Simulations and Designs for Atom Interferometry Experiments

Zhe Luo

M.A. University of Virginia 2018

A Dissertation presented to the Graduate Faculty
of the University of Virginia in Candidacy for the Degree of
Doctor of Philosophy

Department of Physics

University of Virginia
August, 2020

Contents

1 Introduction **1**

1.1 Sagnac Effect 1

1.2 Atom Interferometry 3

1.3 Experiment Apparatus 7

1.4 Previous Work 8

1.5 New Experiment Setup 13

1.6 Scope 14

2 Atom Chip **17**

2.1 Theoretical Design of Atom Chip 17

2.2 Mechanical Design and Fabrication 24

2.3 Testing Chamber Design 30

3 Experiment Programs **34**

3.1 Image Analysis Program 34

3.2 Admiral Program 46

4 Phase Caculations **50**

4.1 Ideal Signal Phase 50

4.2 Numerical Phase Calculation 59

5 Conclusion **66**

Appendices **70**

A AutoCAD Design for testing chamber **71**

B Codes for image analysis program **79**

List of Figures

1.1 Mach-Zehnder interferometer 2

1.2 Sagnac Interferometer 3

1.3 The splitting principle for atom interferometry 5

1.4 The splitting trajectory for atom interferometry 6

1.5 The vacuum system for atom interferometry experiment 7

1.6 Relevant energy level of Rb 9

1.7 Previous atom trap 9

1.8 Absorption image of BEC atom clouds 12

1.9 BIGTOP system concept 14

2.1 Atom Chip 18

2.2 The top view of Spirals 19

2.3 Harmonic polynomial coefficients of the trap potential 22

2.4 Dimensionless anharmonic polynomial coefficients of the trap potential 23

2.5 Relative distances between atom clouds after the second splitting . . 25

2.6 Relative velocitites between atom clouds after the second splitting . . 25

2.7 Top view of the atom chip 26

2.8 Top view of the innermost spiral 26

2.9 Bottom view of the atom chip 27

2.10 illustration of vias and soldering pads 28

2.11 A revised version of atom chip 29

2.12 An overview of testing chamber 31

2.13 The chip holder and bias coil holder 31

2.14 The top and bottom bias coil holder 32

3.1 The image sequence of atom interferometry experiment 35

3.2 An absorption image after the second splitting 37

3.3 A filtered image 38

3.4 A cropped filtered image 39

3.5 A cropped filtered image 40

3.6 The column and row indexes of the black pixels in the binarized image 40

3.7 Cropped BEC packets from the original image 42

3.8 The trajectories of four atom packets after the second splitting . . . 44

3.9 The user interface of BEC image analysis program 45

3.10 A splitting pattern for future interferometry experiments and the labeled result 46

3.11 Another splitting pattern for future interferometry experiments and the labeled result 47

3.12 The user interface of Admiral program 48

4.1 Atom packets after the recombination 51

4.2 Bias field phase effect 51

4.3 Trajectory of atoms in the interferometer 52

B.1 Interferometer outputs after recombination 79

B.2 The updated image analysis program user interface 81

List of Tables

2.1 Spiral parameters in Fig 2.7 25

2.2 Spiral parameters in Fig 2.11 29

2.3 Resistances and inductance for bias coils 32

4.1 Small parameters for laser and trap imperfections 61

4.2 Sensitive parameter terms and corresponding polynomial coefficients. 62

5.1 A-PhI Gyroscope Program Metrics 67

Acknowledgements

I would like to take the time to first thank my advisor Professor Cass Sackett. He is a great mentor and teacher. His ability to patiently answer my questions was invaluable to the experience I developed throughout this thesis project.

I am also grateful to my colleagues for their wonderful collaboration. This project would have been impossible without the support of them, especially Eddie Moan, Seth Berl and Marybeth Beydler.

I am indebted to Professor Chris Goyne, Professor P.Q. Hung and Professor Olivier Pfister for their willingness to serve on my research committee.

Finally and most importantly, I would like to thank my parents for their support and encouragement. They have always been there for me under any situation I was encountering in life.

Abstract

A Sagnac interferometer using Bose-Einstein condensates for rotation sensing is implemented in a harmonic trapping magnetic potential. The trapped cold atom cloud is manipulated by standing wave laser beams to produce two reciprocal interferometers. We designed a new atom chip for confining cold atoms based on double layer spiral copper wires with different chirality. The chip provides pure gradient magnetic field and requires less currents and power consumption. We also designed a testing chamber for the atom chip. The chamber is used to support the chip, adjust trapping frequencies and allow laser beams coming through. Besides, we developed an image processing program and an experiment control program to automate image generations, collections and analysis. These programs will significantly expedite experiment operations. Finally, we analyzed the effects of the imperfections of Bragg lasers and the trap potential on the interferometer phase shifts. The result will be helpful to control the experiment when we use the interferometer for rotation measurements in the future. The ultimate goal is to realize a compact and portable microchip-based atom gyroscope for rotation sensing and inertial navigation.

Chapter 1

Introduction

1.1 Sagnac Effect

In physics, interference is a well-known phenomenon in which two propagating waves superpose to form a resultant wave. Interference effects can be observed with all kinds of waves such as light, acoustic and matter waves. The amplitude of the resultant wave depends on the phase difference between the constituent waves. When two waves are in phase, they will interfere constructively and the resultant oscillation amplitude is the sum of amplitudes of original two waves. When two waves are out of phase, they will interfere destructively and resultant oscillation amplitude is the difference of amplitudes of original waves. In general, the interference pattern can be complicated and a device for measuring interference patterns is called an interferometer.

In a typical optical interferometer (Fig 1.1), the input laser beam is incident upon a beam splitter, which sends the two output beams into the arms of the device. When one arm includes more optical path length than the other, a relative phase develops between the two arms. A final beam splitter recombines the beams, which are then

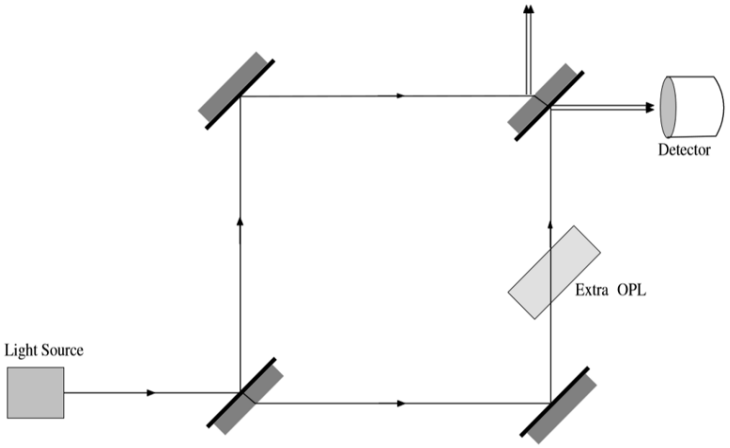


Figure 1.1: Mach-Zehnder interferometer

analyzed at the detector. The relative phase difference results in an interference pattern at the detector.

One way to introduce a phase difference between two interfering arms is to rotate the interferometer apparatus. In a ring interferometer setup (Fig 1.2), an incident beam is split and two beams are made to follow the same path but in opposite directions. When they return to the point of entry two beams are allowed to exit the ring and cause interference. When the interferometer apparatus itself is rotating during this procedure, the counter-rotating pulse will arrive at the end point slightly earlier than the co-rotating pulse, resulting in a phase difference between these two interfering arms. This interfering effect is called Sagnac effect and the related phase difference is called Sagnac phase. According to the Eq (1.1), the Sagnac phase Φ depends on the angular velocity of the interferometer setup Ω, the area enclosed by two interfering arms A [1], the wavelength λ and the velocity v of the interfering field. By measuring the Sagnac phase, we can obtain the angular velocity of the interferometer setup. Therefore, the Sagnac effect is widely used for developing gyroscopes in inertial navigation systems.

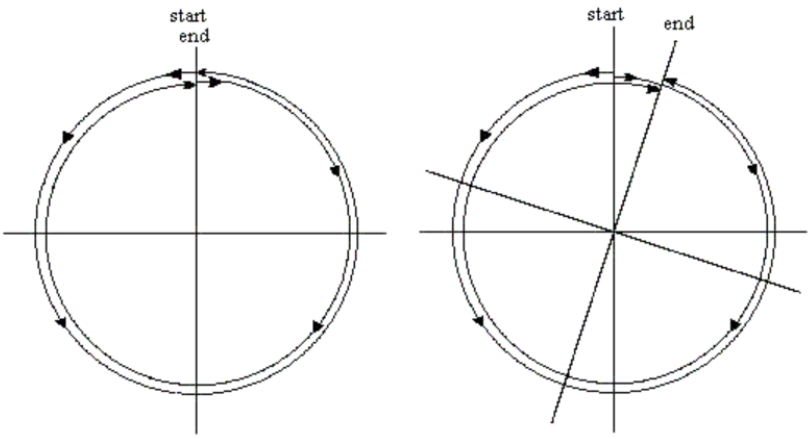


Figure 1.2: Sagnac Interferometer

$$\Phi = \frac{4\pi\Omega A}{\lambda v}. \tag{1.1}$$

1.2 Atom Interferometry

So far ring laser gyroscopes [2, 3] based on Sagnac interferometry effect have been deployed in current inertial navigation applications. One direction to improve current technology is to switch to atom-based gyroscopes. Before starting to discuss Atom Interferometry, I introduce the concept of Bose-Einstein condensation (BEC) at first. In quantum mechanics, all particles are either bosons or fermions. Fermions are forbidden from sharing a quantum state by Pauli exclusion principle. Bosons, on the other hand, are able to share the same quantum state. When a gas of bosons at low densities is cooled to temperatures close to absolute zero, bosons are able to undergo a phase transition to BEC. Under such conditions, a large fraction of bosons occupy the lowest quantum state. The critical transition condition is when the de Broglie wavelength of the particles λ_{dB} becomes comparable to the inter-particle distance,

$n^{-1/3}$ [4], for atomic density n. For a gas in equilibrium at temperature T, the thermal de Broglie wavelength is

$$\lambda_{dB} = \sqrt{\frac{2\pi\hbar^2}{mk_BT}}. \tag{1.2}$$

k_B is the Boltzmann constant, $\hbar$ is the Planck constant. m is the mass. The requirement for BEC transition [5] can be expressed as

$$k_BT(n) = \frac{2\pi\hbar^2}{m}\left(\frac{n}{2.612}\right)^{2/3} \tag{1.3}$$

From Eq (1.3), we can see that the critical temperature depends on the density of the atom cloud used. The phase transition condition can also be expressed through phase-space density. Phase space density is defined as the number of particles contained in a cubic volume with the sides equal to the thermal De Broglie wavelength

$$\rho = n\lambda_{dB}^3 = n\left(\frac{2\pi\hbar^2}{mk_BT}\right)^{3/2} \tag{1.4}$$

The transition to BEC happens when $\rho \geq 2.612$. A combination of density increasing and temperature reducing techniques are used to achieve the transition phase space density [6].

