RF-based Long-range Motion Recognition and Communication System

Spring 2023

Group 45

Joe Luo (luo42)

Zekai Zhang (zekaiz2)

James Tian (zeyut2)

I. Introduction
1. Problem
2. Solution
3. Visual Aid
4. High-level requirements list

II. Design
1. Block diagram
2. Subsystem Overview
3. Subsystem Requirements
4. Tolerance Analysis

III. Cost and Schedule

IV. Ethics and Safety

V. References

Introduction

Problem

While society accelerates into the digital age, an avid demand for more aggressively intimate ways of communication rises especially as the world welcomes a post-covid traumatic recovery. We witness the emergence of many novel products, albeit with mixed reception, that embrace this new concept, such as VR games, Metaverse, and holographic projection. It’s apparent that in modern days we crave information that goes beyond texts, videos, and sounds, but something mobile, three-dimensional, and interactive – for instance, transferring and reproducing motion across a long distance.

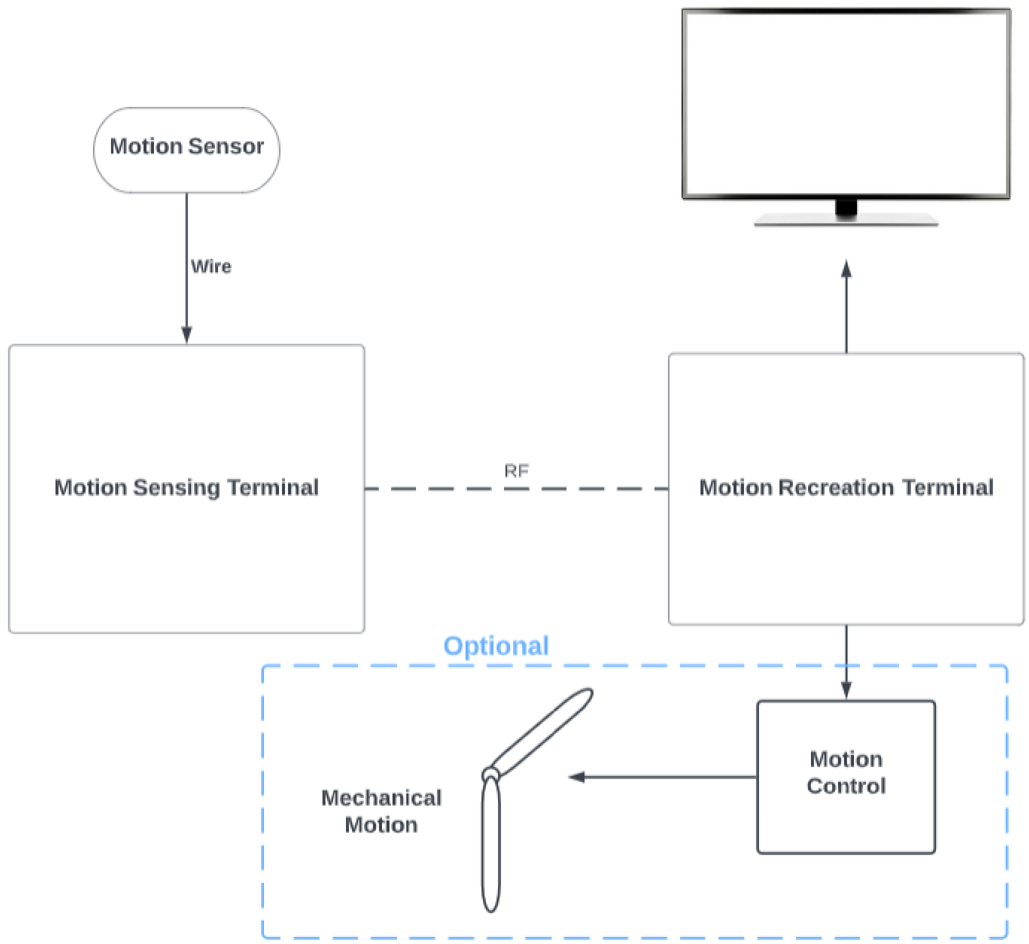
Aside from peer-to-peer communication, a long-range motion communication system can be useful in a variety of scenarios. In a classroom setting, whenever a Physics teacher wants to dig further into a relatively abstract concept, like lattice structure and electron concentration in materials, board and chalk and other variants are their only reliable helpers. However, it would be more engaging for both the lecturer and the learner to see a 3D presentation of the topic in question that's able to move and change at our commands. Likewise, a controller that's able to move extended robot arms can sometimes prove to be ineffective, as not everyone is acquainted with controller maneuvers. It will be much easier to understand and control if one is able to move the arm in real time with points of reference placed on limb joints that match the ones on the machine. Other utilities include but are not limited to workplace security, drone navigation, and smart home. All of which can and will be made simpler with a motion recognition and communication system.

