Instructions to setting up the

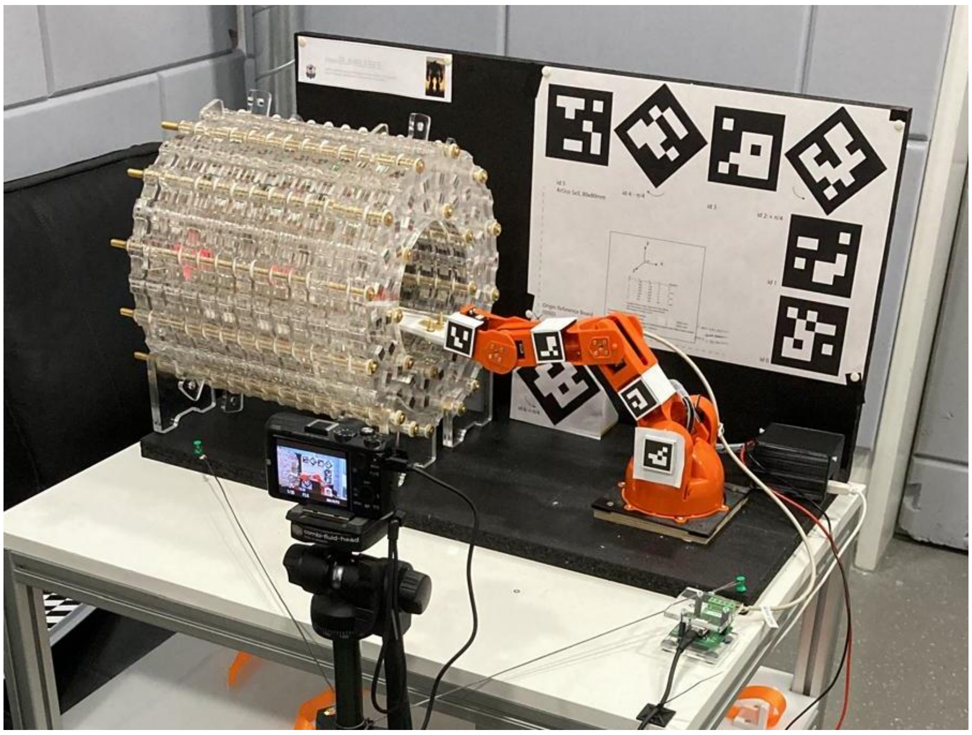
Robot

Project

Easy Scalable, Low-Cost Open Source Magnetic Field Detection System for Evaluating Low-Field MRI Magnets using a Motion Tracked Robot

Pavel Povolni

October 2024



Content

1 Preamble

These instructions describe the setup, start-up and calibration of the robotic arm.
Further information can be found in the publication, as well as in the Appendix and Supplementary Information referenced there.
Software used:

1. PCB design: KiCad 7 (Free)
2. Mechanical design (CAD): Autodesk Inventor 2022 (Commercial). All parts exported as .Steps
3. Embedded software: Arduino IDE 2.2.1 (free)
4. Calibration: Matlab 2021b (w/o expensive toolboxes)

If you have any further questions, please contact us!

2 Mechanical Design Robot

You can find the hardware under:

…\3_5DoF_Robot\31_Hardware

The entire mechanism is shown in the following Figure 1:

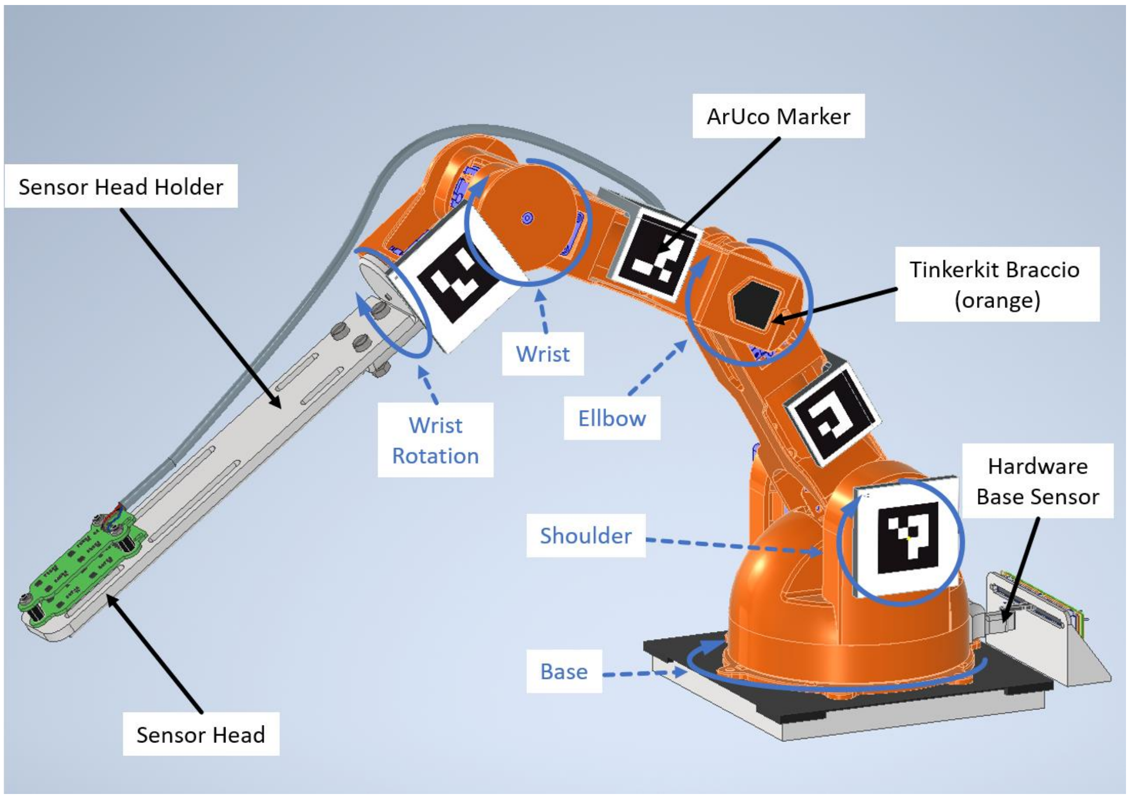


Figure 1: Complete mechanics around the robot (not shown: PowerElectronicDriver housing and motor wiring)

