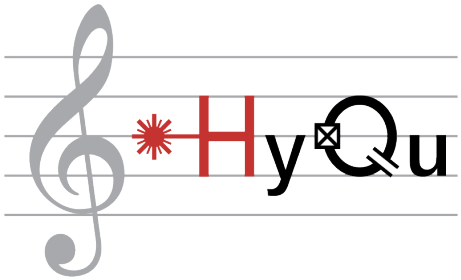
LOCALIZED SOURCE OF QUASIPARTICLES IN SUPERCONDUCTING QUBITS

MASTER THESIS

Giacomo Bisson



Department of Physics, D-PHYS
Hybrid Quantum Systems Group
ETH Zürich

SUPERVISORS:
Prof. Yiwen Chu
Dr. Rodrigo Benevides

September 2022

CONTENTS

1 MOTIVATION & THEORETICAL INTRODUCTION

1.1 MICROWAVE TO OPTICAL TRANSDUCTION

Mechanical resonators can be efficiently coupled to microwave frequency circuits and optical light in the quantum regime. The coupling of several mechanical systems to superconducting (SC) qubits has been demonstrated. These include interactions with propagating surface acoustic waves [1], micromechanical resonators [2] and high-overtone bulk acoustic-wave resonator (HBAR) [3]. Focusing on the high-overtone bulk acoustic-wave resonator, robust strong coupling has been demonstrated between an HBAR and infrared light (IR) in a single mode optical cavity through Brillouin interactions [4]. This makes the HBAR a promising device for transducing quantum information between one of the leading platforms for quantum computing (SC circuit) and a very convenient carrier of quantum information (IR light).
Our approach to building this transducer has been sketched in fig. 1.1.

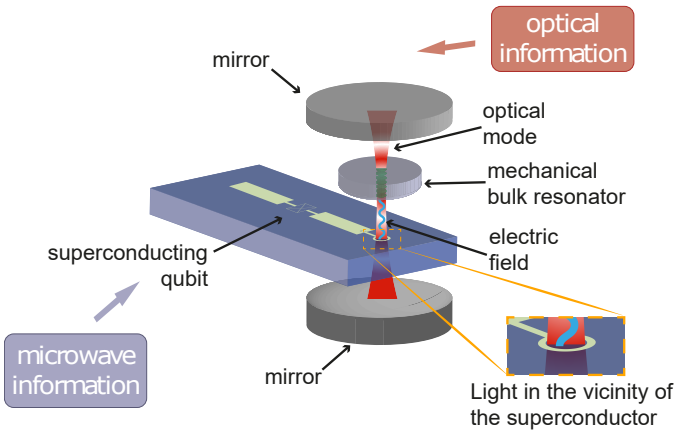


Figure 1.1: Sketch of the microwave to optical transduction device. The superconducting transmon qubit couples to high-overtone bulk acoustic-wave resonator (HBAR) through a piezoelectric film. The HBAR couples to the optical cavity through Brillouin interactions.

The electromagnetic (EM) field in the SC transmon qubit generates phonons in the mechanical bulk resonator through piezoelectricity, either of the HBAR material or using an additional piezoelectric layer on its surface. To increase coupling, the qubit is fabricated with a circular antenna that lies directly below the piezoelectric resonator. The acoustic modes in the mechanical resonator interact with infrared photons in the Fabry-Perot cavity through Brillouin scattering.

An external control laser drives the cavity on resonance. This causes IR light to shine close to the superconducting layer of the qubit antenna. IR photons are absorbed by the Cooper pairs in the superconductor, breaking them and generating quasiparticles; see fig. 1.2. Quasiparticles have many adverse effects on the SC qubit, such as residual excited state population and increased relaxation and decoherence rates [5].

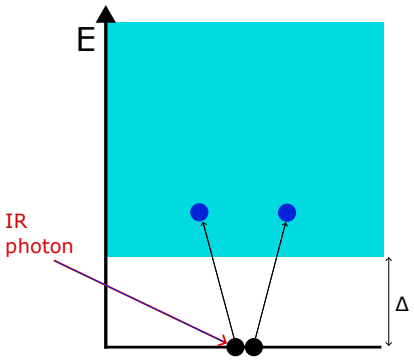


Figure 1.2: Sketch of the band structure of a superconductor. An infrared photon is absorbed by a Cooper pair (in black). This generates two quasiparticles (in blue) because $\hbar\omega > 2\Delta$. In our system, $\hbar\omega \sim 0.8eV$ is the energy of one IR photon and $\Delta \sim 0.2meV$ is the superconducting gap of bulk aluminum.

Before building a working transducer, it is first necessary to investigate the extent of the effects of IR quasiparticles on the superconductor. For this, it is necessary to compare the results of multiple experiments for different laser pulse duration, power, and position on the qubit. The first version of an experimental setup used to explore these features in the group [6] enabled the laser pulse duration and power to be adjusted conveniently. However, the laser beam could only be manually aligned before the experiment outside the dilution refrigerator, making it sub-optimal for comparing experiments for different laser beam positioning. In fact, during the refrigerator cooldown, the setup components contract due to the decreasing temperature, which causes the laser beam to change location. Furthermore, the conditions of the experiments change between cooldowns.

During my master's project, I aimed to extend the previous experimental setup and develop a technique to precisely position the laser beam on the qubit while the setup is mounted in the refrigerator.

1.2 INTRODUCTION TO QUASIPARTICLE DYNAMICS

1.2.1 Quasiparticle Phonon Down-Conversion in Nonequilibrium Superconductor

Once the Cooper pairs in the superconductor absorb a high-energy photon, an energy downconversion process starts.
The energy downconversion occurs in three distinct stages [7]:

- In the first stage, a fast photoelectron of energy E_0 is released. The photoelectrons decay into plasmons very quickly (in the order of fs). Plasmons are unstable and rapidly decay into electron-hole pairs resulting in strongly interacting electrons and holes that thermalize to a characteristic energy E_1.
- In the second stage, the nonequilibrium distribution of electrons and holes lowers its energy to E_2 through electron-phonon scattering. At this stage, a large number of phonons are released.
- Finally, over the third stage, the mixed distribution of quasiparticles and phonons evolves to a quasiparticle distribution centered at the superconducting edge. This third stage lasts much longer than the preceding stages. During the third stage, the quasiparticles can diffuse, tunnel, recombine, and get trapped.

The characteristic energies E_0, E_1, and E_2 depend on the energy of the photon and on the superconductor.
The first two stages are fast. In [7], they calculated that they last less than 2ns combined for aluminum. However, the third stage is longer. During these three phases, the nonequilibrium quasiparticles couple to the qubit degree of freedom and adversely influence it. In the next section, I will describe the main detrimental effects of quasiparticles on a SC transmon qubit.

1.2.2 Qubit Transitions Driven by Quasiparticles

In reference [5], they calculate the transition rates Γ_{if} between the initial and final qubit states $|i\rangle$ –> $|f\rangle$ associated with the tunneling of quasiparticles across the junction of a SC transmon qubit. If the ratio E_J/E_C (E_J and E_C are the Josephson and charging energy of the SC qubit) is smaller than 25, the qubit energy levels depend on the offset charge n_g. See fig. 1.3. Quasiparticles tunneling through the junction cause uncontrollable $\pm 1/2$ jumps in n_g. These jumps cause a series of transitions shown in grey in fig. 1.3. In particular, the rates Γ_{01} and Γ_{10} respectively characterize the transition between the ground and the first excited state ($|0\rangle$ –> $|1\rangle$) and vice versa. On the other hand, transitions that only change the parity of the state contribute to the qubit dephasing. The energy levels of the qubit $\overline{E_1}$ and $\overline{E_0}$ fluctuate due to

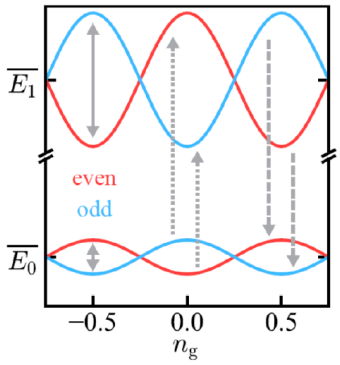


Figure 1.3: Two lowest energy levels of the transmon qubit. The "even" and "odd" labels mark states of opposite charge parity. From [5].

the quasiparticles n_g jumps and therefore the frequency ω_{10} fluctuates by some $\delta\omega = (\delta E_1 + \delta E_0)/\hbar$. These transitions are characterized by the parity switching rates (Γ_{00} and Γ_{11}).
In [5], they calculate all four rates under the assumption that the quasiparticle distribution is described by an effective temperature $T_{eff} << \Delta$ (where Δ is the superconducting gap) and the dimensionless density $x_{qp} << 1$[1]:

$$\Gamma_{10} \approx \frac{16E_J}{\hbar\pi}\sqrt{\frac{E_C}{8E_J}}\sqrt{\frac{\Delta}{2\pi\hbar\omega_{10}}}x_{qp} \tag{1.1}$$

$$\Gamma_{01} = \Gamma_{10} \cdot e^{-\hbar\omega_{10}/T_{eff}} \tag{1.2}$$

$$\Gamma_{00} \approx \Gamma_{11} \approx \frac{16E_J}{\hbar\pi}\sqrt{\frac{T_{eff}}{2\pi\Delta}}x_{qp} \tag{1.3}$$

The rates Γ_{01} and Γ_{10} contribute to the total qubit relaxation rate $1/T_1$. Γ_{11} and Γ_{00} contribute to the total qubit decoherence rate $1/T_2$. Consequently the quasiparticle density x_{qp} can be estimated by measuring T_1 and T_2 of the qubit. In our experiment, only the effects of the rates Γ_{01} and Γ_{10} on the qubit's T_1 can be observed because E_J/E_C is bigger than 25. The dependence of the qubit frequency by the state parity is suppressed.

1.3 MEASUREMENT SEQUENCE

A specific measurement sequence is used to measure the dependence of the qubit's T_1 on the IR generated quasiparticles. This sequence requires multiple measurement steps to be executed after each other.

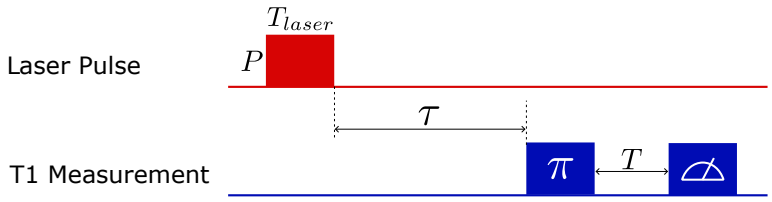


Figure 1.4: Timing diagram of one step of the measurement sequence. The qubit operations are depicted in blue, while the laser operations in red.

One of these steps follows the timing diagram in fig. 1.4. It starts with a laser pulse of power P and length T_{laser} being sent on the qubit surface. Then, after a waiting time τ, it follows the typical qubit T_1 measurement scheme. The qubit T_1 measurement consists of a π pulse that excites the qubit to $|1\rangle$, and after a time T a qubit measurement in the basis $\{|0\rangle, |1\rangle\}$. For one T_1 measurement, this needs to be repeated multiple times with a varying time T.
The sequence described above is repeated multiple times for different τ. The goal is to measure T_1 as a function of τ. Comparing $T_1(\tau)$ to T_1 measured without the laser pulse gives information about the timescale of the quasiparticle dynamics generated by the IR laser photons.

1 The dimensionless quasiparticles density x_{qp} is the density of quasiparticles normalized by the density of Cooper pairs $x_{qp} = n_{qp}/n_{CP}$.