Solution

We propose a duo-terminal system that reads motion data and sends the encoded information through RF communication to the other terminal which deciphers the data and reproduces the motion in real-time with 3D software simulation or mechanical integration like a motor. Built upon a previous project that clones movement data generated from MEMS sensor measurements to 3D animation, we will still work with discrete accelerometer and gyroscope measurements with appropriate sampling rate to ensure a seamless recreation even in a wireless setting.

Two PCBs are needed for each terminal. While most other components for this project will be printed to PCB, for the sake of flexibility of arrangement, IMUs, aka motion sensors, will preferably connect to the rest of the system via STEMMA QT or long wires. Unfettered from the confinement of circuit boards, IMUs in free space can adapt to more situations and diversify the motions they can output. A great example is the VR controllers that accompany most VR headsets.

Visual Aid



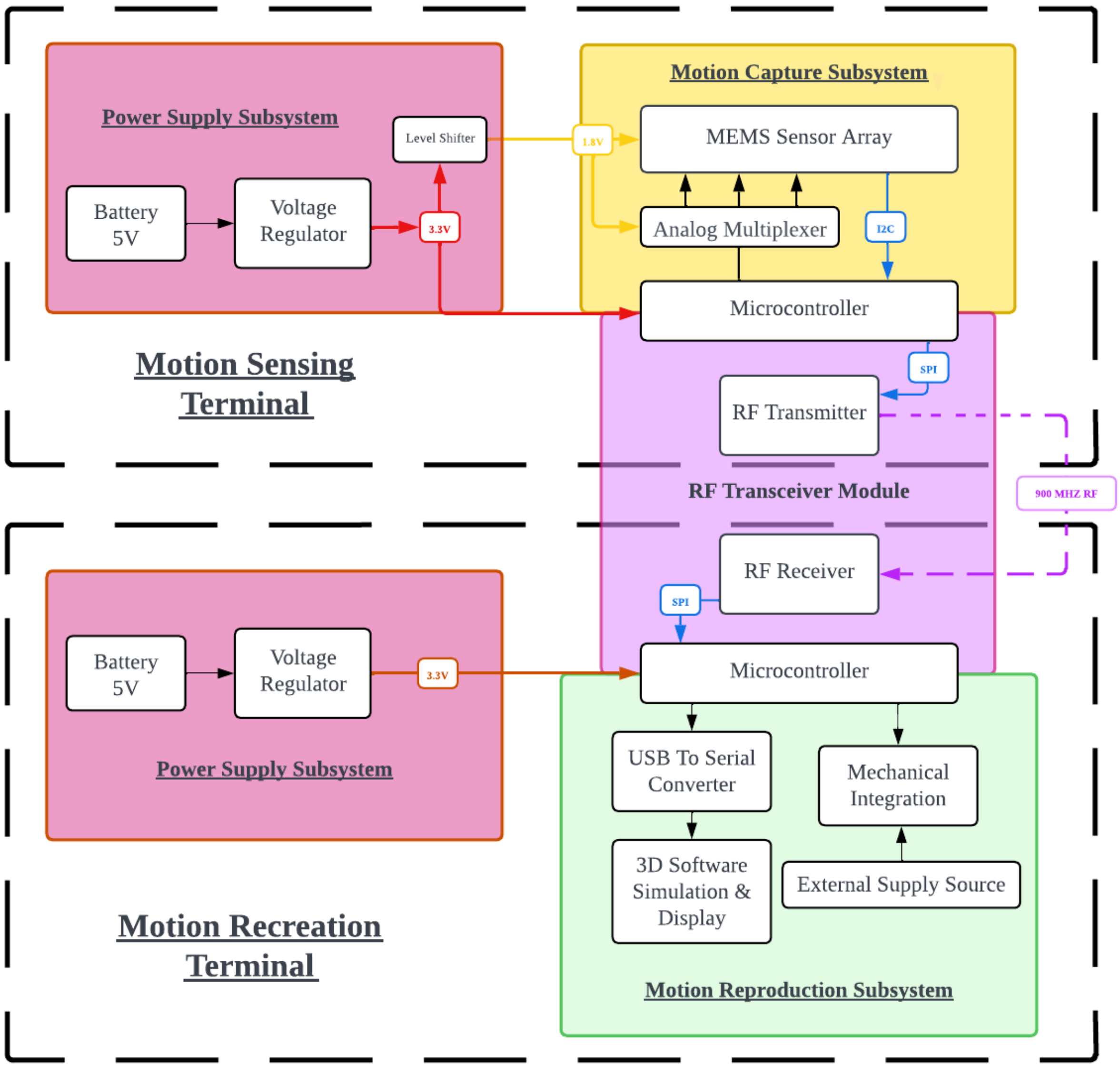
Motion Sensor
Wire
Motion Sensing Terminal
RF
Motion Recreation Terminal
Optional
Mechanical Motion
Motion Control

High-level requirements list:

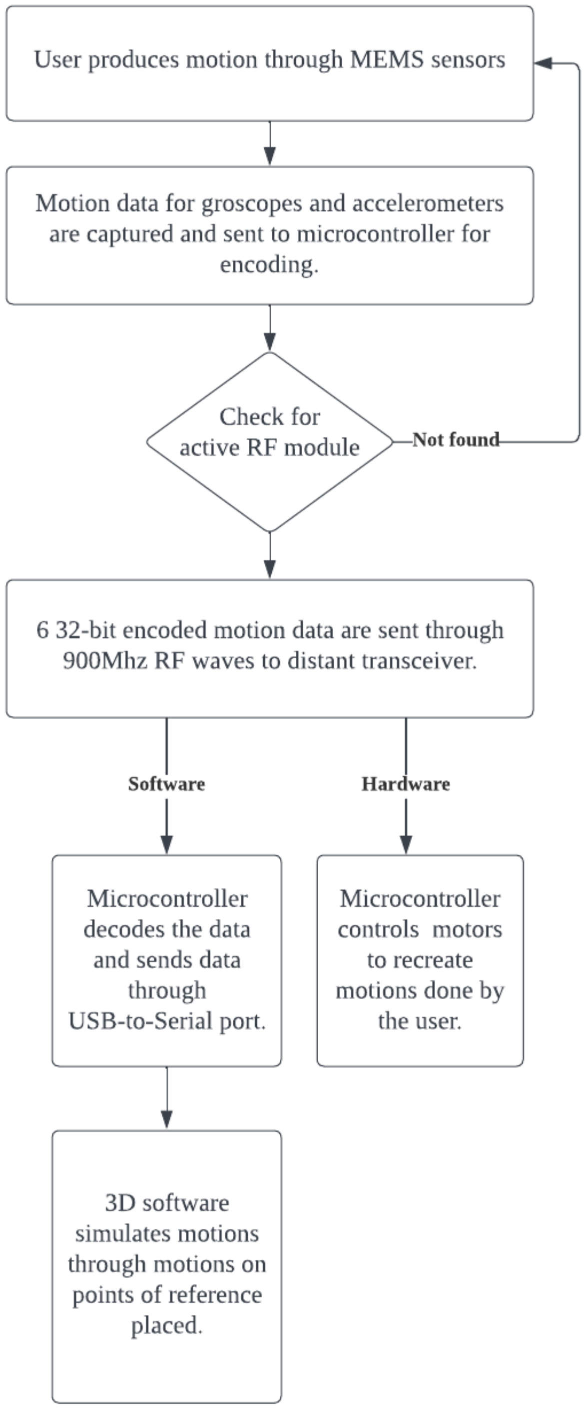
- Positional and rotational motions are captured through MEMS sensors and converted to human–readable data. The MEMS sensor has 6 degrees of freedom. Three for the gyroscope to capture direction and three for the accelerometer to capture movement. For our project we will be connecting the sensors in series to the pcb via wires. This will allow us to move the sensors freely while sending all the data to the pcb.
- RF system is able to function properly and transmits the aforementioned motion data from one device to another at least 0.8-1 mile apart. The RF module will be built into the pcb and connected directly to the main microcontroller.
- The reproduction system at the receiving end is able to dutifully repeat the motion set at the transmitting end on the software end for the minimum success criterion. If met, a 3D printed and/or motor-controlled hardware system can be built to further explore the potential of the project. It will also be implemented with the same pcb design as the motion sensing. The only difference will be in the programming which turns it into a RF signal receiver.

Block Diagram:

The block diagram is primarily divided into two remote terminals as indicated by the dashed box. They communicate via 900mhz RF waves. Each terminal consists of a power supply, a motion capture or reproduction system, and RF module. The transmitting end takes measurements from the MEMS sensor array, while the receiving end decodes the data and commands a software or hardware reconstruction of three-dimensional motion.



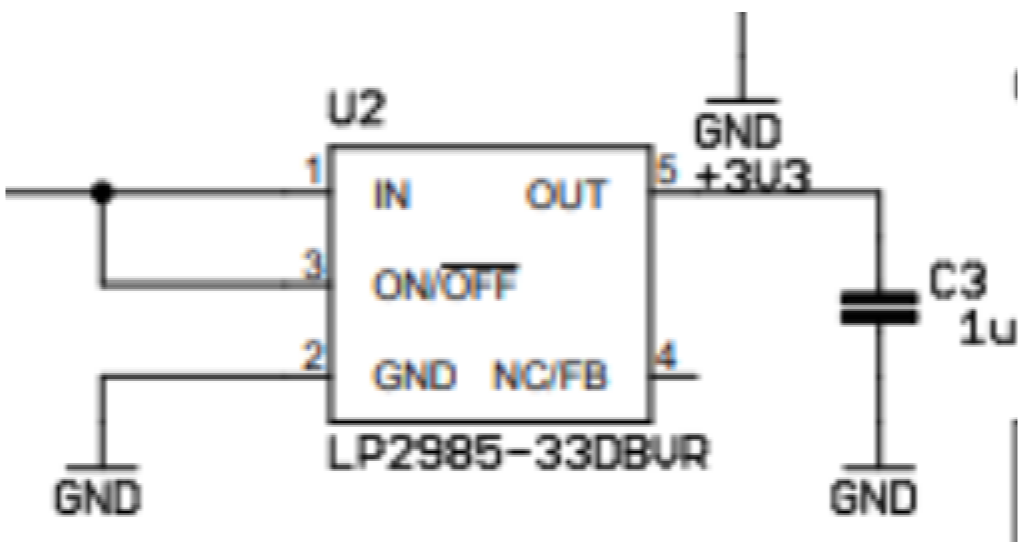
GENERAL-PURPOSE FLOW CHART



POWER SUPPLY:

Both RF components and MEMS sensors require a voltage of 3.3V. We will use AA batteries for the power supply, but onboard LDO from selected microcontrollers will help us control the output voltage. Both terminals use the same form of power supply. The mechanical component needs separate external voltage sources for the motors, preferably batteries as well.