2.1 Design of used Robot (Arduino Tinkerkit Braccio)

We used the Arduino Tinkerkit Braccio (6 Degrees of Freedom) robotic arm for our experiment: https://store.arduino.cc/en-de/products/tinkerkit-braccio-robot

There are a variety of possible robotic arms. Cheap ones start at around €20 (search for "robotic arm" on aliexpress.com, for example) and it seems there is no upper price limit (which of course means that the tolerance of the built-in mechanics is getting better and better).
Depending on how big your budget is, we would actually recommend the Arduino Braccio. Either the commercial product. Or printed out by yourself (CAD is open source). If not, the following mechanics and electronics would have to be adapted. We try to describe the needed changes as good as possible.

2.1.1 Tinkerkit Braccio

The 6th degree of freedom is the gripper (open/close), which was not used in our case. We have therefore removed the gripper. Otherwise there are no mechanical changes to the robot itself.
In the experiments, we found that the shoulder motor (see Figure 2) experiences the highest load and 2 problems occur: Firstly, the motor does not manage to reach all coordinates of the FOV (especially the upper ones) due to limited torque. Secondly, the high torque very easily damages the mechanical mount of the motor axis.
You could install a more powerful servo motor, but this is critical because of the space available and the quality of the mechanical mount.
We have therefore decided to add additional relief in form of a rubber band based pre-tensioner (see Figure 2, no CAD as a simple piece of aluminum with rubber bands is used).

To align the robot basis, a template was printed to hold the star-shaped base. The base plate is a 12mm wooden plate measuring 130x130mm.

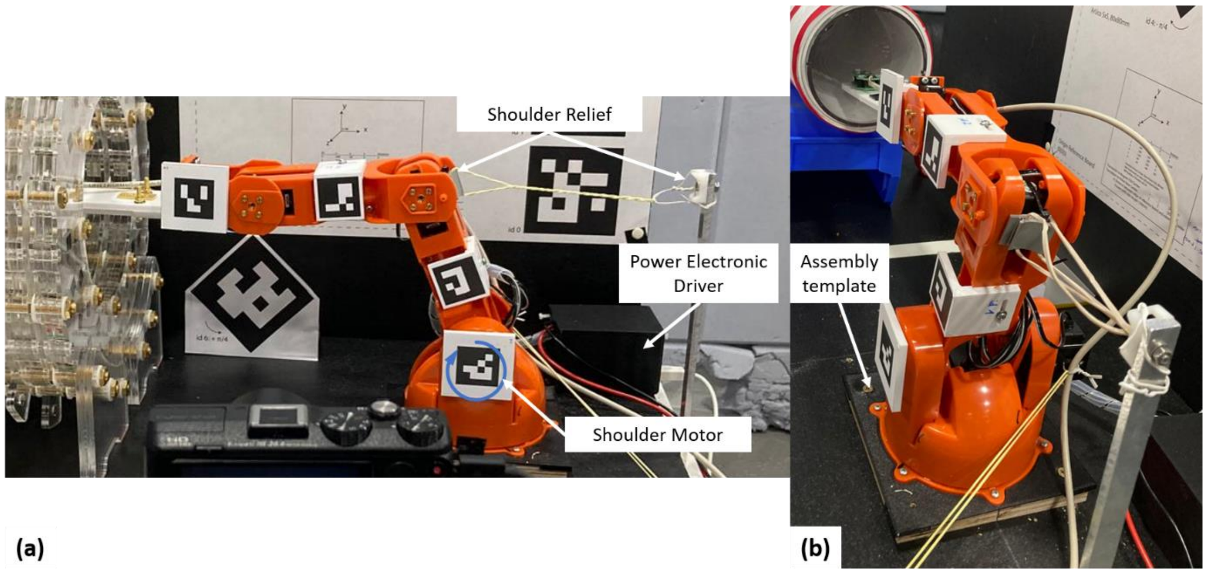


Figure 2: Tinkerkit Braccio with Shoulder Relief (using rubber band)

2.1.2 Sensor Head Holder

The Sensor Head holder consists of two 3D printed parts. The printing material does not matter, but we have had good experiences with PETG.
Depending on what you want to measure, it is advisable to shorten or lengthen this jib.

The holder is mounted on the robot instead of the gripper. This holder is designed mechanically weak on the robot. Take care not to damage anything.

2.1.3 Extract Translation for Motion Tracking

To calculate the kinematics (calculation of the position of the Sensor Head), the exact dimensions of the robot are required (e.g. distance between the markers and the rotation axes).
This is easy to measure in CAD (see Figure 3). However, it seems it’s not possible to export sketches into STEP, so the sketch is only included in the original files.
Autodesk offers a free education license for the used CAD software “Inventor”. Your institute may be covered by this. If not, you will have to redraw it in your program if you want to change anything. If you set up the robot identically, everything is already stored in the software and evaluation.

