Last modified October 2$^{s}$ 2023

Work of Augustin Vercoutere (UCLouvain)

# Readme 3-bar Tensegrity :

This document will give you information on where to find the documents you need, how to run the robot and finally how to assemble one.

In the following folder you will find 4 subfolders:

- CAD : Folder containing all the Solidworks designs permitting to build a rod.
- Code : Folder containing the codes to run on your computer and the codes to upload on the Arduino's of the rod.
- Debug : Folder containing Arduino code that permits to check if the subassemblies of your rod works
- Lib : Folder containing the modified Arduino libraries used in order for the robot to work

## CAD :

This folder contains 2 subfolders :

- Cap&cases : Folder that contains all the cases and caps for the sensor on the strut.
- Strut : Folder that contains a Solidworks assembly of all the elements containing a strut.

## Code :

This folder contains 3 codes and 1 excel sheet :

- Classes_sensor_calibration.py : Code that permits to retrieve all the information from the rods and prints the capacitance of the strain sensors, sends ROS messages to your Tensegrity ROS node and permits to manually turn the motors by hitting specific keys.
- Classes_run_tensegrity.py : Code that permits to retrieve all the information from the rods, computes the PWM to be send to the rods, sends ROS messages to your Tensegrity ROS node and permits to manually turn the motors by hitting specific keys. It can also do the same thing as Classes_sensor_calibration.py by hitting the c key.

- calibration augustin.xls : Excel file used for calibration of the robot, you should specify the value of capacitance of the strain sensors at certain specific length.
- UDP_AccSensor_CapSensor_DcDriver_V2.ino : Code to upload into each rod. The code permits the Arduino 33 iot to connect to the computer hotspot, to retrieve all the sensor information and to send them to the computer.
  Line 10 & 11 : Specifies the hotspot to which the robot must connects itself.
  Line 21-23 : Specifies the number of the rod (0,1,2) and the motors related to it (0-6).
  Line 33 : AD0_VAL must be set to 1 if the I2C address of the ICM20948 is 0x69, 0 if the address is 0x68.
  Line 139,173,192,286 : Specifies the IP address of the hotspot it must connect (it must thus be changed accordingly).

## Debug :

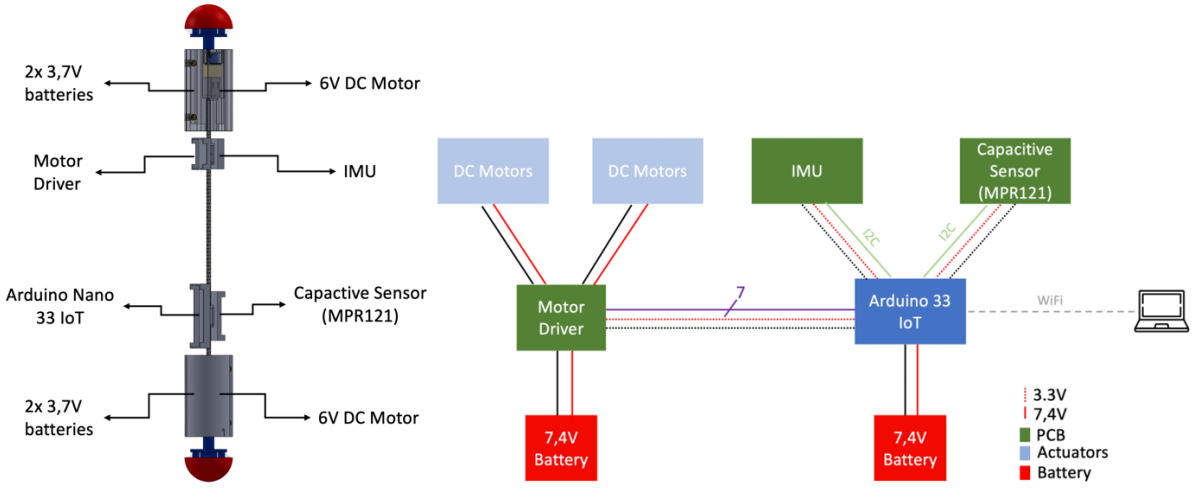
This folder contains 3 Arduino codes that helps to test the modules separately :