Since the first experimental demonstrations of Bose-Einstein condensation [7], it has shown that ultracold matter can play an important role in applications such as atomic clock [8], gravimetry [9, 10, 11] and inertial sensors [12, 13, 14] because of the wave-particle duality. The advantage of atoms for inertial sensing compared to photons can be demonstrated through the calculation of Sagnac phases for both particles. The optical field has wavelength $\lambda_{photon} = 2\pi c/\omega$, while the atomic wavelength is given by $\lambda_{atom} = h/mv$, m is the mass of an atom and v is the atomic velocity. If we take the ratio of phase sensitivity $\Phi_{atom}/\Phi_{photon}$ for the two types of particles, we

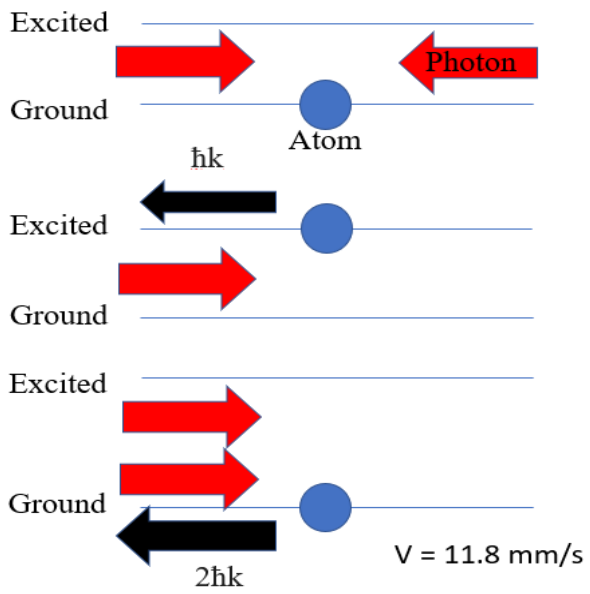


Figure 1.3: The splitting principle for atom interferometry

find a relative phase sensitivity of

$$\frac{\Phi_{atom}}{\Phi_{photon}} = \frac{\lambda_{photon} v_{photon}}{\lambda_{atom} v_{atom}} = \frac{2\pi c^2}{\omega} \frac{mv}{hv} = \frac{mc^2}{\hbar\omega} \tag{1.5}$$

For visible wavelength light and middle-weight atoms, this ratio is on the order of 10^{10}, indicating the fundamental limit on the sensitivity of an atom-based device substantially exceeds that of its photon-based counterpart.

In atom interferometry experiments, we use standing wave laser beams called Bragg laser [15] to split and combine atoms. The atom can absorb a photon from one direction, obtain a $\hbar k$ momentum and go into a virtual excited state. Then the atom is stimulated to emit a photon towards the opposite direction, obtains another $\hbar k$ momentum for itself and goes back to the original ground state. Of course, it can also absorb a photon from the other beam at first and move towards the opposite direction. In our experiment, we use rubidium atoms and the laser wavelength is about 780 nm, which give the splitting velocity $v_B = 2\hbar k/m = 11.8$ mm/s.

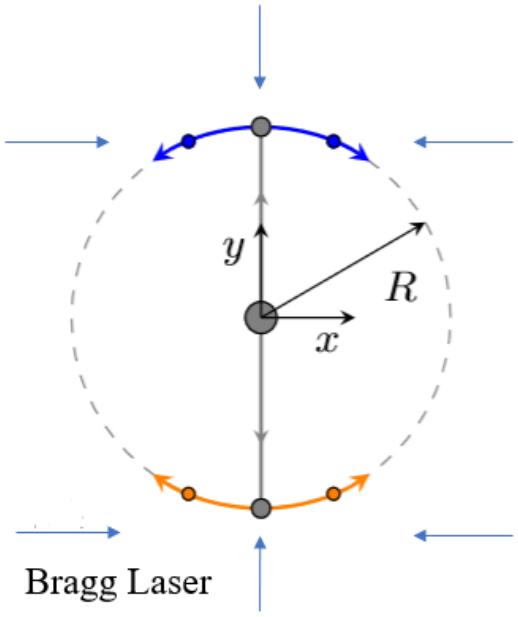


Figure 1.4: The splitting trajectory for atom interferometry

In our atom interferometry experiment [16], the atoms are split twice in a cylindrically symmetric harmonic trapping potential (Fig 1.4). The first split produces two atom packets moving along the diameter of the trap with opposite velocities in y direction. When the packets are slowed down by the harmonic potential and reach to their turning points after a quarter period, the atom packets are split again in x direction. Each atom packet will move around the magnetic trap in a circular orbit. After one period of oscillation, another laser pulse along x direction is applied to recombine the packets and create two interferometers at the top and bottom. The number of atoms that return to rest in each interferometer will depend on the Sagnac phase, which is a differential phase and caused by earth rotation rate, as well as other phases caused by laser noise and background fields that are common to both interferometers. By measuring the Sagnac phase we can obtain the earth rotation rate at our lab location.

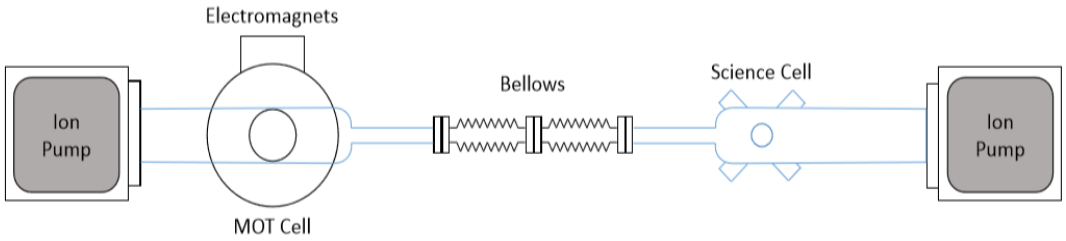


Figure 1.5: The vacuum system for atom interferometry experiment [17]

1.3 Experiment Apparatus

One of the main topics of this thesis is to design a new apparatus for the interferometry experiment. Our current experiment apparatus is a vacuum system [17] (Fig 1.5) including two chambers, MOT cell and Science cell. In MOT chamber we use Magnetic Optical Trap (MOT) [18] to initially capture and cool atoms. Then we use optical pumping to gather all the atoms to a trappable substate [19]. After that we switch to a dc spherical quadruple trap to capture atoms and physically move the atoms to the other chamber with a better vacuum. The magnetic trap is generated by coils mounted on the translation stage. Once the atoms have arrived in the science cell, we cool the atoms further using evaporative cooling [20] and use time-orbiting trap (TOP) [21] to prevent atoms from losing its quantization axis and being ejected from the trap. Finally, we have Bose-Einstein condensates and conduct interferometry experiment in the Science Cell.

Besides the vacuum chambers, we also need lasers to manipulate the atoms. To establish a Magneto-Optical Trap we need to have a cycling transition that when atoms absorb a photon the atoms will spontaneously decay back to the initial state they were in before the photon absorption. The cooling beam used in the MOT operates on one such cycling transition between energy levels $5^2S_{1/2,F=2}$ and $5^2P_{3/2,F=3}$ shown in Fig 1.6. However, there is a small probability that the atoms will be excited by the cooling beam to the other state $5^2P_{3/2,F=2}$, which can decay down to another

S state $5^2S_{1/2,F=1}$. This state is dark to the cooling beam. So a second repumping laser is required to pump the atoms out of the dark state. This repumping laser operates between $5^2S_{1/2,F=1}$ and $5^2P_{3/2,F=2}$ energy levels. In addition, a laser is required operating between the energy levels $5^2S_{1/2,F=2}$ and $5^2P_{3/2,F=2}$ to be used in the optical pumping stage of condensate production. This beam is derived from the MOT laser using an acousto-optic modulato. Finally, a probe beam derived from the MOT laser is required to image the atoms. Prior to the imaging step of the experiment, the atoms are prepared in the ground state $5^2S_{1/2,F=2}$. We utilize absorption imaging and therefore use a beam near-resonant with the same transition used as the cooling beam in the MOT.

1.4 Previous Work

In our previous experiment, the cold atoms are trapped in a magnetic field [12]. The magnetic field of the trap is generated by six coil chips shown in Fig 1.7. They are grouped as three pairs to form a cube. The side length of the cube is 21 mm. Each coil consists of a strip of copper arranged in a spiral pattern on an aluminum nitride substrate. The chips were made using a photolithography technique [22]. We supply the coils with electric currents in each direction [23].

$$I_x = I_0 \sin(\Omega_1 t) \cos(\Omega_2 t) \tag{1.6}$$

$$I_y = I_0 \sin(\Omega_1 t) \sin(\Omega_2 t) \tag{1.7}$$

$$I_{z+} = (I_0 - I_1) \cos(\Omega_1 t) \tag{1.8}$$

$$I_{z-} = (I_0 + I_1) \cos(\Omega_1 t) \tag{1.9}$$

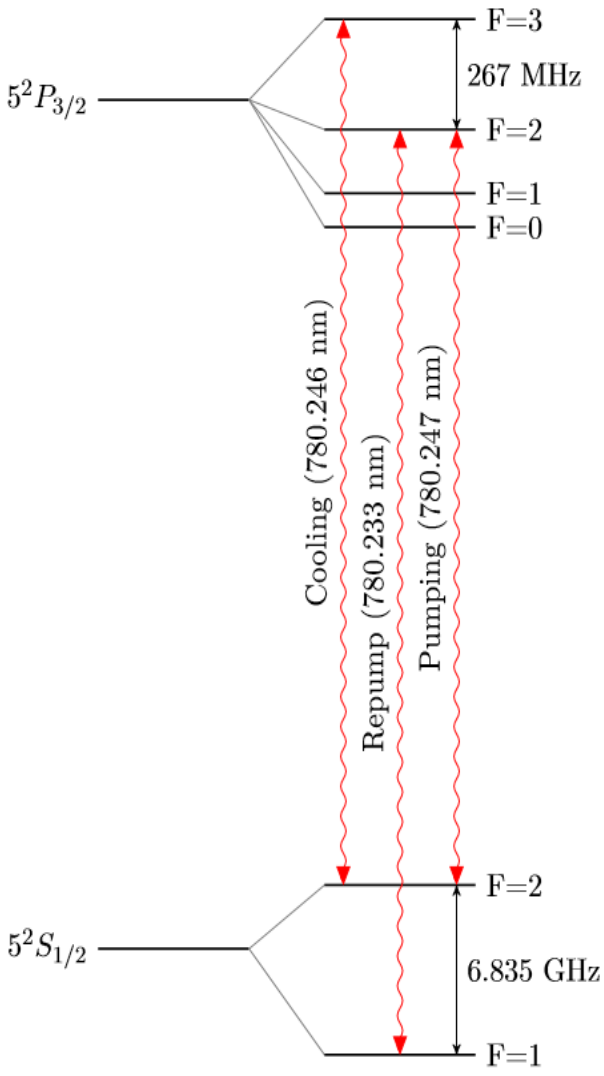


Figure 1.6: Relevant energy level of Rb [12]