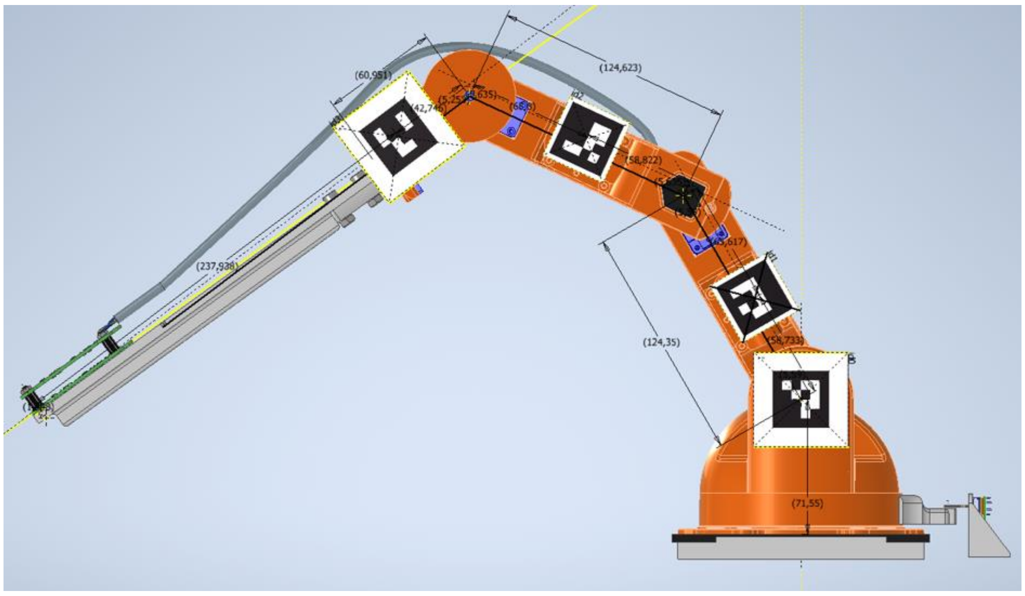


Figure 3: Sketch of exact translations

2.2 Hardware for ArUco Robot Marker

The holders for the ArUco markers on the robot are in the same folder. The holders are 3D printed. The holders for id1 to id4 are designed so that they "automatically" latch into the correct position with the help of lugs. A small self-tapping screw attaches the holder to the robot. The holder for id0 has a bolt on the back that is inserted into a corresponding hole (axis) of the robot (shoulder axis). Push the holder in as far as it will go.
You can find the ArUco markers used under:

...\4_MotionTracking\41_Hardware\411_ArUcoCodes

You can simply print out the markers on paper and gluet them onto the holders. It is important that the orientation of the markers is kept right (see Figure 4. Marker id0 is the wrong in CAD).

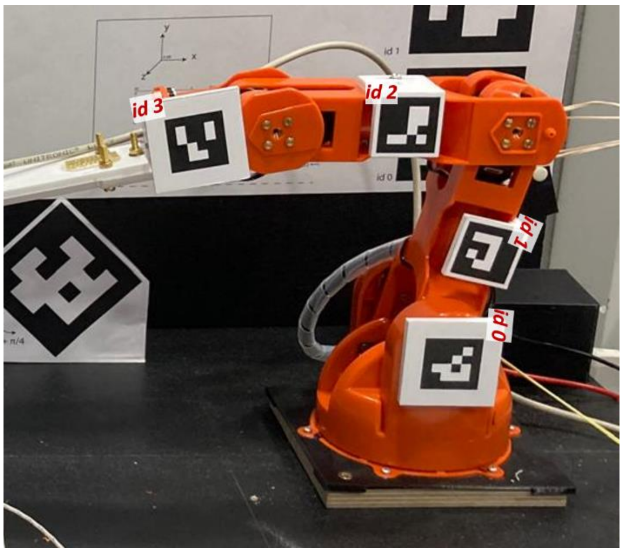


Figure 4 : Orientation of the robot markers

2.3 Hardware for base sensor

To increase Motion Tracking accuracy, an additional sensor is attached to the base (see Figure 5).

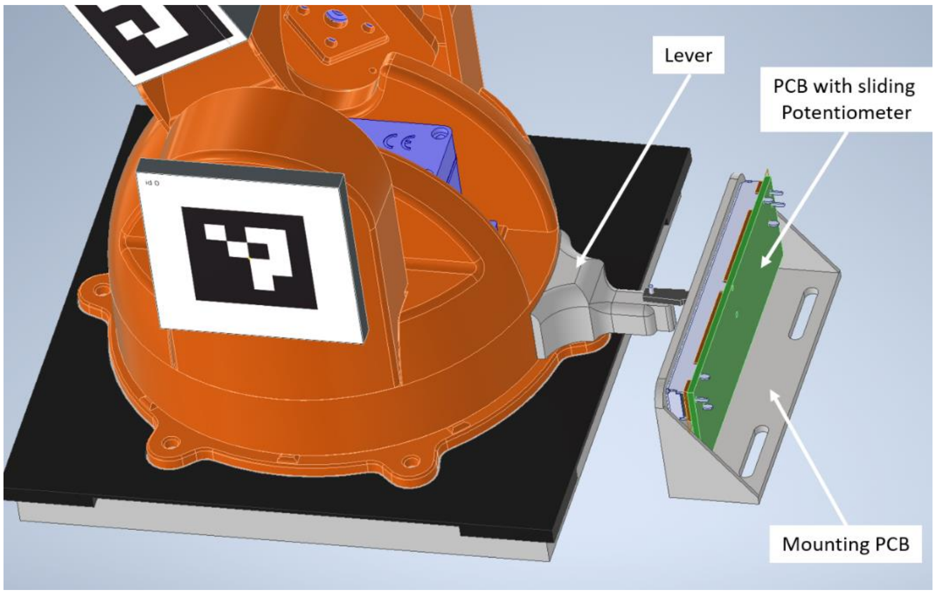


Figure 5 : Base Sensor Parts

This sensor consists of 3 parts:

5. PCB with sliding potentiometer (see 3.2)
6. PCB holder: 3D printing
7. Lever on the robot base: 3D printing

The lever of the potentiometer must be drilled before the circuit board is fitted:
Center, distance 2mm to the front edge, diameter 1mm.
A 1mm pin (e.g. a 1mm-drill) is glued into this hole (connection to the lever). The assembled PCB is attached to the bracket via the mounting points of the potentiometer.

The lever is glued to the base (sand the plastic beforehand). Make sure that the base of the robot is at 0° (neutral) beforehand.
Now clip the PCB incl. holder into the lever via the pin and align the holder parallel to the baseplate of the robot. Make sure that the potentiometer is in the center position (e.g. can be determined using a caliper).