2 IMPROVEMENTS TO THE EXPERIMENTAL SETUP

In the measurement sequence described in 1.3, the laser beam is shone on different parts of the qubit. Many elements of the setup need to be aligned for this to happen. The experimental setup initially used to

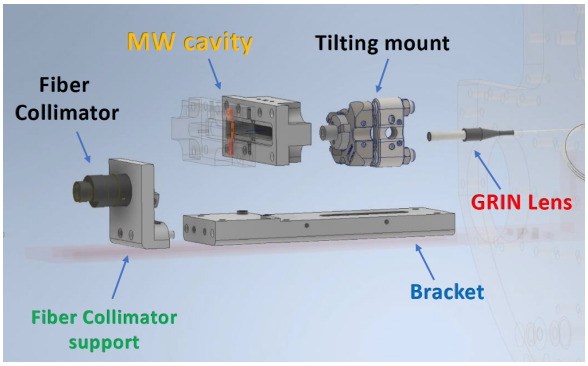


Figure 2.1: Exploded CAD models of the initial experimental setup designed by Francesco during his master thesis [6].

investigate the effects of shining infrared light on a superconducting qubit was built by Francesco Adinolfi during his master thesis [6]. Fig. 2.1 shows the exploded CAD models of the setup components. The SC transmon qubit is mounted in a 3D aluminum cavity in the center of the setup. An input laser beam is emitted by a GRIN (Gradient-index) lens. The laser beam enters the cavity through a small hole and shines on the qubit. It then exits the cavity via an output hole and is collected by a fiber collimator.
Furthermore, Francesco also assembled the microwave components that allow performing control and readout of the qubit and the electronic and optical components for the laser control. See [6] for more information about the experimental setup.
This setup has some limitations that make the alignment of the optical components challenging:

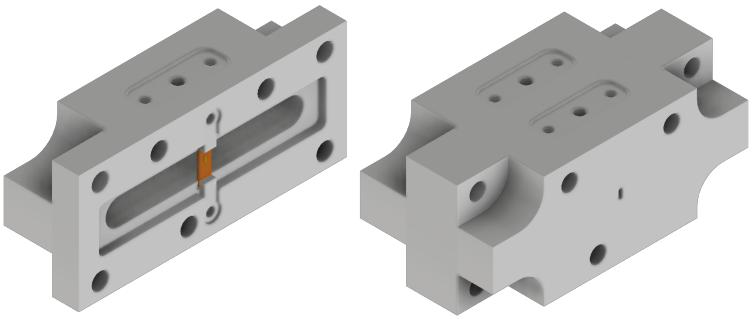
- The hole in the cavity where the laser beam passes through is too limited in size. Therefore, it does not allow to reach all the parts of the qubit.
- The alignment of the input laser beam is done manually by adjusting a tilting mount. Therefore, it can only be aligned before performing the experiment at room temperature on the optical table.

- It is not possible to align the output fiber collimator with the rest of the optics. The collimator is screwed in place and can not be moved.

The first goal of this project was to improve these aspects of the experimental setup.
In this chapter, I will go through the modifications implemented on the setup to make it simpler to align the components.

2.1 THE HOLE IN THE QUBIT CAVITY

The superconductive transom qubit is enclosed in a 3D aluminum cavity. Sending electromagnetic pulses to the cavity allows to control and measure the qubit, as the qubit is capacitively coupled to the cavity [8]. The internal shape of the cavity is such that the resonance frequency of the primary mode of the electromagnetic field confined in it is around 9.3GHz.
In our experiment, compared to a typical microwave cavity, we added input and output holes aligned with the qubit. This allows the laser beam to enter from the input hole, shine on the qubit chip, and exit from the output hole. In fig. 2.2, you can see the 3D models of the left and right cavity pieces and the qubit chip used in this experiment. The qubit is fixed in the center of the left piece with two clamps.



(a) View of the open cavity. **(b)** View of the closed cavity.

Figure 2.2: Inventor 3D models of the aluminum cavity, in grey and the qubit chip, in orange.

As written above, the cavity holes in the previous setup were small. For this reason, aligning the laser beam with specific qubit features was challenging. For example, with the antenna or with the pads.
The first improvement was to increase the dimensions of these holes. Before making this, two different simulations were performed. First, the quality factor of the cavity was computed. The quality factor quantifies how fast the EM energy leaks out of the cavity. If the Q-factor is low, energy leaks out of the cavity faster; consequently, the

qubit coupled to the cavity can also lose energy. [8]. If the holes are too big, the electromagnetic field mode can escape the cavity, reducing the cavity's quality factor. Second, after selecting a hole size that has a negligible impact on the Q-factor, the amount of qubit area the laser beam can cover was computed.

2.1.1 COMSOL Simulation

The software that we chose for the simulation is COMSOL Multiphysics. COMSOL Multiphysics is a simulation platform that provides fully coupled multiphysics and single-physics modeling capabilities.

This software can perform both the Q-factor and the laser beam coverage simulations. Furthermore, it allows the computation of results for multiple hole sizes simultaneously by performing a parameter sweep on the dimensions of the holes. The hole geometry is the same for input and output holes and is very simple. In fig. 2.3, you can see that it is only composed of two circles and a rectangle. The hole has a rectangular shape in the middle because this is approximately the qubit shape. The top and bottom circles make it easy to drill it with a CNC machine.
Thus, the hole size is defined by just two parameters, D and L in fig. 2.3. This makes it convenient to perform a parametric sweep in the COMSOL simulation.

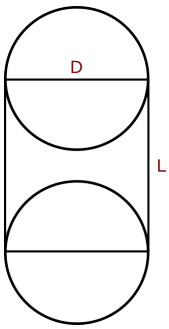


Figure 2.3: Geometry of the cavity holes. The two parameters that determine the hole size are the hole diameter, D in the figure, and hole length, L in the figure.

QUALITY FACTOR: To simulate the electromagnetic field and compute the quality factor, I followed the procedure described in the application *Computing Q-Factors and Resonant Frequencies of Cavity Resonators* [9] from COMSOL. This application guides on how to compute a cavity's Q-factor and resonance frequency.
In [9], the Q-factor of a the EM field is defined as:

$$Q_{fact} = \frac{average\ energy\ stored}{average\ energy\ dissipated} \tag{2.1}$$

The COMSOL *RF Module User's Guide* [10] provides more information about how the Q-factor is calculated in the simulation's eigenfrequency analysis.
The Electric field in the cavity can be written as:

$$\mathbf{E}(\mathbf{r}, t) = Re(\tilde{\mathbf{E}}(\mathbf{r}) \cdot e^{-\lambda t}) \tag{2.2}$$

The eigenvalue $\lambda = \delta - j\omega$ is complex because the electric field can exit the cavity from the holes. The imaginary part ω represents the eigenfrequency, and the real part δ is responsible for the damping. The Q-factor is calculated in terms of δ and ω as:

$$Q_{fact} = \frac{\omega}{2|\delta|} \tag{2.3}$$

To compute the Q-factor of our cavity, the simulation was set up in the following way:

- The mould of the inside of the cavity was used instead of the whole cavity. The hole size was parameterized by its dimensions (D and L, see fig. 2.4). This does not change the results and reduces the computation time because the simulation domain is smaller than if the whole cavity was kept.

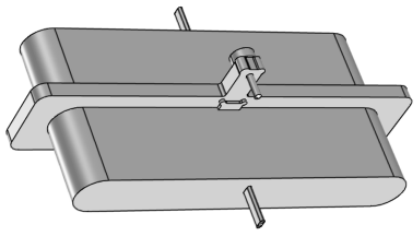
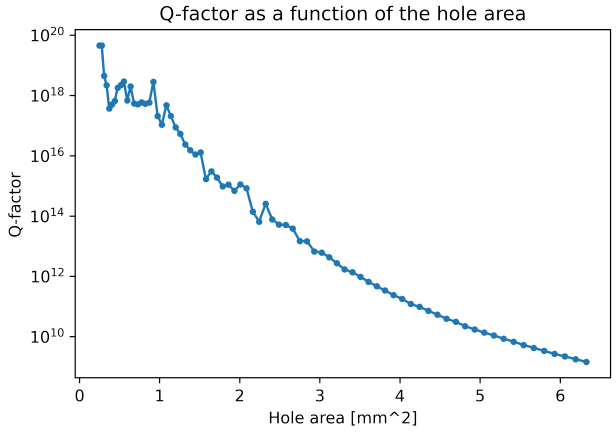
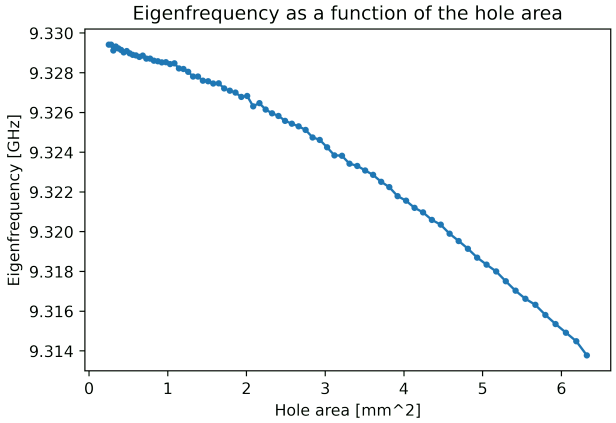


Figure 2.4: Mould of the inside of the cavity for the COMSOL simulation.

- The surface of the mould, except for the hole's surface, was set to *Perfect Electric Conductor* (*PEC*). Our cavity is made of aluminum, with a superconductive critical temperature of around 1.2K [11]. The experiment's temperature is around 8mK, so defining the cavity surface as *PEC* in COMSOL is a good approximation.
- A second-order scattering condition was set to the external hole surface. This accounts for the EM field passing through the hole without reflections. Choosing second-order scattering conditions instead of first-order is advantageous because the reflection is lower [12].
- After some tests, the mesh size was set to the COMSOL setting "normal". This provides a good trade-off between simulation speed and precision in the result.



(a) Log scale plot of the Q-factor.



(b) Plot of the eigenfrequency of the main EM mode.

Figure 2.5: Plots of the computed (a) Q-factor and (b) eigenfrequency from the simulation in COMSOL with a parameter sweeps on the cavity hole size.

A simulation was performed for every couple of D and L in the range from (L=0.3mm, D=0.6mm) to (L=1.8mm, D=2.1mm), with a step size of 0.02mm. Which means a total of 75 couples. The computed Q-factors and the eigenfrequencies of the main EM field mode as a function of the hole area [1] are plotted in fig. 2.5. The Q-factor decreases exponentially fast as a function of the hole area. However, the hole is not the only source of losses in the cavity. If there are multiple sources of losses, the total quality factor of a resonator is computed as [8]:

$$Q_{tot} = (\sum_i 1/Q_i)^{-1} \tag{2.4}$$

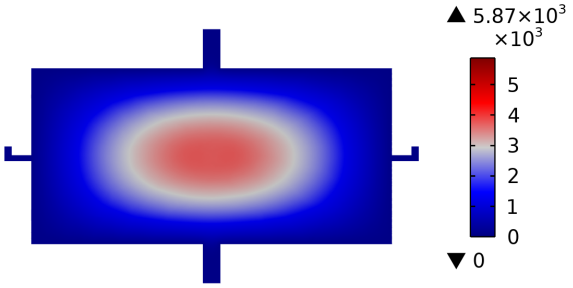
Where Q_i is the quality factor due to the i^{th} loss source.
Rewriting eq. 2.4 and calling Q_{old} the total Q-factor of the cavity

1 $Hole\ area = \pi \cdot (\frac{D}{2})^2 + (D \cdot L)$

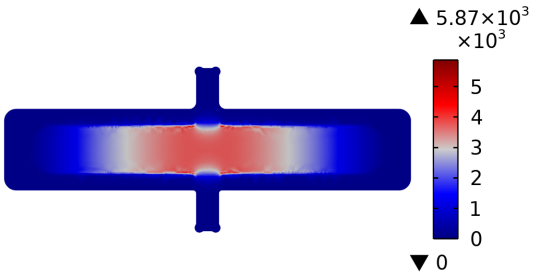
without the hole and Q_{hole} the total Q-factor of the cavity with the hole as the only loss source:

$$Q_{tot} = \left(\frac{Q_{old} + Q_{hole}}{Q_{old} \cdot Q_{hole}}\right)^{-1} \tag{2.5}$$

if $Q_{hole} >> Q_{old}$ then $Q_{tot} \approx Q_{old}$. This assumption holds in our case. Typical experimentally observed Q-factors of our aluminum cavities without a hole are between 10^4 and 10^5. In this case, for the whole range of the hole sizes, the simulated Q-factors Q_{hole} is at least three orders of magnitude bigger than typical Q_{old}. It can be concluded that losses due to the hole in the cavity are negligible compared to other loss sources.
Regarding the eigenfrequencies in fig. 2.5b, the change for the whole range is of around 15 MHz. The cavity resonance frequency is only required to have a frequency which is far from the qubit, which in our case is between 4 to 7 GHz. In conclusion, we could choose all the simulated hole dimensions without degrading the experiment results.



(a) Plot of the EM field norm on a plane that is parallel to the cavity hole's axis.



(b) Plot of the EM field norm on a plane that is normal to the cavity hole's axis.

Figure 2.6: Plot of the EM filed norm in the 3D cavity (a) on a plane parallel to the hole's axis and (b) on a plane perpendicular to the hole's axis. From the COMSOL simulation.

