Extending user interactions in virtual, augmented and mixed reality

1st Juan Rodrigo Ponce Flores
Student of mechatronics engineering
Instituto Tecnologico y de Estudios Superiores de Monterrey
Queretaro, Mexico
juanro.pf@gmail.com

***Abstract*—This document is the description of the process of developing a device programmable with the arduino IDE, able to communicate with an android based VR headset and send the information about the rotation of the arm and with that data be able to determine the location and position of the whole upper limb.**

***Index Terms*—VR, AR, MR, connectivity, upper limb, IMU, hand tracking, 3D printing, arduino, nodeMcu, ESP8266, Wi-fi, bluetooth, unity, kinematics**

I. Introduction

Currently, virtual, augment and mixed reality (VR, AR, and MR respectively) are gaining momentum as engaging technologies that can positively impact learning and skills development in simulated scenarios otherwise impossible, difficult, or impractical to reproduce in real life. The establishment and interest on VR/AR/MR is the result of a growing investment and development in the entertainment industry (led by the video gaming industry) that is providing consumer-level devices and VR-ready computing hardware. The explosion of VR/AR/MR has produced a wide range of immersive and interactive devices that are being creatively and innovatively used in scenarios other than entertainment. However, interactions while wearing a head-mounted display are limited to compatible controllers that can lack adequate body/gestures or other forms of non-verbal communications tracking to create more realistic and meaningful interactions. Current interactive approaches employ and rely on wireless controllers that restrict the inputs when developing serious games, exergames and simulation experiences where psychomotor skills are associated with specific equipment that cannot be mapped to a game pad or joystick.

To increase the range of possible applications for VR/AR/MR, a device that is capable of transmitting information from outside the experience into it would be really useful. For example, applications in which you need to know the position of your arms, follow the motion of your legs, press something and know how hard you are pressing it or how much pressure you are applying, etc. This kind of application could be used in different areas, it could be medical, for learning purposes, for training, etc.

This paper aims to develop and use a wearable system to provide the VR/AR/MR experiences with an input system fed with the position of the users hand, different from the existing ones and compatible with as much VR/AR/MR headsets as possible.

To fulfill this objective we came up with 2 main ideas, the first one is to develop a wearable system that consists in a bracelet with a specific geometry that is going to be followed always by a laser sensor that measures distance, knowing the distance and the angle in which the sensor is looking at, we are going to be able to determine the position of the hand. The main advantage of this idea is that the wearable system would be really comfortable and easy to put on for the user. The main disadvantage is that the laser sensor and the rotating system that would be rotating it has to be placed in the headset, what would make the headset heavier and will probably unbalance it.

The second idea consists on an elbow brace that will have attached 2 IMUs and a last IMU on some kind of chest band, this way we will be able to know the angle of rotation of the arm and chest, knowing this data and the length of the user's arms, chest and other distances we will be able to determine the position of the user's hand. The main advantage of this idea is that with this wearable we are able to know the position of the whole arm, instead of just the hand o wrist. The main disadvantage of this one is that is not as easy to put on as the first one and not as comfortable as well.

II. Literature review

VR (Virtual Reality) is a computer-generated simulation of a realistic experience. Typically VR blocks out the real world (Reality) and replaces it with a Virtual world [1].

AR (Augmented Reality) is a similar concept, but instead of blocking out reality, the computer-generated content is added onto, or embedded into, the real world experience, so that both can be experienced together [1].

MR (Mixed Reality) is like a combination of the two mentioned before, pretty similar to the augmented reality, but with the difference that in mixed reality, objects of the real world can also interact with the virtual objects or the holograms.

Some of the VR/AR/MR headsets include a controller or controllers to make easier or better the experience, some of them are really comfortable and easy to use for the user, some other are not [2]. Or for some MR glasses like the Microsoft Hololens [3], instead of a controller, the user has to do some hand gestures with the hand in front of the glasses to interact with the experience, so the controllers are optional; it comes with a clicker thou, which helps the user with a gesture, so that he doesn't have to do the gesture, this is because for applications or occasions when the user has to be interacting a lot with the experience, he could get tired of maintaining the arm in a position so that he hand stands in front of the glasses.

Some inconveniences of the controllers are that they are exclusive for the headset they were designed for. The main inconvenience that we found is that the controllers were designed for doing a lot of functions, but when it comes to an specific task, these controllers could be limited or even useless.

A lot of people have been working on different projects related with the main topic of this paper, some people have used IMMU's and cameras to determine the position and movement of the upper limb while swimming [4], other have used a Xbox One Kinect and a Myo armband to analyze the motion of the upper limb [5], other people have used 9 IMU's and cameras to track the motion of the upper and lower limb [6], some other example that we saw that was very similar to what we want to do is a wearable system consisting on an IMU and a flex sensor connected to and arduino and sending the data somehow to a game created in unity [7], something really interesting about this last one is that they just used 1 IMU, this is because the figured out that if you already know the orientation of the upper arm, the forearm can only move in 1 axis, that why they used a flex sensor, just to know the orientation of that axis, that information is enough to determine the position of the complete arm.

III. Methods

A. Laser sensor

This method consists on the laser sensor VL53L0X [8], which is a module that provides accurate distance measurement up to 2 meters. The huge advantage of this laser sensor is that it is really small, the size of the module is 2.4 in. by 4.4 in. The sensor is going to be pasted into a servomotor [9] SG90, which at the same time is going to be pasted to another servomotor SG90, to be able to follow a bracelet that the user will be wearing. The advantages of these servomotors are that they are light, weighing just 14.7 g. each and that can rotate 90 degrees on each side. These are going to be attached to one side of the headset, in a position so that they are able to follow the bracelet. The bracelet is going to be 3D printed with a special geometry, so that the sensor can know in which side it should move. The mechanism is going to work with an NodeMcu [10], in which we are going to upload a code so that the laser sensor knows what to do. The program is going to make the sensor detect abrupt changes in the distance, which means it has lost the bracelet and it has to start rotating looking for it again; the geometry of the bracelet is going to help the laser determine if it should rotate left, right, up or down, and as soon as it detects a similar measurement as the last one before losing the bracelet, that means that the bracelet has been found again. All the data is going to be sent via Wi-Fi, using the NodeMcu, which is a programmable circuit board that has also and ESP8266 Wi-Fi module. Having the distance and the angle of both of the servomotor, we are going to be able to determine the position of the hand, respect to the headset.

B. IMUs

This method consists on two MPU6050 sensors attached to an elbow brace, with these sensors we can know the rotation of themselves, one sensor is going to be placed on the upper arm and the other one on the forearm, this way we can know the rotation of both parts of the arm. Also, a 3rd device able to measure rotation is going to be placed on the chest of the user, so that we know the rotation of the trunk. Knowing the rotation of these 3 parts of the body, we can calculate the position of the hand and of the complete arm using inverse kinematics. As on the other method, the NodeMcu is going to be used to get the data and send it to the headset using Wi-Fi.