2.4 Environment / Assembly

The three main components (robot, magnet, ArUco reference) were installed in a reference system (see Figure 6) so that the translation between all components is known. The reference system consists of 2 wooden boards (bottom 750x350mm, rear 750x422mm, thickness 22mm), which are screwed at right angles to each other.

This drawing was created in Autodesk Inventor and unfortunately cannot be exported to STEP (see 2.1.3). The original CAD file is available at:

...\3_5DoF_Robot\31_Hardware\313_EnvironmentCAD_AutodeskInventor

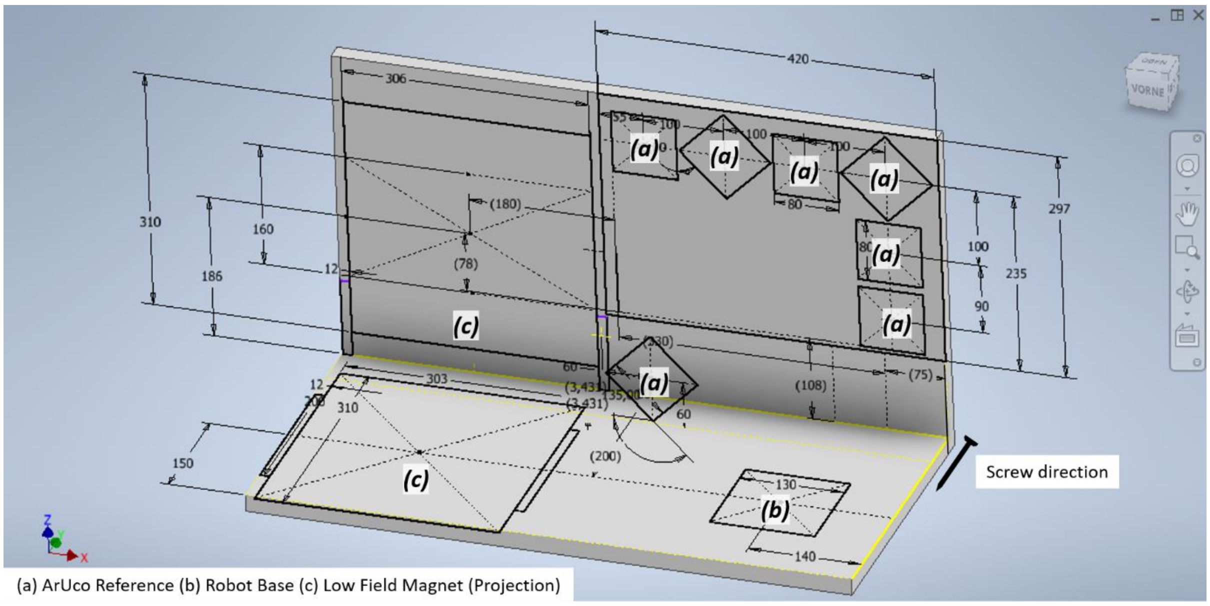


Figure 6 : Setup of the environment with the position of the robot base, the magnet and the ArUco Reference Board (used as a reference for determining the position of the robot markers)

3 Custom Electronics

You can find the PCBs here:

…\3_5DoF_Robot\31_Hardware\315_PCB_ElectronicDrive

2 PCBs are required to operate the robot (see Figure 7). The PCB is two-layered and the design rules are identical to the Hall sensor. To save costs, both PCBs are combined in one layout. Before assembly, the PCB must be sawn apart and separated into 2 PCBs (simultaneous to "1_Hallsensor_manual").

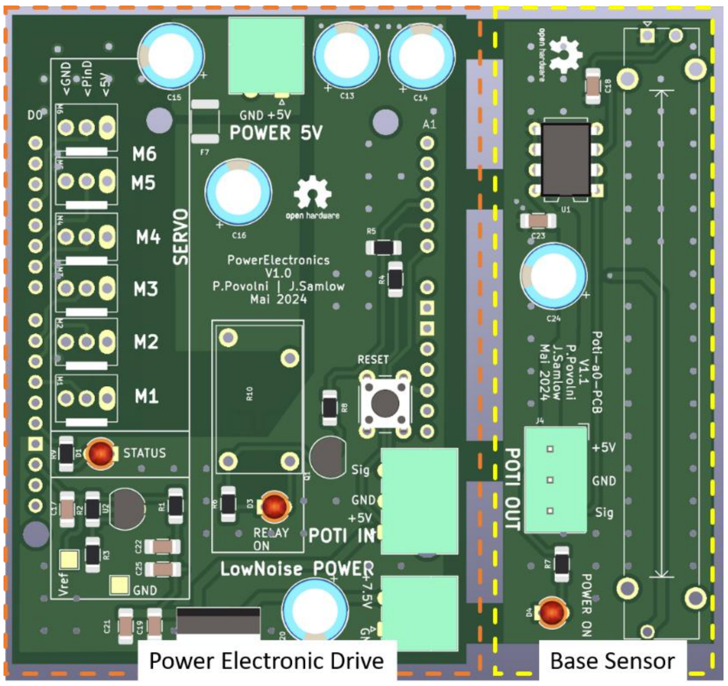


Figure 7 : Common PCB of the Power Electronic and the Base Sensor

3.1 Power Electronic Driver

This PCB is designed as a shield for an Arduino Uno. The Arduino is the controller of the robot (controls the motors, reads the base sensor) and communicates with the central PC on which the measurement is carried out.

This solves a central problem of the Braccio robot: During the boot time of the Arduino, the servo motors vibrate and shake very strongly and destroy the motor mount. Relay R10 therefore disconnects the power supply to the motors during the boot time and only switches it on when the system is in stable operating point.