An intuitive explanation of why the EM field losses are influenced so little by the hole in the cavity can be given by looking at the shape of the mode in the cavity resonator. As you can see in fig. 2.6, the shape of the cavity is such that the EM field is confined to the center of the cavity, and very little of it can escape through the holes.
Furthermore, the cavity holes form a waveguide. The cutoff frequency (i.e. the lowest frequency for which a mode will propagate in it) of a rectangular waveguide is:

$$\omega_c = c\sqrt{(\frac{n\pi}{a})^2 + (\frac{m\pi}{b})^2} \tag{2.6}$$

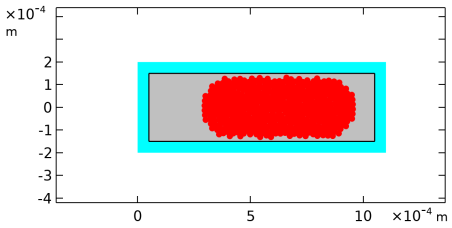
where n, m > 0 are integers indicating the mode number, a and b are the rectangle length and width and c is the speed of light. EM fields with a frequency below ω_c are exponentially suppressed in the waveguide. If the holes of our cavity are approximated by a rectangle the EM modes have a cutoff frequency $\omega_c > c\frac{\pi}{L+D}$, which is $\sim 242\ GHz$ for the biggest simulated hole size. This can explain why the EM energy does not leak much outside of the cavity.

AREA OF THE QUBIT ACCESSIBLE BY THE LASER BEAM: This simulation aims to compute the spot diagram at the qubit location to get an idea of how much of the qubit area can be covered by the laser beam for specific hole sizes.
For this purpose, the Ray optics module is utilized in COMSOL. The full cavity with qubit is imported from Inventor, like the one in fig. 2.2. The light rays are released from a cone with a large enough angle to cover both holes. This cone represents the freedom of movement in the laser beam angle; more on this in the next section. The release cone is located on the same axis as the holes. Furthermore, the distance of the emitted rays from the qubit has been set to 16.2 mm. This corresponds to the focal length of the GRIN lens used in the setup; see section 2.4. Only the light rays that pass through both holes are kept in the spot diagram. This accounts for the fact that the collimator needs to collect the laser light for the qubit imaging algorithm to work; see chapter 3.

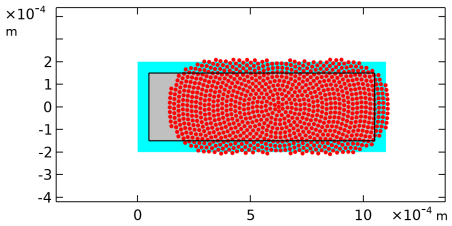
	Hole Dimensions	Area	Quality Factor
1	$D = 0.5\ mm$, $L = 0.8\ mm$	$0.60\ mm^3$	$\sim 6.9E17$
2	$D = 0.8\ mm$, $L = 1.1\ mm$	$1.38\ mm^3$	$\sim 1.5E16$
3	$D = 1.2\ mm$, $L = 1.5\ mm$	$2.93\ mm^3$	$\sim 6.7E12$

Table 2.1: Selected hole dimensions in the COMSOL ray optics simulation.

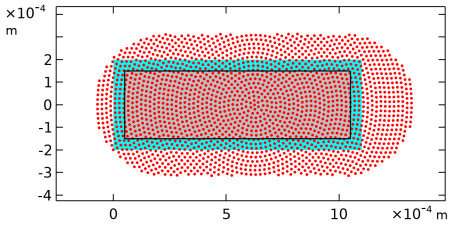
Three different hole dimensions were chosen for this simulation, which are listed in table 2.1. The expression *"comp1.gop.fs==4"* is added in the filters of the spot diagram to select only the light rays that go through both holes. In the fig. 2.7 you can see the three simulated spot diagrams:



(a) Hole size: $D = 0.5\ mm$, $L = 0.8\ mm$.



(b) Hole size: $D = 0.8\ mm$, $L = 1.1\ mm$.



(c) Hole size: $D = 1.2mm$, $L = 1.5mm$.

Figure 2.7: The light rays that pass through both cavity holes generate spot diagrams on the qubit surface. The spot diagrams have been computed for three different hole dimensions. The red dots are the light rays. The gray area is approximately where the qubit is situated. Finally, the cyan area around the qubit represents the imprecision in the position of the qubit due to the manufacturing tolerances. The plots are obtained from the COMSOL ray optics simulation.

- For the old hole size ($D = 0.5\ mm$, $L = 0.8\ mm$) the laser beam is able to only cover a small portion of the qubit. Fig. 2.7a.
- The hole dimensions $D = 0.8\ mm$ and $L = 1.1\ mm$ allow to shine the beam on most of the qubit area. Fig. 2.7b.
- The biggest simulated hole ($D = 1.2\ mm$, $L = 1.5\ mm$) would enable to shine the laser beam everywhere on the qubit area. Fig. 2.7c.

Based on the simulation results showed in this section, we opted for a hole's size of $D = 0.8\ mm$, $L = 1.1\ mm$ for the following reason. As you can see from table 2.1, there is a difference of almost five orders of magnitude between the Q-factor for these hole dimensions and the bigger one. This should not matter, based on equation 2.1.1.

However, these hole dimensions are big enough and we decided to be conservative, in case there are other factors we did not consider.

2.2 ADDING THE JPE ACTUATOR STAGE

One of the major improvements in the new setup is the ability to move the laser beam while the setup is mounted in the dilution refrigerator. This is possible thanks to the *CRYO TIP-TILT-PISTON STAGE (CTTPS1/2)* system from *JPE* [13].
This section illustrates how we modified the old setup to include the *CTTPS1/2* system.
Fig. 2.8 shows a 3D model of the *CTTPS1/2* system. The device mainly

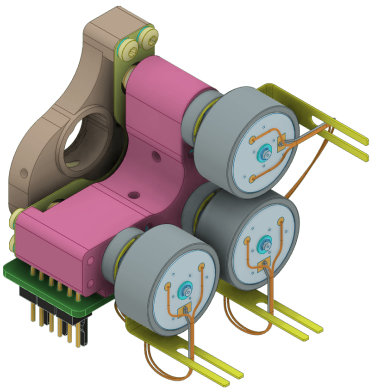


Figure 2.8: 3D model of the JPE CTTPS1/2 system. From JPE's website.

comprises a front metal plate that houses the optics (in our case, the GRIN lens holder that emits the laser beam), and three cryo piezo actuators (*CLA2201*) that can individually push on the three corners of the metal plate. Because the front metal plate is held in place by a metal membrane, it can be tilted and moved back and forth.
Another component of the JPE stage is the small PCB (in green in fig. 2.8), where the cables of the piezo actuators are soldered to three male Molex 2-pin connectors.

2.2.1 JPEs Bracket

I used the CAD software *Autodesk Inventor 2021* for designing a new bracket to mount the *CTTPS1/2* with the other components of the setup. Then, the D-PHYS Mechanical Workshop used aluminum to manufacture the bracket with a CNC machine based on the drawings and the *.step* files. In fig. 2.9, you can see the 3D model of the bracket. The main features that have been included are:

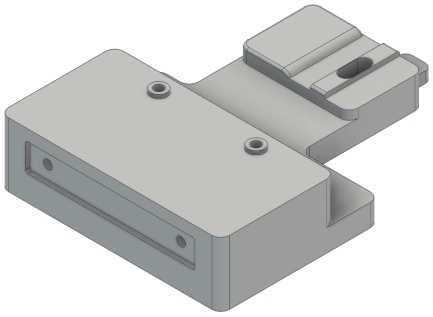
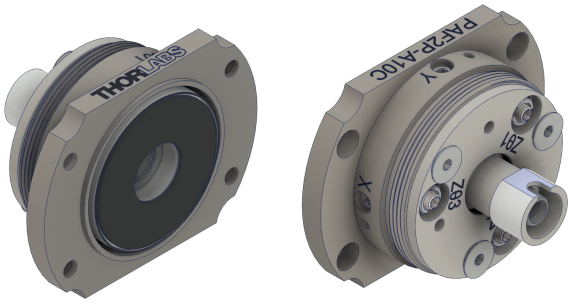


Figure 2.9: 3D model of the new bracket designed to house the *CTTPS1/2* and the other setup components. From the Inventor's design.

- All the components of the setup are aligned on the same axis. The holes in the cavity are aligned with the center of the optic mount on the CTTPS1/2 stage.
- Where the JPE stage is mounted, metal tracks replicate the shape on the bottom of the CTTPS1/2. This assures that the stage is aligned with the bracket just by placing it on the tracks and screwing it in place.
- There is a slot (on the right side, in fig. 2.9) that allows mounting the JPE stage with its PCB facing down.
- The mounting holes for the cavity are kept the same as in the old bracket.
- The height of the bracket is such that the setup can be mounted inside the MU-metal can that shields the setup from magnetic fields when mounted inside the dilution refrigerator.

2.3 NEW FIBER COLLIMATOR

One of the problems with the old setup was that the collimator that collects the output IR light was screwed in place. It was impossible to align it with the laser beam and the cavity holes. Which resulted in low light collection efficiency.
To solve this issue, we opted for a more sophisticated fiber collimator: the *PAF2P-A10C FiberPort Collimator* from THORLABS. This device has many degrees of adjustment that enable it to achieve a more precise alignment. A 3D model of the collimator is shown in fig. 2.4. The screws labeled as $Z\theta 1, 2, 3$ can be used to adjust the angle and distance of the collimator lens to the fiber. The screws labeled X and Y allow to shift the lens in the X and Y directions [14].



(a) Front view. **(b)** Back view.

Figure 2.10: 3D model of the THORLABS PAF2P-A10C FiberPort Collimator. Form the THORLABS website.

I designed a simple mount where the fiber collimator can be attached. The center of the collimator is aligned with all the optical components once the mount is screwed onto the bracket.

2.4 NEW GRIN LENS

A Gradient-index (GRIN) lens exploits a gradient of the refractive index of a material to collimate light. In our setup, the GRIN lens is used to collimate the laser beam out of the input fiber. The old setup already had a working GRIN lens; however, we decided to assemble a new one with a smaller beam waist on the focus point.

The following components from THORLABS were used for our GRIN lens:

- A *GRIN2315A* lens. This lens has an 8° face angle on one of the sides. This prevents the reflected light from entering into the fiber.
- A *SMPF0115-APC* single mode fiber pigtailed ferrules with connector, which also has an 8° angle on one face. The advantage of this ferrule is that the fiber is jacketed, which makes it much more resistant to breakage than bare fiber.
- A glass sleeve for gluing the GRIN lenses and ferrule in place.

Fig. 2.11 shows the plot of the razor blade measurement of the GRIN lens assembly that I built. Its specifications can be found in table 2.2. These specifications fit well with the experiment requirements. In particular, the focal length of 16.2mm is very close to the distance between the GRIN lens surface and the qubit.

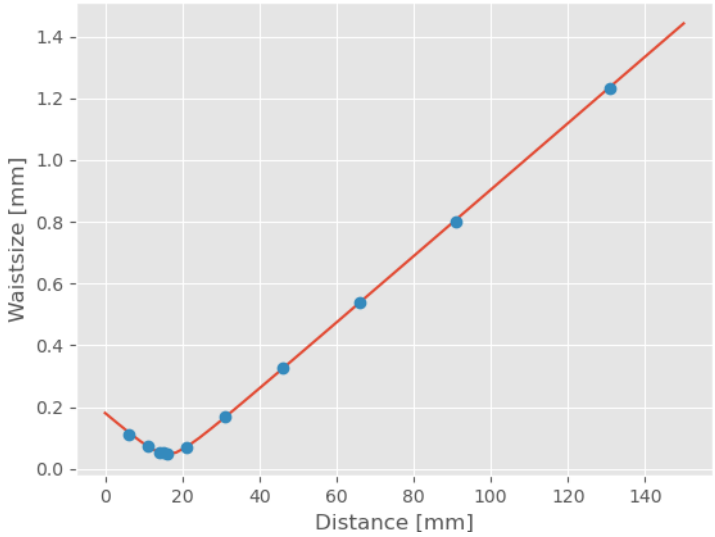
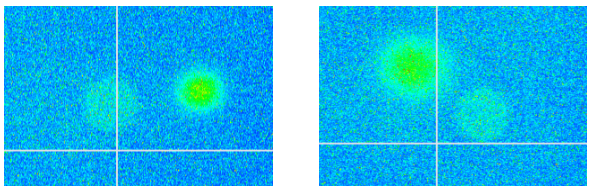


Figure 2.11: Plot of the measurements of the beam waist w.r.t. distance from the GRIN lens, obtained using a razor blade measurement. The blue dots are the measurements. The orange line is the fit of the measurements. Obtained with the Beam Measurement Python script used in the HYQU group.

Focus position $[mm]$	16.21 ± 0.1
Waist $[\mu m]$	47.13 ± 0.3
Rayleigh range $[mm]$	4.38 ± 0.1

Table 2.2: Specifications of the new GRIN lens.



(a) New GRIN lens. **(b)** Old GRIN lens.