IV. Development

A. Laser sensor

After doing some testing with the first method, we realized that the rotation speed of the servomotors is not as fast as we need them to follow the typical human motion. So, if the user makes a quick movement, the laser sensor will be lost and unable to find the bracelet again. To solve this problem, a potential solution would be to change the servomotors for faster ones.

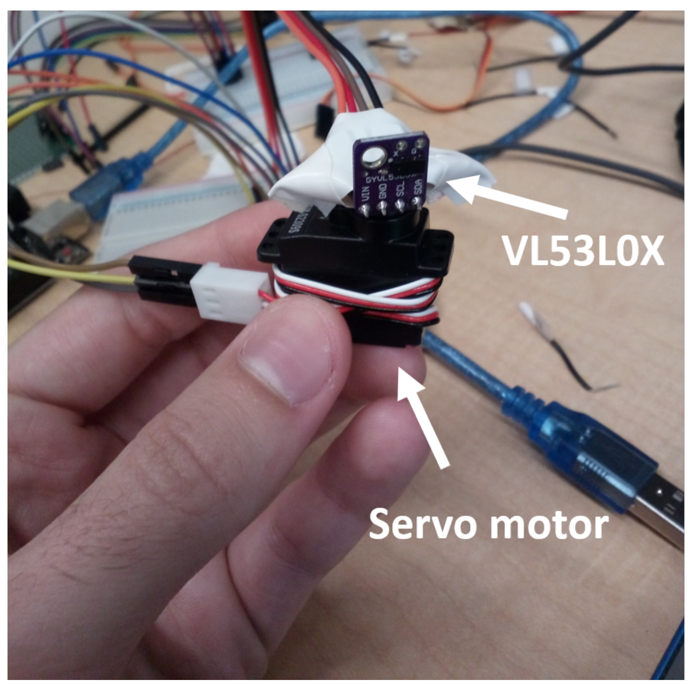


Fig. 1. Elbow brace.

B. IMUs

To implement the second method, first we had to test the IMUs and see if those where going to work, to do this we took an example and 2 libraries developed by jrowberg, we modified a little the code of one of the example codes to connect 2 IMUs instead of just 1, which wasn't that hard thanks to the good documentation he provides in his code, the code provides us with the quaternions of each IMU, after this we made the connection between Unity and Arduino using Wi-Fi, to do this, both devices have to be connected to the same network, after observing that both things worked separately, we combined them, assigning the values obtained by the Arduino to the objects of the Unity application.

To test the application we had available a Lenovo Mirage Solo, which is an android based Virtual Reality headset that has included a controller that has some buttons and also has inside some IMUs, which we used as the 3rd. rotation sensor required for this method, the one that goes on the chest of the user.

To do the physical prototype we used an elbow brace to which we sew the cases of the Arduino (which was placed on a tiny breadboard) and the IMUs, which we 3D printed after doing the designs according to the specifics measurements of the devices.

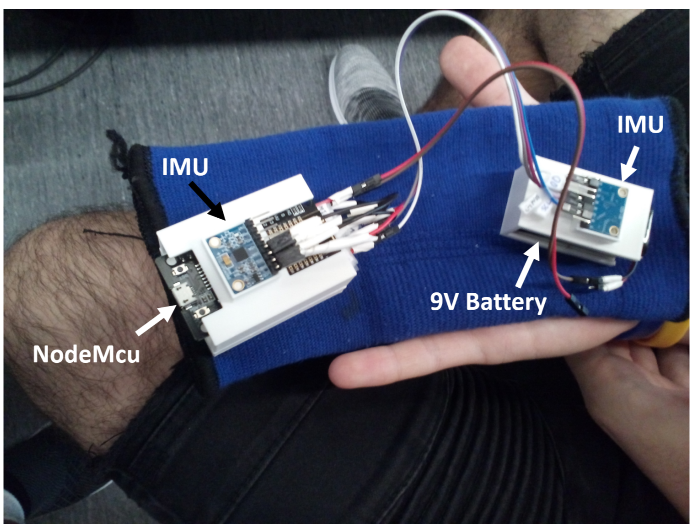


Fig. 2. Elbow brace.

For the controller we designed an adaptable chest band, to which we sew as well the controller case that we designed and 3D printed.

C. Complications

Before testing with the Lenovo Mirage Solo we tried to develop the application for the Microsoft Hololens, which are a MR smart glasses. The main reason of choosing the Hololens is because the only way to interact with these smartglasses is with hand gestures, and in order to interact with them, the user has to put his hand in front of the smartglasses and perform the gesture, which after doing it for a while, the user

Fig. 3. Chest band.

gets tired of the position. The smartglasses have included a clicker that allows the user to do click and drag task without putting the hand in front of the smartglasses and performing the gesture, this device is really useful and partially fixes the problem mentioned before. But in order to do the experience more comfortable and realistic to the user, it will be useful to have a device that allows the headset to know or determine the position of the hand without having to look at it. One clear example is if you are in a simulation and want to grab a virtual object that is not in your visual field, you have to look at the object and the perform the click gesture or push the button of the clicker to grab it, whilst in real life, if you know the location of the object, you can just grab it, without having to look at it before.

The idea was to use the same wearable that we developed with the Hololens, the problem was the limitations with the connectivity, unfortunately the ”shared experience” section of the documentation of these smartglasses was not available when we were testing and the methods to connect with other devices that used to work before, don't work anymore because of the updated libraries and APIs, that is why we had to desert with the Hololens and work with an Android based headset.

V. Results

After 3 months of working in the project, we developed the wearable system, it works completely wireless, connecting the NodeMcu to a 9 volts battery. Also in Unity we created a simple app in which the user can see a representation of his upper limb that replicates the movements of his own.

VI. Conclusions

The connection between an android based VR headset and the NodeMcu was a complete success, sending data between

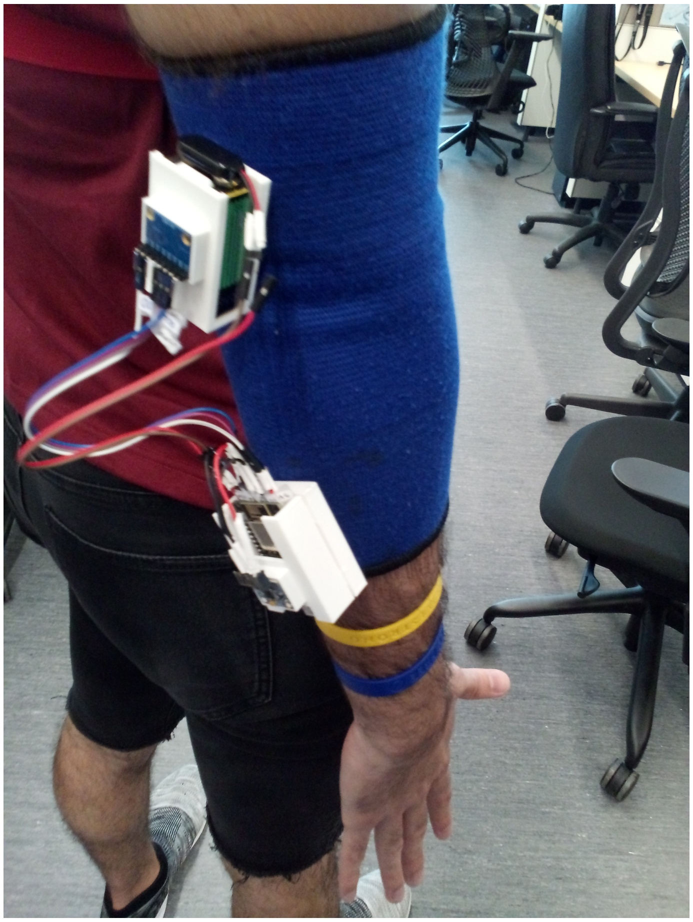


Fig. 4. Elbow brace placed.

