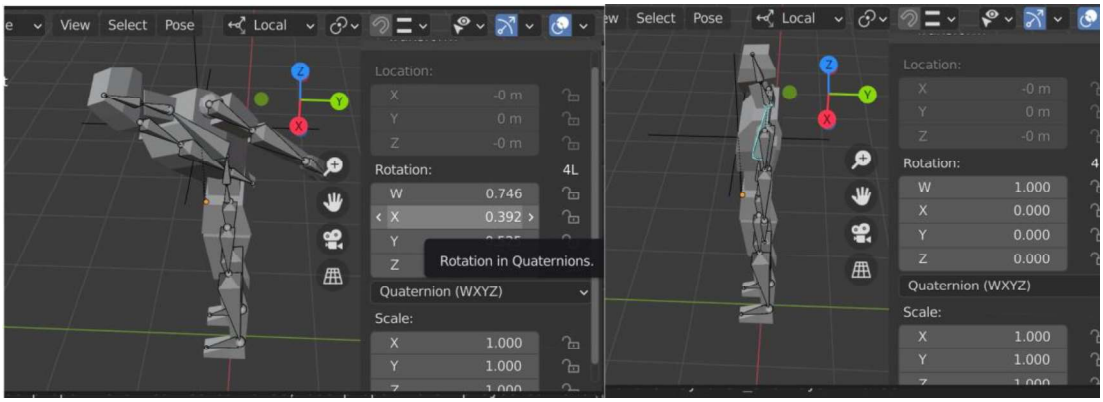

Figure 3.5: Before and After Calibration of the Chest IMU

During calibration, the bias quaternion is constantly updated with each iteration of the code meaning that the final bias quaternion will be as result of the latest IMU quaternion and the reference quaternion which is (1,0,0,0).

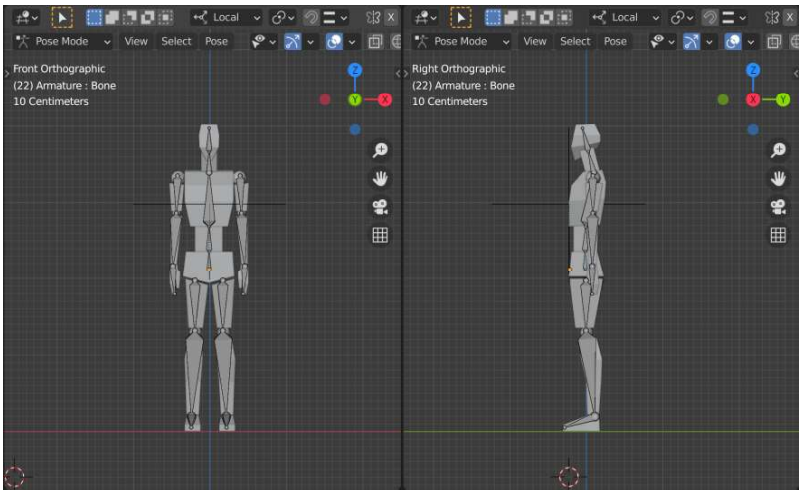

Figure 3.6: Standing Pose During Calibration

Chapter 5: Testing and Results

5.2 Calibration

For the testing phase, the chest IMU data was recorded by plotting the raw quaternions and the corrected quaternions on the same graph in real time to show the behaviour of the IMU and to show that the calibration process was necessary. The head, chest and upper limbs were also recorded, and their quaternions were mapped onto the bones of a 3D character. The real time posture of the joints was saved in real-time as well for playback.

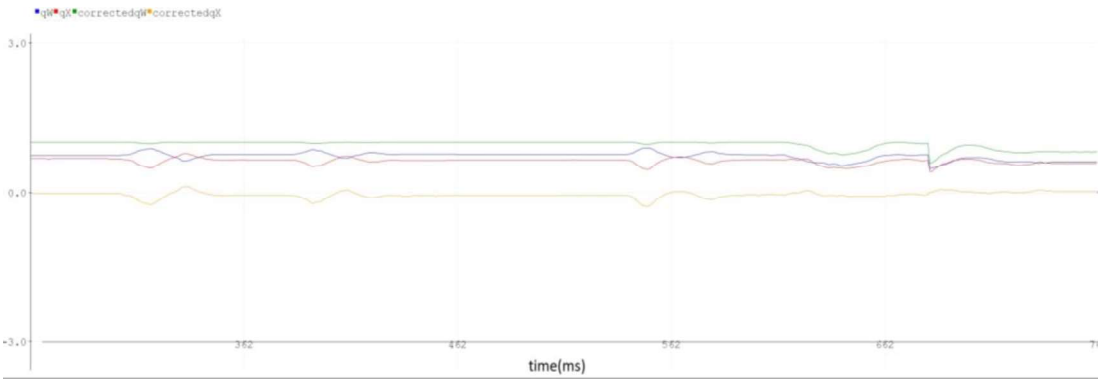

Figure 4.1: Raw IMU Data and Corrected IMU Data of The Chest IMU.

