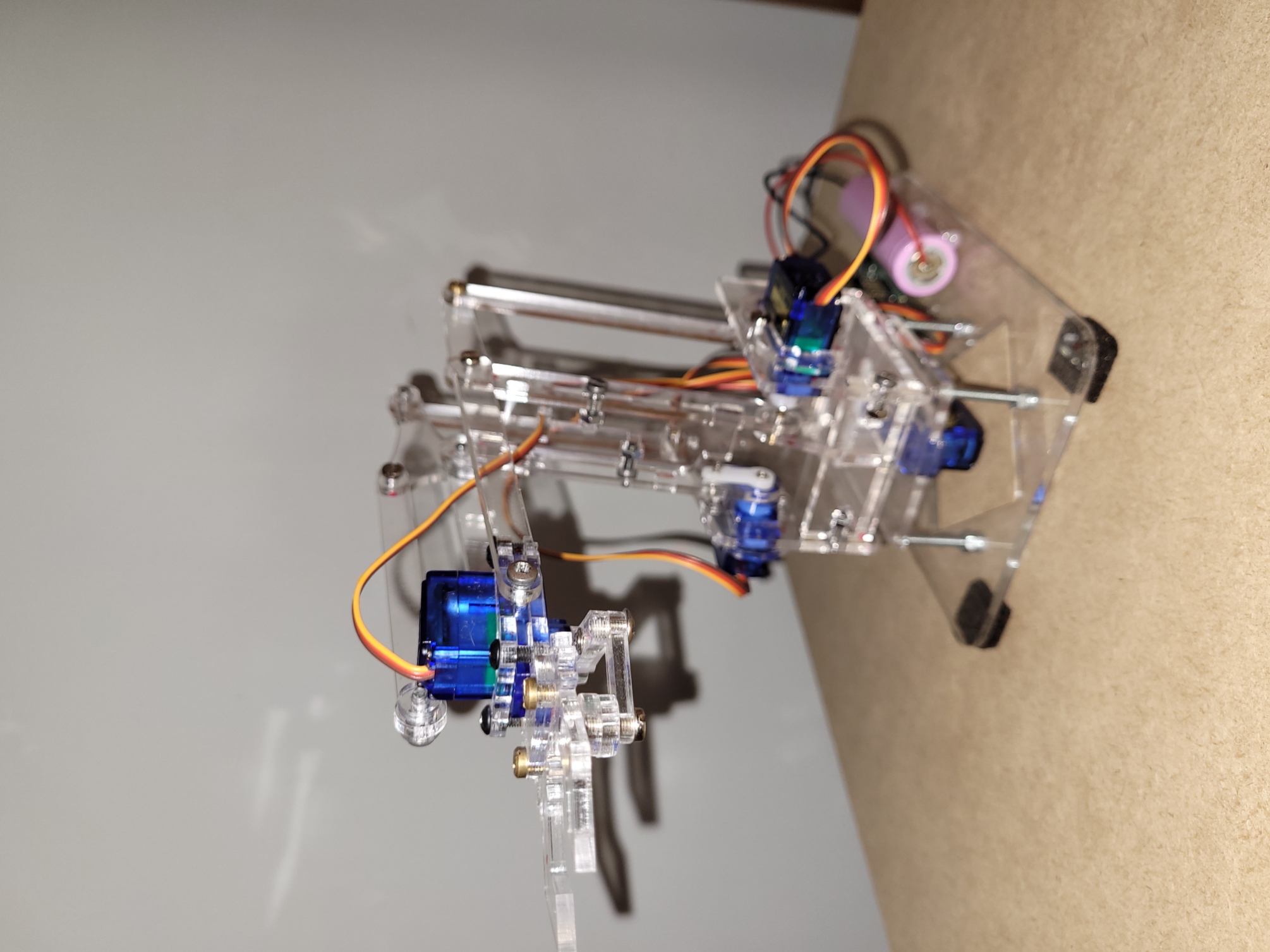
Robotic Arm firmware with ROS integration



Robot Programming Project Report

Flavio Galasso 2034377

18/02/2022

# MEARM Robot

The MEARM robot is an open source robotic arm, used for teaching kids about robotics. The cost, ease of assembly and easy availability of components make it the best choice for an introduction to robotics, especially for children. The robot was designed to be controlled by a joystick with an arduino library provided by the creators themselves.