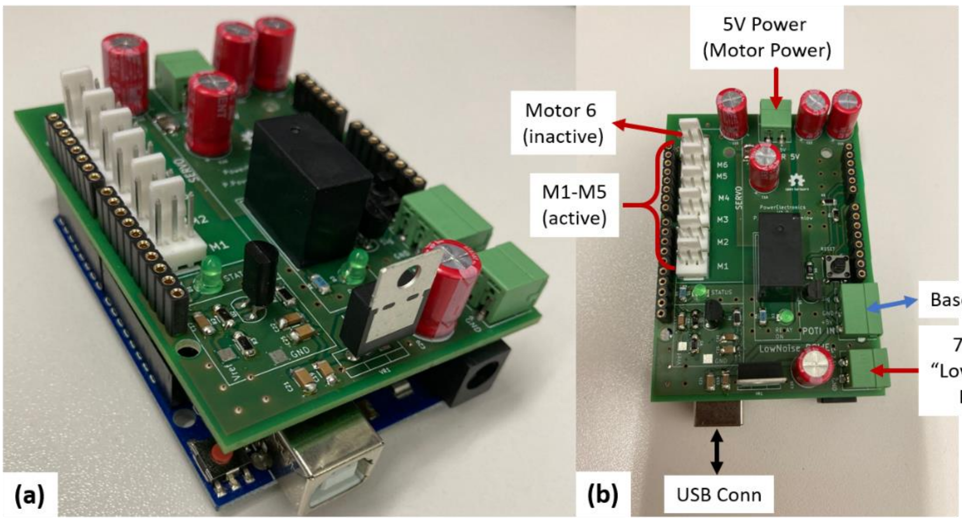


Figure 8: Photos of the assembled PCB with wiring

Soldering the PCB should not be a problem due to the large components used. There is only one special thing on the back that needs to be paid attention to (see Figure 9):

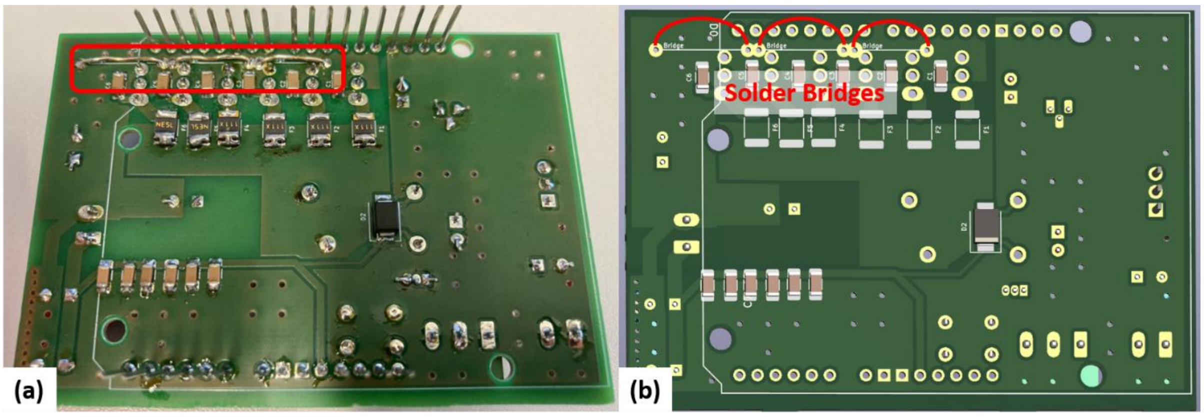


Figure 9: Solder Bridges on Backside of the PCB.

Two voltage sources are required:

1. One 5V | 4 A (voltage source for power electronics and motors)
2. One 7.5-12V | 0.5A "Low Noise" (voltage source for analog electronics). The noise requirements are not particularly high. A normal laboratory power supply works perfectly

This PCB should also work for any other robot arm , which uses servo motors. In this case, it is best to test it with the motors loosely wired (not installed in the arm) to check everything.

The board is plugged onto an Arduino Uno. Any copy of the Arduino Uno (e.g. from Adafruit etc.) works the same way.

3.2 Base sensor

The base sensor PCB board is also very simple and can be soldered in just a few moments. Just make sure that the potentiometer is mounted on the correct side:

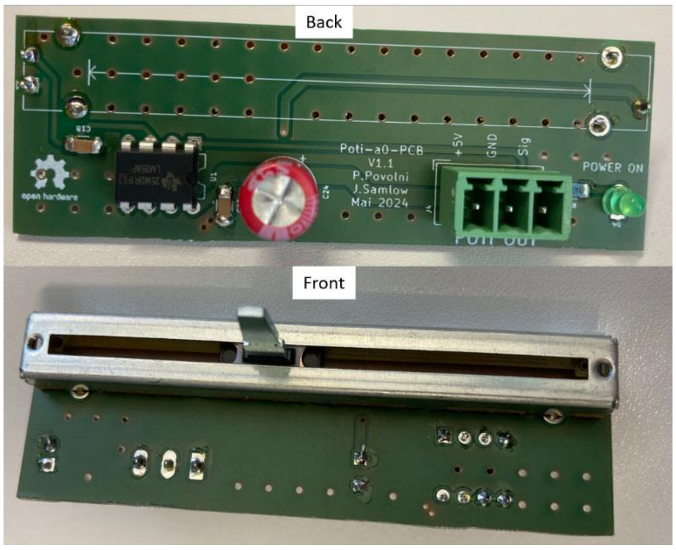


Figure 10: Front & Back of Base Sensor PCB

4 Embedded software

You can find the software here:

…\3_5DoF_Robot\32_Software\321_ArduinoCode

The script is optimized for the robot used. Presumably it works for other robot types in the same way, but only if servo motors are installed there. Other motors (e.g. stepper motors) cannot be controlled.

The software processes the commands received serially from the central computer and uses them to control the motors. In addition, the base sensor is read out and the measured angle is sent back to the computer.