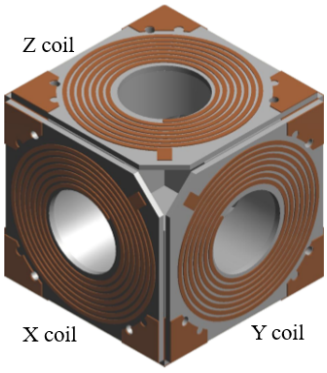


Figure 1.7: Previous atom trap

The current I_0 is used to provide a bias field and I_1 is used to provide a gradient field. Based on Biot-Savart law, we can get such expressions for gradient and bias field.

$$B_{bias} = B_0 \begin{pmatrix} \sin(\Omega_1 t)\cos(\Omega_2 t) \\ \sin(\Omega_1 t)\sin(\Omega_2 t) \\ \cos(\Omega_1 t) \end{pmatrix} \tag{1.10}$$

$$B_{quad} = B_1^{'} \cos(\Omega_1 t) \begin{pmatrix} x/2 \\ y/2 \\ -z \end{pmatrix} \tag{1.11}$$

where Ω_1 is $2\pi \times$ 10 kHz, Ω_2 is $2\pi \times$ 1k Hz. Then we calculate the magnitude of the sum of these two fields. We extract out a coefficient B_0 and consider all internal terms except 1 as ϵ and calculate its Taylor expansion to the second order.

$$\begin{aligned} |\overrightarrow{B}| &= |\overrightarrow{B}_{bias} + \overrightarrow{B}_{quad}| \\ &= B_0\{1 + \frac{B_1^{'}}{B_0}\sin\Omega_1 t\cos\Omega_1 t(x\cos\Omega_2 t + y\sin\Omega_2 t) - \\ &\quad 2\frac{B_1^{'}}{B_0} z(\cos\Omega_1 t)^2 + \frac{B_1^{'\,2}}{B_0}(\cos\Omega_1 t)^2(\frac{x^2+y^2}{4} + z^2)\}^{1/2} \end{aligned} \tag{1.12}$$

$$\sqrt{(1+\epsilon)} \approx 1 + \frac{1}{2}\epsilon - \frac{1}{8}\epsilon^2 + ... \tag{1.13}$$

And since the frequency of magnetic fields are much larger than the oscillation frequency of atoms, we calculate the time averaged magnetic field.

$$\langle f \rangle = \lim_{T\to+\infty} \frac{1}{T}\int_0^T f(t)dt \tag{1.14}$$

Then we multiply time averaged field with Bohr magneton and include gravitational potential to get the expression for harmonic trap.

$$U_{total} = \mu_B B_0 + mgz - \frac{1}{2}\mu_B B_1^{'} z + \frac{1}{2}m\omega_x^2 x^2 + \frac{1}{2}m\omega_y^2 y^2 + \frac{1}{2}m\omega_z^2 z^2 \tag{1.15}$$

We can adjust the potential term including gradient field $B_1^{'}$ and make it canceled with the gravitation potential mgz. And Eq (1.16) and (1.17) shows the trapping frequencies in x, y and z direction, which indicate that the magnetic trap is cylindrically symmetric.

$$\omega_x = \omega_y = \left[\frac{2\mu_B}{m}\left(\frac{7}{128}\frac{B_1^{'2}}{B_0}\right)\right]^{1/2} \tag{1.16}$$

$$\omega_z = \left[\frac{2\mu_B}{m}\left(\frac{1}{16}\frac{B_1^{'2}}{B_0}\right)\right]^{1/2} \tag{1.17}$$

Magnetic traps provide potentials that act as a simple container for storing cold atoms. In addition to conventional traps made from macroscopic coils, the concept of magnetic microtraps have been introduced [24, 25]. These traps use miniaturized current conductors for generating the magnetic field of almost arbitrary geometry. The initial motivation for the development of microtraps was the strong confinement that results from miniaturization of the trap. However, by using chip technology, additional tools and devices can be integrated so that today microtraps are discussed in a far more general context. Promising areas of application include model systems for three- and one-dimensional quantum gases [26], quantum information processing with neutral atoms [27], integrated atom optics [28, 29], matter-wave interferometry [30, 31], etc.

For now, we're making a chip trap to replace the cube trap. The new trap should be more stable and enable faster BEC production. Besides, the above expression for the trapping potential is an ideal case. In reality the magnetic trap can differ from the cylindrically harmonic potential. Possible reasons include: actual bias fields

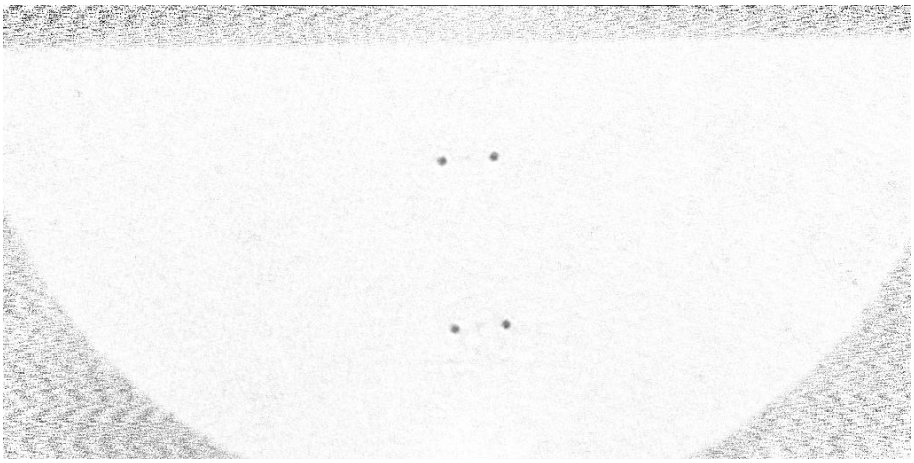


Figure 1.8: Absorption image of BEC atom clouds

aren't perfectly uniform, the actual gradient isn't perfectly linear, actual coils aren't all identical, the leads to the coils will contribute to the field, and there are other conducting objects around which can generate eddy current fields.

If the trapping potential isn't perfect, there are other contributions to the phase which need to be understood. In a later chapter we explore how the imperfections of Bragg laser beams and trapping potential affect the Sagnac phase.

Our research group realized the interference trajectory about two years ago [16]. I contributed to the progress by deriving the analytical form of Sagnac phase and comparing to the experiment result [32]. We want to derive the actual magnetic potential from trajectories of atom packets and compare with the ideal potential. In order to do that, we need to extract the locations of the atom clouds from the trajectories. Before we realized this circular interference trajectory, we have limited amounts of images like this (Fig 1.8) for processing. The BEC atom clouds are imaged using absorption imaging. Absorption imaging presents dark absorption profiles on a bright background. A probe beam tuned to the transition as shown in Fig 1.6 illuminates the atoms and the surrounding area. The photons in the region around the atoms move through the space without obstruction. The photons in the region

intercepted by the atoms are absorbed. The amount and the area of absorption depends on the density and the size of the atom cloud respectively. The shadow caused by the absorption can be analyzed to obtain the density and the size of the atom cloud.

In order to get the positions and sizes of BEC clouds from an image, we have to manually select a square region around each packet and fit the pixel values and the coordinates to a Gaussian function. As we are getting more and more images and more clouds from a single image for our interferometry experiment, we need a faster image analysis program to quickly extract the important information from an image dataset. And we'll be taking more data faster, we need a more automated experiment control system. Therefore, two experiment programs are built to satisfy the requirements.

1.5 New Experiment Setup

The efforts included in this thesis are used to meet the requirements of an Atomic-Photonic Integration(A-PhI) program: a Bragg Interferometer Gyroscope in a Time-Orbiting Potential (BIGTOP). BIGTOP is a rotation sensor using a Bose-Einstein Condensate to implement two reciprocal Sagnac interferometers in a magnetic trap.

The BIGTOP system, illustrated in Fig 1.9, is designed to rapidly produce BEC in a compact device while maintaining the vacuum quality necessary for sensitive interferometry. BIGTOP will produce a condensate in 5 seconds followed by a 5 second measurement cycle. Each interferometer will enclose one or more circular orbits of area 11 mm^2.

BIGTOP's dual Magneto-Optical Trap (MOT) design collects rubidium atoms

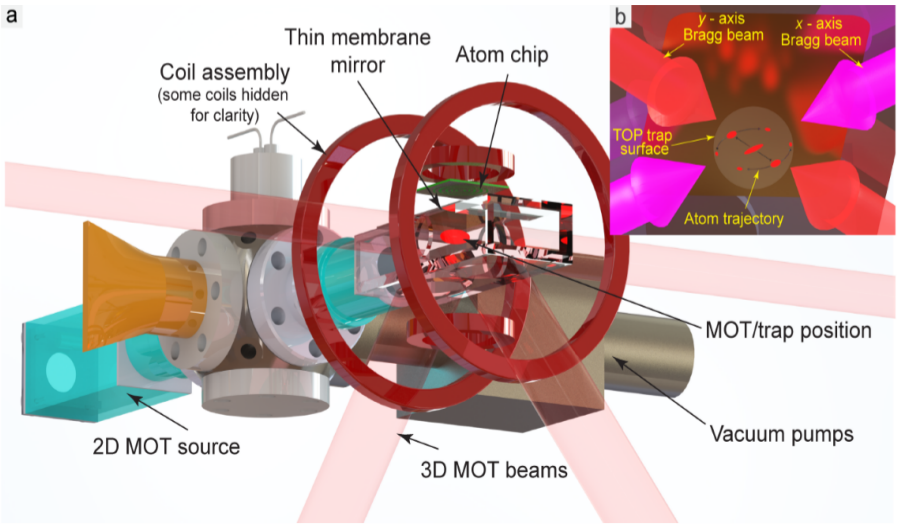


Figure 1.9: BIGTOP system concept. (a) Overview of system components, with some coils and beams omitted for clarity. (b) Detail of trap region with interferometry scheme [33]

from an alkali vapor and transfers them to an ultra-high vacuum (UHV) cell. The 3D MOT is produced near a thin mirrored surface of the UHV cell. On the non-vacuum side of the thin surface, strong magnetic fields are produced by a planar chip with micro-fabricated current carrying wires. The field from this atom chip is complemented by an array of magnetic coils oriented orthogonally around the UHV cell. These produce a static magnetic trap which captures the cold atoms and allows for efficient evaporative cooling. The static trap is subsequently converted to a TOP trap in which a BEC forms. Finally, the trap confinement is weakened to enable large packet separation during the interferometry cycle.

