**Lego and ROS Manipulator Assignment Report**

**ASE-9406 Robot Manipulators : Modelling, Control and Programming**

Group 17

Maxime Etienne, 273556

Prasun Biswas, 267948

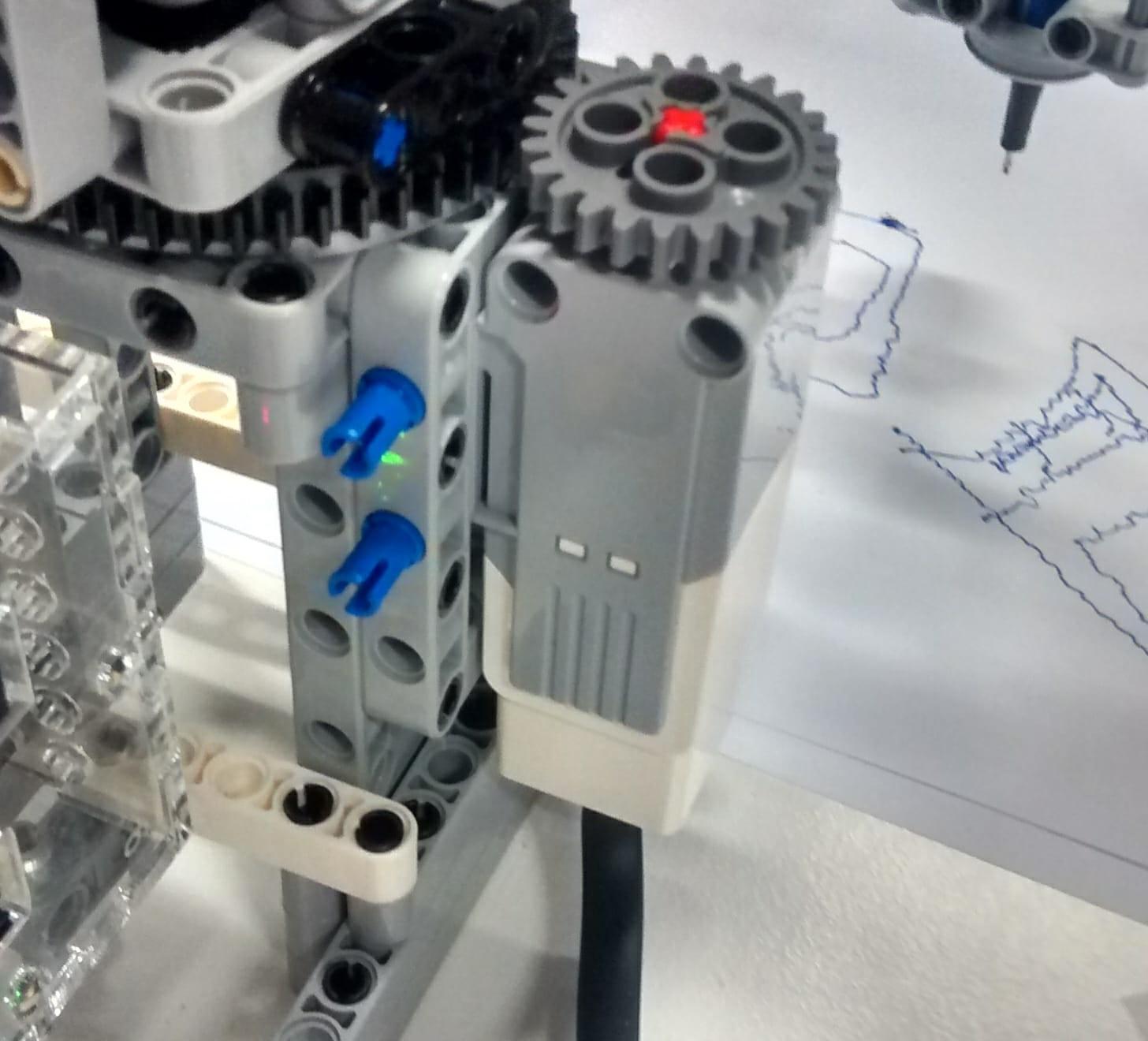
Aleksi Järvelä, 234017

1. **Building the robot**

We chose to build a SCARA robot. Indeed, this model seemed to us to be the simplest to build and control, for it has 3 joints to control, 2 revolute and one prismatic.

