ElConRoM Documentation

Release 1.0

Andreas Fognini

Jun 29, 2018

CONTENTS

CHAPTER

ONE

INTRODUCTION

ElConRoM stands for **El**ectronically**Con**trolled**Ro**tation**M**ount. The objective of this project was to build an easy to use, accurate, and inexpensive rotation mount for waveplates to be used in quantum tomography experiments. This has been achieved by assembling a hollow core motor from PCBMotors in a 3D printed housing. We used the 22 mm free center motor with 5760 counts per revolution. The motor is controlled with a Python library.

This manual describes in detail how to assemble the 3D printed housing and the code to control the motor.

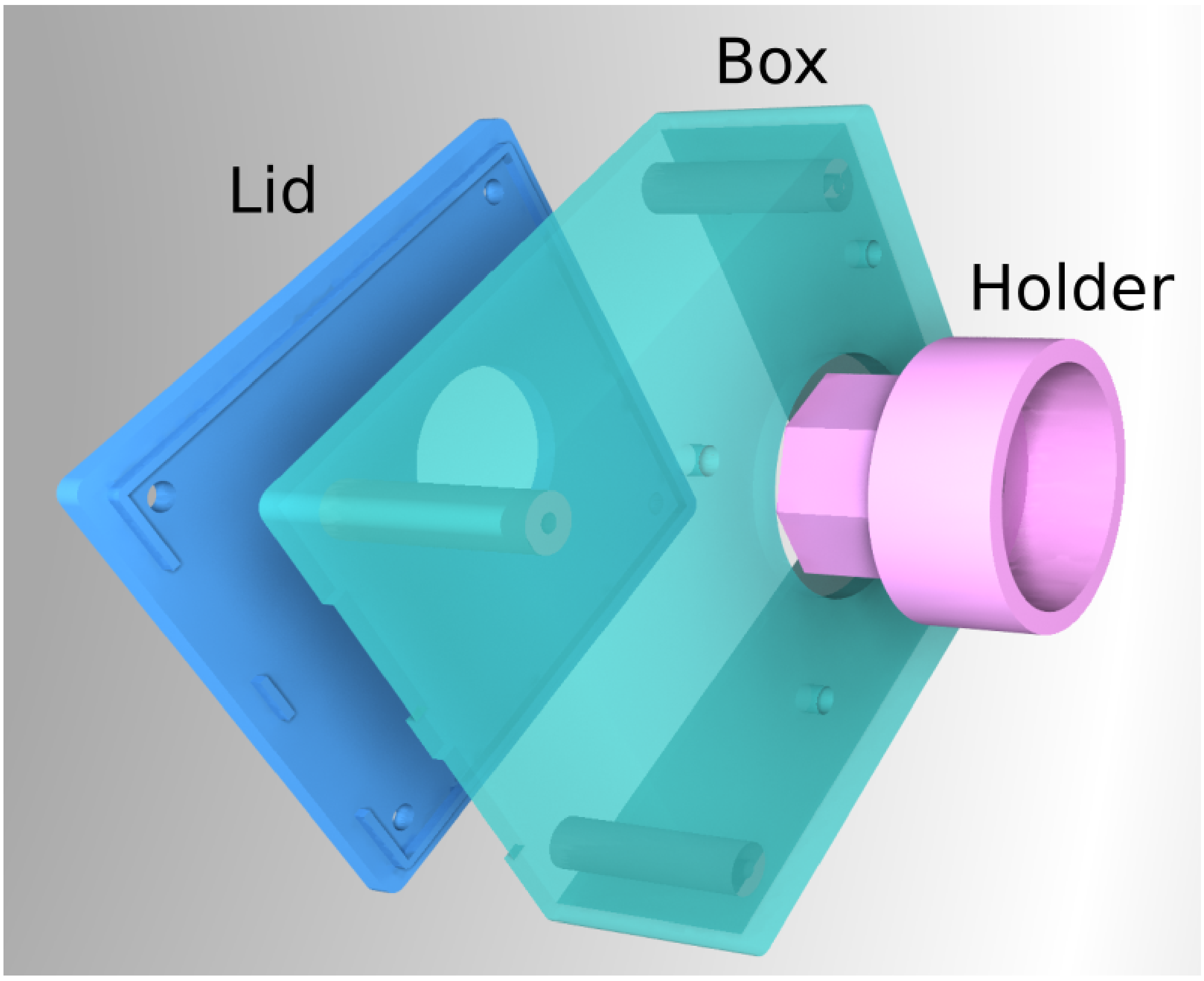

Figure 1.1: Rendered image of the three parts of the housing; the lid, the box, and the holder.

CHAPTER

TWO

MANUFACTURING THE MOTOR HOUSING

Figure 1.1 shows a rendering of the three parts which from the motor housing - the lid, the box, and the waveplate holder.