1.6 Scope

The scope of this thesis is to describe the simulations and designs of a new atom chip apparatus for the next generation interferometry experiment in our lab. The objectives of my study includes: 1. To design an atom chip which provides pure gra-

dient magnetic field for trapping atoms and a testing chamber which adapts to the chip and overcomes the thermal variation problem. 2. To built experiment programs which expedite image generations, collections and analysis. 3. To analyize the contributions of the imperfections of Bragg laser and trap potential to the interferometer phase shifts.

The thesis consists of four additional chapters beyond the current introductory chapter:

Chapter 2: Atom Chip: In this chapter, we start with the theoretical design of atom chip which includes its geometric pattern and simulations of the generated magnetic fields and required currents. Then we will introduce its mechanical design and fabrication in detail. We include the AutoCAD drawings and explanations for each part such as soldering pads and vias. We also describe the design of a testing chamber for the atom chip. AutoDesk Inventor drawings and explanations for each component such as atom chip holder, bias coil holder and base are included.

Chapter 3: Experiment Program: This chapter mainly includes descriptions for two experiment programs. The first one is an image analysis program, which is developed to quickly extract positions and sizes of atom packets from their trajectories. Then we introduce the user interface and related principles: image denoise, image binarization, clustering algorithm, object tracking and atom packets labeling for different trajectories. The second experiment program is mainly used to automate image generations and collections. At last we introduce the user interface and functions.

Chapter 4: Sagnac Phase Calculation: In this chapter we explore the imperfections of Bragg laser beams and trapping potential. We introduce small parameters to express the directions of laser beams and derive the ideal Sagnac phase expression at first. Then we incorporate some anharmonic terms into the magnetic trap potential

and see how the Sagnac phase varies according to small parameters.

Chapter 5: Conclusion: This chapter summarizes the results of all previous chapters. Additionally, future improving directions are discussed.

Chapter 2

Atom Chip

For our future atom interferometry experiment, we want to replace the previous coils cube with a two-dimensional atom chip (Fig 2.1) to reduce the power consumption and volume. It also has better thermal stability and produces a tighter trap for fast BEC production. We want to use this chip in our interferometry experiment for rotation sensing.

The chip is made of aluminum nitride clad with copper foil that is cut into conducting traces. We need this chip to make the gradient field for the TOP trap and a tight gradient for BEC production. The chip will be surrounded by other coils which provide the bias field. We hope the size of the chip is within 35 mm and its power consumption is within 10 Watts. I will discuss the design procedure in this chapter.

2.1 Theoretical Design of Atom Chip

On this chip the strips of copper are in Archimedes spiral pattern (Fig 2.2). Its radial distance increases linearly with its polar angle: $r = r_0 + \theta p^{'}$, where r_0 (mm) is the

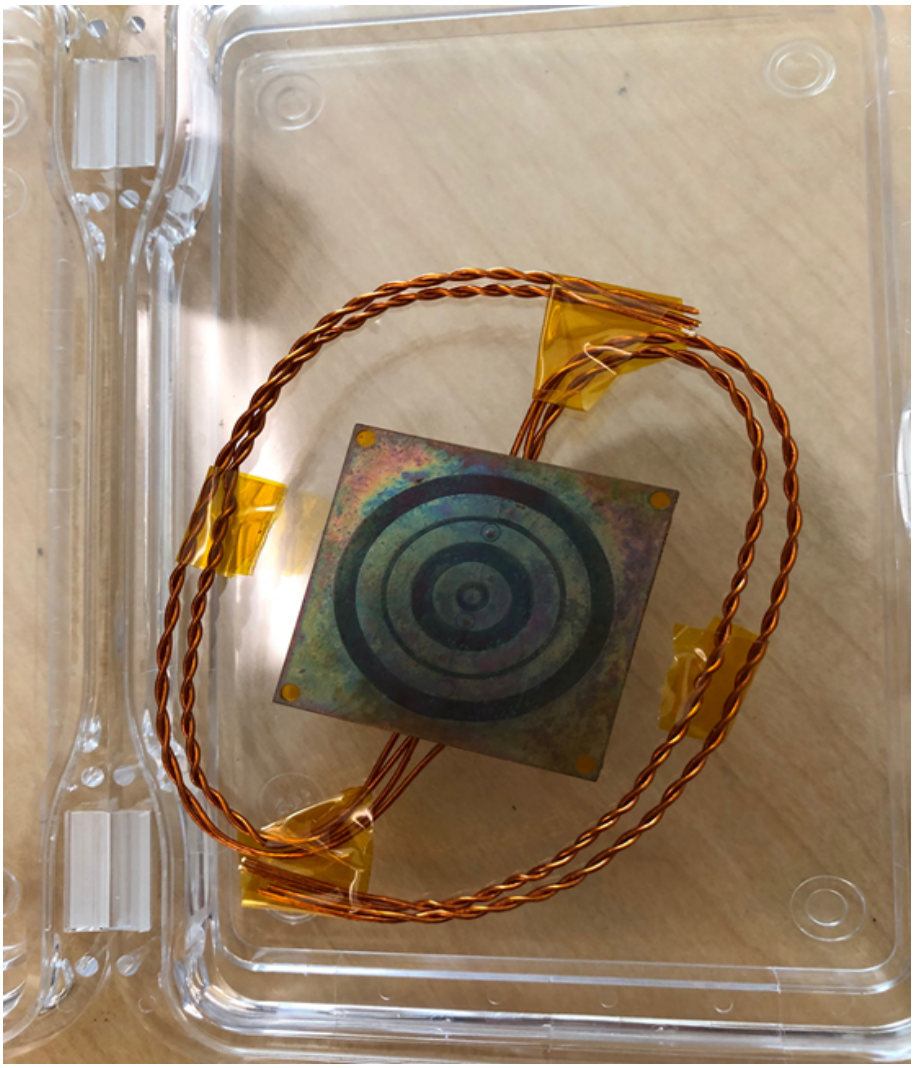


Figure 2.1: Atom Chip

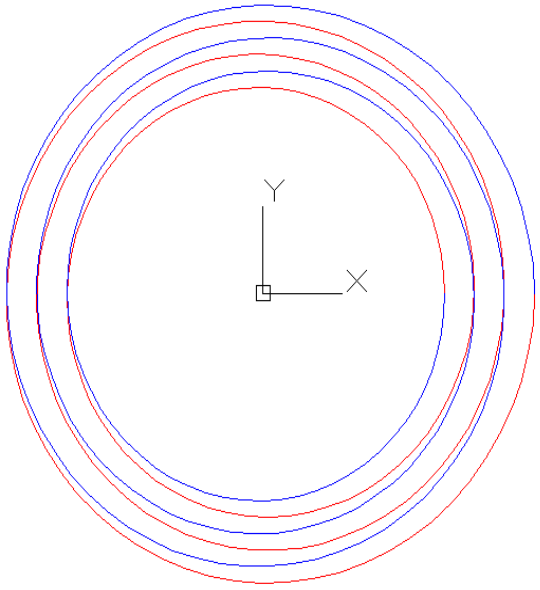


Figure 2.2: The top view of Archimedes Spirals. The red and blue spiral represent two layers respectively. The red one is at the top and the blue one is at the bottom

initial radial distance and θ (rad) is the polar angle. $p^{'} = p/2\pi$ where p (mm/turn) is the pitch of the spiral. It has two layers on the top and bottom of the chip with different chirality to reduce its asymmetry. The two layers also reduce the required current and power compared to a single layer. These two layers have the same r_0 but their pitches are opposite to each other.

Based on Biot-Savart law,

$$\overrightarrow{dB} = \frac{\mu_0 I}{4\pi} \frac{d\overrightarrow{l} \times \overrightarrow{R}}{R^3} \tag{2.1}$$

we can calculate the magnetic field caused by this structure. For example, we assume a spiral is located on x-y plane. A small current line segment $d\overrightarrow{l}$ is expressed as $(dx, dy, 0)$. In polar coordinate system, it is $((\cos\theta dr - r\sin\theta d\theta), (\sin\theta dr + r\cos\theta d\theta), 0)$. Besides, its position is $(r\cos\theta, r\sin\theta, 0)$. To calculate the magnetic field at any point in space (X, Y, Z), the vector pointing from the current line segment to the space point $\overrightarrow{R}$ is $(X–r\cos\theta, Y–r\sin\theta, Z)$. Therefore, the magnetic field dB in

each direction is

$$dB_x = \frac{\mu_0 I}{4\pi} \frac{Z(\frac{\sin\theta}{2\pi}p + r\cos\theta)d\theta}{((X - r\cos\theta)^2 + (Y - r\sin\theta)^2 + z^2)^{3/2}} \tag{2.2}$$

$$dB_y = -\frac{\mu_0 I}{4\pi} \frac{Z(\frac{\cos\theta}{2\pi}p - r\sin\theta)d\theta}{((X - r\cos\theta)^2 + (Y - r\sin\theta)^2 + z^2)^{3/2}} \tag{2.3}$$

$$dB_z = \frac{\mu_0 I}{4\pi} \frac{(Y\cos\theta - X\sin\theta)\frac{d\theta}{2\pi}p + (-Yr\sin\theta - Xr\cos\theta + r^2)d\theta}{((X - r\cos\theta)^2 + (Y - r\sin\theta)^2 + z^2)^{3/2}} \tag{2.4}$$