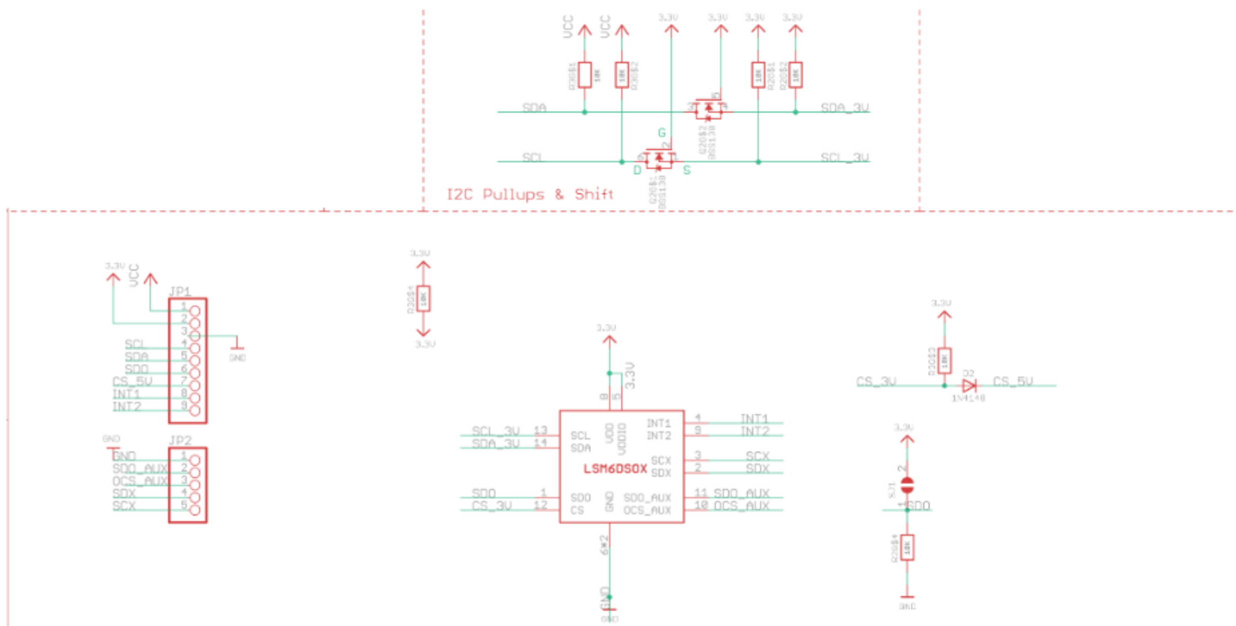
Although the typical operating voltage for the sensors and radio modules are around 1.8V, they can operate under 3.3V up to 3.7V. An LDO is needed at the input source to regulate incoming voltage and keep the rails at 3.3V.



MOTION-CAPTURING SUBSYSTEM

This system consists of two or more IMUs that are in free space. The LSM6DSO32 6-DoF Accelerometer and Gyroscope IC will fulfill the need. Since we are aiming to use STEMMA QT as the connector, the I2C communication protocol is favored. This subsystem in particular will likely reuse some of the codes and concepts developed in a previous project that can be found here: https://wiki.illinois.edu/wiki/pages/viewpage.action?pageId=785286420 [1].

A comprehensive schematic is as below:



RF TRANSMITTER SUBSYSTEM

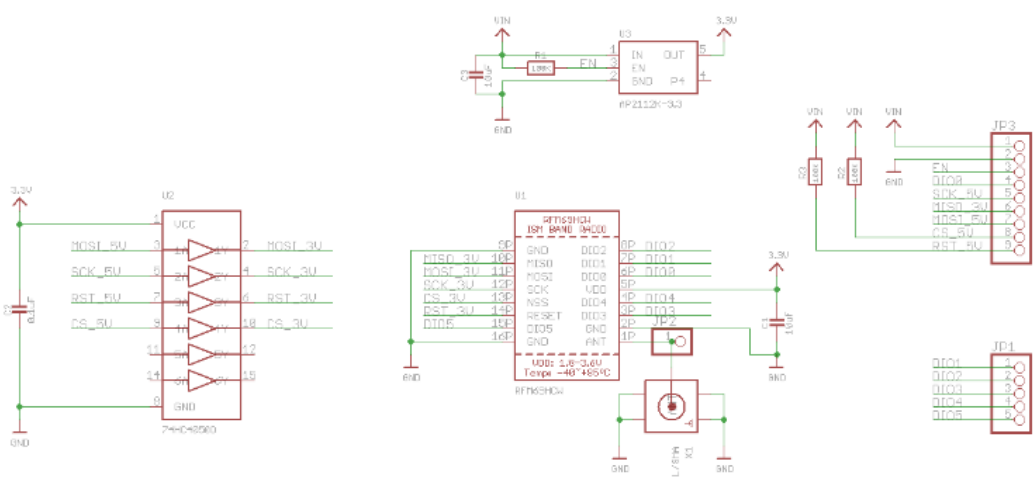
Measurements from the registers of LSM6DSO32 will be sent to the Arduino or an SPI and I2C-enabled microcontroller that processes the information and packs them into a 16 to 32 bit code with at least 4 bits for position and 4 bits for rotation for each three-dimensional axis. The code sent via SPI interface will be transmitted wirelessly through RFM95 transceiver with external Antenna connector.

RF RECEIVER SUBSYSTEM

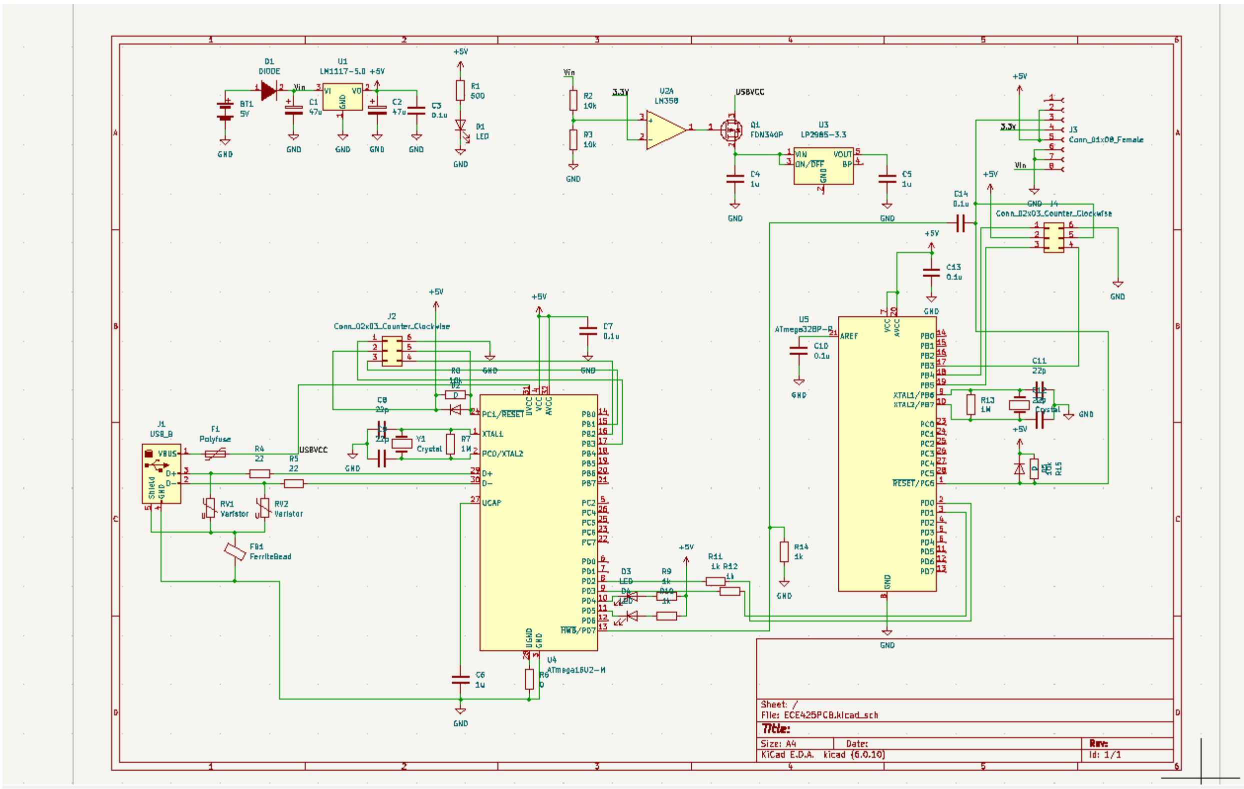
Information sent through the transmitter will be recovered by another RFM95 module connected to another SPI-enabled microcontroller. The microcontroller will analyze and unpack the code to extract the information and prepare the data for respective motion recreation.