Figure 2.12: Pictures of the laser beam spot (brighter spot in the picture) compared to the qubit antenna (dimmer spot) for the new GRIN lens (a) and the old one (b). Obtained for the same laser power with the IR microscope built by Francesco Adinolfi [6]

2.5 OVERVIEW OF THE WHOLE SETUP

To conclude this chapter, I will give an overview of the complete setup and all the improvements. In fig. 2.13, you can see a schematic representation of the setup. To summarize, the major improvements from the old setup are:

- The input laser beam can be tilted thanks to the JPE CTTPS. This way, it is possible to remotely move the laser spot on the qubit chip.
- The holes in the cavity allow the laser beam to reach areas of the qubit that were not accessible before.
- The fiber collimator can be aligned with the rest of the optics yielding a higher coupling efficiency to the output fiber.

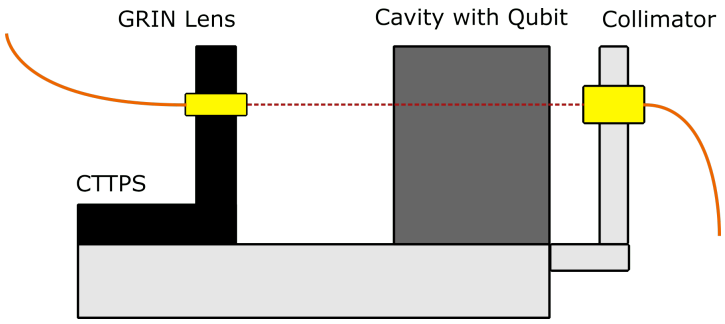


Figure 2.13: Schematic drawing of the whole setup. On the left, the input fiber (orange) brings IR laser light to the GRIN lens (yellow). The GRIN lens is mounted onto the CTTPS (black), which can tilt the GRIN lens angle. The collimated laser light (dashed red) exits the GRIN lens and passes through the cavity holes. Inside the cavity, it shines on the qubit, and then it exits the cavity. On the right side, the collimator (yellow) couples the laser light back into the output fiber (orange).

The setup components can be screwed together on the new bracket. Table 2.3 contains a list of all the screws that are necessary to assemble the setup.

Component	Type	Screw length	Quantity
CTTPS onto Bracket	M3	10-13 mm	1
Cavity onto Bracket	M2	16 mm	2
Collimator Mount	M2.5	10 mm	2
Collimator onto Mount	M2, lens	4 mm	4
Bracket onto Fridge	M3	6 mm	5

Table 2.3: List of screws needed to assemble the new setup.

In the next chapter, I will present an algorithm that exploits the new improvements to reconstruct an image of the qubit from the output collected light.

3 QUBIT IMAGING

The amount of light that passes through the cavity hole gives information about where the laser beam is shining on the qubit chip. If the laser beam is shining on the aluminum layer of the qubit, less light comes out compared to when the laser beam passes through the sapphire chip, which is almost transparent to infrared light. It is possible to use the output light intensity to reconstruct an image of the qubit and use this image to position the laser beam precisely on areas of the qubit chip. The idea is to move the laser beam in an ordered

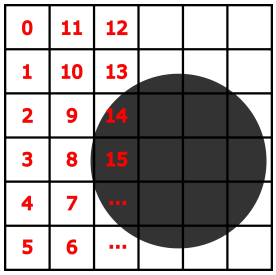


Figure 3.1: Schematic representation of how the qubit imaging works. The red numbers show the positions and order with which the laser beam is moved on the qubit chip. The grey circle represents the antenna of the qubit. It is darker than the background because less light comes out of the cavity hole when the laser beam shines on the qubit. In this example, the image reconstructed is 6x6 pixels.

manner on a grid with the JPE actuators (see section 2.2) and measure the light intensity for every point with a power meter. Finally, these measurements can be visualized as an image, for example, in fig. 3.1, you can see a schematic example of how it would work for a 6x6 image.

The imaging algorithm described above is not as straightforward to implement as it may seem because of some limitations and imperfections in the setup. In the following sections and subsections, I will go through a detailed explanation of how the JPE stage works, its issues, and the algorithms I developed to stem these issues. This chapter deliberately contains a lot of technical details about the code in the hope that it could serve as a guide for the people who will use my code in the future.

3.1 CHALLENGES WITH THE JPE STAGE

3.1.1 Synchronization of the Laser Beam Movement with the Measurements

The actuators in the JPE CTTPS system (see section 2.2) are controlled by the Cryo Positioning System Controller (CPSC), which needs to be connected to a Windows PC via USB or Ethernet. A Command Line Interface program (cacli.exe) needs to be called from the cmd terminal to move the actuators. See the software manual [15] for more information. The full command that needs to be executed is:

```
cacli MOV [ADDR] [DIR] [FREQ] [RSS] [STEPS] [TEMP] [STAGE] [DF]
```

where:

- $ADDR$: Address of the module corresponding to the controller slot. For example, $ADDR = 1$ for actuator number one.
- DIR : Direction of movement: set to 1 for positive movement and 0 (zero) for negative movement.
- $FREQ$: Step frequency of the actuator, in Hz.
- RSS : Relative actuator step size parameter.
- $STEPS$: Number of actuation steps. Ranges from 0 to 50000.
- $TEMP$: Temperature of the environment in which the actuator is used, in Kelvin.
- $STAGE$: Type of actuator. In our case it is set to "CTTPS1/2".
- DF : Drive factor. Can have a value from 0.1 to 3.0. In normal operating conditions, set this value to 1.

The JPE controller does not send any feedback when the actuators start or stop making the steps. This makes it impossible to use an Analog to Digital Converter (ADC) that takes one measurement synchronized with every step. The only option is to execute the MOV command, wait until the actuator has finished moving, and finally take a measurement. Based on tests, the maximum amount of time that it takes to execute on the on the command line a MOV command of $STEPS$ and $FREQ$ is: $T = STEPS/FREQ + 0.5[S]$.
I decided to wrap the cacli MOV command in a Python script. With Python, the MOV command can be executed on the cmd terminal by using the *subprocess.run*() function. For example:

```
subprocess.run("cacli MOV 1 1 600 100 500 293 CTTPS1/2 1",
               check=True)
```

Then I wrote a $move()$ function that takes into account the amount of time that it requires for the actuator to stop moving ("T" from above). For example, for actuator one, it looks like this:

```
def move_1(steps):
    if steps > 0:
        subprocess.run(f"cacli MOV 1 1 {freq} 100 {steps}
                       {temp} CTTPS1/2 {df}", check=True)
        # wait for it to finish
        time.sleep(steps / freq + 0.5)
    if steps < 0:
        subprocess.run(f"cacli MOV 1 0 {freq} 100 {-steps}
                       {temp} CTTPS1/2 {df}", check=True)
        # wait for it to finish
        time.sleep(-steps / freq + 0.5)
```

As you can see, if $move\_1(steps = 0)$ is called, nothing happens. This is because the actuators would move an infinite amount of steps if the MOV command was executed with $STEPS = 0$.
In the same Python script, it is also possible to take a light intensity measurement with the THORLABS PM100D power meter by using the library *pyvisa* and the instrument class written here in the group. The command that is used to take a power meter measurement in watts is:

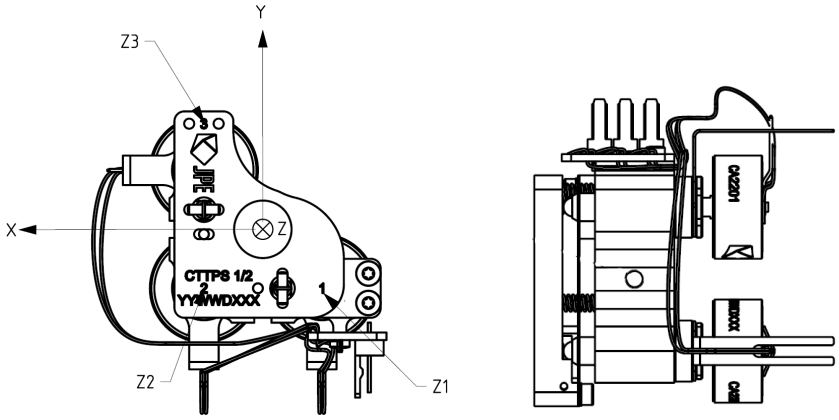
```
rm         = pyvisa.ResourceManager()
PowerMeter = instr.Thorlabs_PowerMeter(rm)
power      =  PowerMeter.get_power(unit='W')
```

It is then possible to synchronize the movement of the JPE actuators and the power meter measurement by first calling the function $move(steps)$ and afterwards $PowerMeter.get\_power(unit =' W')$. This ensures that every measurement is taken an exact number of steps after the previous one. This synchronization is fundamental for having an image with precise information about where the qubit was with respect to the JPE laser beam movement.

3.1.2 Moving the Laser Beam on a Grid

A second challenge that we had to face to obtain images is the difficulty to move the laser on a grid. This is because the JPE CTTPS works by pushing independently on one of 3 corners of a metal plate, see fig. 3.2 [1]. The laser beam is emitted by the GRIN lens mounted at the center of the plate. This kind of system does not allow for precise movement of the laser beam on the x and y-axis. Based on tests, actuator one can move the laser beam reliably on the y-axis. However, actuator three is unreliable for moving the laser beam on the x-axis.

1 Note that in our setup, the JPE stage is mounted with the PCB facing the side and not the bottom like in fig. 3.2. Our x and y axis are inverted compared to fig. 3.2(a).



(a) JPE Stage front drawing **(b)** JPE stage side drawing

Figure 3.2: Drawings of the JPE actuator stage from [16]. The JPE stage is mounted in our setup with the PCB facing the right side (a). This means that our x and y axes are swapped compared to figure (a).

What turned out to work better is using the transformation matrix provided by the manufacturer [16]:

Here, Rx has been swapped with Ry compared to [16] to account for the fact that in our experimental setup, the x and y axis are swapped.

$$\begin{bmatrix} z1 \\ z2 \\ z3 \end{bmatrix} = \begin{bmatrix} -C & +C & 1 \\ -C & -C & 1 \\ +C & -C & 1 \end{bmatrix} \cdot \begin{bmatrix} Ry \\ Rx \\ Z \end{bmatrix} \tag{3.1}$$

Where:

- z1, z2, z3 are the displacements of the actuators one, two and three in mm.
- Rx and Ry are the x and y angles in radians of the plate's z axis with respect to the z axis when the plate is in the nominal position.
- Z is the displacement of the plate in the Z direction in mm.
- C is a constant: $C = 15.35\ mm/RAD$.

Setting the input vector to $(Ry = 0,\ Rx = 0.1RAD,\ Z = 0)$ gives:

$$\begin{bmatrix} z1 \\ z2 \\ z3 \end{bmatrix} = \begin{bmatrix} +1.535\ mm \\ -1.535\ mm \\ -1.535\ mm \end{bmatrix} \tag{3.2}$$

This means that it is necessary to move actuator one a positive amount and actuators two and three a negative amount to move the laser beam in the positive x direction. Based on the above considerations, the Python function that moves the laser beam on the x-axis can be written as:

```
def move_x(steps):
    if steps > 0:
        subprocess.run(f"cacli MOV 1 1 {freq} 100 {steps}
                       {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 2 0 {freq} 100 {steps}
                       {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 3 0 {freq} 100 {steps}
                       {temp} CTTPS1/2 1", check=True)
        time.sleep(steps / FREQ + 1.5)
    if steps < 0:
        subprocess.run(f"cacli MOV 1 0 {freq} 100 {-steps}
                       {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 2 1 {freq} 100 {-steps}
                       {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 3 1 {freq} 100 {-steps}
                       {temp} CTTPS1/2 1", check=True)
        time.sleep(-steps / freq + 1.5)
```

The Python function that moves the laser beam on the y axis is simpler:

```
def move_y(steps):
    if steps > 0:
        subprocess.run(f"cacli MOV 1 1 {freq} 100 {steps}
                       {temp} CTTPS1/2 1", check=True)
        time.sleep(steps / freq + 0.5)
    if steps < 0:
        subprocess.run(f"cacli MOV 1 0 {freq} 100 {-steps}
                       {temp} CTTPS1/2 1", check=True)
        time.sleep(-steps / freq + 0.5)
```

3.1.3 Difference of Movement for the Positive and Negative Direction

Because of the way how the JPE stage is built, the displacement of an actuator is not the same if it moves in the positive or negative direction (the direction is set by $DIR = 1$ or $DIR = 0$ in the *cacli MOV* command, see 3.1.1). For our JPE stage, the movement difference arrives even to more than 10%, for example, for actuator one. This makes it impossible to map the qubit with the laser beam with good precision. In the imaging algorithm, the laser beam needs to scan the columns of the image back and forth. Referring to fig. 3.1, if the JPE stage moved 10% more in the positive direction, the column scanned upward would be as long as 6.6 pixels of the downwards one. Moreover, when the laser gets to position 11 of fig. 3.1 it would not be aligned anymore with the starting position 0. The result is an image where the qubit is deformed, and the picture's grid cannot be used as a way to map the space precisely.