The left side of the graph shows the Chest IMU during the calibration phase. Represented by the green line, you can see that the corrected W quaternion remains stable at 1 even as the raw W quaternion data, which is represented by the blue line, changes. The W value is shifted to 1 regardless of the orientation of the IMU, which is why the user is supposed to be pose that the animated character is in. This helps establish a reliable reference for the motion capture after calibration has been done.

5.2 Power Consumption

For mobile devices, it is important to ensure that they have adequate energy to be able to perform the job given to them to the best of their abilities. The esp32 will be in active mode throughout since data will be transferred wirelessly and as fast as possible. Therefore, all the resources available by the microcontroller will be used. Reading the 15 Bno055 IMUs at 100Hz and wireless communication via the Wi-Fi 802.11 b/g/n module will consume the most power. The power consumption was measured using an oscilloscope and estimated to be 282.3mA. Given that the power bank is rated at 25800mAh, it will be more than enough to record long sessions.

5.3 Comparisons to Existing Motion Capture Unit in The Market.

The specifications of the suit proposed in the picture was compared the Rokoko Pro Suit and the Xsens MVN animate product line to see how it fared amongst the “big boys” in the market. The specifications were taken from their official websites[6][7].

	Proposed Motion Capture Unit	Rokoko Smartsuit Pro	MVN Awinda starter (Standard Performance)	MVN Awinda (Intermediate Performance)	MVN Link (High Performance)
Range	150m	100m	20m	50m	150m
Update Rate	60Hz	50Hz	60Hz	60Hz	240Hz
Battery and Power	External Power Bank	External Power Bank	6h	6h	8-10h
Comms	Wi-Fi + Ethernet Cables	Wi-Fi + cables	Radio Protocol (Awinda)	Radio Protocol (Awinda)	Wi-Fi
Receiver	Wi-Fi Router	Wi-Fi Router	Awinda Dongle	Awinda Station	Wi-Fi Router
Hardware	15 sensors	19 sensors	17 sensors	17 sensors	17 sensors
Charging	External Power Bank	External Power Bank	USB cable	Charging Station	USB Cable
Software & Plugins	Blender	Rokoko Studio, Unreal, Unity, Blender, Maya, MoBu, Houdini, Cinema4D, iClone	MVN Animate, Motion Cloud, Unreal, Unity, Blender, Maya, IIoudini, Cinema4D, iClone, Motion Builder, Pixotope, 3Ds Max, Notch, Fabric Enginc	MVN Animate, Motion Cloud, Unreal, Unity, Blender, Maya, IIoudini, Cinema4D, iClone, Motion Builder, Pixotope, 3Ds Max, Notch, Fabric Enginc	MVN Animate, Motion Cloud, Unreal, Unity, Blender, Maya, IIoudini, Cinema4D, iClone, Motion Builder, Pixotope, 3Ds Max, Notch, Fabric Enginc
Price	$570.93*	$2495.00	$4,218.89	$7845.44	$13980.35

Table 2.1: Proposed Motion Capture Suit Against Suits in the Market

Figure 4.2: Demonstrating The 3D Tracking Performance of The Inertial Motion Capture Unit

Chapter 6: Discussion and Conclusion

In this paper we proposed the design of a cheap motion capture unit to be used for animation. It achieves the aim of recording close-to-accurate motion capture using low-cost sensors and microcontroller. It has good enough refresh rate for traditional animation which is shot in 24fps. The system is designed to be worn by straps on the head and limbs. Whilst the chest build is placed in a vest with the power bank protected in an encasing attached to a custom-made belt.

Although the suit, is cheap, energy efficient and has a good measurement quality, there are still shortcomings that need to be addressed in future works. The best motion capture suits are their ones that are unobtrusive to the extent that the user barely feels their presence. Ethernet cables are thick and relatively heavy and could get in the way when users perform complex motions. Still keeping the idea of a wired internal system, the ethernet cables could be replaced with lighter wires that are still capable of detaching easily. Another shortcoming is that the calibration system could be worked on later in the future as it relies on the user having the patience to be in a standing pose for a few seconds. The final reading during calibration is the final quaternion bias that will be used in the entirety of the motion capture. This means that if the user loses concentration at the final second of the calibration period, the entire calibration process must start again. Perhaps, an average of all the quaternion readings during calibration should be taken into consideration. But extra computation will also increase the execution time which is not something we want. Therefore, establishing the balance between accuracy and speed will be necessary in the future works to come.