RADIO MODULE SUBSYSTEM SUMMARY

Although divided into two parts, the two transmitting and receiving subsystems consist of the same circuit setup in terms of radio ICs. A schematic demonstrating an experimental setup is shown



POWER AND CONTROL SYSTEM SCHEMATIC



MOTION REPRODUCTION SUBSYSTEM

The motion reproduction subsystem ideally consists of two major parts – software and hardware. The software section is realized by receiving data sent through the serial port from the microcontroller at the receiver's end. The Unity 3D engine will decode the information and animate a 3D model in a fashion similar to the previous project done by Joe Luo mentioned above. (https://wiki.illinois.edu/wiki/pages/viewpage.action?pageId=785286420) [1] The hardware component consists of a mechanical integration that's able to recreate simple directional movements, like 3D printed structures or pulse-controlled continuous rotation servo motors (FS90R) that rotate on a 2D plane on a scale dependent on the degrees of rotation of MEMS sensors.

A sample vector-based motion-direction simulation code written in C# for Unity is shown below:

```
// Update is called once per frame
Unity Message | 0 references
void Update()
{
    //logValues();

    //Set finger bottom rotation
    bottomJoint.transform.eulerAngles = bottomEulerAngles;
    //print("Bottom Joint Dir: " + bottomJoint.transform.up);

    // Calculate Joint Position from gryo and accelero inputs
    Vector3 middleBottomJointPos = bottomPosition + fingerBottom.transform.up * fingerBott
    Vector3 topMiddleJointPos = tipPosition - Quaternion.Euler(tipEulerAngles) * Vector3.u

    // Get finger middle rotation
    Vector3 fingerMiddleDir = Vector3.Normalize(topMiddleJointPos - middleBottomJointPos);
    //print("Finger Middle Dir: " + fingerMiddleDir);

    // Set finger middle rotation
    middleBottomJoint.transform.up = Quaternion.LookRotation(Vector3.forward, fingerMiddle
    //print("Middle Bottom Joint Rotated to: " + middleBottomJoint.transform.up);

    // Set finger tip rotation
    topMiddleJoint.transform.eulerAngles = tipEulerAngles;

    // Lock condition values
    tipPosition = fingerTip.transform.position;
    // tipEulerAngles = topMiddleJoint.transform.eulerAngles;
    bottomPosition = fingerBottom.transform.position;
    // bottomEulerAngles = fingerBottom.transform.eulerAngles;

    //print("||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||");
}
```

Below is the requirement and verification table for all the subsystems:

Requirements	**Verifications**
• The RF Transmitter Subsystem must transmit measurements from the LSM6DSO32 to the Arduino or SPI/I2C-enabled microcontroller wirelessly through RFM69HCW transceiver with external Antenna connector.	• Connect the RF Transmitter Subsystem to the Arduino or SPI/I2C-enabled microcontroller and LSM6DSO32. Verify that the measurements from the LSM6DSO32 are being sent to the Arduino or microcontroller via the SPI or I2C interface. • Then, pack the measurements into a 16 to 32 bit code with at least 4 bits for position and 4 bits for rotation for each three-dimensional axis. Record the packed code as read from the microcontroller via serial debugging.

Requirements	Verifications
• The RF Transmitter Subsystem must transmit data with minimal error or loss.	• Ensure the RF Transmitter Subsystem is connected to the receiver subsystem. Record the packed code as read from the receiver subsystem via serial debugging. • Move the RF Transmitter Subsystem to a location with greater distance or obstructions between it and the receiver subsystem. Record the packed code as read from the receiver subsystem via serial debugging. Confirm that there is minimal error or loss in the transmitted data.
• The RF receiver subsystem must be able to recover information sent through the transmitter wirelessly.	• Connect the output of the RF transmitter module to the input of the RF receiver module. Transmit 6 32-bit codes containing information about the three axes of gyro and accelerometer from the transmitter and confirm that the data is received at the output of the receiver. Use an oscilloscope to ensure that the signal is clean and free from interference.
• The RF receiver subsystem must be able to connect to an SPI-enabled microcontroller to analyze and unpack the code received from the transmitter.	• Connect the output of the RF receiver module to the input of the SPI-enabled microcontroller. Transmit a known code from the transmitter and confirm that the same code is received by the microcontroller. Use a logic analyzer to ensure that the code is correctly unpacked and extracted by the microcontroller.
• The microcontroller must be able to prepare the data for respective motion recreation.	• Program the microcontroller to analyze the received code and prepare the data for motion recreation. Verify that the prepared data accurately represents the motion information that was originally sent by the transmitter. Use a motion sensor to confirm that the recreated motion matches the original motion sent by the transmitter.
• The motion sensing system must measure the linear acceleration and angular velocity in three dimensions	• Secure IMUS on a hand and manually move the IMUs in different directions and rotations. Record the data and verify that the linear acceleration and angular velocity in all three dimensions are being measured by the IMUs.
• The motion sensing system must use I2C communication protocol for data transmission	• Connect the LSM6DSO32 IC to the microcontroller using I2C protocol. Verify that the data is being transmitted between the microcontroller and IMUs using an I2C sniffer or logic analyzer.
• The software section must be able to receive data and animate a 3D	• Connect the microcontroller at the receiver's end to the computer via serial port. Verify that the software section