What you need to change is commented in the software. Here are the most important parameters:

1. WRIST_ROTATION:
   Set this rotation so that the Sensor Head is oriented as horizontally as possible. The value is then no longer changed in the measurement
2. Block StartUp position:
   Moving the robot arm into the magnet in the center of the FOV is a very complicated trajectory and has to be simulated extensively (taking into account the mechanical tolerances) so that there is no crash and the sensor is not damaged. In the end, we decided to position the robot manually (motors de-energized) so that the sensor is already close to the center of the FOV. This means that the robot moves to the startup position is small. The startup position is approximately the center of the FOV (does not have to be exact). You can calculate the

necessary angles using the scripts in the "4_Mapping_manual" manual, for example. These angles are saved in the parameters:

   - alpha_Base | alpha_Shoulder | alpha_Ellbow | alpha_Wrist

3. alphaBase_Lin_a1 & alphaBase_Lin_a2:
   Linearization factors from chapter 5.

4.1.1 Library

A self-written library is used for control:

…\3_5DoF_Robot\32_Software\321_ArduinoCode\libs

You must first install this in your environment. You can find the instructions here:

https://docs.arduino.cc/retired/hacking/software/Libraries/

4.1.2 Usage (Serial Commands)

Serial connection with 115200 baud.

To set the motors, the following example serial command is sent from the host PC:

```
+090,+120,+180,+060
```

In this case, the base is rotated to an angle of 90°, the shoulder to 120°, the elbow to 180° and the wrist to 60°.

The Arduino script only checks whether the motors are being operated within their mechanical limits. A crash with the magnet is not checked! Therefore, pay attention to which position is sent.

The base angle is averaged over 10 measurements. The serial response of the script after the robot has moved is:

```
...Braccio Angle. Base = 85.61| Shoulder = +120| Elbow = +180| Wrist = +60
```

Where the base angle is the measured value (2 decimal places). The other angles correspond to the angles from the command of the host PC.

5 Calibration of Base Sensor

An accurate measurement of the base sensor angle is essential for to get the full potential of the Motion Tracking and to produce a good mapping result.
There are 2 possibilities:

1. Very precise measurement of the mechanical lengths of the potentiometer and precise assembly of the PCB-part to the robot
2. Calibration of the sensor

Option 1 is difficult (we tried, but were not satisfied), so we ended up using option 2

5.1.1 SetUp

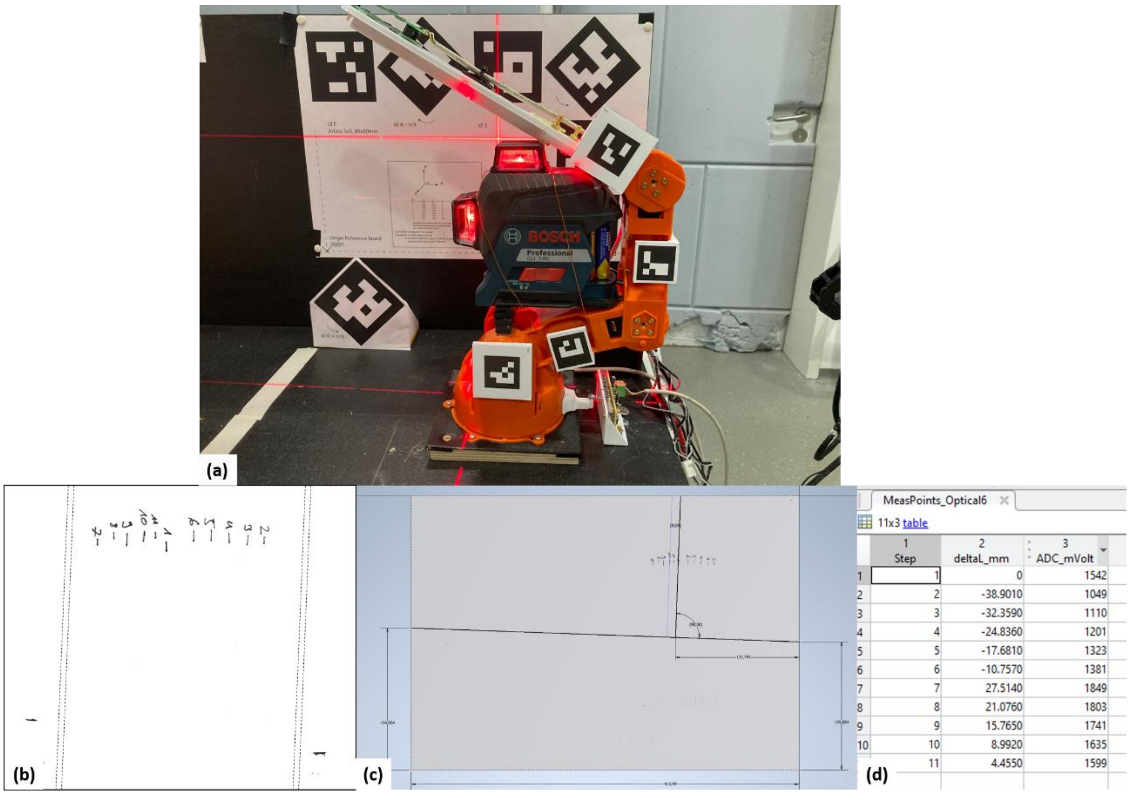
For the calibration, we provisionally attached a 3D rotation laser (self-leveling deactivated) to the robot (see Figure 11). Alternatively, an usual laser point can be used and mounted to the robot. A piece of DinA3 paper is attached at a distance of 60 cm on which the laser beams are projected and the position is marked (see Figure 11b).

The robot controller (see Ch. 4) is programmed with a new Arduino script that makes calibration easier ("Arduino_Robot_CheckAlpha0_withPoti.ino"). The script is similar to the original script. Only the base angle is changed during the measurement. It is programmed in such a way that 10 measuring points in both directions are carried out from a starting point in 1° steps.
Alternatively (as done in our case), the motors can be de-energized (5V supply switched off) and the robot can be turned by hand.
It is important that the base angle set in step 1 (first line) is 0°. The distances of the other measurements are then measured relative to this line.