3.2 JPE ACTUATORS CALIBRATION

The main idea, to compensate for the difference of displacement, is to add a parameter to the Python functions $move\_x()$ and $move\_y()$. This parameter is called RPN_i (i=X, Y), which stands for Ratio Positive Negative:

$$RPN = \frac{displacement\ in\ positive\ direction}{displacement\ in\ negative\ direction} \tag{3.3}$$

It is then possible to execute the MOV command with the parameter $STEPS$ equal to:

- if $DIR = 0$, $STEPS = n\_steps$
- if $DIR = 1$, $STEPS = -int(n\_steps \cdot RPN)$

Where DIR sets the direction in the MOV command and n_steps is the number of steps that the actuator needs to move.
For example, if actuator one moves 10% more in the positive direction, then RPN_1=1.1. And when moved in the negative direction, the steps will be 10% more. The RPN parameter effectively compensates for the missing factory calibration of the JPE actuators.
The improved $move\_y()$ function looks like this:

```
def move_y(steps):
    if steps > 0:
        subprocess.run(f"cacli MOV 1 1 {freq} 100
        {steps} {temp} CTTPS1/2 1", check=True)
        time.sleep(steps / freq + 0.5)
    if steps < 0:
        subprocess.run(f"cacli MOV 1 0 {freq} 100
        {-int(steps*RPN_Y)} {temp} CTTPS1/2 1", check=True)
        time.sleep(-int(steps*RPN_Y) / freq + 0.5)
```

The compensation is a bit more complex for the $move\_x()$ function because, in this case, all three actuators are moved simultaneously. The first strategy that comes to mind is to calculate the RPN for every actuator separately and then move $STEPS = -int(n\_steps \cdot RPN)$ instead of just $-n\_steps$ every time $DIR = 0$ in the MOV command. Equation 3.2 states that the absolute displacement needs to be the same for all three actuators. However, based on tests, every actuator moves slightly differently if asked to move the same amount of steps and direction. And there is no way of calculating how much this difference is. For this reason, I opted for another more practical strategy. The compensated $move\_x()$ function is implemented in the following way:

```
def move_x(steps):
   steps = steps / XY_RATIO
    if steps > 0:
        subprocess.run(f"cacli MOV 1 1 {freq} 100
        {int(steps*RNP_X_Y)} {temp} CTTPS1/2 1", check=True)
```

```
        subprocess.run(f"cacli MOV 2 0 {freq} 100
        {int(steps)} {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 3 0 {freq} 100
        {int(steps)} {temp} CTTPS1/2 1", check=True)
        time.sleep(steps / FREQ + 1.5)
    if steps < 0:
        subprocess.run(f"cacli MOV 1 0 {freq} 100
        {-int(steps*RPN_X)} {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 2 1 {freq} 100
        {-int(steps*RPN_X)} {temp} CTTPS1/2 1", check=True)
        subprocess.run(f"cacli MOV 3 1 {freq} 100
        {-int(steps*RPN_X)} {temp} CTTPS1/2 1", check=True)
        time.sleep(-int(steps*RPN_X) / freq + 1.5)
```

The differences between this compensated function and the one in 3.1.2 are:

- *steps* is divided by XY_RATIO. This is because if the functions $move\_x()$ and $move\_y()$ were called with the same amount of steps, the pixels of the generated image would not be squares. The laser beam would move more in the x direction than in the y because all three actuators are used simultaneously with $move\_x()$, while only one is used for $move\_y()$. This parameter cannot be calibrated. It can only be adjusted by eye.

- If $steps < 0$ all three *cacli MOV* commands have $STEPS = -int(steps \cdot RPN\_X)$. Independently of having $DIR = 0$ or $DIR = 1$. This is because in the case of the $move\_x()$ formula of RPN is:

$$RPN\_X = \frac{positive\ x\ displacement}{negative\ x\ displacement} \tag{3.4}$$

  where the positive and negative laser beam displacements are caused by moving all three actuators simultaneously.

- During tests, it turned out that the previous compensation parameters were not enough for moving reliably back and forth on the x-axis. While issuing $move\_x()$ commands, the laser moved slightly also on y. For this reason, the parameter RNP_X_Y was introduced. For $steps > 0$, the number of steps that actuator one moves is $STEPS = int(steps \cdot RNP\_X\_Y)$. This compensates for the unwanted y movement.

3.2.1 Calibration Script

After showing which parameters are used to compensate for the missing manufacturer calibration, I will go through the code I wrote to calculate them automatically.
To begin, the class *JPEsClass*, situated in the file *jpe_class.py*, contains

the *move_x*() and *move_y*() functions with all the parameters that were previously discussed. This way, to use the JPE actuators in a Python script, it is just needed to import the class and create an object from it. The object can move the laser beam with the JPE stage in x and y just by telling it how many steps to move:

```
from jpe_class import JPEsClass

JPEs = JPEsClass()
JPEs.move_y(steps)
JPEs.move_x(steps)
```

This simplifies the use of the JPE stage and removes the annoyance of manually setting up all the parameters in the *cacli MOV* command every time. The script that performs the calibration is called *calibration_image_jpe.py*. It calibrates the JPE stage movement by repetitively moving the laser beam back and forth on the x or y-axis, while recoding the output optical power with the THORLABS PM100D power meter. Then it generates and analyses an image from the data. Finally, it suggests the new optimal value for the compensation parameters *RPN_i*. These new values need to be inserted in the file *jpe_class.py* in the *JPEsClass*. The script can perform three kinds of calibration that need to be performed in order.

CALIBRATION 'Y': With this setting, the laser beam scans a column back and forth a predefined amount of times with the function *move_y*(). It saves the data in a 2D matrix, where the element $(i,\ j)$ is the i^{th} power meter measurement of the j^{th} time it scanned the same column. If j is even the measurement are taken with the actuator moving in the negative direction, if j is odd the actuator was moving in the positive direction. Because the script scans the same column, again and again, this data contains information about the difference between the positive and negative movement of *move_y*(). Here what comes into help is that the fiber collimator coupling efficiency changes throughout the column. It is possible to choose a power-meter measurement value in the first column and track how it shifts in the following columns. This is exactly how the script analyses the 2D matrix with the data. Supposing that the 2D matrix is NxN:

1. It starts by selecting and storing the value at the position $(int(N/2), 0)$. This is the value in the middle of the first column. Lets call it $value0$.

2. For every column, it checks which is the element with a value closest to $value0$. It saves the i^{th} coordinate of such element in an array. The array index corresponds to the matrix's column number (j^{th} coordinate). After repeating this procedure for all the columns, this array contains the function of the shift of $value0$ during the scan.

3. Finally, it plots the function in the array and performs a linear fit of it. The slope of the fitted function tells how RPN_Y needs to be adjusted. The new optimal value for RPN_Y will be:

$$RPN\_Y_{new} = RPN\_Y_{old} \cdot (1 + \frac{slope\ of\ fit}{int(N/2)}) \tag{3.5}$$

An example can make it easier to understand this formula. Suppose that the JPE stage moves 5% more in the positive direction. Before the calibration $RPN\_Y_{old} = 1$. Suppose that the script has been set up to generate a 20x20 2D matrix with the data. This setting has been simulated with Python for proof of concept and better visualization, and the result is in fig. 3.3. A difference of 5% means that the element at position (10,0) will shift up one pixel every time the laser beam scans two columns. This is what can also be observed in fig. 3.3. Con-

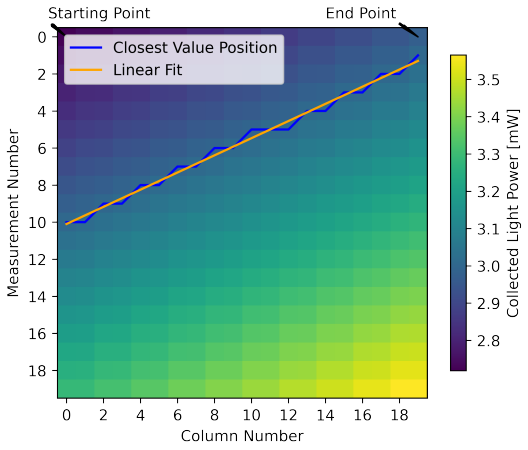


Figure 3.3: Simulated Y calibration image with the same analysis as done by the calibration script. For this simulation the *move_y()* function moves 5% more in the positive direction. The pixel color represents the optical power measurement in that position.

sequently, the slope of the linear fit is approximately 0.5. Which, using equation 3.6, gives the expected $RPN\_Y_{new} = 1.05$. After applying this new RPN value and executing the simulation again, you can see in fig. 3.4 that *move_y()* has been successfully calibrated.

CALIBRATION 'XY': This setting is used to calculate the new optimal RNP_X_Y parameter. Mostly, it works in the same way as the 'Y' calibration. The only difference is that every time it reaches the bottom of a column, it executes the following for loop:

```
for _ in range(0, 4):
    JPEs.move_x(N_STEPS_PER_MMT)
    JPEs.move_x(-N_STEPS_PER_MMT)
```

This moves the laser beam back and forth four times on the x-axis for a distance of $N\_STEPS\_PER\_MMT$. $N\_STEPS\_PER\_MMT$ is

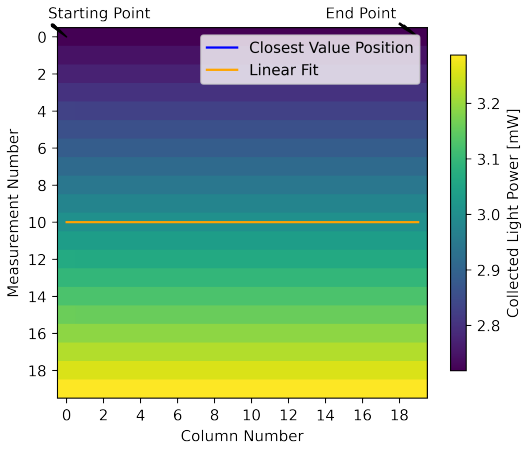


Figure 3.4: Simulated Y calibration image with the correct RPN_Y compensation parameter.

equivalent to the number of steps needed to move to the next pixel of the picture. If $move\_x(N\_STEPS\_PER\_MMT)$ causes a shift in the y direction, a result similar to the one in fig. 3.3 will be observed.
The new parameter is calculated as:

$$RNP\_X\_Y_{new} = RNP\_X\_Y_{old} \cdot (1 + \frac{slope\ of\ fit}{4}) \tag{3.6}$$

The 'Y' calibration must be performed successfully before 'XY'. Otherwise, the observed y shift would be a sum of the shift caused by $move\_y()$ and the one by $move\_x()$.

CALIBRATION 'X': This setting is used to obtain the new optimal RPN_X. It works exactly in the same way as the 'Y' calibration, with the only difference that $move\_x()$ is used instead of $move\_y()$.

3.3 QUBIT IMAGING

The qubit imaging script *precision_scan_jpe_v2.py* performs an automated scan of the qubit with the JPE stage by moving the laser beam in the x and y direction on a grid. It follows the same pattern as fig. 3.1. It works by scanning a column in the y direction and then moving to the next column, which is scanned in the opposite direction. While scanning, it measures the output optical power with the THORLABS PM100D power meter. A picture is constructed by assigning the light intensity measurement to each pixel.
To run this script, it is necessary to configure three parameters:

- $N\_STEPS\_PER\_MMT$: Defines how many steps are given as a parameter to the $move\_x(steps)$ and $move\_y(steps)$ functions. It effectively sets how much distance there is between one pixel

and the next one. Note that the JPE actuators have a different step size depending on the temperature. At room temperature, a good value for this parameter is 500. At 4K, a good value is 2500.

- NX_MMTS : Defines how many measurements will be done in the x direction. It corresponds to the number of columns of the generated image.
- NY_MMTS : Defines how many measurements will be done in the y direction. It corresponds to the number of rows of the generated image.

One last parameter that can be changed, but is often not necessary, is $SCAN\_DIR$. If $SCAN\_DIR = 1$, the order that the scan follows is the same as fig. 3.1. If $SCAN\_DIR = 0$, the order is reversed, i.e., the scan would start from the last pixel and end on the first one.