Fig. 6. Back view.

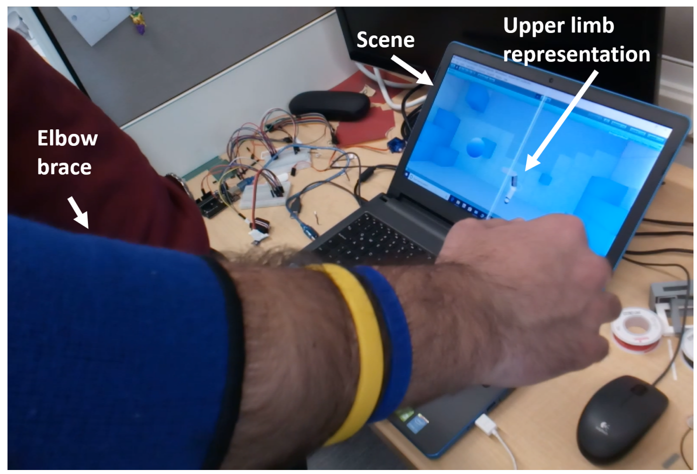


Fig. 7. Application view.

Fig. 5. Front view.

this two devices via Wi-Fi is really useful for the development of different controllers for the headsets. Mirroring the movement of the user is just one simple application that can be given to the device, but all kind of pick and place tasks, dancing applications, or any app where you need to know the position of the upper limb can be developed and used for the device.

VII. Future work

For future work on the same area we can start with putting some LED's on the device that turn on or of depending on the status of the device, sometimes, the IMUs doesn't initialize correctly and that causes the device to not be able to connect to the headset, some other times there is a problem only with the Wi-fi connection, so putting 2 or 3 LED's that indicate which is the problem would make it easier and faster to solve it; most of the time the problem is solved with just resetting the NodeMcu, but depending on the problem sometimes you need to disconnect one of the IMU's, or both and connect them again.

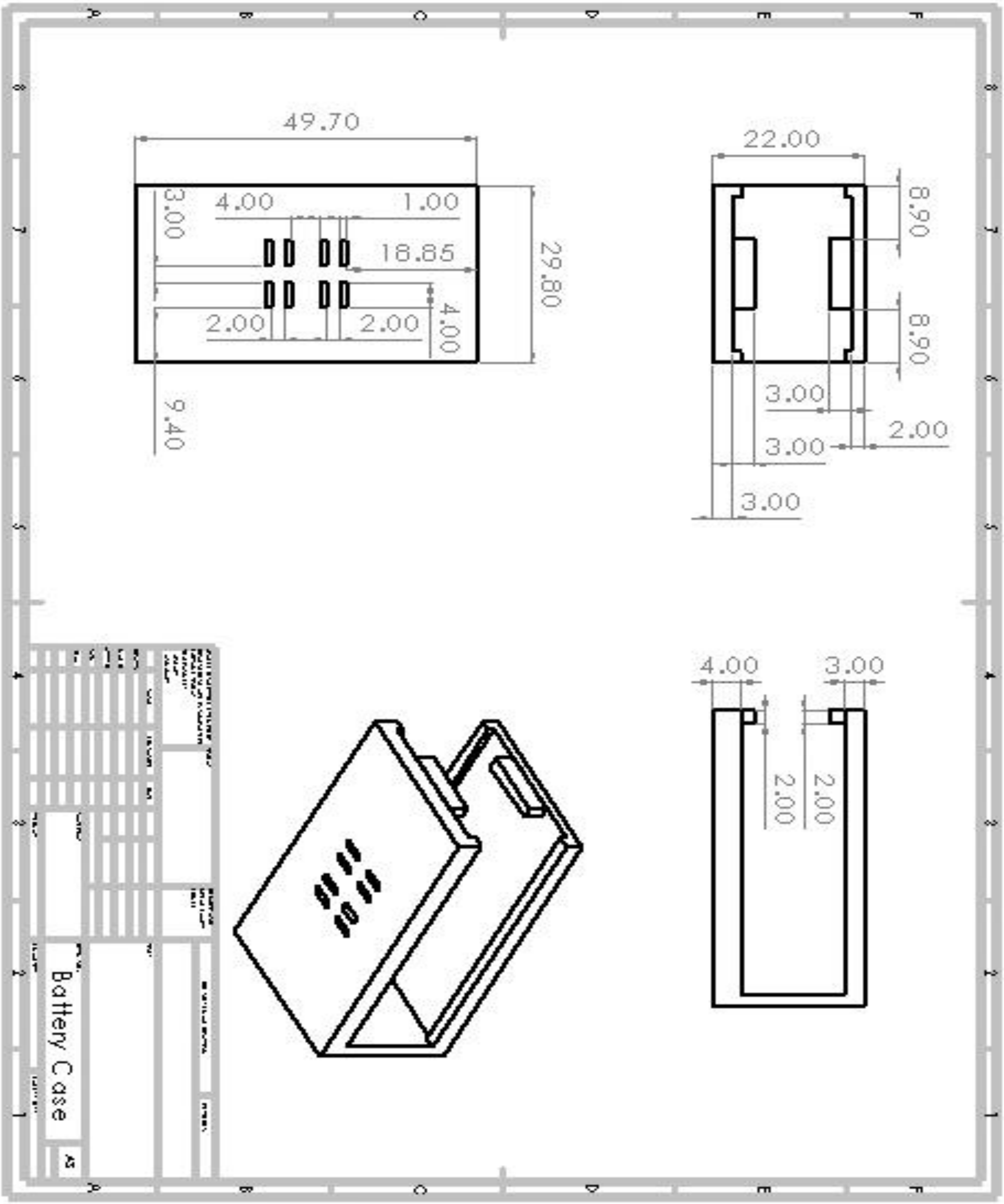
Other idea is to put some buzzer or vibration mini motor on the elbow brace, so the app could send feedback to the user, maybe if if in the application you need to pick and place something, the elbow brace could vibrate when you are touching the object or when you are on the location you need to place it; it would be useful for some applications.

The improvement of the app and the calibration of the IMU's is another thing we would like to work in; because of the time we had to do this project and the obstacles we found along the road, we didn't have time to do it perfect, right now the calibration of the IMU's is not the best and also the app doesn't mirror the movement of the upper limb as smooth as we would like to.