The measuring sequence is as follows (in our case with de-energized motors):

1. Start Arduino, open a serial window on the PC (Arduino or Putty or Hterm or similar). Baud rate 115200 Bd
2. Move base to 0° position
3. Send a command serially on the PC (text doesn't matter) – this starts the measurement
4. The Arduino measures the base sensor 10 times and outputs raw values serially. We are interested in the part "uout_x = 2.1225V(Raw..."
5. Mark the laser beam on the paper
6. Rotates the base to the next position
7. Continue with point 3 until measurement is complete. We have approached a total of 10 positions

You can either save the serial window and average the voltage measurements for each point afterwards. Or you can calculate the average during the measurement itself. In addition, the horizontal laser beam is marked on the paper (see next chapter).



	1 Step	2 deltaL_mm	3 ADC_mVolt
1	1	0	1542
2	2	-38.9010	1049
3	3	-32.3590	1110
4	4	-24.8360	1201
5	5	-17.6810	1323
6	6	-10.7570	1381
7	7	27.5140	1849
8	8	21.0760	1803
9	9	15.7650	1741
10	10	8.9920	1635
	11	4.4550	1599

Figure 11: Setup for calibrating the base sensor. (a) Mounted 3D laser (b) Scan of the DinA3 sheet on which the laser projection was marked at the different base angles (the scan was scaled down here in the picture). (c)/(d) The image imported into CAD and measured.

5.1.2 Evaluation / Linearization

For linearization, the marked laser projection must be evaluated and the distance between the points measured.

This can be done simply by scanning the sheet and importing it into a CAD program. There, the distance between the markings and point 1 is measured (see Figure 11c). Make sure that the sign is correct:

- e.g. point 2 means a negative base angle
- The measured voltage decreases due to the potentiometer lever
- The sign of the distance is therefore negative

The measured markers and ADC voltages are combined in a table (in Matlab, Figure 11d).

The Matlab script "Evaluation_ArduinoLog.m" can be used to calculate the linear relationship from the measured values.

This will automatically calculate the linearization coefficients that are used during the mapping in the Arduino script (see Ch 4).

6 Power Supply

The robot is supplied with 2 voltage levels:

- 5V|5A, which are used to supply the motors.
- 7...25V|0.5A for the supply of the analog electronics (which are reduced to 5V on the PCB from chapter 3.1 with a “low-noise” L7805 voltage regulator)

In the simplest case, 2 adjustable laboratory power supplies are used for the power supply (in our case). However, fixed voltage sources can also be used without any problems.

The setup is sensitive to power cuts, as the motors in the robot are constantly running and powered. If the voltage drops briefly, the robot arm collapses and could damage the mounted sensor, for example.
If you want to use this setup in an area where the power supply is not guaranteed (e.g. at exhibitions, outdoor, or in areas with an unreliable grid), operation with a battery is possible without any problems. One possible solution is described here:

6.1 Power Supply Using conventional 12V Lead-Acid-Battery

For an exhibition, our setup was operated without access to a power socket.
The designed battery driven power supply is shown schematically in Figure 12. The circuit was built into a casing (see Figure 13 and Figure 14).

Essentially, the setup consists of a 12V battery (e.g. from UPS, motorcycles, cars, RC etc.), a battery guard (protection against deep discharge) and a 5V DCDC. A simple voltmeter was also added to see the charge level.
The used parts used are listed in Table 1.
The 5V|5A input of the robot is connected to the DIN connector (any other connector will do as well).
The 7...12V input is connected directly to the Charge input (to the 4mm banana plugs).

Experience has shown that the robot requires a current of between 400-700mA (depending on position and weight) to hold the position. Theoretically, this can be up to 5A for a short time during movement (limited by the fuses on the power electronics board).
Assuming an average current consumption of 1.5A and a measurement duration of 180 minutes, this results in a energy requirement of 22.5Wh. Assuming a (very poor) efficiency of the DCDC of 80%, this results in 28.1W. Therefore, a lead battery (12V) with a capacity of 3Ah is completely sufficient for a 3h measurement (a 6Ah motorcycle battery is installed in the picture).

The batteries can be charged with a laboratory power supply unit via the “Charge input connectors”.

It is also possible to use the battery only as a buffer (quasi as a UPS):
A laboratory power supply unit is connected to the Charge input (14V, >2A, CC operation if necessary, diode in series to protect against backfeeding). This supply constantly charges the batteries. The robot is supplied via the battery guard and the DCDC.
If the grid power fails and the laboratory power supply unit switches off, the robot is supplied by the battery without any interruption.
The camera we use (see instructions "3_Motion_Tracking_manual") has sufficient battery capacity for a 3h-mapping. It is advisable to use a camera with a USB charging input (power supply via power bank possible).
During mapping (see instructions "4_Mapping_manual"), the computer does not have to calculate anything intensive. A battery-powered laptop should be able to do this without any problems.