Requirements	Verifications
model.	receives the data correctly and that it is able to decode the information. • Using those inputs, verify if the 3D model simulates motions in the real world within a reasonable range.
• The hardware component must be able to reproduce movements on a scale dependent on the degrees of rotation of MEMS sensors.	• Vary the degrees of rotation of the MEMS sensors and record the resulting movements of the hardware component. Verify that the hardware component is able to reproduce movements on a scale dependent on the degrees of rotation of the MEMS sensors.
• The power supply must meet required average and peak voltage	• Test the average and peak voltage, 3.3V and 5V respectively, using a multimeter.
• The hardware component must be able to utilize pulse-controlled servo motors and 3D printed connection parts. (Extra parts, extension to the project if finished early)	• Connect the pulse-controlled continuous rotation servo motors to the hardware component. Verify that the hardware component is able to rotate freely in their own dimensions and reach limitations in a safe manner. • Design and print 3D structures and connect them to the hardware component to connect the rest of the electronic parts.

Tolerance Analysis

Admittedly, the power consumption of RF devices has always been a concern for radio module circuits. Since the motion update needs to be real-time, the radio modules should rarely enter idle or sleep modes, which may cause additional power consumption. To estimate the rough power consumption in the presence of RFM95W and additional IMUs, we take power ratings, idle current, and power supply into account. The master equation [3] for total consumed energy is :

$$ETotal = ESleep + EActive$$

From which, we deduct the model we use to calculate the power consumption.

$$ESleep = PSleep \cdot TSleep$$

$$EActive = EWU + Em + Eproc + EWUT + ETr + ER$$

Since we are mostly concerned with the energy for wake-up of the LoRa transceiver and energy of transmission or reception mode as explained above but not microcontroller ratings here, we can simplify the equation to

$$Etotal = EWUT + ETr + Er$$

The consumed energy EWUT during transceiver wake-up duration TWUT is given by:

$$EWU = PON(fMCU) \cdot TWU$$

The consumed energies of transmit and receive mode are expressed respectively as:

$$ETr = (PON(fMCU) + PTr) \cdot TTr \qquad \text{(Eq. 1)}$$

$$ER = (PON(fMCU) + PR) \cdot TR \qquad \text{(Eq. 2)}$$

Of which $Tr = Nbit \cdot Tbit$, where Nbit is transmitted bits and the duration of one transmission. To obtain the metrics for relevant parameters, we use the equation for bit-rate (in bits per second) as shown below:

$$Rbit = SF \cdot BW/2^SF \cdot CR \qquad \text{(Eq. 3)}$$

Where CR represents a controllable coding rates, and the rest are displayed in the tables for RFM9X below:

Part Number	Frequency Range	LoRa^{TM} Parameters			
		Spreading Factor	Bandwidth	Effective Bitrate	Sensitivity
RFM92	860 - 1020 MHz	7 - 12	125 - 500 kHz	0.3 - 20 kbps	-137.5 dBm
RFM93	860 - 1020 MHz	7 - 9	125 - 500 kHz	1.7 - 20 kbps	-130 dBm

Symbol	Description	Conditions	Min	Typ	Max	Unit
IDDSL	Supply current in Sleep mode		-	0.1	1	uA
IDDIDLE	Supply current in Idle mode	RC oscillator enabled	-	1.5	-	uA
IDDST	Supply current in Standby mode	Crystal oscillator enabled	-	1.4	1.6	mA
IDDFS	Supply current in Synthesizer mode	FSRx	-	4.5	-	mA
IDDR	Supply current in Receive mode	*LnaBoost* = 00	-	10.5	-	mA
IDDT	Supply current in Transmit mode with impedance matching	RFOP = +20 dBm, on PA_BOOST RFOP = +17 dBm, on PA_BOOST RFOP = +13 dBm, on RFO pin RFOP = + 7 dBm, on RFO pin	- - - -	125 90 28 18	- - - -	mA mA mA mA

The crystal oscillator wake-up time for RFM9X is 250us. Substituting Eq. 3 into Eq. 1 and Eq. 2 and factoring desired parameters into the equation, we thereby obtain the energy consumption for a complete communication for 10 seconds as what is illustrated below:

ETr = (250us * 3.3V + 1.5mA*3.3V) * [10*300*1000/2^10*1/(4+4)] * 10 = 8.81J

Hence, the average power consumption for transmission is given by 8.81J/10s = 0.881W, which is reasonable and within the range of power we estimated for the radio module circuit. Same goes for the receiving terminal.

Another point of interest is whether the bit rate supplied by the chip is sufficient for our information to be properly transmitted. Our measurement demands six axes of data, spanning the three from the accelerometer and the three from the gyroscope. The accelerometer is selected to have a ±8g sensitivity. However, considering the limit of human motion and accessibility of motors in the hardware reconstruction, we limit the maximum acceleration transmitted to ±255m/s^2. The gyroscope, on the other hand, outputs degrees ranging from -180 to 180. As a result, we need at most 32 bits for each of the axes and one data header and footer, with a sampling rate of 1Hz, at least

$$\text{Total Bits} = 36 * (6+2) = 288 \text{ bits}$$

are needed for one successful transmission per second. The calculation from Eq.3 nets us with a bit rate of 366.21 bits/second, which is well above the needed transmission rates of data.