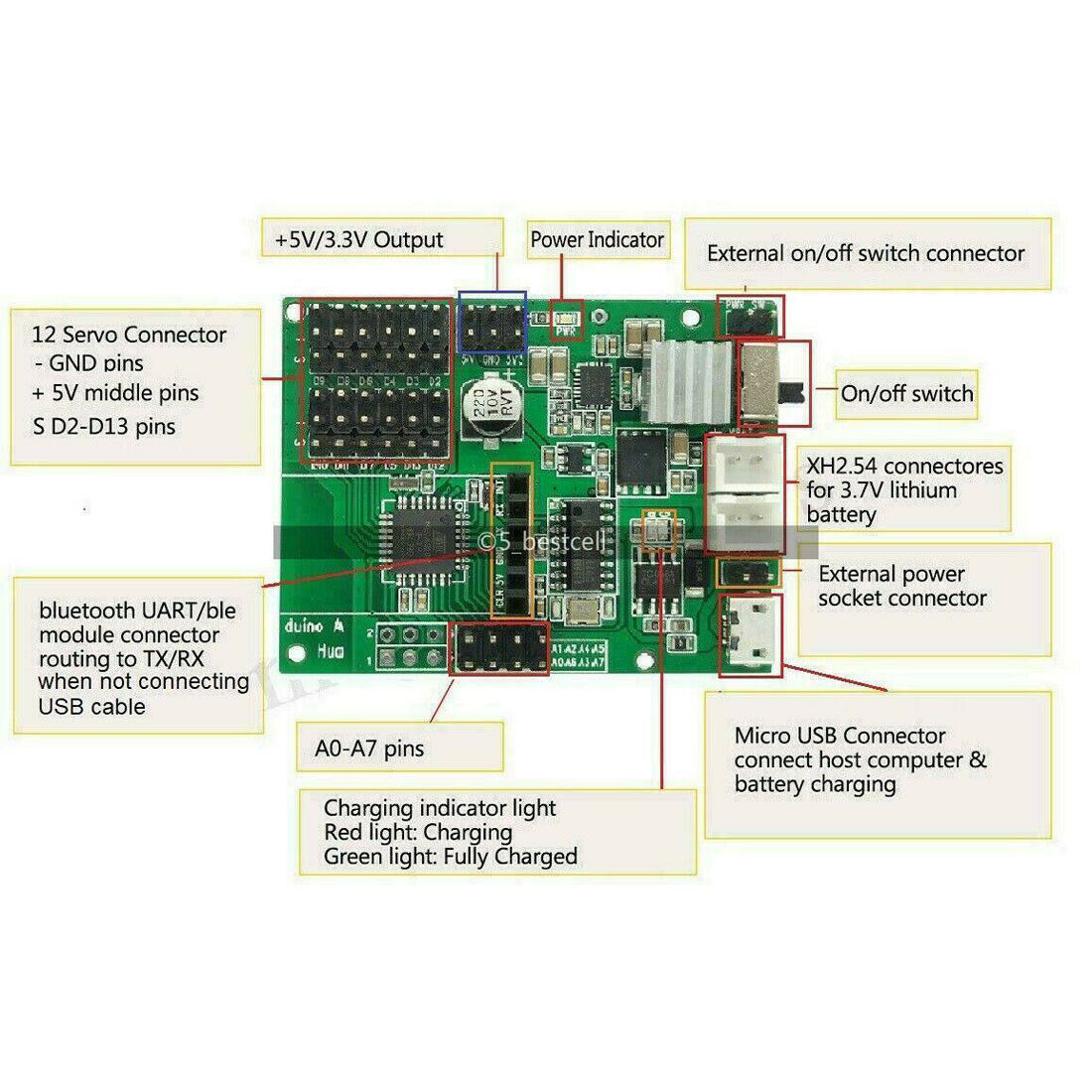
The aim of my project and its future implementations is to use only the mechanical structure to implement a more advanced controller, capable of solving direct and inverse kinematics problems and able to communicate with ROS, giving access to easy teleoperation and visualization of the robot joints.