The integration of more IMU's on the elbow brace would give us more precise information about the rotation of the arm, for example, the controller attached to the chest band has inside 9 IMU's, which makes it very precise with it's information.

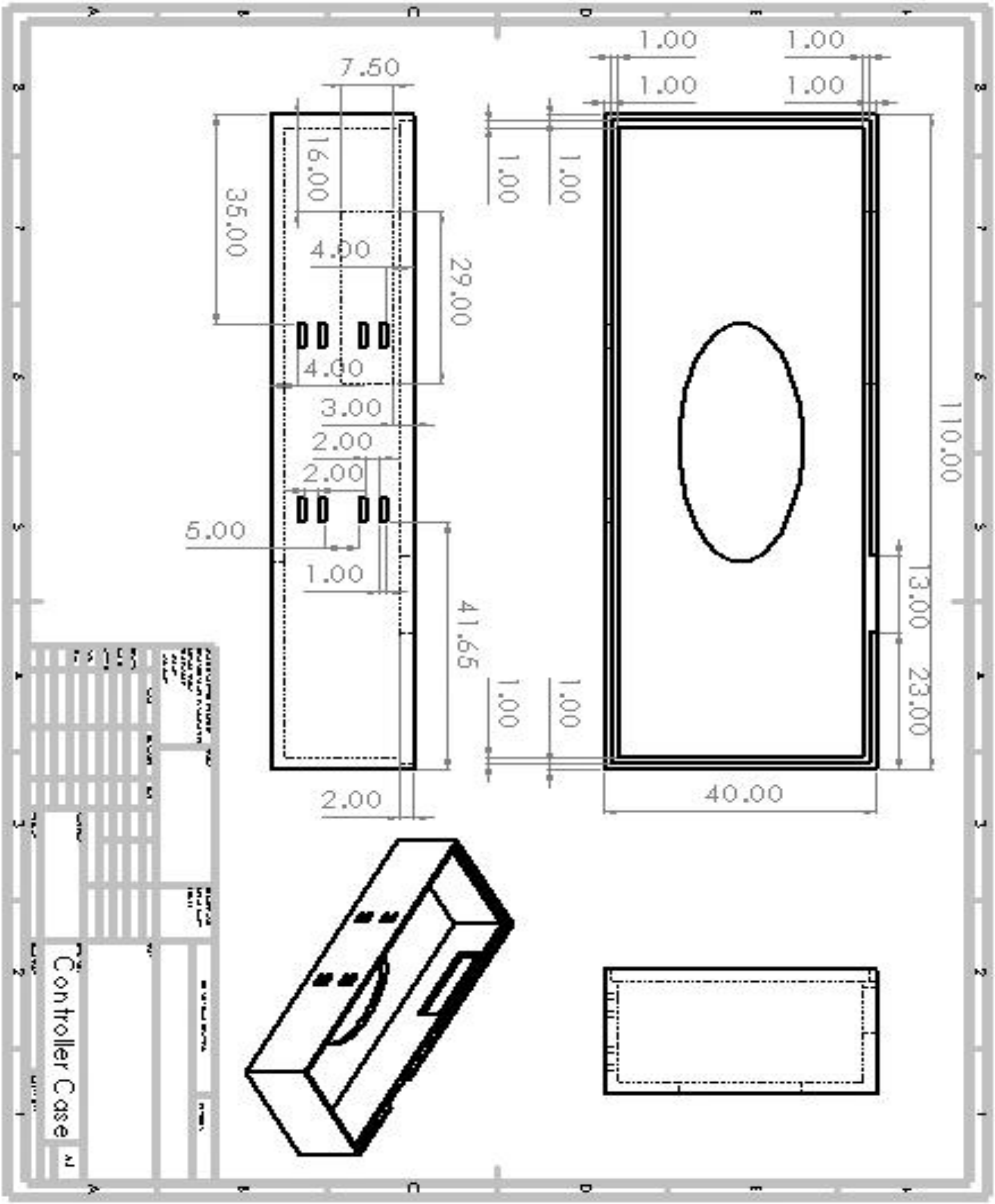
Also, for some future applications, we don't wanna stay just on receiving data of the upper limb, with the Wi-fi connectivity between the devices working, the applications are a lot, for example for walking in a VR experience there could be a device that the user could use in which the movement of his legs or feet determine whether he is walking or not. Another example could be with a pressure sensor determine how hard the user is pressing something in the experience. Another one could be an application that gets the user's heart rate using a different sensor, this one could be useful in apps were the user might get tired.

APPENDIX A



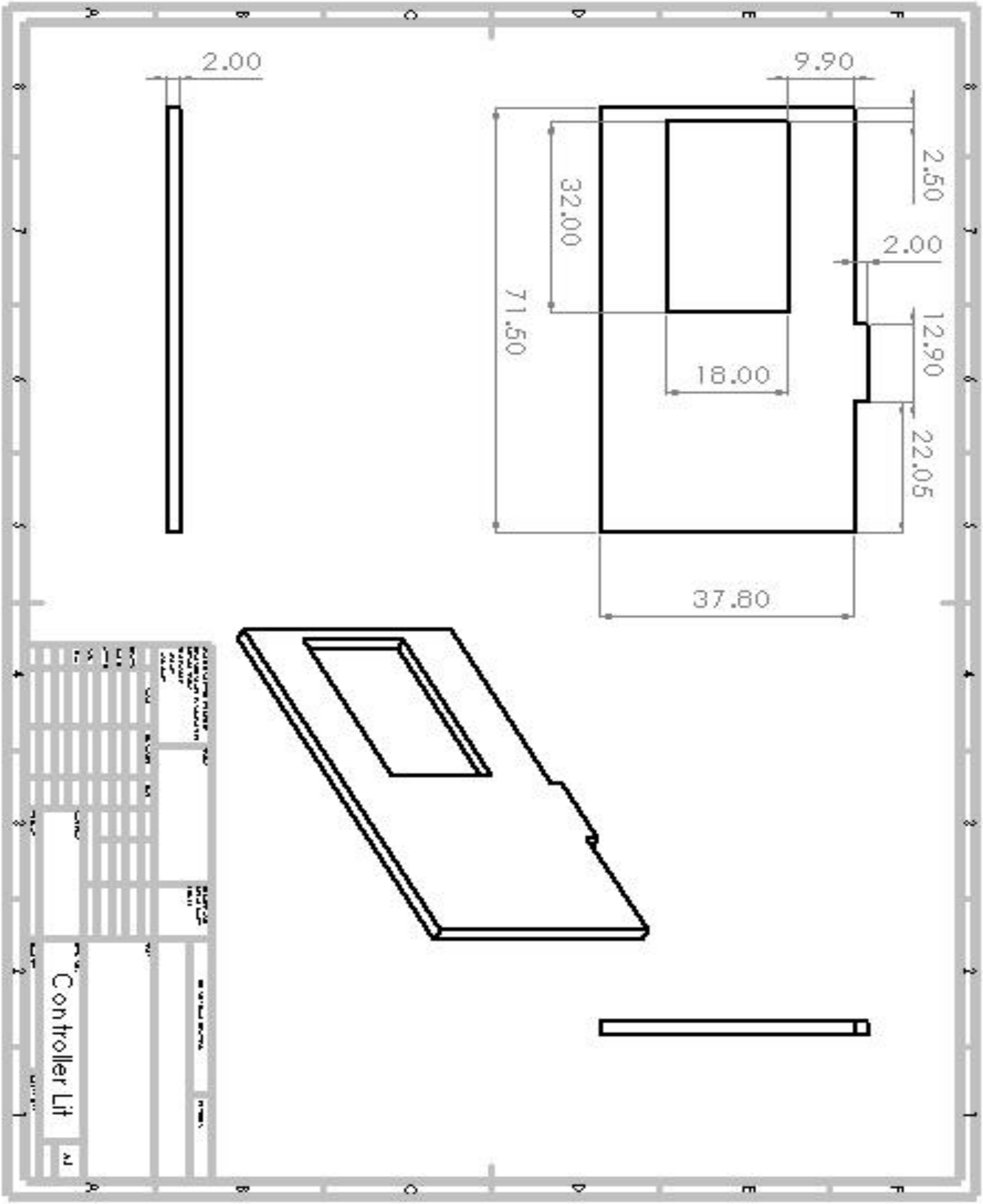
49.70
4.00
1.00
3.00
18.85
29.80
2.00
2.00
4.00
9.40
22.00
8.90
8.90
3.00
2.00
3.00
3.00
4.00
3.00
2.00
2.00
Battery Case