Cost and Schedule

Cost

Name	Rate	Hours	Total (R*H*2.5)
Joe	$33	$210	$17325
James	$33	$210	$17325
Jason	$33	$210	$17325
			$51985

Item	Part Number	Quantity	Cost	Total
Microcontroller	ATmega328P	2	$2.86	$5.72
Microcontroller	AT mega16U2	2	$3.48	$6.96

Radio Transceiver	RFM95W	2	$7.51	$15.02
Voltage Regulator	LP2985	2	$0.359	$0.72
MEMS Sensor	LSM6DSOX	3	$8.26	$24.78
Small circuit components (Resistors, capacitors..)	N/A	20	$10	$10
USB to Serial Converter	ATmega16u2	1	$3.59	$3.59
Qwiic Cable	JST PH connector	9	$12.93 (total)	$12.93
Analog Multiplexer	MC74HC251A	1	$0.57	$0.57
Total				$80.29

Grand Total: $52045.47

Schedule

The following is a rough schedule of the project

<table>
<tr><td>Week</td><td>Jason</td><td>Joe</td><td>James</td></tr>
<tr><td>2/20</td><td colspan="2" rowspan="2">> Breadboard circuit test
> Narrowing designing ideas
> Deciding circuit parts</td><td>> pcb schematic draft</td></tr>
<tr><td>2/27</td><td>> Finalize schematic, pcb layout draft</td></tr>
<tr><td>3/6</td><td colspan="2">> Parts order
> Finalize software selection</td><td>> Finalize layout</td></tr>
<tr><td>3/13</td><td colspan="3">> Spring break (start soldering if parts received)
> Initial testing on devices and integrated circuits</td></tr>
<tr><td>3/20</td><td colspan="3">> Soldering, first round pcb testing
> Continued testing on existing system. Functional radio transfer of information on</td></tr>
</table>

	IMUs should be established. Data is exhibited on serial ports on computer		
3/27	> Initial testing on pcb once received (reorder pcb if required)	> Further testing with existing system and IMUs > Finalizing machine shop order and design	> Initial testing on pcb once received (reorder pcb if required)
4/3	> 3D engine simulation and mechanical component simulation and testing		> Final pcb testing
4/10	> Project complete and prepared for demo		

Ethics and Safety

When working on a project, it is important to consider the ethical and safety issues that may arise both during development and from the potential misuse of the project. Adhering to the IEEE Code of Ethics [2] and ACM Code of Ethics, which emphasize the importance of avoiding harm to others and upholding integrity, is crucial to avoiding ethical breaches. To ensure compliance, it is necessary to review relevant safety and regulatory standards such as state and federal regulations, industry standards, and campus policy.

This project has been approved, which means the general purpose of this project is ethical and safe. However, it is essential to continue monitoring and evaluating its ethical and safety implications throughout the development process. Additionally, potential safety concerns in the project should be identified and addressed in order to minimize any potential harm to users or others.

While our project is generally safe, there are a few minor safety concerns that should be addressed. Below is a lab safety instruction that elaborate on those issues:

Lab Safety Instructions for the Project:

1. Small Motors:

- The motors should be mounted securely to prevent any movement while in operation
- The power supply to the motors should be turned off when not in use
- The motors should not be operated in the presence of flammable substances

2. Moving Robot Arm:

- Only team members with related experiences should operate the robot arm
- A safety perimeter should be established around the robot arm
- The power supply to the robot arm should be turned off when not in use

3. Small Batteries:

- Batteries should be kept away from heat sources and flammable substances
- Batteries should be stored in a cool, dry place when not in use
- If any battery is damaged or leaking, it should be disposed of according to the manufacturer's instructions

The above lab safety instructions sufficiently protect both users and developers from unsafe conditions by providing guidelines for the safe handling and use of the small motors, moving robot arm, and small batteries. By training personnel on proper handling procedures, establishing safety perimeters, the risk of injury or harm is significantly reduced. Turning off power supplies and storing batteries properly also helps prevent accidents related to electrical or chemical hazards. The instructions also emphasize the importance of avoiding the presence of flammable substances, which could cause a fire or explosion. By following these guidelines, both users and developers can work safely with the project's equipment and minimize any potential safety risks.

References

1. Unity-enabled gesture motion capture (Joe Luo) - ECE 395 Advanced Digital Projects Lab - Illinois Wiki. [Online]. Available: https://wiki.illinois.edu/wiki/pages/viewpage.action?pageId=785286420. [Accessed: 09-Feb-2023].
2. “IEEE code of Ethics,” IEEE. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 09-Feb-2023].
3. Adafruit RFM95 LoRa Radio Breakout. [Online]. Available: https://learn.adafruit.com/assets/85363 [Accessed: 23-Feb-2023].
4. Adafruit LSM6DSO32 IMU. [Online]. Available: https://learn.adafruit.com/assets/85363 [Accessed: 23-Feb-2023]
5. Arduino Uno Rev3-Schematic . [Online]. Available: https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf [Accessed: 23-Feb-2023]