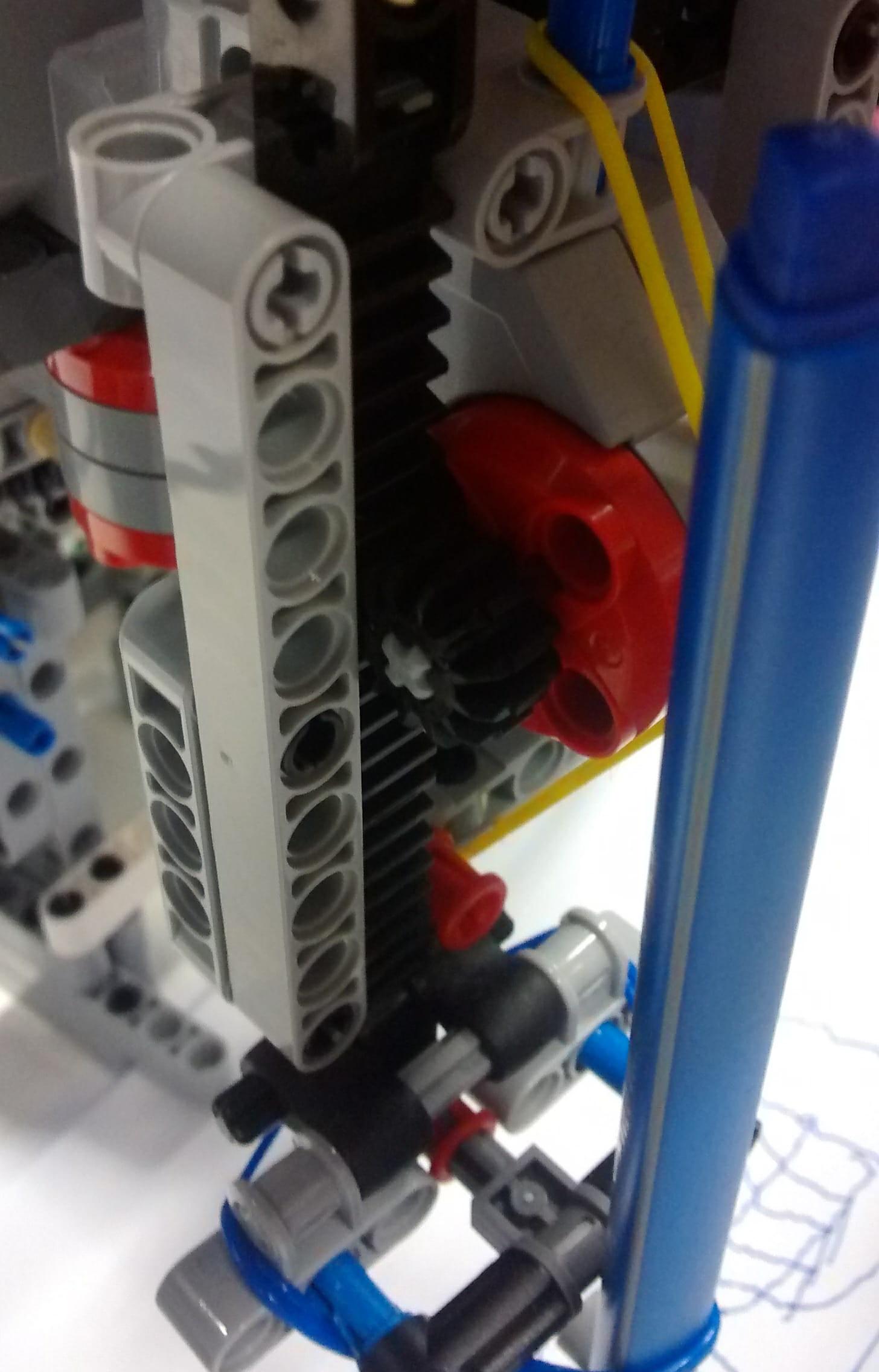
We added a pen with a medium soft point at the end of the tool. Maybe a softer point would have been better, but we needed a pen with a thin body, and we didn’t have anything better.

We used gears for transmission in joint 1 (revolute). With a transmission coefficient inferior to one, it enabled us to get more precision.