APPENDIX B

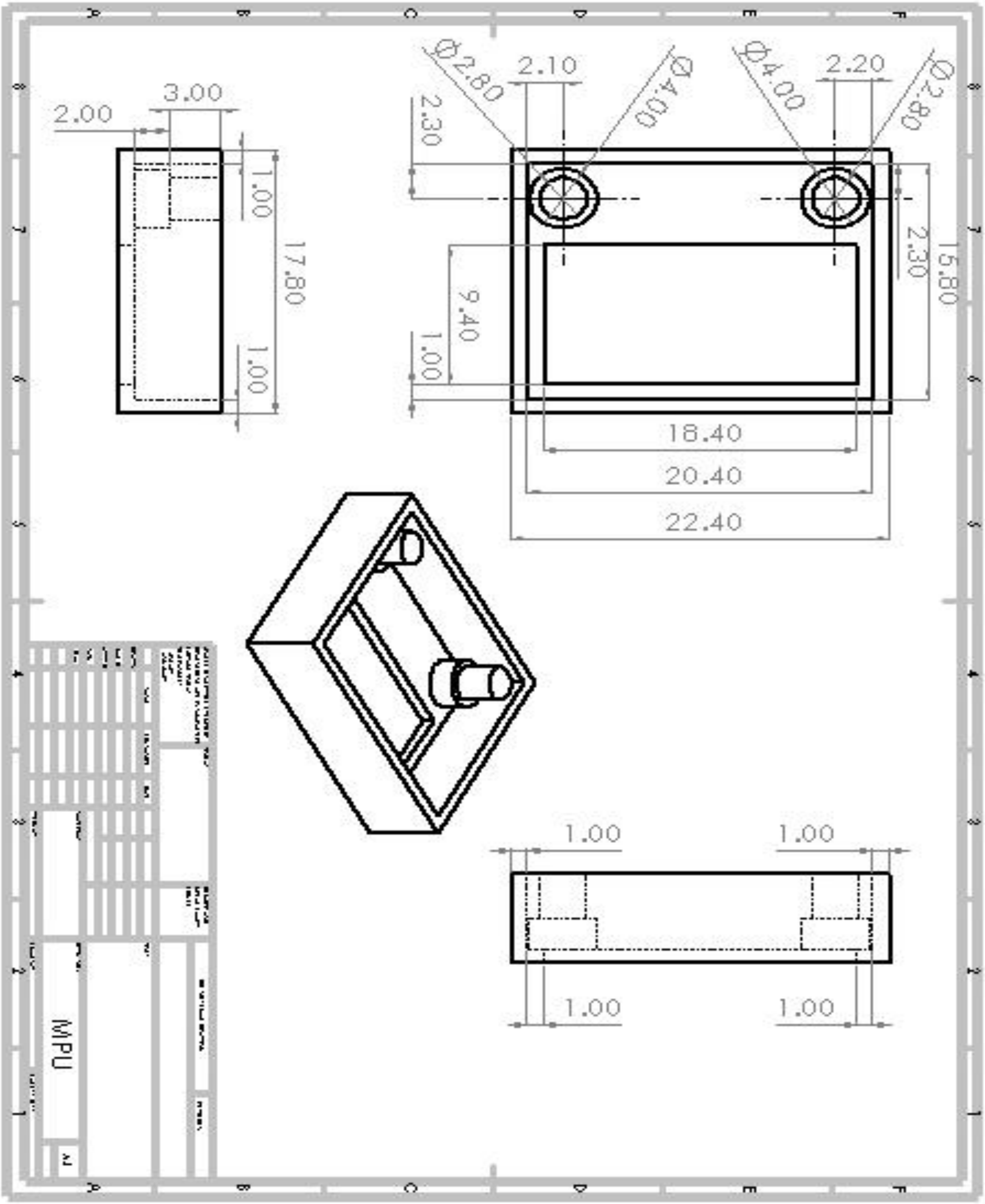


7.50
16.00
35.00
4.00
29.00
4.00
3.00
2.00
2.00
5.00
1.00
41.65
2.00
1.00
1.00
1.00
1.00
1.00
1.00
1.00
1.00
1.00
1.00
110.00
13.00
23.00
40.00
Controller Case