Where r can be substituted with $r_0 + \theta p^{'}$. We use two sets of this structure for interferometry experiments. Their numbers of turns are both 20. Their pitches are both 0.12 mm/ turn. This is set by the resolution of the fabrication process. Because we want to make the size of outer spiral set no larger than 15 mm, we set its starting radius to be 12.6 mm. For the inner spiral set, we get the intuition from the magnetic field on the axis of a current loop that maximizes the gradient at the desired trap point. The magnetic field of a current loop is proportional to $R^2/(z^2 + R^2)^{3/2}$, where R is the radius of the loop and z is the distance on z axis from the loop center. We set $z = 2$ mm, take second order derivative respect to R and make the curvature field equal to 0. This gives R = 4.6 mm, which should be the ideal average radius of inner spirals. But the actual average radius is larger in order to accommodate the size of the innermost spiral(introduced later) for making BEC. And we want the inner spiral to match the loop as well as possible. So we decide that the starting radius of the inner spiral set is 4.25 mm. The distance between the spirals at the top and bottom is 0.76 mm. We will be using two spirals to produce pure gradient at the location of the atoms. We assume I_1 and I_2 are currents for the inner and outer spiral set respectively and z is the distance above the center of top spirals. At the center of the trap potential we want to adjust I_1 , I_2 and z to make bias field B_z and curvature field $B_z^{''}$ equal to 0 and gradient field $B_z^{'} = 31$ Gauss/cm to be canceled with gravity, which simplifies the analysis and operation if the chip provides just the gradient for

the TOP trap.

$$B_z(I_1, I_2, z) = 0 \tag{2.5}$$

$$B_z^{'}(I_1, I_2, z) = \frac{2mg}{\mu_B} \tag{2.6}$$

$$B_z^{''}(I_1, I_2, z) = 0 \tag{2.7}$$

The calculation result shows that I_1 and I_2 are 0.5 A and 1 A respectively. The distance z is 2 mm.

Based on these currents and the distance, we can calculate the magnetic trap potential. First of all, we calculated the magnetic field generated by these spirals at 125 ($5 \times 5 \times 5$) space points within 0.1 mm. Then we follow the same idea in the introduction chapter to calculate the time averaged superposition of the magnetic field B_{ave} generated by spirals and the bias magnetic field.

$$B_{ave} = \frac{1}{2\pi}\int_0^{2\pi}((B_{bias}\cos 10\Theta\cos\Theta + B_x\sin 10\Theta)^2 + (B_{bias}\cos 10\Theta\sin\Theta) + B_y\sin 10\Theta)^2 + (B_{bias}\sin 10\Theta + B_z\sin 10\Theta)^2)^{1/2}d\Theta \tag{2.8}$$

In the introduction chapter the time averaged field calculation is integrated over time. Here we replace $\Omega_2 t$ with Θ. Then $\Omega_1 t$ is 10Θ. At last, we multiply the time averaged field with Bohr magneton and plus gravitational potential energy to get the trap potential at each space point. We fit the trap potential with the space coordinates (X,Y,Z) using polynomial functions. Our experiment will run the bias magnetic field between 4 to 20 Gauss. We want to see how the magnitudes of harmonic potential terms vary with magnetic fields. Fig 2.3 shows results for the change of harmonic

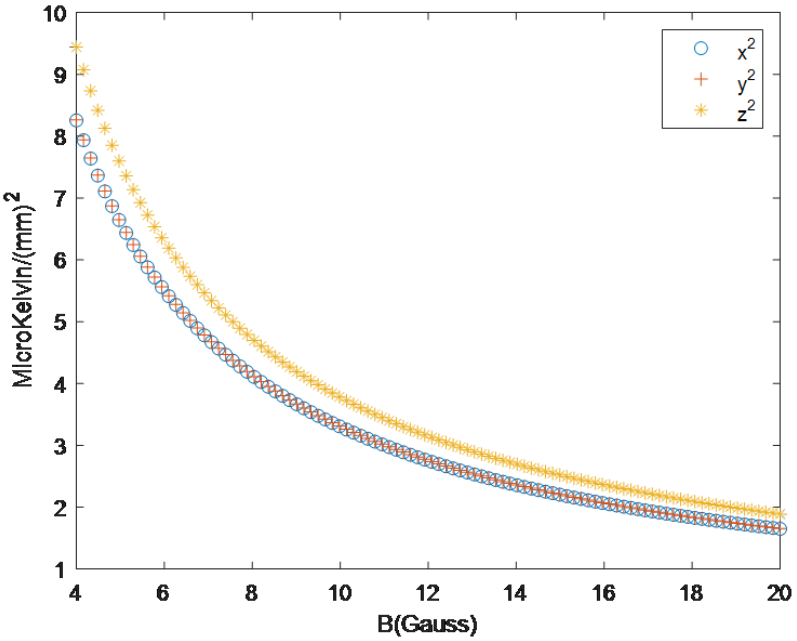


Figure 2.3: Harmonic polynomial coefficients of the trap potential

polynomial coefficients respect to bias magnetic field.

We can see the coefficients for x^2 and y^2 term are the same and different from z^2 term, which indicates that the trap potential is cylindrically harmonic.

Besides harmonic terms, we also need to consider anharmonic terms which may cause the atom packets to miss each other when they are supposed to recombine. The polynomial coefficients for harmonic terms represent $\frac{1}{2}m\omega^2$. Given ω, the amplitude of the harmonic oscillation $R = v_B/\omega$. v_B is the splitting velocity 11.8 mm/s. Then we can obtain dimensionless anharmonic polynomial coefficients by rewriting expressions for the third and fourth order anharmonic terms. For example, the third order term ax^3 can be written as $a^{'}\frac{1}{2}mv^2(\frac{x^3}{R^3})$, where $a^{'}$ is the dimensionless polynomial coefficient for x^3 term. The fourth order term bx^4 can be written as $b^{'}\frac{1}{2}mv^2(\frac{x^4}{R^4})$, where $b^{'}$ is the dimensionless coefficient for x^4 term. In this way we can compare how significant those anharmonic terms are compared to harmonic terms.

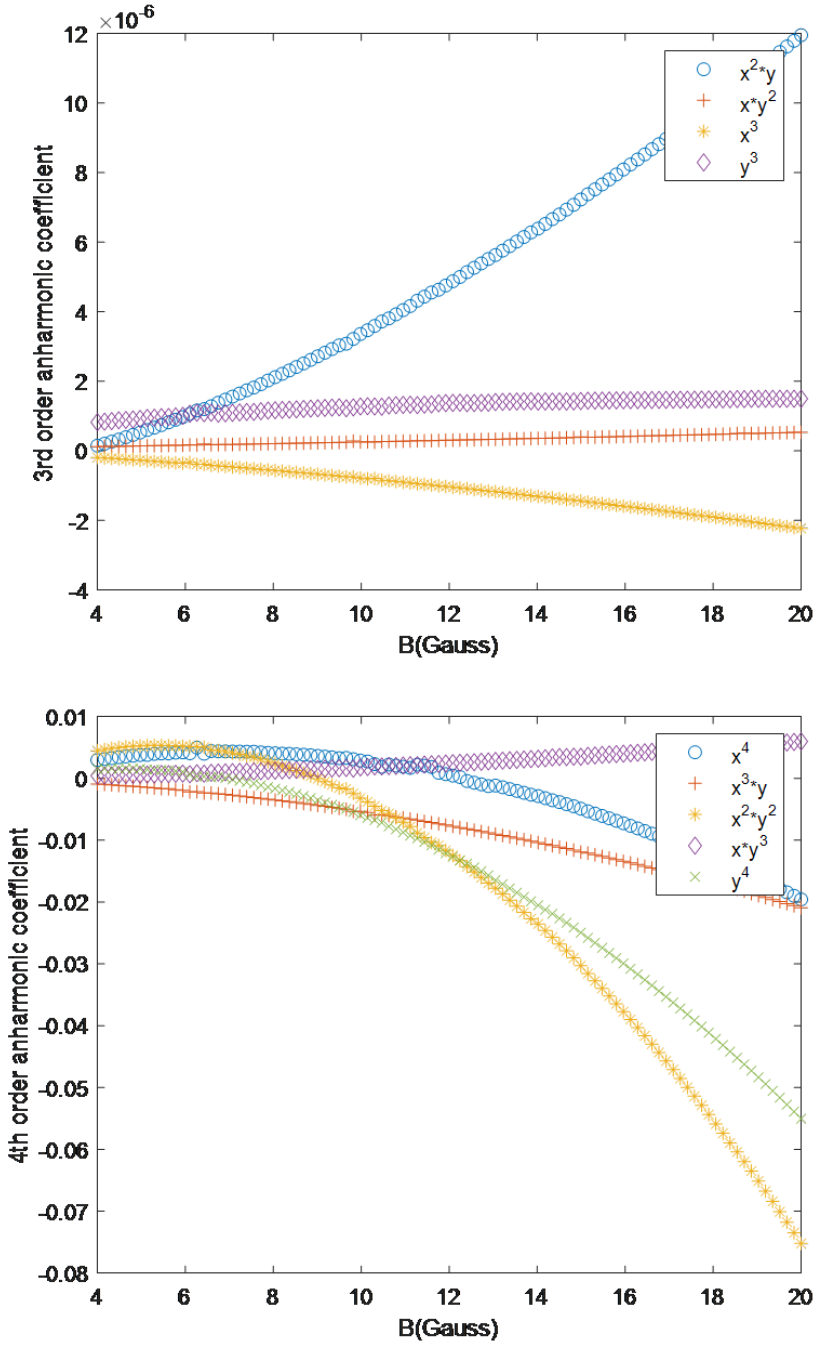


Figure 2.4: Dimensionless anharmonic polynomial coefficients of the trap potential

As you can see in Fig 2.4, the trap potential mainly consists of second order harmonic terms. There are some higher order anharmonic terms as well, but the coefficients are much smaller (2 order less) than the harmonic coefficients.

The trajectories of the interferometer must meet up in order to get interference. We try to simulate the effects of anharmonic terms on the trajectories of atom clouds. Assuming that the atom clouds move inside the trap potential according to classical Newton equations, we plot the change of distances between atom clouds respect to time after the second splitting in Fig 2.5 and the corresponding change of magnitudes of relative velocities between atom clouds respect to time in Fig 2.6 for a 6.37 Hz trap. We found that when the atom clouds are supposed to recombine after one period of oscillation, the relative distances between the atom clouds at the top and bottom are 1.05 μm and 1.01 μm respectively. And the relative velocity percentage changes between the atom clouds are 0.13% (30.7 μm/s) and 0.19% (44.8 μm/s) respectively. According to Thomas-Fermi theory [34], the size of BEC cloud L is 18.6 μm, which is larger than the relative distances between atom clouds. And the velocity width is $\hbar/mL = 40$ μm/s, which is in the same order with the velocity change caused by the trap potential.