References

[1] Sturman, D. J. (1999, March 13). A brief history of motion capture for computer character animation. Retrieved April 01, 2021, from http://www6.uniovi.es/hypgraph/animation/character_animation/motion_capture/history1.htm.

[2] Lum, R. (2019, January 09). A Brief History of Motion Tracking Technology and How it is Used Today. Retrieved October 06, 2020, from https://medium.com/@lumrachele/a-brief-history-of-motion-tracking-technology-and-how-it-is-used-today-44923087ef4c.

[3] What is motion capture? What can I use motion capture for? (2020, April 23). Retrieved April 01, 2021, from https://www.vicon.com/about-us/what-is-motion-capture/.

[4] The term IMU stands for “Inertial Measurement Unit. (2020, March 06). What is IMU? Inertial measurement Unit Working: Arrow.com. Retrieved April 14, 2021, from https://www.arrow.com/en/research-and-events/articles/imu-principles-and-applications.

[5] Blue Trident IMU: Inertial sensor by Vicon: BIOMECHANIC TRACKING. (2020, November 25). Retrieved April 14, 2021, from https://www.vicon.com/hardware/blue-trident/.

[6] Mvn animate. (n.d.). Retrieved April 15, 2021, from https://www.xsens.com/products/mvn-animate.

[7] Quality motion capture in one simple suit. (n.d.). Retrieved April 15, 2021, from https://www.rokoko.com/products/smartsuit-pro.

[8] S. Salehi, G. Bleser, N. Schmitz and D. Stricker, "A Low-Cost and Light-Weight Motion Tracking Suit," 2013 IEEE 10th International Conference on Ubiquitous Intelligence

and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing, Vietri sul Mare, Italy, 2013, pp. 474-479, doi: 10.1109/UIC-ATC.2013.22.

[9] F. Destelle et al., "Low-cost accurate skeleton tracking based on fusion of kinect and wearable inertial sensors," 2014 22nd European Signal Processing Conference

[10] R. Liu, L. Peng, L. Tong, K. Yang and B. Liu, "The Design of Wearable Wireless Inertial Measurement Unit for Body motion Capture System," 2018 IEEE International Conference on Intelligence and Safety for Robotics (ISR), Shenyang, China, 2018, pp. 557-562, doi: 10.1109/IISR.2018.8535742.

[11] S. R. Kadam and S. N. Pawar, "Development of Cost-Effective Motion Capture System based on Arduino," 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2020, pp. 1-6, doi: 10.1109/ICCMC48092.2020.ICCMC-0001.

[12] Raghavendra, P., Sachin, M., Srinivas, P. S., & Talasila, V. (2017). Design and development of a real-time, low-cost imu based human motion capture system. Lecture Notes in Networks and Systems, 155-165. doi:10.1007/978-981-10-3935-5_17

[13] T. Taunyazov, B. Omarali and A. Shintemirov, "A novel low-cost 4-DOF wireless human arm motion tracker," 2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob), 2016, pp. 157-162, doi: 10.1109/BIOROB.2016.7523615.

[14] Global, B. (2021, April 13). BNO055. Retrieved April 15, 2021, from https://www.bosch-sensortec.com/products/smart-sensors/bno055/.

[15] Z. Lin, Y. Xiong, H. Dai and X. Xia, "An Experimental Performance Evaluation of the Orientation Accuracy of Four Nine-Axis MEMS Motion Sensors," 2017 5th

International Conference on Enterprise Systems (ES), Beijing, China, 2017, pp. 185-189, doi: 10.1109/ES.2017.37.

[16] Teja, R. (2021, March 08). Introduction to ESP32: Specifications, ESP32 DEVKIT Board, Layout. Retrieved April 15, 2021, from https://www.electronicshub.org/getting-started-with-esp32/.

[17] CH robotics. (n.d.). Retrieved April 18, 2021, from http://www.chrobotics.com/library/understanding-euler-angles.

[18] Kuipers, J. B. (2007). Quaternions and rotation sequences: A primer with applications to orbits, aerospace, and virtual reality. Princeton, NJ: Princeton University Press.

[19] Townsend, K. (n.d.). Adafruit bno055 Absolute orientation sensor. Retrieved April 19, 2021, from https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/device-calibration.

[20] Zhang, F. (1998, May 19). Quaternions and matrices of quaternions. Retrieved April 18, 2021, from https://doi.org/10.1016/0024-3795(95)00543-9.