APPENDIX C

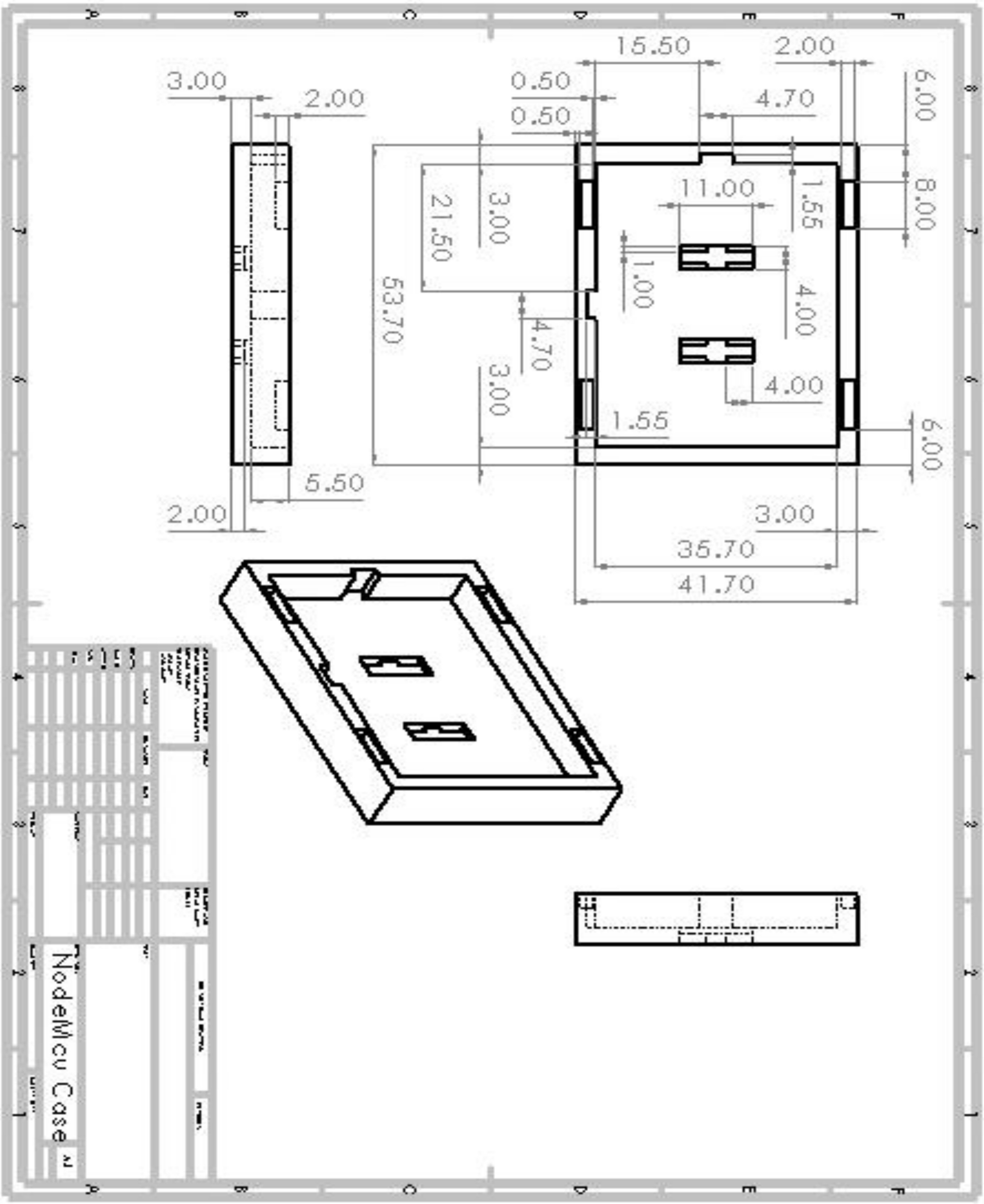


APPENDIX D

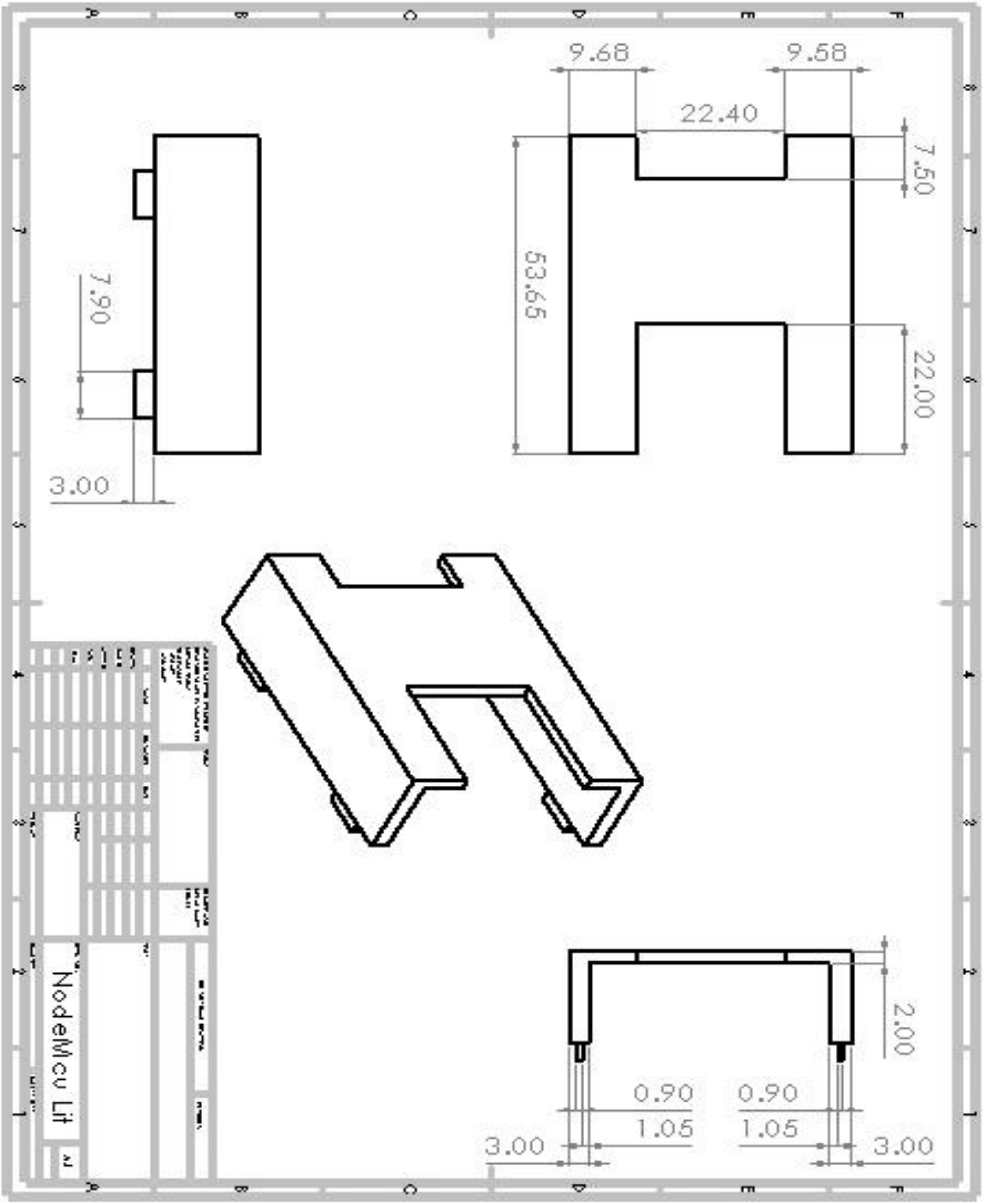


2.00
3.00
1.00
17.80
1.00
Ø2.80
2.10
Ø4.00
Ø4.00
2.20
Ø2.80
2.30
2.30
15.80
9.40
1.00
18.40
20.40
22.40
1.00
1.00
1.00
1.00
MPU

APPENDIX E


15.50
2.00
3.00
2.00
0.50
0.50
4.70
6.00
11.00
1.55
8.00
21.50
3.00
53.70
1.00
4.00
4.70
3.00
4.00
1.55
6.00
5.50
2.00
3.00
35.70
41.70
NodeMcu Case