Therefore, the velocity errors caused by unharmonic terms are significant and could limit the performance of interferometers, which requires further study.

2.2 Mechanical Design and Fabrication

After finishing the theoretical simulation for the atom chip, we need to consider its actual mechanical design for the fabrication purpose. Its mechanical design is shown in Fig 2.7 and Fig 2.8. It is designed in AutoCAD. The size of the chip is 35.34 mm.

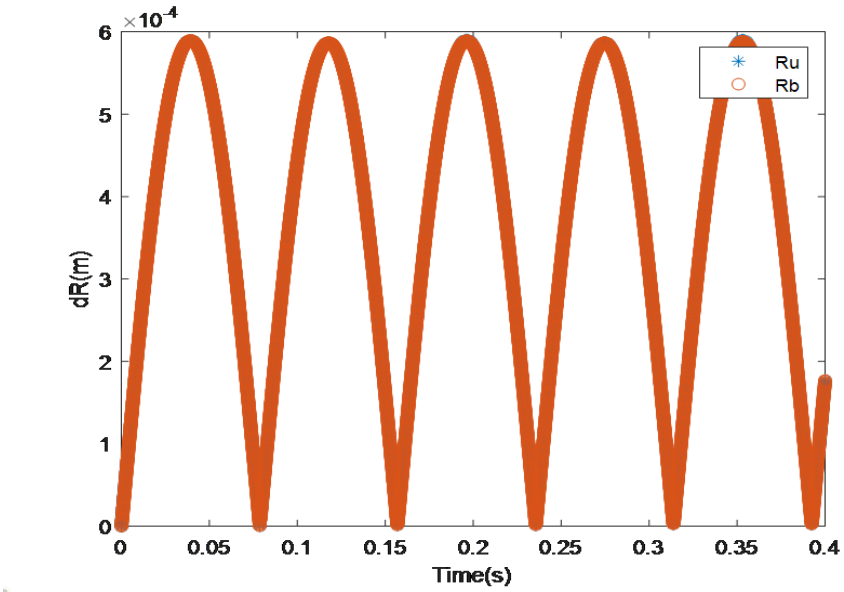


Figure 2.5: Relative distances between atom clouds after the second splitting

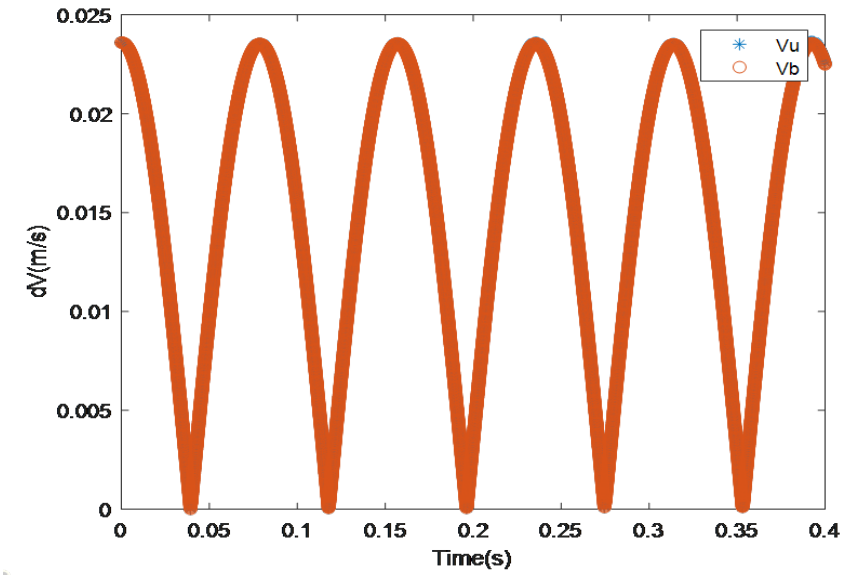


Figure 2.6: Relative velocitites between atom clouds after the second splitting

Spiral	Number of turns	Starting radius(mm)	Ending radius(mm)	Length(mm)
1	5	1.23	1.83	48.07
2	20	4.25	6.65	684.87
3	5	9.29	9.89	301.28
4	20	12.6	15	1734.16

Table 2.1: The number of turns, starting radius and ending radius for each spiral structure in Fig 2.7. The chip thickness is 0.76 mm

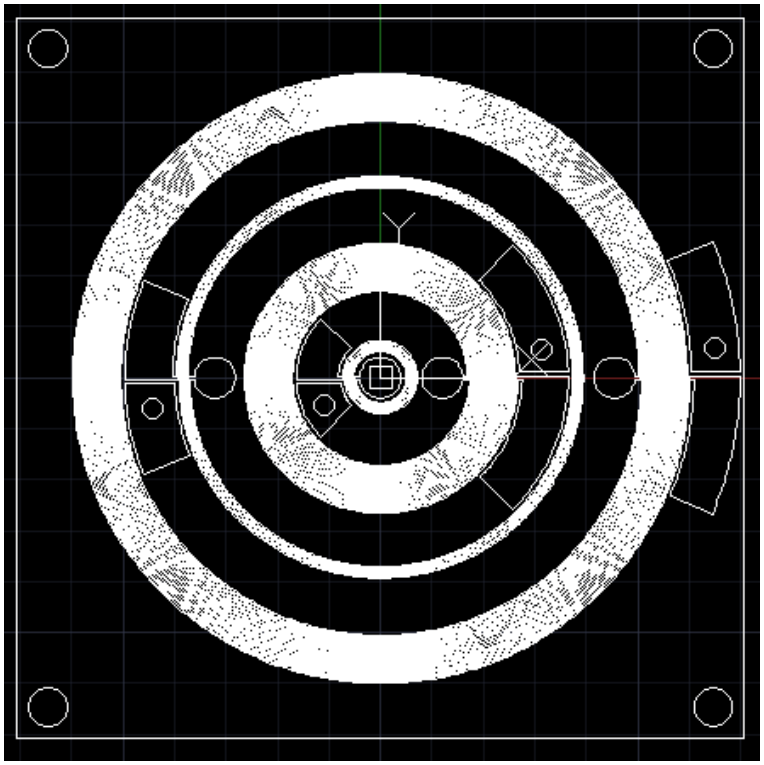


Figure 2.7: Top view of the atom chip

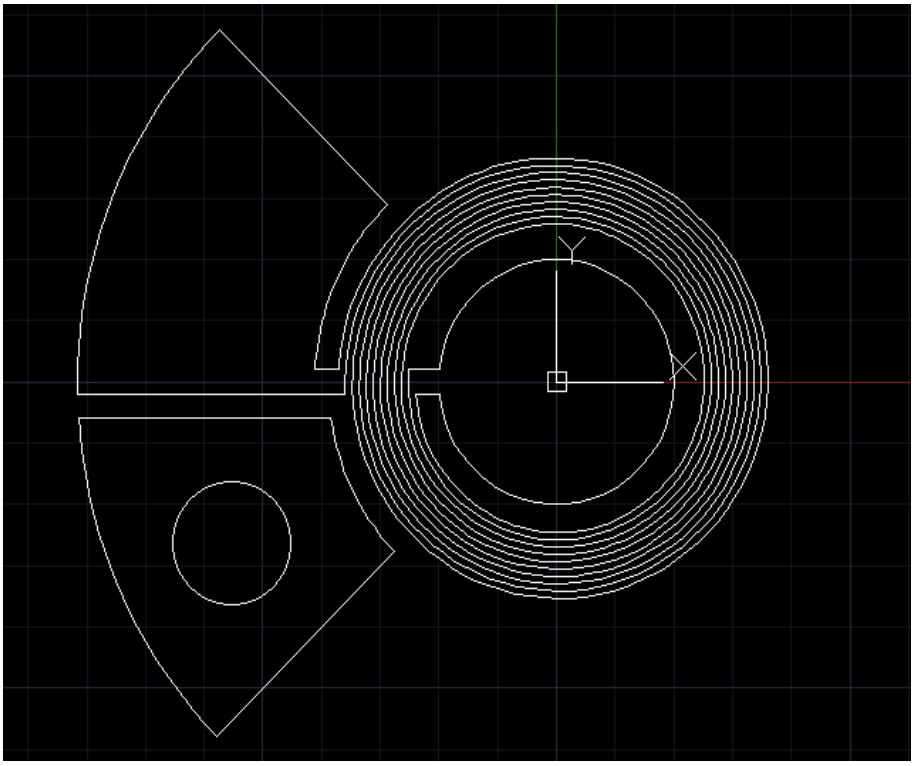


Figure 2.8: Top view of the innermost spiral

Figure 2.9: Bottom view of the atom chip

Fig 2.7 is the top view of the chip. We have four sets of the anticlockwise spirals. The innermost spiral is used to provide a strong gradient field for evaporative cooling when we make BEC. Once a condensate is produced, the chip field will be relaxed to produce a weaker trap for interferometry. The starting radius of the innermost spiral is 1.23 mm and It has 5 turns. The two thickest spirals are for interferometry experiment. They both have 20 turns of spiral. The pitch of all spirals is 0.12 mm/turn. The width of copper wire is 0.06 mm and the width of the channel between wires is also 0.06 mm. The thickness of copper wire is 0.125 mm. Considering the resistivity of copper is 1.68×10^{-8} (Ohm $\cdot$ m), the power consumption of the chip for interferometry experiments is about 4.27 W. Another spiral between the two thickest spirals is used to provide a degree of freedom and help control the shape of the trap potential. Its starting radius is 9.29 mm and it has 5 turns. Fig 2.9 shows the bottom view of the atom chip. Those spirals are still drawn anticlockwisely. But since during