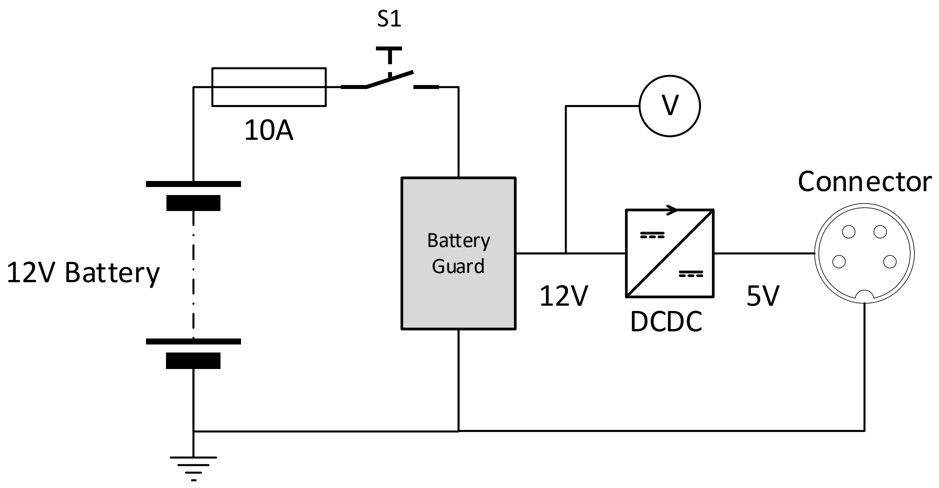


Figure 12: Schematic of the circuit

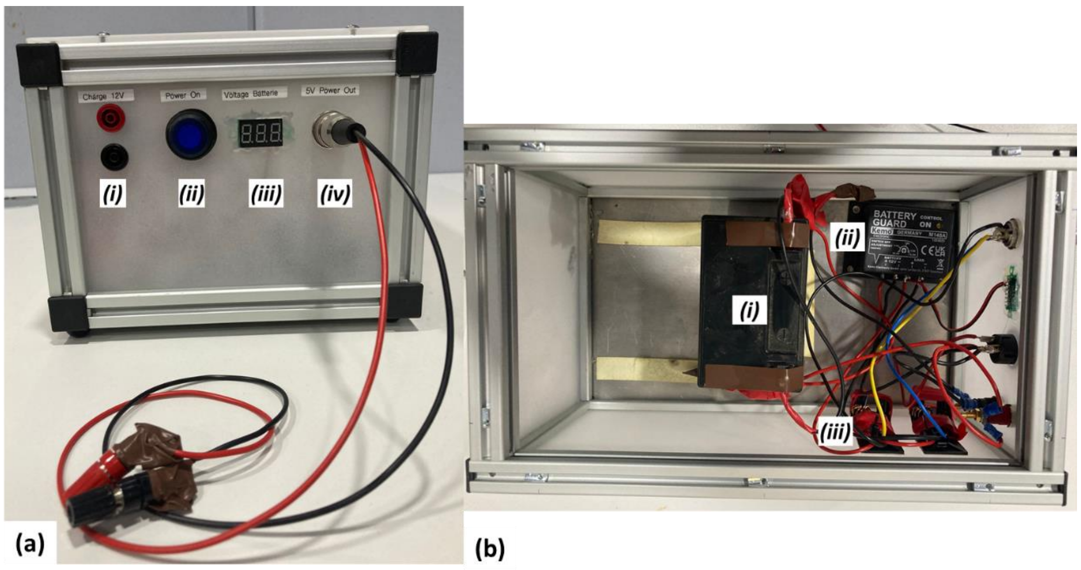


Figure 13: Assembled setup. In our case, built into a housing made of aluminum profiles and acrylic glass. A simpler housing will do it as well. (a) Front side. (i) Charging input, (ii) on/off switch, (iii) voltmeter, (iv) 5V output. (b) Inside. (i) 12V battery, (ii) battery guard, (iii) DCDC.

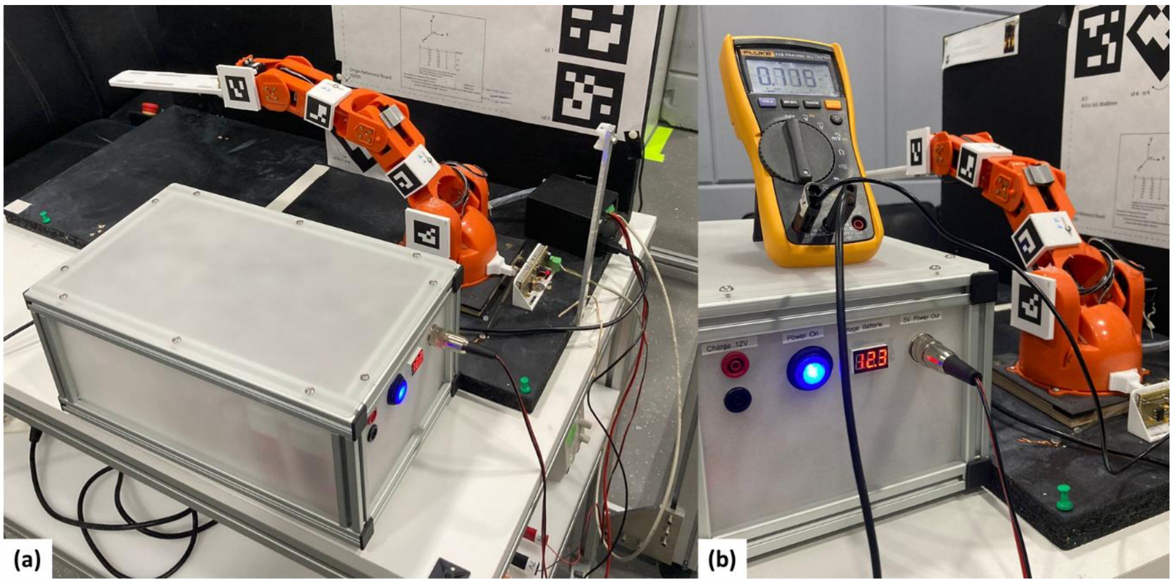


Figure 14: Power supply of the robot via the battery. Robot stands still in one position, approx. 700mA current is required for this.

Table 1: Components used. The exact components are not critical and any other similar component can be used (especially the battery - just take what is available). The order number and link are from the shop Conrad Electronic SE from Germany.

Component	Description	Order Number *	Link*
Voltmeter	JOY-IT VM330	2435975 - 62	Link
Battery Guard	Kemo Electronic M148A	190095 - 62	Link
DCDC	JOY-IT SBC-Buck02 In 9-35V, Out 5V 5A	2582436 - 62	Link
Connector (illuminated)	TRU COMPONENTS 1587920	1587920 - 62	Link
Banana connectors	TRU COMPONENTS 6239C26DN	1565268 - 62	Link
Battery 12V	Yuasa YTB9	3134958 - 62	Link