3.4 COUPLING EFFICIENCY OF THE COLLIMATOR

All the code described until now is still insufficient for generating a precise and clear image of the qubit. The background needs to be approximately constant for the qubit to be visible. In reality, this is not the case. The THORLABS fiber collimator's coupling efficiently varies in space. It has a maximum, which can be adjusted with some screws. Moving the laser beam away from the maximum results in less light being collected.
There are two issues with this. First of all, the position of the maximum coupling efficiency cannot be changed arbitrarily. It is possible to position it close to the qubit antenna, but we were not able at the moment to align it precisely. Second, suppose the maximum is not aligned perfectly with the antenna. In that case, the spatial change in the coupling efficiency function is more significant than the change caused by the laser beam moving behind the qubit. This makes it impossible to distinguish the qubit from its background. To solve this issue, I developed an image processing algorithm that is able to normalize the image without exact knowledge of the coupling efficiency function.

3.4.1 Image post-processing algorithm

This algorithm assumes that the spatial function of the collimator's coupling efficiency is radially symmetric around the axis that passes through the maximum. In fig. 3.5 there is a plot of a function that fulfills this assumption.
This assumption holds when all the optical elements of the setup are aligned on the same axis. In fact:

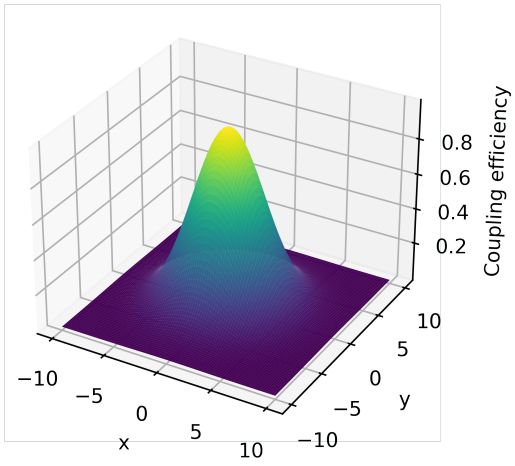


Figure 3.5: Example of a radially symmetric spacial function where the symmetry axis passes through the maximum. The plotted function is $f(x,y) = \exp\left(-\frac{x^2+y^2}{18}\right)$.

- The laser beam is radially symmetric.
- The fiber collimator's optics is radially symmetric if the lens is in the nominal position.

This is not always true because the setup elements are not necessarily on the same axis because of their tolerances. For example, how the JPE stage is designed doesn't allow fixing the GRIN lens metal holder perfectly aligned and centered. Despite this, as it will be shown in 4.1.5, the image processing algorithm turned out to be effective.
If the radial symmetry assumption is fulfilled, the coupling efficiency has the same value on circumferences centered on the maximum. This corresponds to the intersection between a plane parallel to the x-y plane and the 2D coupling efficiency function. In this case, it is possible to normalize the generated image by checking the distance from the maximum. The file *picture_processing.py* contains a Python implementation of the algorithm.

REFERENCE NORMALIZATION ARRAY: The algorithm starts with an array of reference values that contains the coupling efficiency values[2], where the index is the distance from the maximum. For example:

$$\begin{bmatrix} 1, & 0.8, & 0.6, & 0.4, & 0.2 \end{bmatrix} \tag{3.7}$$

means that the coupling efficiency is one on the maximum, 0.8 for the pixels with a distance from the maximum of one, 0.6 for the pixels

2 Note that what is measured is the output optical power which is directly proportional to the coupling efficiency if the laser beam is not shining on the qubit:
output optical power = coupling eff. $*$ input optical power

with a distance of two, and so on. This array is used for normalizing all the pixels of the image. In this example, all the pixels that have a distance of one from the maximum would be divided by 0.8, all the pixels that have a distance of two from the maximum by 0.6, and so on. After this normalization process, the qubit will emerge from the background.
There are some catches to this algorithm:

- The array that contains the calibration values needs to be extracted somehow from the image.
- The radial symmetry assumption does not hold perfectly. Note that this makes it even more important to have a precise calibration. The pixels' location should not shift throughout the image.

The first issue is solved by defining some portions of the image where it is certain that the laser beam was not shining on the qubit. The algorithm then takes care of calculating the distance of each pixel in these portions and averaging the pixel values to produce the normalization array. Notice that these portions need to cover all possible distances from the maximum. Otherwise, it would not be possible to normalize all the pixels of the image.

AUTOMATIC OPTIMAL CENTER SEARCH: The solution to the second issue is to perform an automatic search that finds the maximum point (i.e., the center of the normalization circumferences) that gives the best normalization results. Because the radial symmetry is not perfect, this point is a virtual maximum point, different from the real one. This point can even be outside the image.
This procedure only works if there is a quality measure that quantifies how well it is possible to visually distinguish the qubit from the background in the normalized picture.
There are mainly two factors that determine how well the image has been normalized:

- How close all the pixel values are to one. If the normalization is perfect, all the pixel values are one, except where the qubit is. The pixels where the qubit is situated have a value smaller than one, but this area is usually a small portion of the whole image. The distance from one can be computed as:

$$\sum_{i,j}(image[i,j]-1)^2 \tag{3.8}$$

  where (i, j) are respectively the row and column indices of the pixels in the processed image. image[i, j] is the value of the pixel (i, j).

- If the normalization is perfect, all the values of the post-processed image are smaller or equal to one. However, during tests, a few values were much bigger than one, probably because of imperfections in the radial symmetry assumption. To avoid this from happening, the distance of the maximum value from one is also included in the quality measurement as:

$$(1 - max)^2 \cdot n\_rows \cdot n\_columns \cdot weight \tag{3.9}$$

  Here max is the maximum pixel value in the post-processed image. $(n\_rows \cdot n\_columns)$ gives a weight equal to the total number of pixels in the picture to $(1 - max)^2$. This is necessary because otherwise, having a max in a 3x3 picture would give the same result as having the same max in a 30x30 image. $weight$ is a parameter that is used to tune the result.

Putting together equations 3.8 and 3.9 it is possible to implement a Python function that evaluates the quality of the image in the following way:

```
def evaluate_quality(image):
    p = 0 # quality = 1 / p
    max_value = 0
    for i in range(0, image.shape[0]):
        for j in range(0, image.shape[1]):
            p = p + (image[i, j] - 1)**2
            max_value = max(max_value, image[i, j])
    quality = 1 / (p + ((1 - max_value)**2 * image.shape[0] *
         image.shape[1] * weight_max))
    return quality
```

Everything is now ready for the image processing algorithm. To recap, the steps that the algorithm performs are:

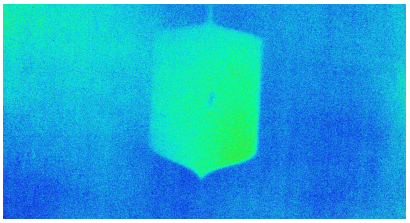
1. It starts on the first point in the area where we told it to search for the optimal maximum.
2. It generates the reference normalization array based on the distance between the pixels in the normalization areas and the current optimal maximum point.
3. It normalizes the whole image with the reference normalization array.
4. It evaluates the quality of the image.
5. It performs the above steps for all the points in the area of the optimal maximum search.
6. Finally, it outputs the image for which the quality measure was the highest.

As you will see in the next chapter, this algorithm gave consistent results when used to post-process qubit images.

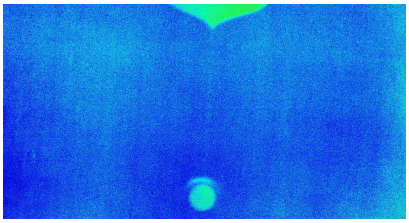
4 TEST OF THE NEW SETUP

The algorithms described in the previous chapter were developed and improved during experimental tests. The challenges described in chapter 3 arose during this time.
In this chapter, I will present the experimental tests which show that the improved experimental setup and the Python scripts achieved the desired results. The sections illustrate the tests in chronological execution order. Subsequent tests needed the previous ones to be successful.

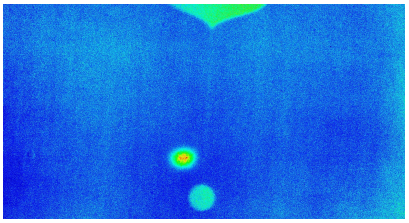
4.1 TESTS AT ROOM TEMPERATURE



(a) Microscope picture of the superconducting qubit pad.



(b) Microscope picture of the qubit antenna with the laser beam positioned on the antenna.



(c) Microscope picture of the qubit antenna with the laser beam positioned outside of the antenna.

Figure 4.1: Pictures of the qubit antenna and pad obtained with the IR microscope (built by Francesco Adinolfi [6]) during the room temperature alignment, while the qubit was mounted inside the cavity. The brighter spot in picture (c) is the laser beam. In picture (b) a halo can be seen behind the antenna. It is caused by the laser beam shining behind it.

The quasiparticles experiment is performed in the dilution refrigerator. However, before testing the setup in the refrigerator, we carried out several tests on the optical table at room temperature. This way,

we could access and modify the setup without the inconvenience of opening the refrigerator and dismounting the experimental setup.

4.1.1 Alignment with the Qubit Antenna

The first task we carried out was the alignment of the laser beam with the qubit antenna. In the proposed transducer the laser beam shines close to the antenna; see 1.1. The quasiparticles experiment is performed with the laser beam shining in the proximity of the antenna or onto it such that the results can apply to the transducer as well.
The alignment can be done with the help of the IR microscope built by Francesco Adinolfi during his master thesis [6]. The experimental setup without the collimator is mounted on an XYZ translation stage with the output cavity hole facing the microscope. The translation stage allows the hole to be placed on the microscope focus point. The microscope is equipped with a laser beam profiler that streams the image on a PC. Once the image of the qubit can be seen on the PC, it is possible to align the laser beam with the qubit antenna by sending movement commands to the JPE actuators. In fig. 4.1, you can see three pictures taken from the microscope during the alignment.
The starting point of the imaging algorithm is in the top left corner; see chapter 3. Therefore, the alignment with the antenna can be considered correct if the laser beam is positioned similarly to fig. 4.1 (c).

4.1.2 Qubit Imaging with the Power Meter

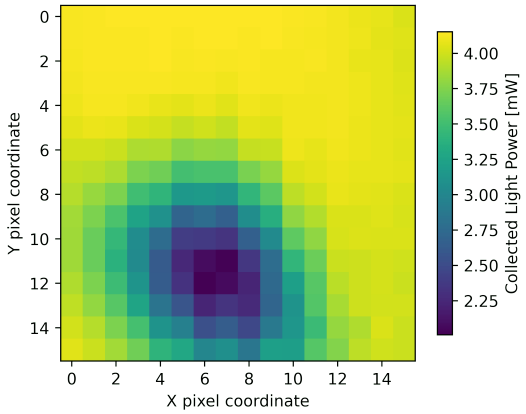


Figure 4.2: First successful attempt at the antenna imaging performed using the power-meter photodiode instead of the fiber collimator. The color-map represents the collected optical power in mW.

After aligning the laser beam with the antenna, we tried the qubit imaging script. Unfortunately, the first imaging attempts were unsuccessful because, at that time, I had not yet implemented the image processing algorithm. For this reason, the first successful antenna im-

ages were obtained by removing the collimator and placing the power meter photodiode sensor in front of the output hole. An image of the antenna, reconstructed using this method, is shown in fig. 4.2. As you can see, the power meter records less power when the laser beam is shining on the antenna. The power is approximately half when the laser beam is on the center of the antenna (2.01 mW vs. 4.13 mW). However, the antenna shape shifts slightly throughout the picture due to the JPE actuators not being calibrated.

I show this initial result even though this method is not used in the experiment because it can serve as a comparison to the images taken with the collimator mounted. It shows how much difference the steep coupling efficiency of the collimator makes.

4.1.3 Collimator Alignment

If the collimator is aligned correctly, the maximum collection can go up to 90% with our setup. However, the coupling efficiency function of the collimator is steep. Moving an amount equal to twice the antenna diameter from the maximum can decrease the collection by one order of magnitude. A precise alignment is necessary to have a high enough light collection on the whole range of the imaging. The maximum coupling efficiency must be inside the scan area or very close to it. To achieve this, I first aligned the laser beam with the antenna (see subsection 4.1.1). Then, I followed the alignment procedure on the THROLABS website [17] to align the collimator to the laser beam position.
There is an issue with this procedure: the qubit antenna does not lay precisely on the axis to which all the optical elements are aligned. Consequently, the laser is tilted by an angle when it is shining close to the antenna, which prevents the maximum collimator collection from being close to the antenna. A possible solution is to move the qubit chip in its slot manually. Unfortunately, this method is imprecise because of the inconvenience of moving the qubit chip manually by a few tens of micrometers. However, for now, this is the only option.
Here is an example of the result that I managed to obtain with this method:

- The laser power was adjusted such that 5.25 mW were recorded coming out of the output cavity hole without the collimator. In the first step, the laser beam was positioned like in fig. 4.1 (c).
- The collimator was mounted. Before aligning it, the collected power was 0.82 mW, equivalent to a 16% coupling efficiency.
- After of few iterations of realigning the laser beam and the collimator and repositioning the qubit, the coupling efficiency reached 84%.