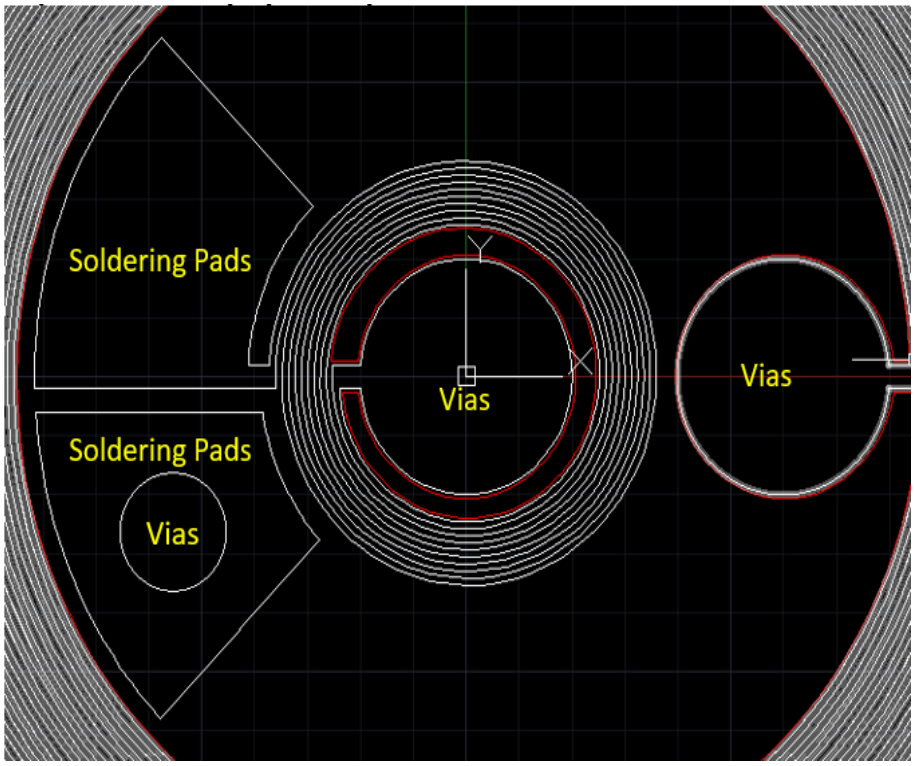


Figure 2.10: illustration of vias and soldering pads

the fabrication the chip will be flipped 180 degree, the spirals from the bottom view will become clockwise from the top view.

Besides the spirals composed of copper wires, we also need soldering pads and vias connected with spiral copper wires as shown in Fig 2.10.

Currents will flow into and out of the spirals through the soldering pads. And they flow between top and bottom layer through these vias. The distance between the edges of soldering pads and spiral copper wires is 0.2 mm. The radius of vias is 1 mm. The four circles near the corners of the chip represent screw holes. Their diameters are 1.85 mm. This chip was finally fabricated by research staff from Air Force Research Lab (AFRL) using laser mills. For now the outermost spiral is non-functional due to a soldering problem.

Besides the chip we already received from AFRL, some revised chips are cur-

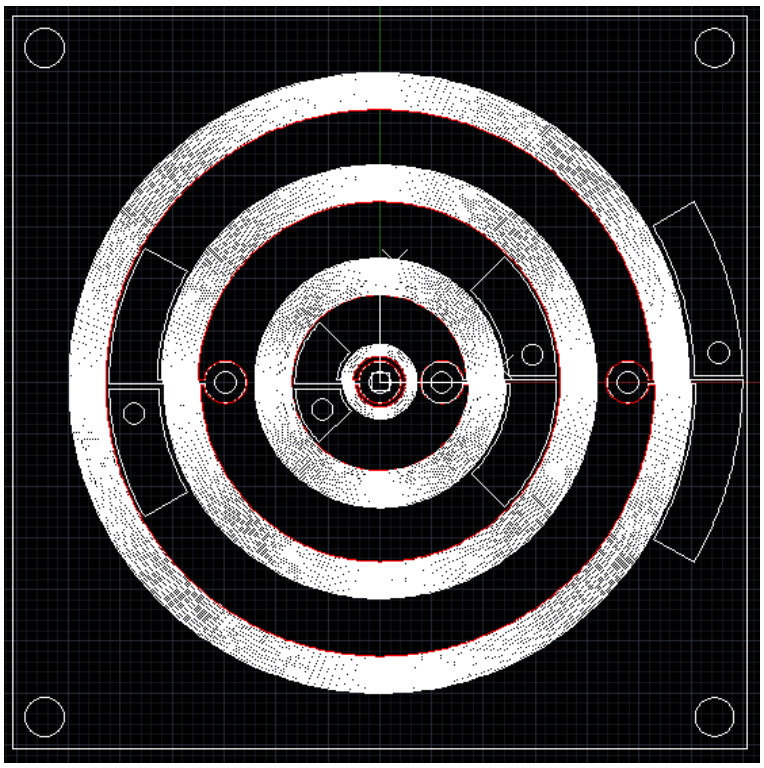


Figure 2.11: A revised version of atom chip

Spiral	Number of turns	Starting radius(mm)	Ending radius(mm)	Length(mm)
1	5	1.23	1.83	48.07
2	15	4.25	6.05	485.38
3	15	8.69	10.49	903.84
4	15	13.2	15	1328.89

Table 2.2: The number of turns, starting radius and ending radius for each spiral structure in Fig 2.11

rently under design and fabrication. For example, Fig 2.11 shows the top view of a chip which has the innermost spiral structures for making BEC and other three spiral structures for interferometry experiments. The innermost spiral structure has the same parameters compared to before. For the other three spiral structures, the middle one has more number of turns compared to the previous design, which can provide more freedom to control the magnetic field. The starting radii of three spiral structures are 4.25 mm, 8.69 mm and 13.2 mm respectively. All of them have 15 turns and the same pitch: 0.12 mm/turn.

2.3 Testing Chamber Design

After we received the atom chip from AFRL, we want to test the chip in our current experiment setup. In order to do that, we need to insert the chip in a testing chamber. The chamber requires ultra high vacuum compatibility, support for bias fields and good thermal heat sinking.

Fig 2.12 is an overview of the design of the testing chamber. Our current experiment apparatus suffers from thermal variations. A temperature rise of about 100 degrees made the trap parameters fluctuate. To get rid of this problem, this new chamber uses Shapal Hi-M Soft as the material. Shapal Hi-M Soft is a hybrid type of machinable Aluminum Nitride ceramic that offers high mechanical strength, good thermal conductivity (92 W/(m $\cdot$ K)) and low thermal expansion coefficients (4.8×10^{-6} /K). This chamber includes one chip holder, two bias coil holders for each direction, one bottom base and one top base. The atom chip is placed in the middle of the chip holder shown in Fig 2.13. And four bias coil holders are inserted around the chip holder. The chip is rotated by an angle in order to prevent the screw heads at the corners from blocking the Bragg lasers coming from x and y direction. The chip holder has a channel beneath the chip for connecting the chip to external wires. The coils are wired around the bias coil holders to provide the bias fields in x and y direction. Bragg beams pass through bias coil centers, and BEC atoms are inserted through diagonal openings between coil holders.

Coils for z direction bias fields will be inserted from below and above shown in Fig 2.14. The top z coil holder has a hole that allows the probe beam to go through and be reflected out of the chamber. And it is inserted on the top base. The bottom z bias coil holder is inserted on the chip holder. Finally, the chip holder is inserted on the bottom base. This whole structure will be placed in our science cell. This testing

