Measuring and Tracking Arm Range of Motion With IMU's

Jalen Gumayagay, Department of Electrical Engineering

Mentor: Dr. Norali Pernalete, David Flores, Eric Tran

Kellogg Honors College Capstone Project: RSCA 2024

Introduction

Partial paralysis is a potential side effect from various injuries and diseases, from damage to the spinal cord to strokes. Part of a patient's recovery process from partial paralysis involves having a physical therapist measure the range of motion (ROM) of their affected limbs to see how much they can move. Typically, an instrument called a goniometer is used to make these measurements, and a physical therapist operates one by having the patient move their limb to a certain level and lining up the goniometer's two arms to measure the angle. For our project, we wanted to improve this process by creating an easy-to-use, easy-to-wear electronic module that can track the patient's ROM and display a simple three-dimensional version of their movement that allows the therapist and patient to monitor the patient's improvements over time. In doing so, we aimed to make the recovery process easier for both therapist and patient to help the patient recover in a faster and more effective manner.

Design Guidelines

1. Module is comfortable to wear
2. Module is simple to use and operate
3. Module is inexpensive and easy to replicate or mass produce
4. Module outputs numerical values describing changes in yaw, pitch, and roll values of patient's arm movements
5. Module tracks patient's arm movements and displays a simplified 3D version of the movements on a computer.

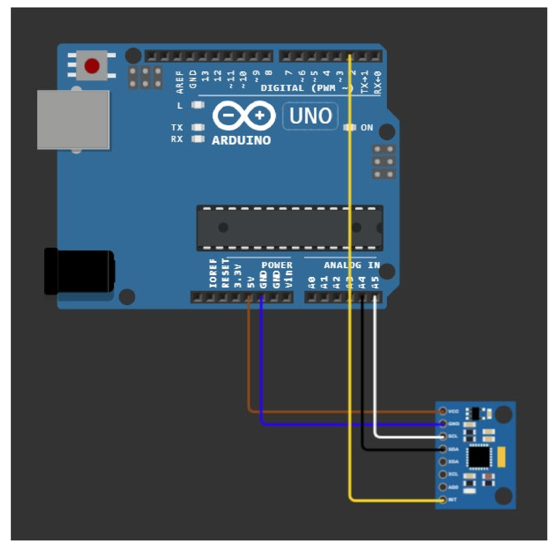
Components/Software

MPU-6050 Inertial Measurement Unit (IMU): Combination of accelerometer and gyroscope; functions as the main motion tracking sensor.

Arduino Uno: Microcontroller, communicates with the MPU-6050 and sends the data received from it to the computer.

Arduino IDE: Used for programming and uploading code to Arduino and printing out numerical changes in yaw, pitch, and roll.

Processor IDE: Used to display simplified 3D movement tracking of the arm.



Wire Color	Arduino Connection	MPU-6050 Connection
Blue	5V	VCC
Brown	GND	GND
Yellow	Digital 2	INT
Black	Analog 4	SDA
White	Analog 5	SCL

Figure 1: Main schematic

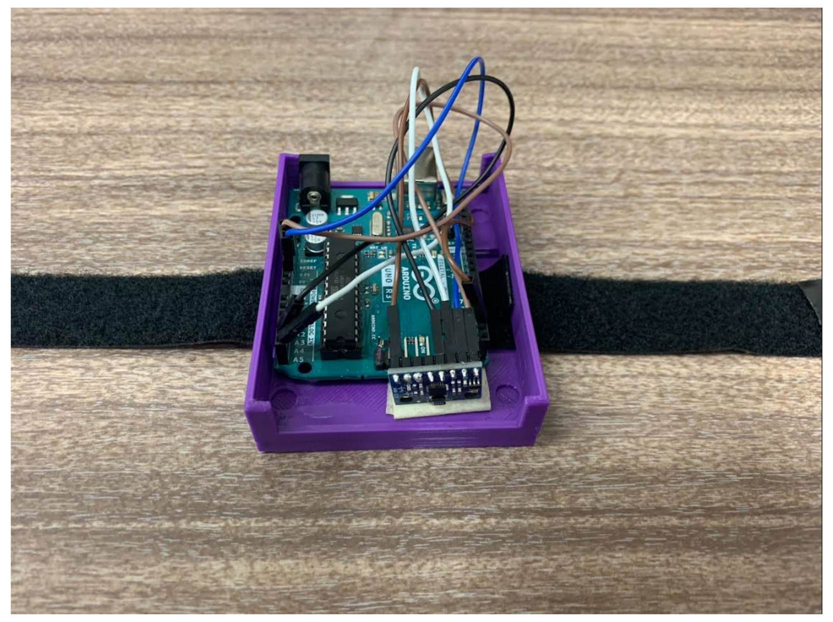


Figure 2: Constructed module

Procedure

Initial Research

Our module needed two main electronic components to implement our desired measuring and modeling capabilities. The first is a motion-tracking sensor that can detect the arm's movement and change in orientation. The second is a microcontroller that can communicate with the sensor and send the information it receives from it to a computer. I researched various sensors and microcontrollers and analyzed how they could work together to achieve this goal before choosing the MPU-6050 for the sensor and the Arduino Uno for the microcontroller.