APPENDIX F



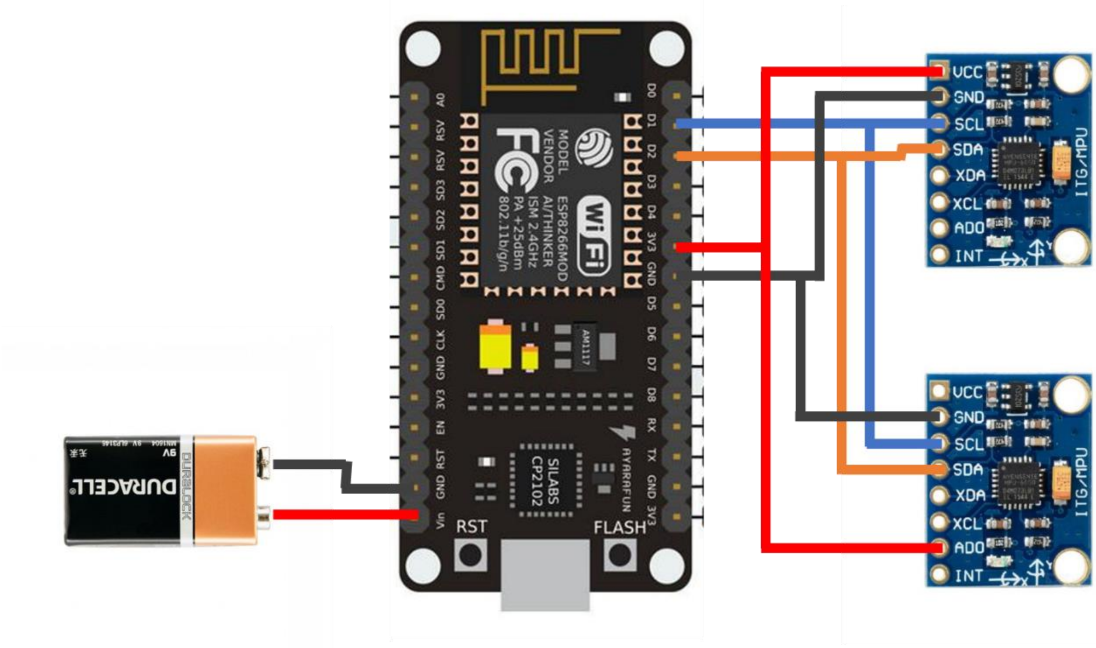
9.68
9.58
22.40
7.50
53.65
22.00
7.90
3.00
2.00
0.90
1.05
0.90
1.05
3.00
3.00
NodeMcu Lit

APPENDIX G

TUTORIAL

You need:

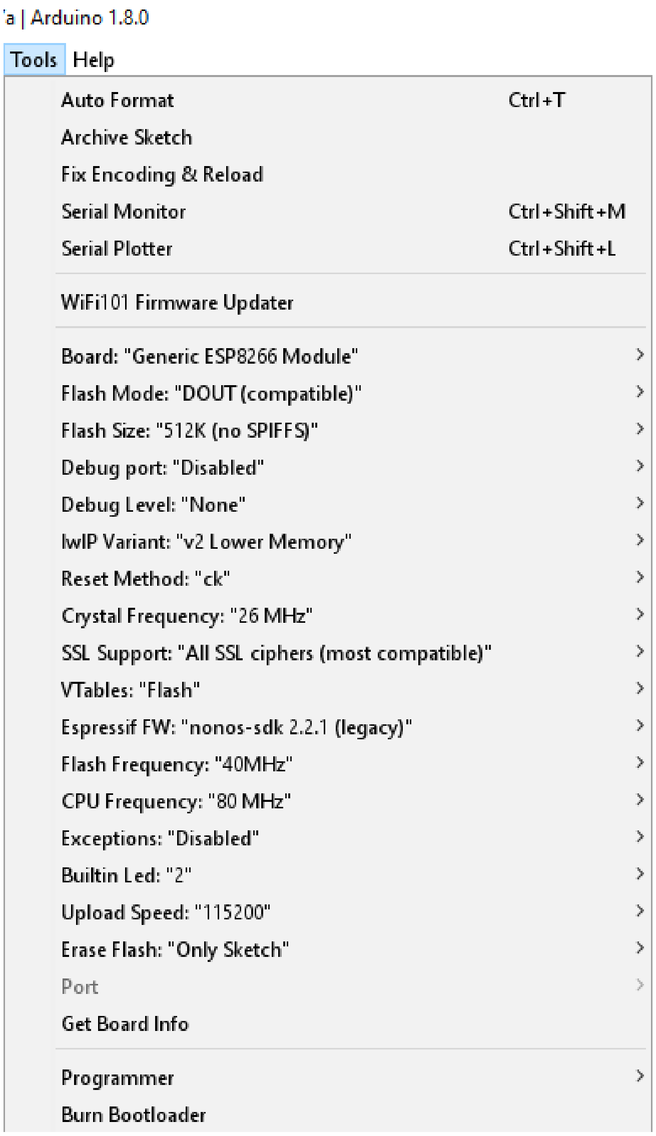
- 1 - NodeMcu (ESP8266)
- 2 - MPU6050
- 1 - 9V battery



Steps:

1. Download and install Arduino IDE (https://www.arduino.cc/en/main/software)
2. Follow the next tutorial to download and install the required libraries (https://www.youtube.com/watch?v=GjHxwDLBh00 min 1:00 to 2:16)
3. Follow the next instructions: https://developers.google.com/vr/develop/unity/get-started-android
4. Download Android Studio (https://developer.android.com/studio/?gclid=EAIaIQobChMIk77f1vvx4wIVmx-tBh34JQK4EAAYASAAEgJwMPD_BwE)
5. Open Android Studio and go to **configure** -> **SDK Manager** -> **SDK Platforms**
6. Then select the platform you will be working with (Android 8.0 if you are using Mirage SOLO), click apply and follow the installation instructions.
7. Follow the next tutorial (https://www.youtube.com/watch?v=NEo1WsT5T7s min 2:09 to 3:44)