### Hardware

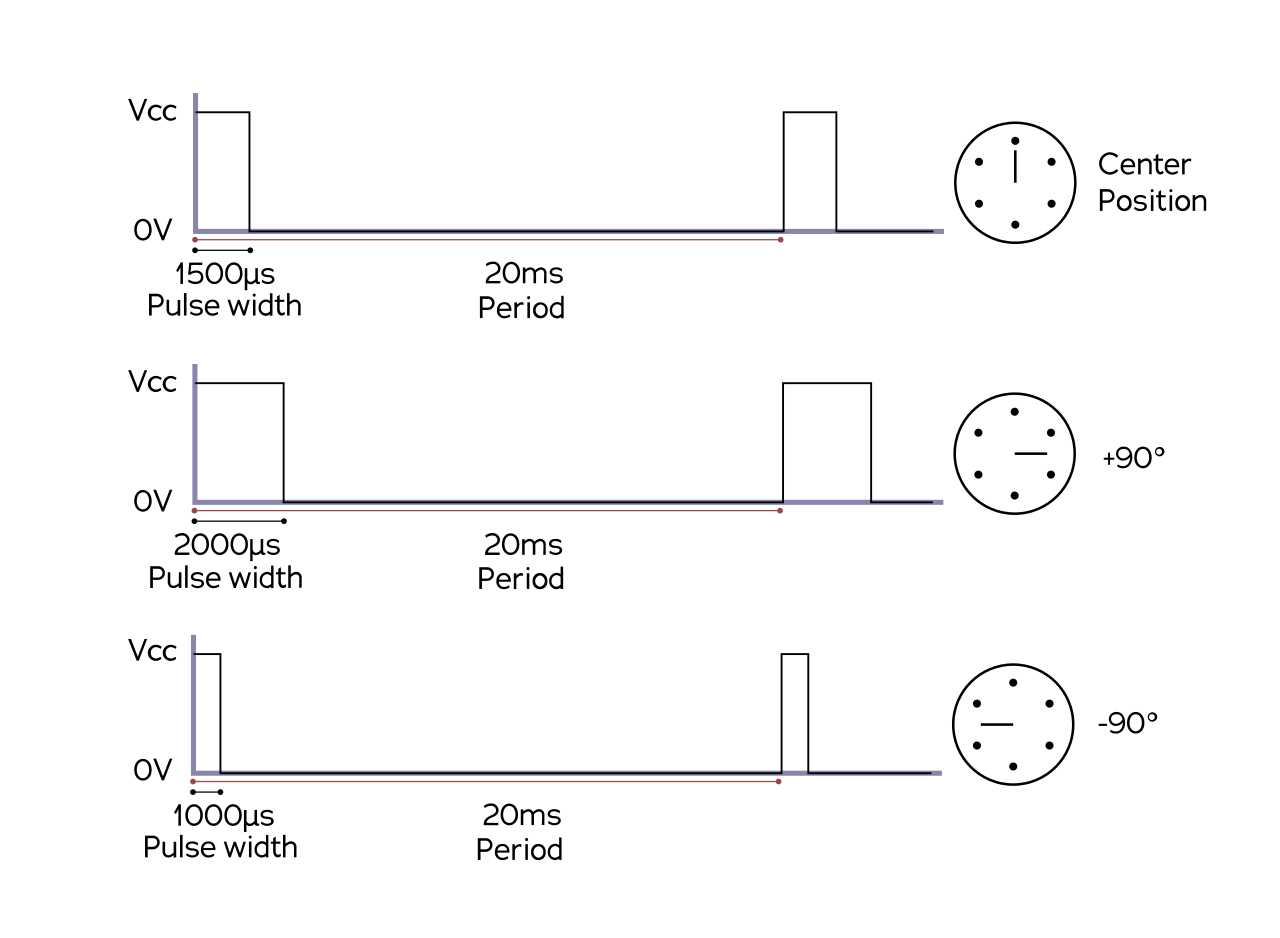
1. 9G SG90 Servo x 4
2. 18650 LI-ION battery x 1
3. A4 sheet of plexiglass x1
4. Huaduino board x1

The robot structure has been laser cut in 3 revisions using the open source .DXF diagrams and version 0.4 has been chosen after several sessions of assembly and testing of the movement of the servomotors.

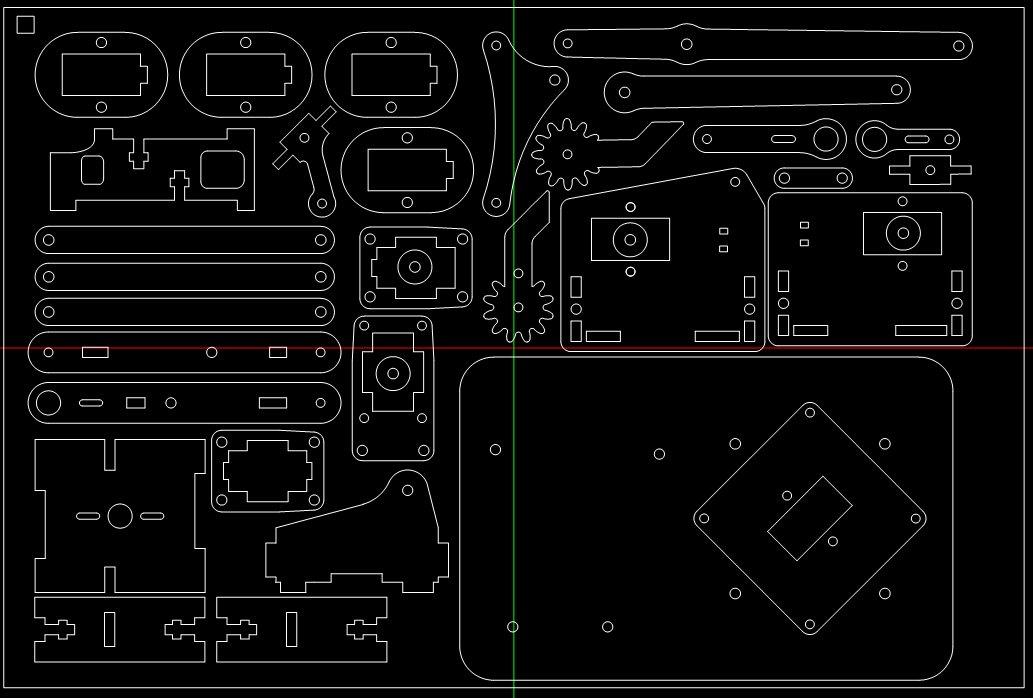
The board Huaduino has been choosen because it includes an AVR Atmega328 and a breakout of 12 servo-ready connectors. This board also includes a 3.7v 18650 battery charger and 12A step-up driver for powering all the 5v servos.