The MPU-6050 is a three-axis accelerometer and gyroscope combination that suited our need for a sensor that can detect motion and changes in orientation in a three-dimensional space. It is also inexpensive and readily available, allowing us to stay within our budget and acquire more as back-ups.

The Uno is a microcontroller board which consists of an ATmega328P microcontroller chip and both digital and analog input and output pins. Like the MPU-6050, the Uno is popular and versatile, with an abundance of examples and libraries. The analog pins are also useful for establishing the I2C lines that enable communication and data exchange between the Uno and the MPU-6050. Additionally, the Uno can be powered by a USB connection and can in turn power the MPU-6050. Therefore, we can use the same computer we are using to program the Uno to power both it and the MPU-6050.

Design and Construction

We used Tinkercad to design and 3D print a hard plastic case for the electronics. This case is also what is fastened to the patient's arm using adjustable hook-and-loop fastener strips. The brown and blue wires provide power from the Arduino to the MPU-6050. The black and white wires set up the I2C communication lines and enable the devices to communicate with each other. The SDA pin (black wire) carries the data being transmitted between the two devices. The SCL pin (white wire) synchronizes the data transfer between the devices and ensures that the timing of the data transmission between them is in sync.

Programming/Uploading Code

Programming and uploading the code to the Arduino and the MPU-6050 proved to be the most challenging part of this project. We used Arduino libraries uploaded to GitHub by Jeff Rowberg as the basis of the code, but when we were trying to upload it, it consistently failed. We first suspected this to be caused by a faulty wire connecting the computer to the Arduino, but when we replaced it with another, the issue wasn't fixed. We then thought our MPU-6050 may have been faulty, but when we tried another one, that didn't resolve it either. The issue turned out to be caused by the jumper wires connecting the Arduino to the MPU-6050 not maintaining a consistent connection throughout the upload progress. To address this, we soldered the pins of the jumper wires to the MPU-6050, which provided a stable connection for the code to be uploaded.

Testing

To test our module, we operated it using the same procedure we would expect a physical therapist to if they were using it on an actual patient. We strapped the module to our team member's arm. The operator then connects the power cable to the Uno and uploads the code to the Uno using the Arduino IDE on the computer they are using. The Uno will be initialized and begin displaying the yaw, pitch, and roll values based on the data it is receiving from the MPU-6050 on the IDE's serial monitor. Afterwards, the operator can run a separate program using the Processor IDE. This separate program creates a simplified three-dimensional version of the patient's arm using basic shapes that tracks the movement of their arm.

Figure 3: Testing constructed module

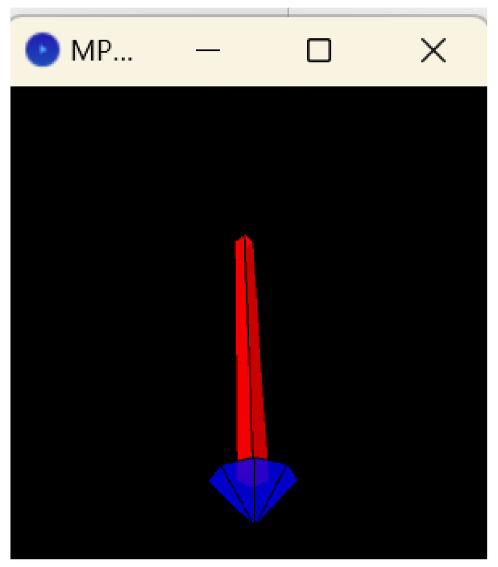


Figure 4: 3D representation of arm

Future Improvements

Having the yaw, pitch, and roll values is good, but they might not be very intuitive for the therapist and patient to understand. If there was a singular angular measurement the display outputted instead, this might be easier for the therapist and patient to read and make analyses. Additionally, our three-dimensional representation of the arm only consists of a single joint, meaning we can only represent motion from either the shoulder or the elbow, not both simultaneously. One possible way to incorporate this might be to include a second module with another Uno and MPU-6050 and to link the shapes of both in the Programmer IDE. Lastly, the measurements could suffer from gyroscope drift, which is a build-up of readings from the gyroscope that get progressively further away from the true values. Over time, this could cause the MPU-6050 to give inaccurate values for the yaw, pitch, and roll. This could be addressed by applying a Kalman filter, which combines measurements from the MPU-6050 with predictions of the MPU-6050's orientation to give an accurate estimation of its actual orientation. With some modifications, our project could theoretically work for tracking the ROM of a patient's leg, as well.