*Fig [1] : Gear transmission for joint 1*

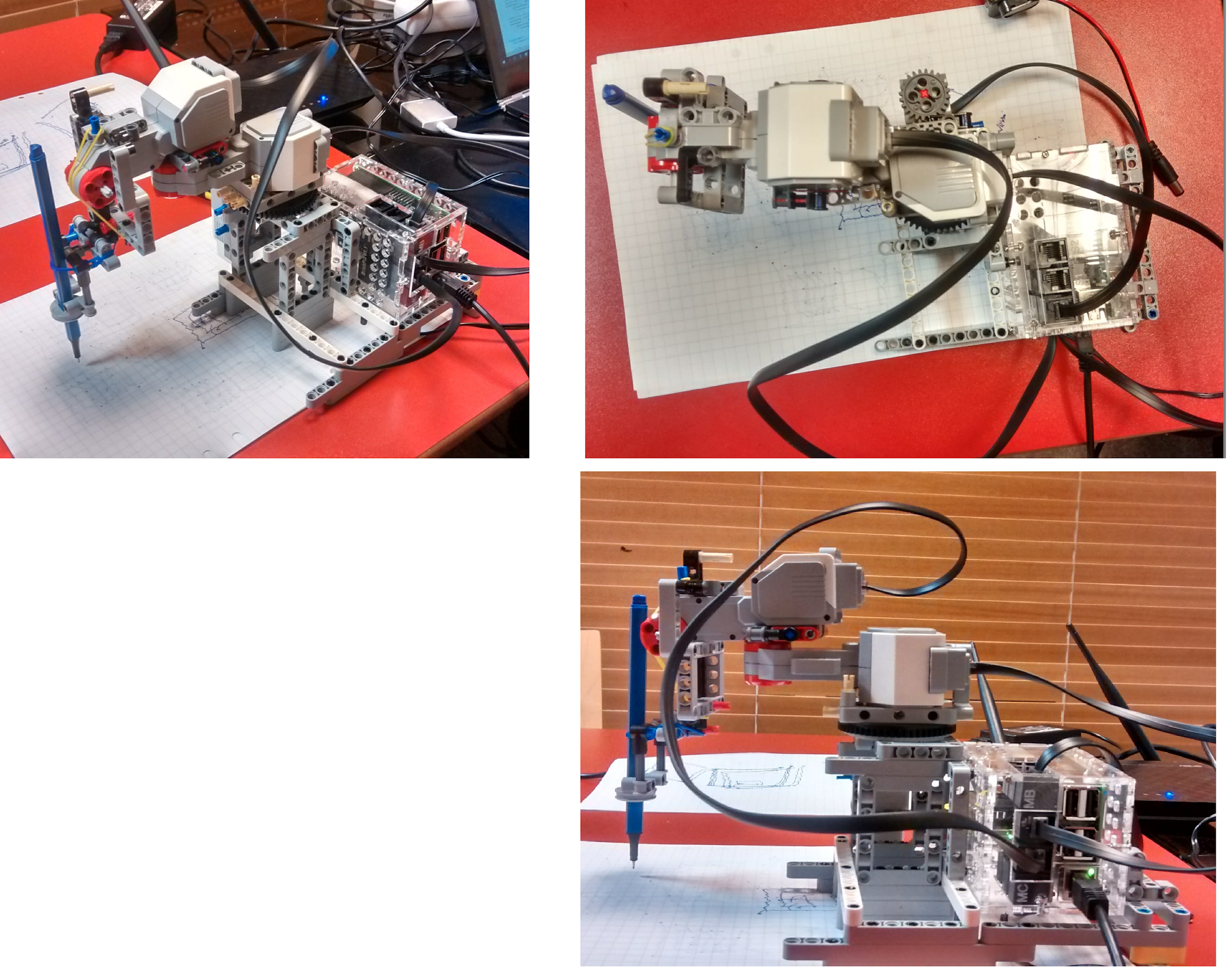
Also we used a rack and pinion transmission between motor 3 and the tool, to create a prismatic joint.



*Fig [2] : Rack and pinion transmission for prismatic joint*

A difficulty we encounter while constructing this robot was the optimization of weights. Indeed we wanted most of the weight to be supported be the main structure, and we needed the arm to be as light as possible. Also while constructing the robot, we tried to make it as rigid as possible in order to avoid unwanted vibrations at the end point.

At first we were not using any gears in the first revolute joint, and we had some problems with this joint precision? Therefore we decided to modify our structure and to add this gear transmission.



*Fig[3] : Our Lego Robot*

1. **Modifying C++ code**

In the C++ code, we modified joint definition with type of joint, angle and direction of the joint.

We also defined the kinematics chain. We considered the world frame under the first joint. Then we assigned the other frames in the center of the motors and at the point of the pen for the last position. We used a slide caliper to measure the distances in order to be as precise as possible.

1. **Modifying Python code**

In the python code, we basically defined a CallbackVariant function where our trajectory was defined.

We designed joints scale values according to our gear ratios for each joint.

First we tried to draw straight lines and see how we could tune different parameters so that we get the straightest line possible. We tuned PID using the following method : we set all parameters to 1, then we added some P parameter, then D and finally I. We decided to use different PID values for each motor because we felt like we had more control this way.

We also tried to modify speed of motors and number of points in trajectory, because we find out these parameters also had an impact on our drawing. That’s maybe the part of the assignment that took us the most time, because it was hard to find out what values would be the best for all these parameters by doing guesses.