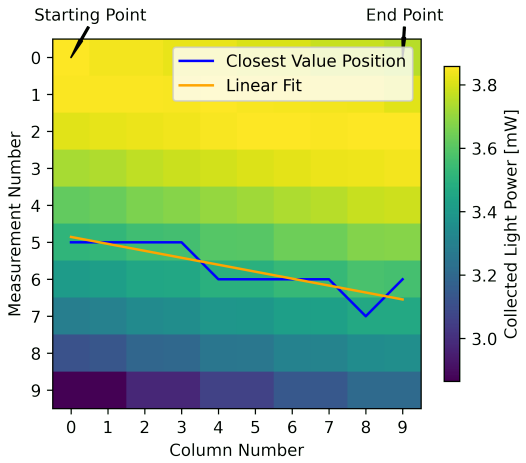
After the alignment of the collimator was complete it was possible to proceed with the JPE actuators calibration.

4.1.4 Calibration with the Collimator

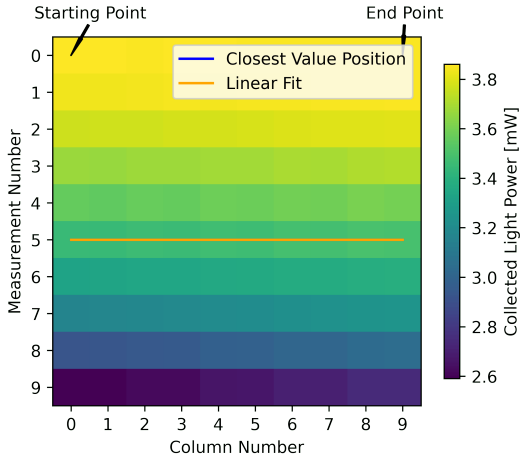
The calibration of the JPE actuators needs to be performed before the qubit imaging. As explained in section 3.4.1, it helps with the validity of the radial symmetry assumption upon which the post-processing algorithm is based.
The automated calibration is performed by the Python script that was implemented during this project. The script executes some measurements, generates an image, and analyses it to give the new optimal value of the compensation parameter (RPN_i). See the description of fig. 4.3 and subsection 3.2.1 for more details about how the algorithm works. In the ideal case, the RPN_i parameter calculated by the script should be correct after the first iteration. However, we observed during tests that getting a well-calibrated actuator takes a few iterations. We think this might be due to the actuators warming up while moving, which could modify the optimal compensation parameter.
In fig. 4.3 you can appreciate the results of a calibration of the Y actuator. Fig. (a) shows an uncalibrated Y actuator which becomes very well calibrated (b) after a few iterations (in this case four iterations) of the calibration algorithm.
As a reminder from subsection 3.2.1, the order in which the calibration is executed is: Y, XY and finally X. Another two calibration examples for XY and X are shown in A.1. As you can see the calibration effectively decreases the relative shift between positive and negative direction of the actuators.

4.1.5 Qubit Imaging with the Collimator

After performing the alignment and the calibration, it is finally possible to proceed with the qubit imaging with the collimator mounted. Chapter 3 already describes how the imaging algorithm works. Here I will show the results we obtained. As you can see from fig. 4.4 (a), it is not clear where the qubit antenna is in the original picture. There is only a shadow of the antenna. However, the antenna appears after executing the post-processing algorithm (see 3.4.1). From this figure, you can also observe the steepness of the collimator coupling efficiency function. The collected power on pixel $(0,\ 0)$ is 2.7 mW. On the opposite corner of the image, this value goes down to 90 μW. This shows how important the alignment is.

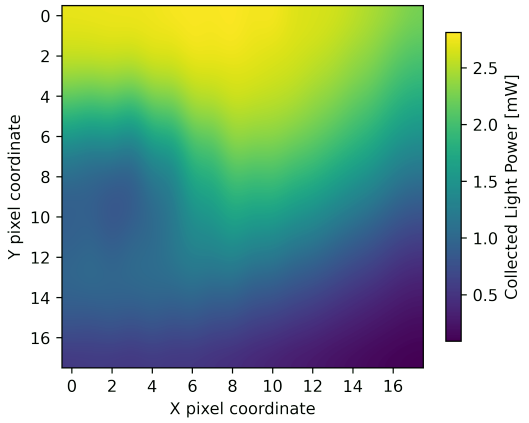


(a) Calibration data after the first calibration procedure.

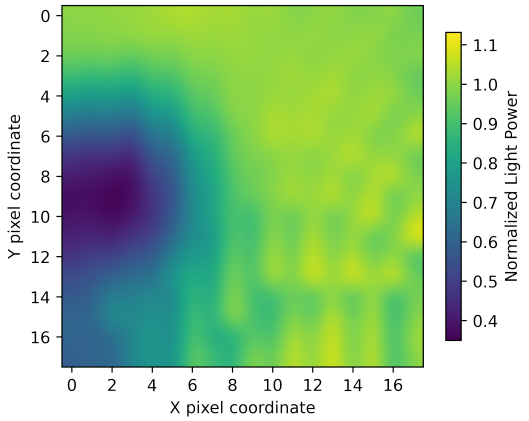


(b) Calibration data after a few iteration of the calibration procedure.

Figure 4.3: Data generated during the calibration of the JPE Y actuator. The image is generated by moving the Y actuator along a column in the negative direction while measuring the collected output light power (in this case, it took ten measurements per column). Then the actuator scans the same column back while taking measurements. Then it repeats for a fixed amount of iterations. The image's blue and orange plots are the results of the algorithm analysis. The algorithm starts by choosing the middle power measurement on column zero (in this picture at coordinates (5,0)). The blue line shows the value closest to the initially chosen one for subsequent columns. This quantifies how much the columns shifted between each other. The orange line is a linear fit of the blue plot. The slope of the orange line obtains the new RPN_Y. See 3.2.1 for more information about the algorithm.



(a) Original Image.



(b) Image after the post processing.

Figure 4.4: Image of the data generated by the qubit imaging script. Visualized with the *interpolation* $='$ *bicubic*$'$ option. As you can see the antenna is not visible in the original picture (a). However it is clearly visible in the the post-processed one (b).

4.2 TESTS IN THE DILUTION REFRIGERATOR

After completing all the tests presented above, we mounted the setup in the dilution refrigerator. The quasiparticle experiment must be carried out at a temperature of 8 mK. However, the imaging of the qubit antenna needs to be performed at 4 K. In fact, the JPE actuators warm up the refrigerator substantially because, during one antenna imaging, they move continuously for approximately half an hour. The refrigerator has enough cooling power at 4 K to maintain a stable temperature. In our tests, we observed an initial increase in temperature, which stabilized at 12 K. The pressure sensors remained in the optimal range during this time. However, imaging the qubit at 8 mK would probably damage the refrigerator because, at this temperature, the cooling power is less than 12 μW [18]. The idea is then to image the qubit antenna at 4 K and position the laser beam in a specific location.

Cooldown to 8 mK and carry out the quasiparticle experiment. Then repeat this procedure for every new laser beam location.

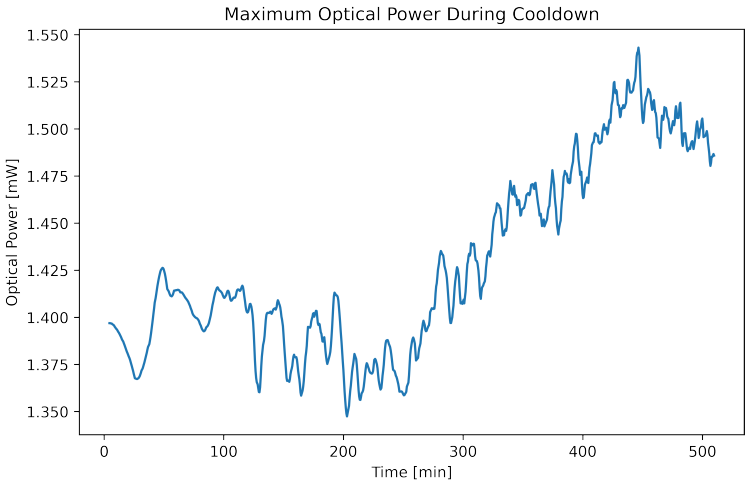
We tried to obtain an image of the antenna during two refrigerator cooldowns. However, we never managed to because of a series of issues.

4.2.1 First Cooldown

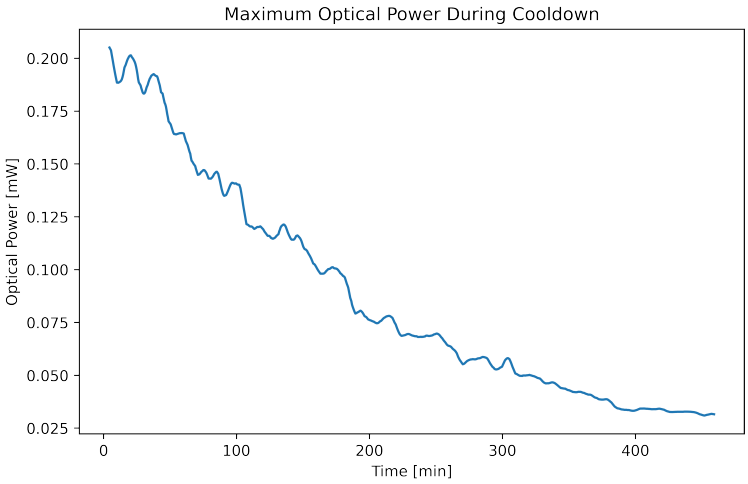
During the first cooldown, the laser beam got moved too far from the hole while testing the JPE actuators movement, and it was challenging to find it again. The collimator must collect light to know if the laser beam is moving closer to the maximum collection point or further from it. If the light collection increases during the JPE actuator's movement, then the movement direction is towards the maximum. If there is no light collection, it is impossible to tell where the laser beam is moving. Unfortunately, during the first cooldown, we could not recover the laser beam position and collect any data.
Furthermore, we observed that the collimator's coupling efficiency significantly decreased during the cooldown. Therefore, we further investigated this issue during the following cooldown.

4.2.2 Collimator's Coupling Efficiency During Cooldown

During the refrigerator cooldowns, the setup components contract because of the temperature change. This causes the collimator coupling efficiency to decrease significantly. We decided to track this decrease during the second cooldown with a Python script that measures the output optical power and automatically re-aligns the laser beam with the collimator after N measurements. The automatic realignment is straightforward: the script moves the laser beam with the JPE CTTPS until it finds the point with the highest optical power collection.
In fig. 4.5, you can see the script's data during the second refrigerator cooldown to 4 K. Interestingly, as shown in fig. 4.5 (a), in the first eight hours of the cooldown, when the refrigerator reached a temperature of 40 k, the contraction of the components improves the alignment. After, the output optical power gradually decreases. At the end of the cooldown (see fig. 4.5 (b)), the output optical power was approximately 30 μW, ~ 50 times smaller than the initially measured 1.4 mW. We think this could be caused by the collimator adjustment screws that misalign its lens due to the temperature change. Another possible cause is that the bracket, where all the components are mounted, was manufactured in aluminum. However, the JPE CTTPS and the collimator are made of titanium and stainless steel. Different materials have different temperature coefficients, making them contract differently.



(a) Data of the first eight and a half hours of cooldown. The starting temperature of the refrigerator is 293 K. The last data point was recorded for a temperature of approximately 40 k.

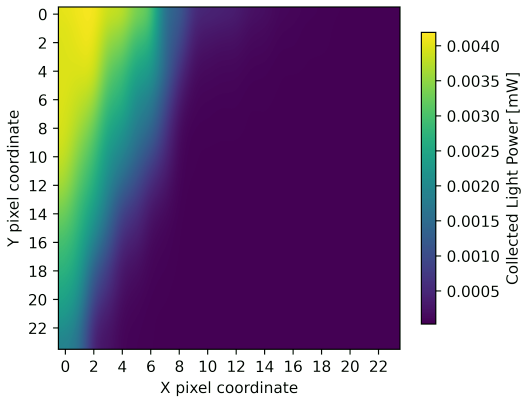


(b) Data taken in the morning approximately nine hours after the computer stopped recording the data from (a).