To allow barrier free access to the CAD file the program FreeCAD has been used to design the parts. All 3D parts were printed in PLA but the choice of material is not critical. To build this project you will need the following tools/parts:

2.1 Tool/Part list

To build the ElConRoM project the following tools/parts are necessary:

- 3D printer or printed parts
- A hollow core motor from www.PCBmotor.com (22 mm free center motor with 5760 counts per revolution)
- cutter knife
- 3 mm tap
- 2.5 mm, 3.5 mm, and a 6.5 mm drill
- three M3 set screws

2.2 The housing

The housing can be made in three easy steps:

- 3D print the lid (/CAD/Lid.stl) and the box (/Cad/Box.stl) and clean the edges with a cutter knife.
- The four stands of the box are drilled with 2.5 mm and tapped to hold M3 screws.
- A hole can be drilled in the side of the box to mount an optical post.

After all that preparation work we are ready to assemble the motor in its new housing. The motor can be mounted easily in the box by removing its four screws and screwing long ones back in through the four holes next to the main optical opening of the box. Then, the lid is mounted and the assembly of the housing is finished.

2.3 The waveplate holder

The waveplate holder or referred later as holder is manufactured in two steps:

- 3D print the holder (/CAD/Holder.stl) with 100% filling and clean the edges with a cutter knife.
- Make three M3 taps radially and equally spaced into the holding ring. These taps will later be used by three setscrews to hold the waveplate in place, compare Figure 2.1 where two of the three setscrews are clearly visible.

The waveplate holder is mounted to the motor by a press fit. To scope with low tolerances from 3D printed parts, the holder which will be pressed into the motor is not round, but has an octagonal shape to have only eight well defined supporting points. Now we will discuss the fitting procedure to prepare the holder for mounting:

With a cutter knife the edges of the octagon are slightly conically reduced. In this way, the waveplate holder can be easily put into the motor and by a light press will hold within the motor's bearing. Care is necessary here to not overbend the springs within the motor. Figure 2.1 shows a picture of the assembled motor housing.

After the waveplate holder is mounted in the motor, the waveplate can be mounted in the holder by adjusting the three setscrews.