8. In the Arduino IDE go to **tools** and check everything is like this:



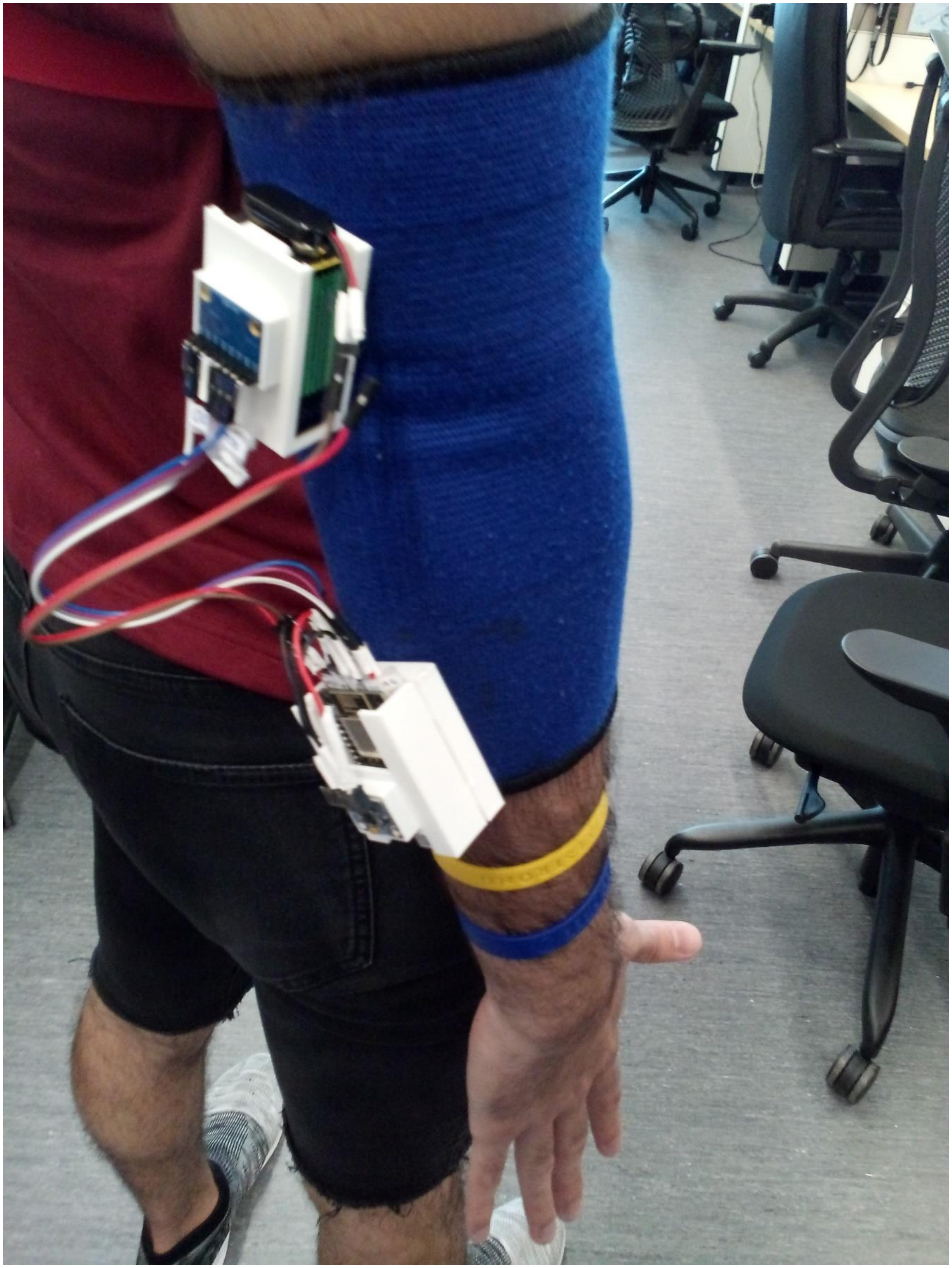
9. Connect the NodeMcu to the computer using the USB cable, then open the code: *"Multiple_MPU_Quaternions_WiFi.ino"* using the Arduino IDE.
10. Change the values of the variables ssid and password (lines 7 and 8) for your own data and then go to tools, select the port in which your Arduino is connected and upload the code to your NodeMcu.
11. After uploaded the code go to **tools**, open the serial monitor and change the baud to 115200, then press the RST button in the NodeMcu and wait until it sends you a message like this: "Use this URL to connect: http://192.168.43.208"
12. After that, go to unity and create a new project, then right click on **assets** -> **Import Package** -> **Custom Package** and select the *"Summer_Research.unitypackage".*
13. After importing everything in the package open the scene named "PrimeraPrueba", then on the Hierarchy click once on the CubeRoom, after that go to the Inspector

and change the value of the IP Address for the one that the serial monitor gave you (192.168.43.208 for example) and the Port should be 80.

14. Then connect your VR device to the computer, go to **File** -> **Build Settings** and make sure you followed correctly the step 3, if everything is as it should be (player settings and platform) you just have to change the Build System to Gradle, click on Add Open Scenes and click on Build And Run. Then the app will open automatically in your VR device.

To make sure the app runs as expected, first you must connect the NodeMcu (before opening the app in the VR device; if it is open, make sure to close it. In the case of the Mirage SOLO you must go to **settings** -> **apps**, then select the app and click on FORCE STOP), wait about 10 seconds and then open the app. Then put the controller inside its case and press the "O" button for 3 seconds.

Initial Position before connecting the NodeMcu (maintain this position until the app is running).



Initial Position before connecting the NodeMcu (maintain this position until the app is running).

Initial Position before connecting the NodeMcu (maintain this position until the app is running).

REFERENCES

[1] S. Mann, T. Furness, Y. Yuan, J. Iorio, and Z. Wang, "All reality: Virtual, augmented, mixed (x), mediated (x, y), and multimediated reality," *arXiv preprint arXiv:1804.08386*, 2018.

[2] I. Choi, E. Ofek, H. Benko, M. Sinclair, and C. Holz, "Claw: A multifunctional handheld haptic controller for grasping, touching, and triggering in virtual reality," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, p. 654, ACM, 2018.

[3] D. C. Ogdon, "Hololens and vive pro: Virtual reality headsets," *Journal of the Medical Library Association: JMLA*, vol. 107, no. 1, p. 118, 2019.

[4] S. Fantozzi, A. Giovanardi, F. A. Magalhães, R. Di Michele, M. Cortesi, and G. Gatta, "Assessment of three-dimensional joint kinematics of the upper limb during simulated swimming using wearable inertial-magnetic measurement units," *Journal of sports sciences*, vol. 34, no. 11, pp. 1073–1080, 2016.

[5] S. S. Esfahlani, B. Muresan, A. Sanaei, and G. Wilson, "Validity of the kinect and myo armband in a serious game for assessing upper limb movement," *Entertainment Computing*, vol. 27, pp. 150–156, 2018.

[6] A. Filippeschi, N. Schmitz, M. Miezal, G. Bleser, E. Ruffaldi, and D. Stricker, "Survey of motion tracking methods based on inertial sensors: A focus on upper limb human motion," *Sensors*, vol. 17, no. 6, p. 1257, 2017.

[7] K. Zhu and L. Shi, "Motion control in vrreal-time upper limb tracking via imu and flex sensor," *Report submitted to Stanford University*, pp. 1–5, 2016.

[8] N. Laković, M. Brkić, B. Batinić, J. Bajić, V. Rajs, and N. Kulundžić, "Application of low-cost vl53l0x tof sensor for robot environment detection," in *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, pp. 1–4, IEEE, 2019.

[9] N. Cameron, "Servo and stepper motors," in *Arduino Applied*, pp. 157–176, Springer, 2019.

[10] A. C. Bento, "Iot: Nodemcu 12e x arduino uno, results of an experimental and comparative survey," *International Journal*, vol. 6, no. 1, 2018.