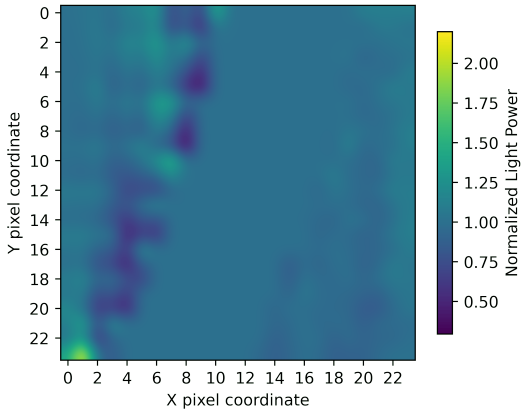
Figure 4.5: Plots of the output optical power measured during a cooldown to 4 k. One measurement was taken every 30 seconds. The laser beam was automatically re-aligned to the maximum every 5 minutes. Unfortunately, after the first eight and a half hours of data collection (a), the computer crashed. After approximately nine hours the script was restarted, it continued recording data until the refrigerator reached a temperature of 4k (b).

4.2.3 Second Cooldown

Once the refrigerator reached a temperature of 4 K, we faced another issue. Out of the three JPE actuators, one stopped moving, one moved less than expected, and only one moved as expected. We think this issue could have been caused by friction on the actuators. The white ceramic plates covering the actuators' interiors fell off, which might have



(a) Original Image.



(b) Processed Image.

Figure 4.6: Image generated by the qubit imaging script with the setup at 4 k in the refrigerator during the second cooldown. Visualized with the *interpolation* $=$′ *bicubic*′ option. As you can see, the antenna is not visible in both the original picture (a) and the post-processed one (b).

prevented the actuators from moving. We tried many combinations of the MOV parameters (see 3.1). We tried changing the driving factor DF and the temperature parameters. However, nothing happened. In the end, we performed a qubit imaging with a modified script version, which only used the two working actuators. In fig. 4.6, you can see the image we generated and its processed version. The antenna is not visible. The darker spots in the processed image are due to imperfect normalization. Unfortunately, we could not take more data because the laser beam was lost again.

5 CONCLUSION AND OUTLOOK

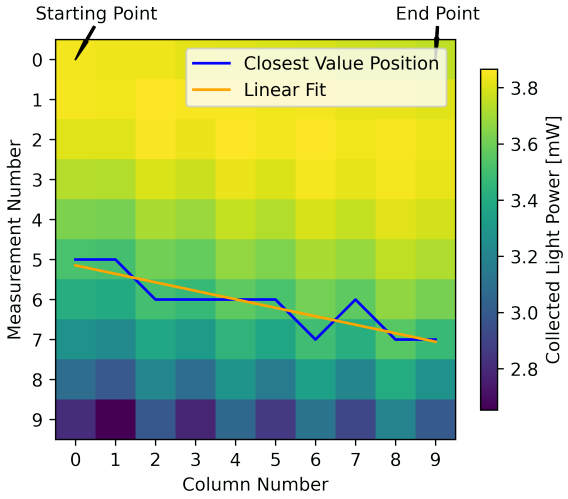
During the course of this project, I extended and improved the experimental setup for measuring quasiparticles generated in a superconducting qubit by an infrared laser. The new setup offers more flexibility due to the JPE actuators. Moving the laser beam position while the setup components are mounted inside the dilution refrigerator is now possible. Furthermore, I designed and implemented:

- A calibration algorithm that increases the precision and reliability of the JPE actuators' movement.
- An imaging algorithm that reconstructs a picture of portions of the qubit by using the output collected optical power and the JPE stage movement.
- A post-processing algorithm that extracts the qubit from the image background by automatically finding the optimal normalization of the image.

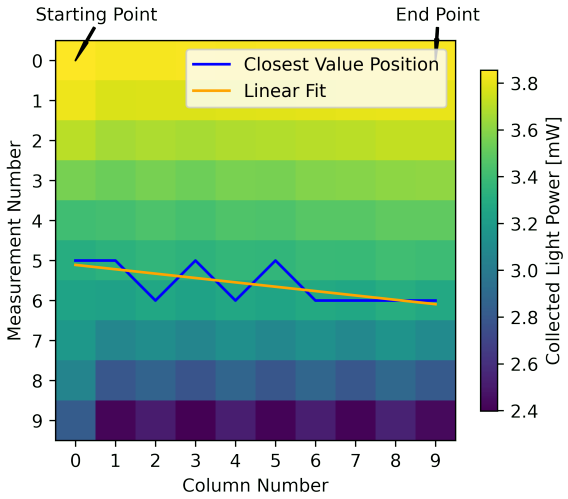
These three algorithms were extensively tested at room temperature with the improved setup. Unfortunately, due to technical problems with the JPE actuators, we could not image the qubit antenna while the components were mounted in the dilution refrigerator at 4K. However, we believe that once the technical problems with the JPE actuators are sorted out, it will be possible to image the qubit antenna in the dilution refrigerator. This will allow investigating the effects of quasiparticles on a superconducting qubit, not only for different laser pulse duration and power but also with the laser beam precisely positioned on specific qubit locations.

A APPENDIX: MORE MEASUREMENT RESULTS

A.1 JPE CALIBRATION

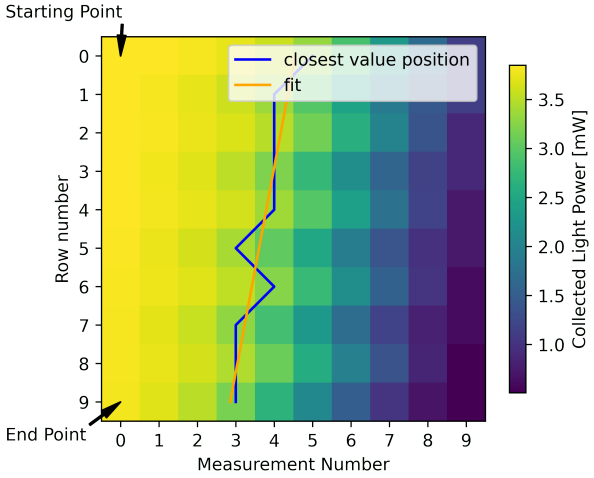


(a) Before calibration.

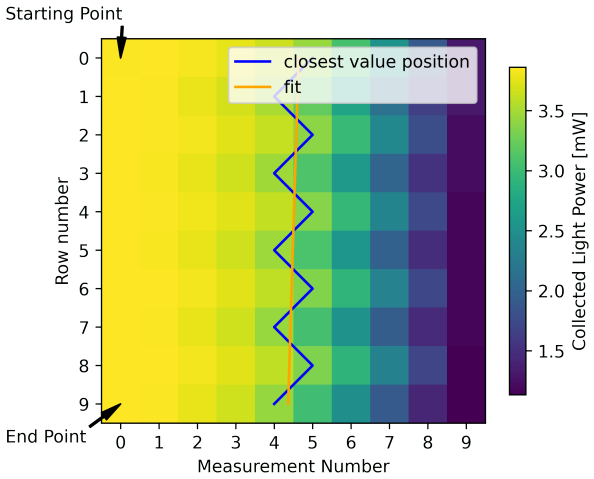


(b) After one calibration.

Figure A.1: Data generated during the JPE XY calibration. The blue line shows the values closest to the value chosen by the analysis. This quantifies how much the columns shifted between each other. The orange line is a linear fit of the blue plot. The slope of the orange line obtains the new RNP_X_Y. See 3.2.1 for more information about the algorithm.



(a) Before calibration.

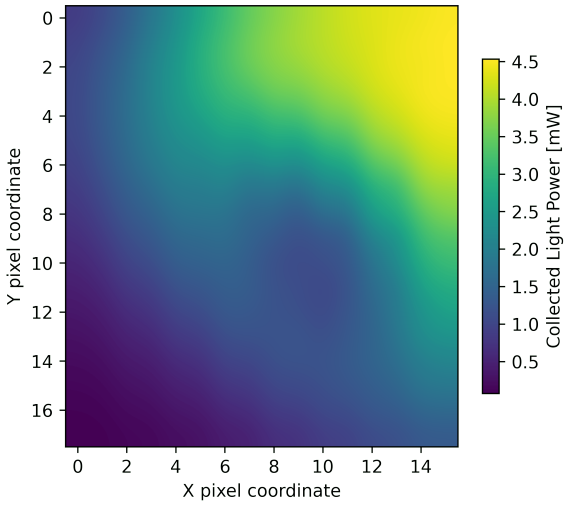


(b) After one calibration.

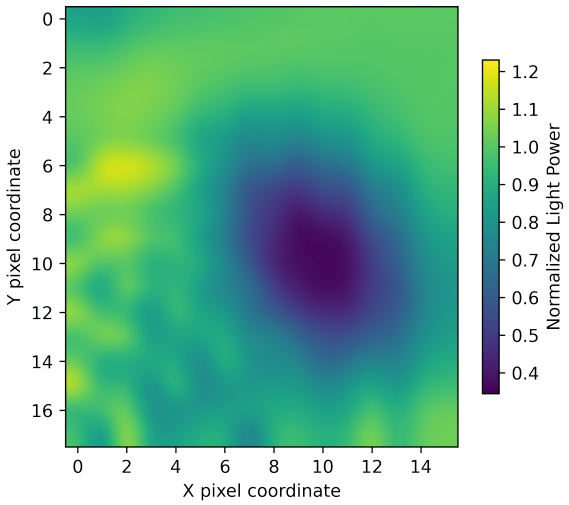
Figure A.2: Data generated during the JPE X calibration. The blue line shows the values closest to the value chosen by the analysis. This quantifies how much the rows shifted between each other. The orange line is a linear fit of the blue plot. The slope of the orange line obtains the new RNP_X. See 3.2.1 for more information about the algorithm.

A.2 ANTENNA IMAGING

In this section I show one more example of antenna imaging performed with the collimator.



(a) Original Image.



(b) Processed Image.

Figure A.3: Image of the data generated by the qubit imaging script. Visualized with the *interpolation* $=' bicubic'$ option. As you can see the antenna is not visible in the original picture (a). However it is clearly visible in the the post-processed one (b).

BIBLIOGRAPHY

[1] Gustafsson M. and Aref T. et al. "Propagating phonons coupled to an artificial atom." In: *Science* 346.6206 (2014), pp. 207–211. DOI: 10.1126/science.1257219.

[2] LaHaye M., Suh J., and Echternach P. et al. "Nanomechanical measurements of a superconducting qubit." In: *Nature* 459.7249 (2009), pp. 960–964.

[3] Chu Y., Kharel P., and Yoon T. et al. "Creation and control of multi-phonon Fock states in a bulk acoustic-wave resonator." In: *Nature* (2018). DOI: https://doi.org/10.1038/s41586-018-0717-7.

[4] Kharel P., Chu Y., and Kittlaus E. et al. "Multimode strong coupling in cavity optomechanics." In: *arXiv* (2018). DOI: 10.48550/ARXIV.1812.06202.

[5] Glazman L. and Catelani G. "Bogoliubov quasiparticles in superconducting qubits." In: *SciPost Physics Lecture Notes* (2021), p. 031.

[6] HYQU Group Francesco Adinolfi. *Master Thesis*. 2021.

[7] Kozorezov AG, Volkov AF, and Wigmore JK et al. "Quasiparticle-phonon downconversion in nonequilibrium superconductors." In: *Physical Review B* 61.17 (2000), p. 11807.

[8] Alexandre Blais, Arne L Grimsmo, Steven M Girvin, and Andreas Wallraff. "Circuit quantum electrodynamics." In: *Reviews of Modern Physics* 93.2 (2021), p. 025005.

[9] COMSOL. *Computing Q-Factors and Resonant Frequencies of Cavity Resonators*. URL: https://www.comsol.com/model/computing-q-factors-and-resonant-frequencies-of-cavity-resonators-9618.

[10] COMSOL. *RF Module User's guide*.

[11] Michael Tinkham. *Introduction to superconductivity*. Courier Corporation, 2004.

[12] COMSOL. *Scattering Boundary Conditions for Wave Electromagnetics Problems*. URL: https://www.comsol.com/blogs/using-perfectly-matched-layers-and-scattering-boundary-conditions-for-wave-electromagnetics-problems/.

[13] JPE. *CRYO TIP-TILT-PISTON STAGE*. URL: https://www.jpe-innovations.com/cryo-nano-products/cryo-tip-tilt-stage-with-piston-motion-cttps/.

[14] THORLABS. *PAF2 Series FiberPort Collimators, User Guide*.

[15] JPE. *CPSC Software User Manual, Software v7.3.20210802*.

[16] JPE. *CTTPS Interface drawings*.

[17] THORLABS. *Fiber Collimator Alignment Procedure*. URL: https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=2940.

[18] BLUEFORS. *Technical Specifications of the Dilution Refrigerator*. URL: https://bluefors.com/products/ld-dilution-refrigerator/#technical-specifications2.