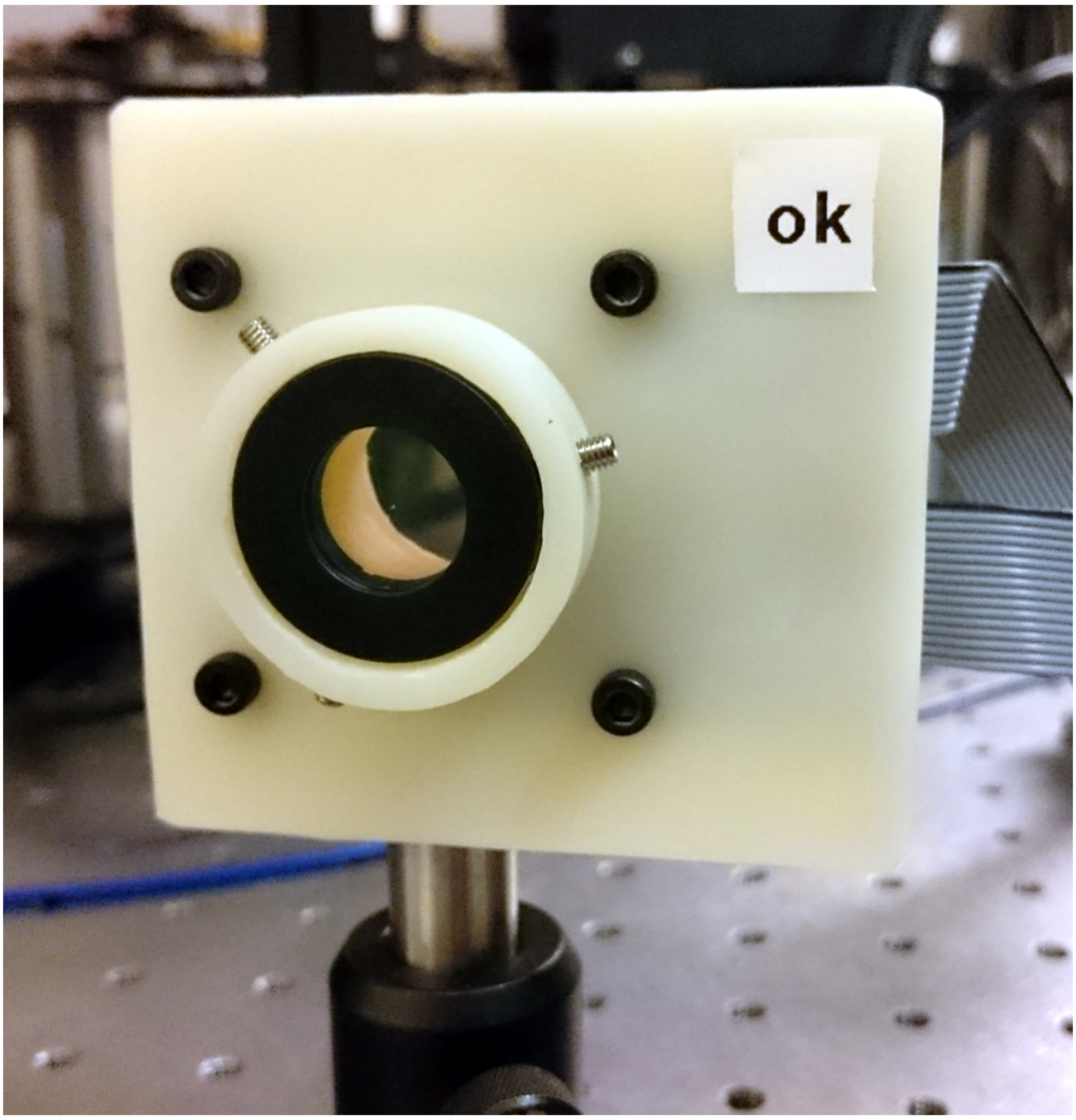

Figure 2.1: A picture of the assembled housing. The holder is equipped with a waveplate. Two of the three setscrews are clearly visible.

CHAPTER

THREE

USAGE OF THE ELCONROM LIBRARY

The usage of the ElConRoM library is outlined in an example of moving motor 1.

```python
from elconrom import ElConRoM
import time

motor = ElConRoM()

#Select motor number 1
motor.number(1)

#Home motor
motor.home()

#Set it to 90 degrees
motor.abs_position_degrees(90)

#Wait one second
time.sleep(1)

#Drive back to zero degrees
motor.abs_position_degrees(0)
```

It can be necessary to reduce the speed to reach more accurate stopping behavior. For example:

```python
motor.speed(70)
```

CHAPTER

FOUR

INSTALLATION

The ElConRoM Python library can be installed by the provided setup script.

You can install it either by:

```
sudo python setup.py install
```

or by:

```
sudo python3 setup.py install
```

depending on your Python installation. Note, ElConRoM needs Python 3 or higher.

You can test if the installation worked by importing the library:

```
import elconrom
```

If that did not give you an error the installation worked successfully.

CHAPTER

FIVE

MODULES DESCRIPTION

In the following the ElConRoM module is described. It consists of one classe:

```
* ELConRoM
```

This class controlles the motor.

5.1 ElConRoM module

class elconrom.**ElConRoM**(*com_port='/dev/ttyUSB0'*)

Bases: `object`

Implements an interface to communicate with the PCBMotor controller.

Parameters **com_port** (*string*) – Serial communcation port path.

abs_position_degrees(*s*)

Move to absolute position in degrees.

Parameters **s** (*float*) – Position in degrees.

abs_position_steps(*s*)

Move to absolute position in units of steps.

Parameters **s** (*float*) – Position in steps. Is rounded to integer while processing.

degrees_to_steps(*d*)

Convert degrees to steps.

Parameters **d** (*float*) – Number of degrees.

Returns Number of steps.

Return type float

home()

Home motor. Homes the motor by finding the home marker. Sets this position to zero.

number(*m=1*)

Select motor to address.

One controller can controll several motors. By stating the number a specific motor is selected. Note: Counting starts at 1.

Parameters **m** (*int*) – The Motor's number which should be controlled.

query_position()

Query position.

The position is not queried from the controller but from bookkeeping of the sent steps. Therefore, not to accurate.

Returns The absolute position in steps.

Rtyep float

`rel_position`(*s*)
Move relative in units of steps.

Parameters **`s`** (*`float`*) – Number of steps. Converted to integer in function.

`send`(*cmd*)
Send the commands to the controller. Sends the commands and receives the controller's reply which is returned.

Parameters **`cmd`** (*`string`*) – Command for the PCBMotors

Returns Reply from the motor controller.

Return type string

`speed`(*scale*)
Set speed.

The speed of clock and counter clockwise rotation is adjusted to the same value.

Parameters **`scale`** (*`int`*) – 0..255

`steps_to_degrees`(*s*)
Convert steps to degrees.

Parameters **`s`** (*`float`*) – Number of steps.

Returns Number of degrees.

Return type float

`verbocity`(*state*)
Set verbocity.

Parameters **`state`** (*`bool`*) – Verbocity active when True.

CHAPTER

SIX

LICENSES

6.1 Source Code

The source code of the ElConRoM library is licensed under GPLv3:

6.2 Documentation

This documentation is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.

INDEX