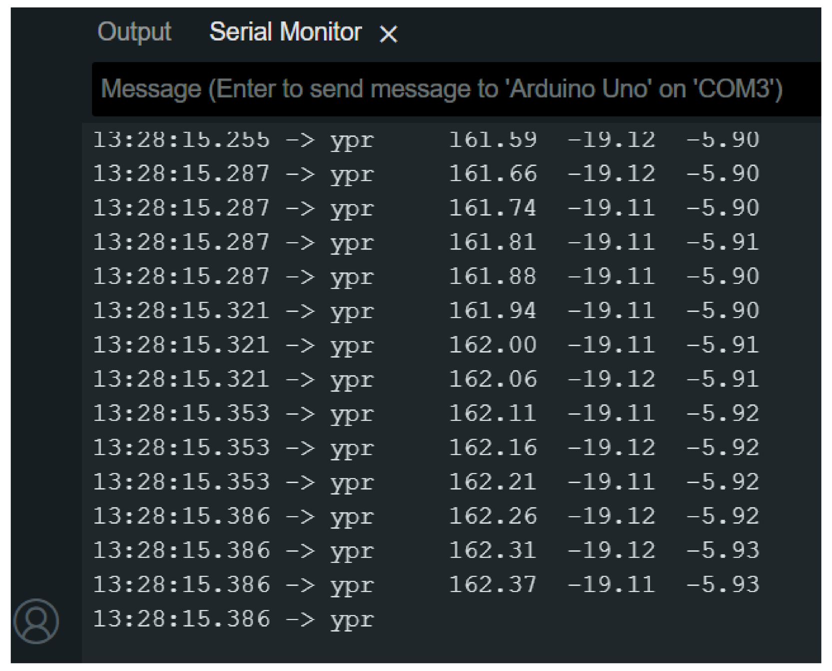


Figure 5: Yaw, pitch, and roll values measured by the MPU-6050

Conclusion and Acknowledgements

In conclusion, our team successfully designed a motion-tracking module that physical therapists could use to measure the ROM of a patient's arm and keep track of their progress throughout the recovery process. We designed it to be comfortable, relatively inexpensive, and easy to replicate or mass produce so if it were ever to be manufactured, it could help a greater number of people at lower costs. This also came with its drawbacks, such as slightly lower tracking accuracy and the module being subject to gyroscope drift. Although our module worked as intended, there is still plenty of room for improvement, such as filters to reduce errors and more modularity so the same module could also be used to track the ROM of a patient's leg, if needed. With more time and research, our project could potentially become a viable and inexpensive way for physical therapists to monitor their patients' ROM.

We would like to thank our Faculty Advisor, Dr. Norali Pernalete, for the valuable insight, guidance, and feedback she provided over the course of our work. We would also like to thank Jeff Rowberg, whose Arduino libraries on GitHub were of immense help in developing our module. We would also like to thank the College of Engineering and its faculty for providing us with this project opportunity. Finally, we thank the Kellogg Honors College, the Office of Government and External Affairs, Career Center, and Alumni Affairs for hosting this year's RSCA Conference and giving us this opportunity to share our work.

References

[1]B. Sears, "What is Range of Motion?," *Verywell Health*, Nov. 11, 2011. https://www.verywellhealth.com/overview-range-of-motion-2696650

[2]"10.8: What Does an Inertial Measurement Unit Measure?," *Engineering LibreTexts*, Apr. 25, 2021. https://eng.libretexts.org/Bookshelves/Mechanical_Engineering/System_Design_for_Uncertainty_(Hover_and_Triantafyllou)/10%3A_Vehicle_Inertial_Dynamics/10.8%3A_What_Does_an_Inertial_Measurement_Unit_Measure%3F (accessed Dec. 12, 2023).

[3]S. Harris, "Inertial Measurement Unit (IMU) - an introduction," *Advanced Navigation*, Feb. 12, 2023. https://www.advancednavigation.com/tech-articles/inertial-measurement-unit-imu-an-introduction/

[4]I. Beavers, "The Case of the Misguided Gyro | Analog Devices," *www.analog.com*, Mar. 2017. https://www.analog.com/en/analog-dialogue/raqs/raq-issue-139.html

[5]PricewaterhouseCoopers, "Behind the numbers 2019: Healthcare and medical cost trends: PwC," *PwC*, 2019. https://www.pwc.com/us/en/industries/health-industries/library/behind-the-numbers.html

[6]A. Montero, L. Hamel, M. Brodie, and A. Kearney, "Americans' challenges with health care costs," *Kaiser Family Foundation*, Jul. 14, 2022. https://www.kff.org/health-costs/issue-brief/americans-challenges-with-health-care-costs/

[7]"i2cdevlib/Arduino at master · jrowberg/i2cdevlib," *GitHub*. https://github.com/jrowberg/i2cdevlib/tree/master/Arduino (accessed Dec. 13, 2023).