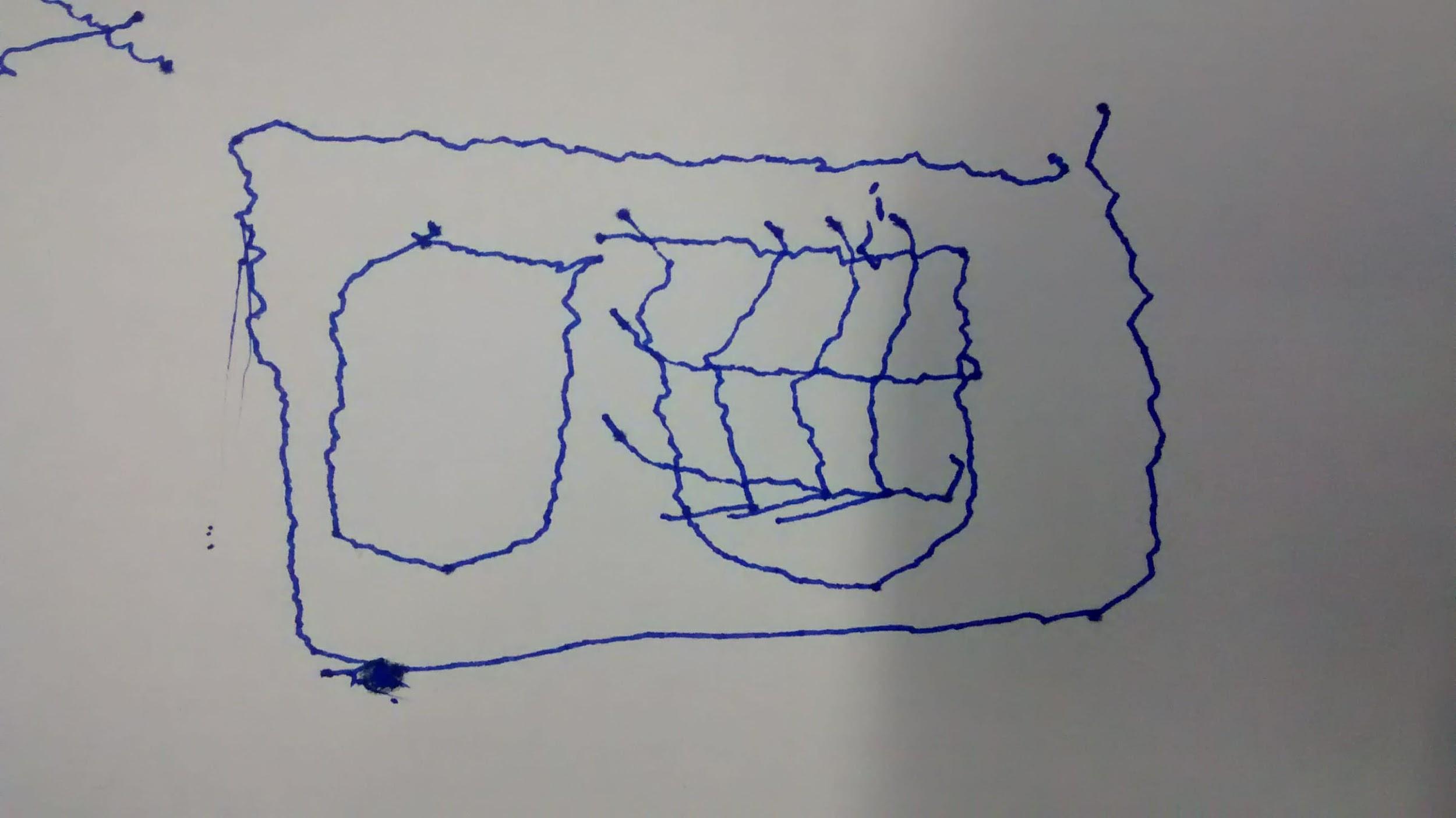
When we were able to draw straight lines, we started to plan a trajectory with which we would be able to draw 3 different mobiles on the same sheet of paper.

1. **MATLAB commands**

We needed to use matlab to command which of the 3 mobiles the robot would have to draw. To do so, we created a “number” topic and publish it in the matlab code, and then subscribe it in the Python code.

1. **Results and limits of our robot**

At the end, our robot is able to draw all the figures. But still, the lines aren’t very straight, especially on one axis.



*Fig [4] : Mobile Variant 1*

It feels like if our robot was bigger, we would have been able to draw bigger figures and then these oscillations on the lines would be relatively less important. Moreover maybe adding a gear transmission for second joint would have enable us to achieve more precision. Also the arm of the robot was vibrating a lot, which might be the reason why our lines weren’t so straight. But it was complicated to combine that with our need to keep the arm of the robot as light as possible.