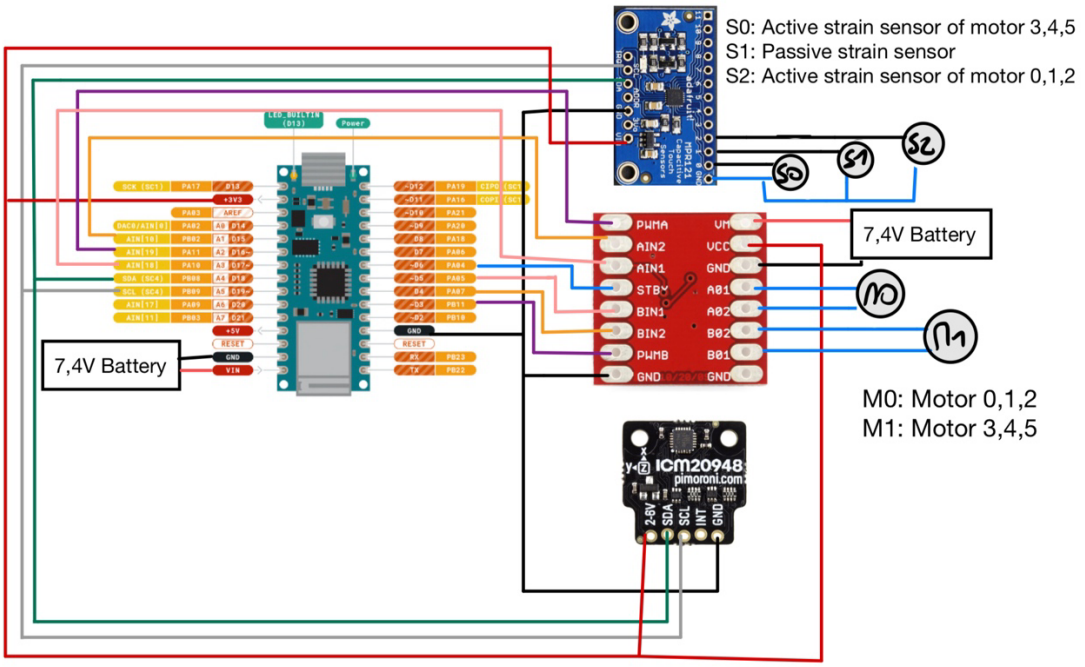
- i2c_scanner.ino : Code that scans the connected devices to the I2C of the Arduino 33 iot.
- ICM20948_DMP_Quaternions.ino : Code that tests if the ICM20948 is correctly connected.
- MotorTestRun.ino : Code that tests the 2 motors of the rod.

## Lib :

In this folder you will find a modified library used for the ICM20948. Please replace it with the according library's you should install for the Arduino codes to work.

## How to build a rod :






On the previous images you can find the design of the rod, an electric scheme as well as a diagram of connection for the different modules. The upper motor of the rod will always be the smallest motor e.g. : Rod 2 has motor 0 and 5 and thus the upper motor will be 0.
The active strain sensor of each motor will be printed as it equivalent in the alphabet e.g. : The active strain sensor related to motor 0 will be called a, the one for motor 1 will be called b, …
Concerning the passive strain sensor will be printed as I,H or G depending on which rod it is connected to !

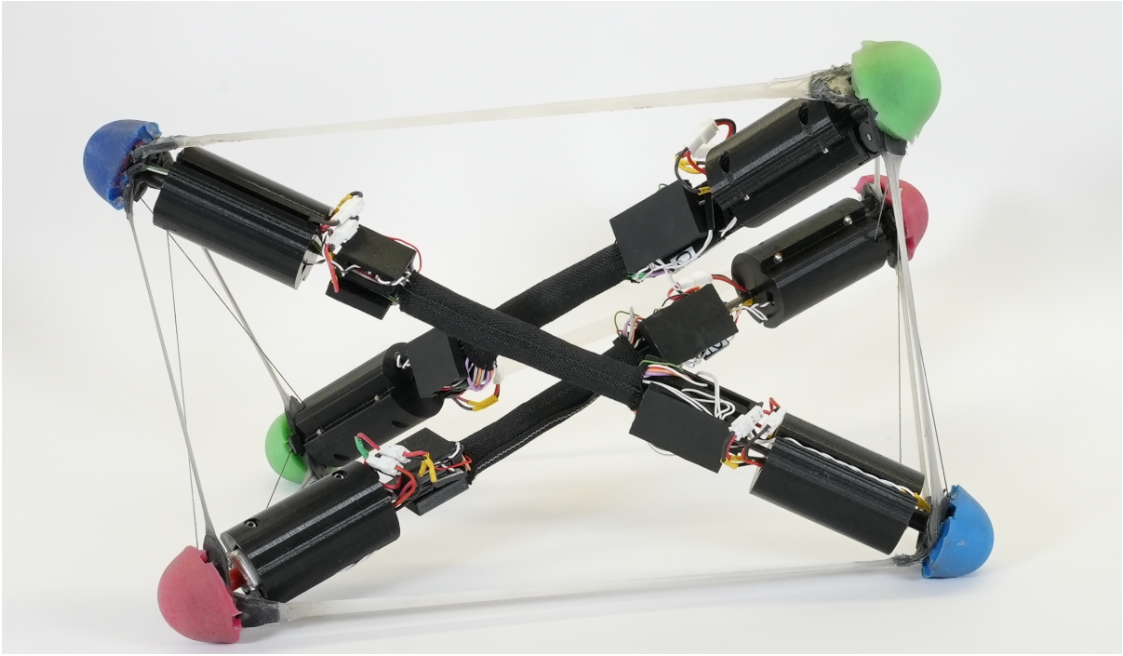
Reminder, different connections might work but must be changed accordingly in the different codes!
Second reminder, you should update the firmware of the Arduino 33 iot before using it!

## How to build a 3-bar Tensegrity:

Once you've build 3 different rods, the different motors, strain sensors and color for the RGB-D camera for each rod are divided as the following :

- Rod 0 (red) has sensors C, E, and I (2, 4, and 8) and motors 2 and 4
- Rod 1 (green) has sensors B, D, and H (1, 3, and 7) and motors 1 and 3
- Rod 2 (blue) has sensors A, F, and G (0, 5, and 6) and motors 0 and 5

This should give you the following 3-bar tensegrity robot :



Be careful, the assembly of the tensegrity should follow the same position of the rods in order for it to roll perfectly! Before trying a roll, the robot should be calibrated, and the key “r” should be used during the use of the code Classes_run_tensegrity.py in order to have the perfect shape.