For the servomotors the cheapest and most available on the market were chosen, which turned out to be largely inaccurate, with little torque and with a very high backlash due to the plastic gears.

**1.1.2 Assembly**

Refer to this link for the correct assembly of the robotic arm: <https://www.instructables.com/Pocket-Sized-Robot-Arm-meArm-V04/>



# Arduino Firmware

Requirements: Arduino IDE 1.8.19, [RosSerial Arduino Library 0.9.1](https://downloads.arduino.cc/libraries/github.com/frankjoshua/Rosserial_Arduino_Library-0.9.1.zip)

-Clone git repo: <https://github.com/redmodder/ros_roboticarm>

-Open file \Arduino\_Code\RoboticArm\RoboticArm.ino

-Before Compilation:

1. Open file: Rosserial\_Arduino\_Library-0.9.1\src\ros\msg.h
2. Change all occurrences of #include <cstring> to #include <string.h>
3. Change all occurrences of std::memcpy() to memcpy()

-Arduino IDE board setup: Arduino Nano, Atmega328P (old Bootloader)

-Insert the usb cable of the huaduino board in the computer

-Power up the huaduino board from its switch (Left->Right = ON)

-Select the right COM port in the Arduino IDE Tools submenu

-Submenu Sketch->Upload to compile and automatically upload the firmware

# RosCpp Package

Requirements: Linux Ubuntu 20.04.3 LTS, Ros Noetic, roscpp 1.15.13, rosserial\_arduino 0.9.2

-Clone git repo: <https://github.com/redmodder/ros_roboticarm>

-In a new terminal, run (and keep running):

1. roscore

-In a terminal, inside the main repo folder, run and keep running:

1. cd Ros\_Code
2. catkin build
3. source /devel/setup.bash
4. rosrun RoboticArm\_ServoPublisher RoboticArm\_ServoPublisher\_node

-In a new terminal, run (and keep running)

1. rviz

-Inside rviz program, open the configuration file config.rviz from the main folder of the cloned repo

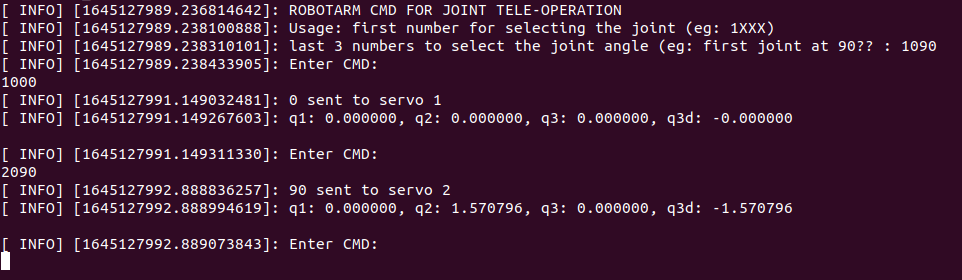
-Connect the huaduino board

-In a new terminal, run (and keep running):

1. rosrun rosserial\_arduino serial\_node.py

# Operation and Test

When every step has been successfully executed, the robot should move from resting position to an armed position when the rosserial\_arduino package is launched pointing to the correct COM port (I didn’t have to do it, the port was automatically found). When there are no linked connections between ros and the arm, it should go in a resting position with the end-effector pointing to the ground. To operate the robot joints, input a 4 digit number in the terminal that has “RoboticArm\_ServoPublisher” package running, where the first digit is the number of the joint, and the other 3 are the angle expressed in positive degrees from 000->090->359.



Everytime a new command is executed, the robot should move accordingly to the commanded angle and a new /tf representation will be shown in the rviz environment. Please note that the input angles are NOT the ones sent to the servos, they represent the denavit hartemberg parameters and they will be constrained according to the respective joint angle limits and then they will be converted in the right servo angles inside the robot firmware.