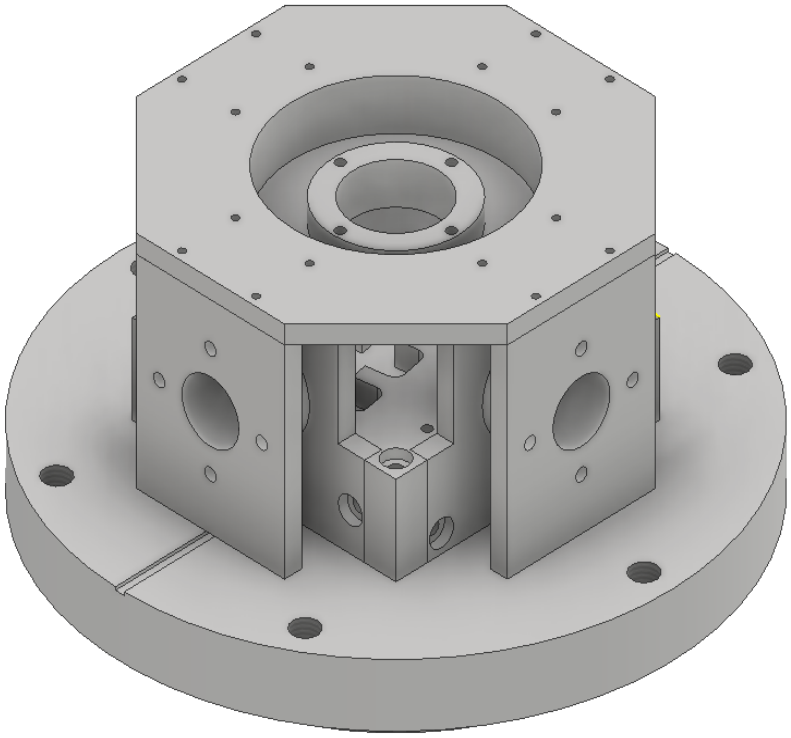


Figure 2.12: An overview of testing chamber

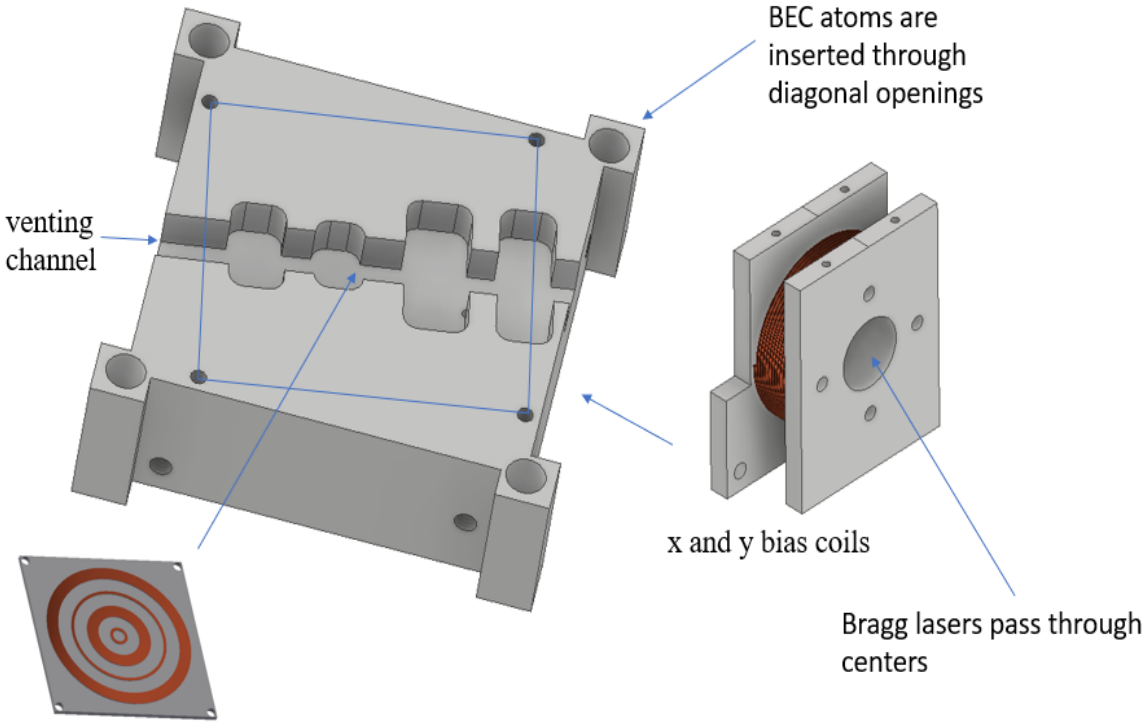


Figure 2.13: The chip holder and bias coil holder

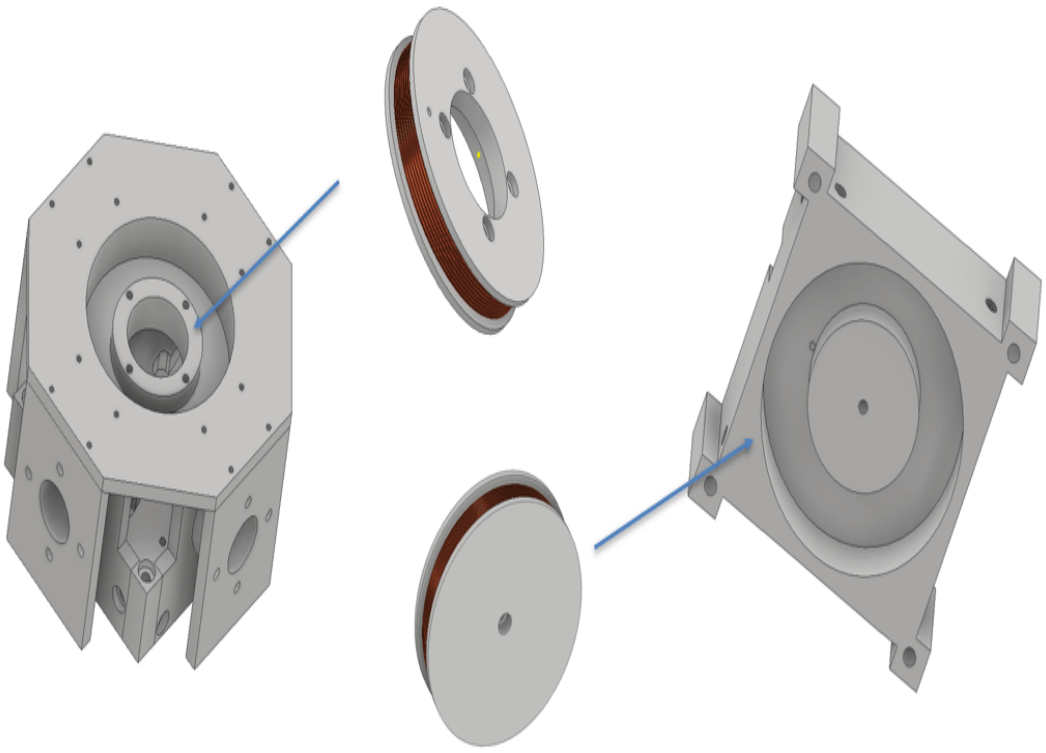


Figure 2.14: The top and bottom bias coil holder

Coil	A	B	C	D	Zt	Zb
Resistance (Ohm)	0.238	0.241	0.231	0.229	0.243	0.229
Inductance (μH) at 10 kHz	105.2	110.1	108.4	106.8	25.5	31.9

Table 2.3: Resistances and inductance for bias coils

chamber is designed using Autodesk Inventor. The details of design parameters of the testing chamber are attached behind in Appendix A.

After we wrapped coils on bias coil holder for each direction, we measure some parameters for bias coils such as resistance, inductance (Table 2.3) and field/current ratio. Coil A, B, C and D are used for x and y bias field. Coil Zt and Zb are for z bias field.

For x and y bias coils, the field/current ratio at the trap center is 7.06 Gauss/Amp. And for z bias coils it is 5.74 Gauss/Amp. When the bias field is 20 Gauss, the power consumption of six bias coils is about 3.32 W. The total power consumption of the

chip and bias coils is about 7.59 W.

To sum up this chapter. A new atom chip is simulated for its performance on the interferometry experiment. It is designed and constructed based on double layer spiral copper wires with different chirality. And the supporting chamber for testing the atom chip is also designed

Chapter 3

Experiment Programs

The new atom interferometry experiment will take data more quickly, so we need more automated ways to run it and analyze data obtained

3.1 Image Analysis Program

In order to derive the actual magnetic potential from trajectories of atom packets and compare with the ideal potential, we need to quickly extract the locations of the atom clouds from an image dataset. Until now, we have been doing this by hand. We have to manually select a square region around each atom cloud and fit the pixel values and corresponding coordinates to a Gaussian function. To automate the image analysis procedure, we come up with a BEC tracking method. Here we consider a sequence of images making up a trajectory (left to right, top to bottom) in Fig 3.1.

Fig 3.2 shows an example image after the second splitting with four atom packets. It is a grayscale absorption image. The absorption imaging for a cold atom cloud is illuminated by a laser beam which is resonant with it. The absorption of light by the

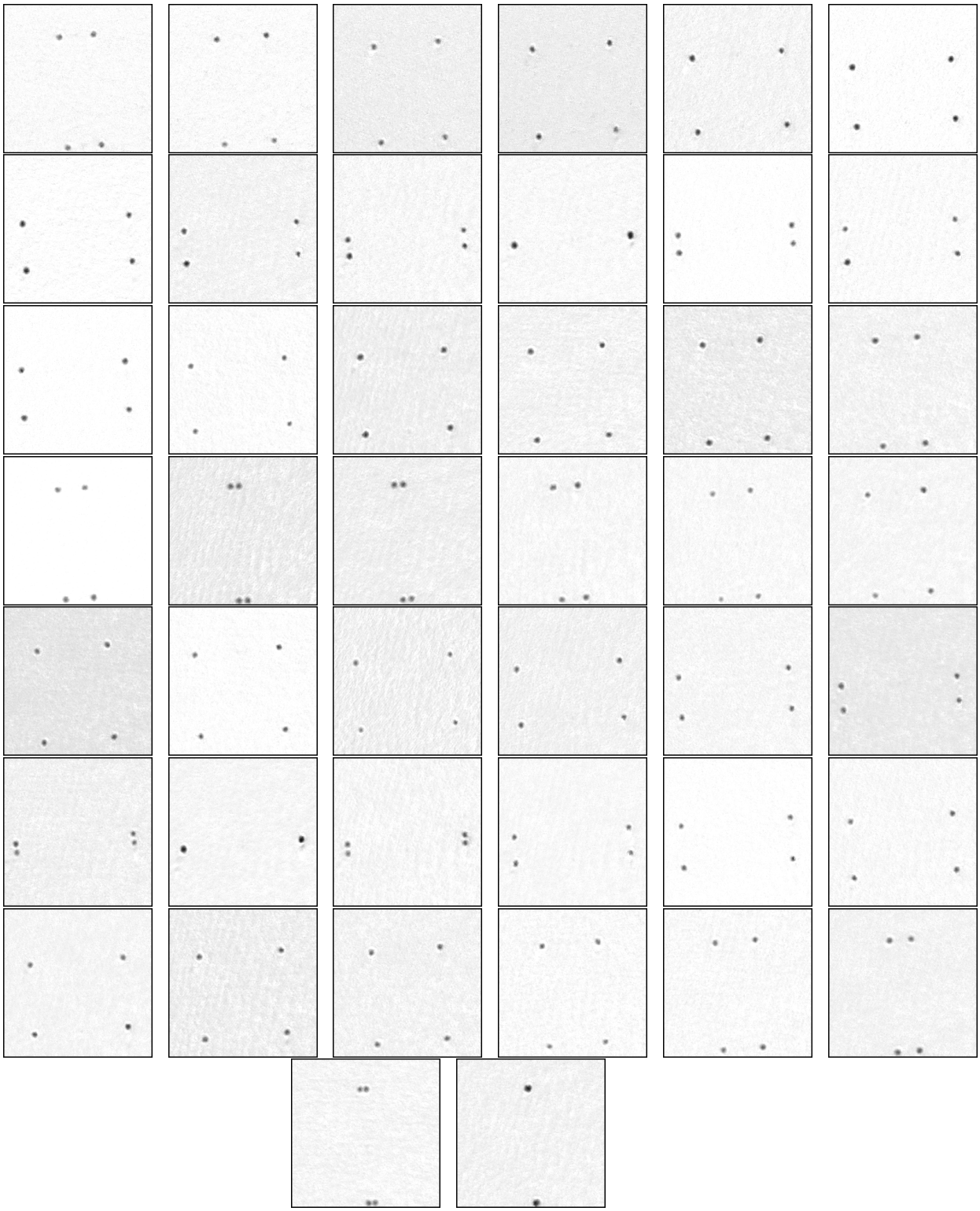


Figure 3.1: The image sequence of atom interferometry experiment

atoms casts a shadow which is then imaged onto a CCD camera. The effect of the absorption caused by atoms on the intensity of the laser beam is given by

$$I = I_0 e^{-n(x,y)\sigma} \tag{3.1}$$

Where I_0 is the intensity of the incident laser beam, I is the intensity of the output beam, $n(x,y) = \int \rho(x,y,z)dz$ is the atom cloud intensity and σ is the absorption cross section [35]. The absorption cross section is given by [36]:

$$\sigma = \frac{\sigma_0}{1 + 4(\frac{\Delta}{\Gamma})^2 + \frac{I_0}{I_{sat}}} \tag{3.2}$$

Where σ_0 is the on resonance cross section. Δ is the detuning of the laser field from the atomic resonance. Γ is the nature decay rate of the excited state. And I_{sat} is the saturation intensity.

Three images are taken for each absorption imaging measurement: the "atom" image, the "no-atom" image and the background image. At first, the background image is taken with the probe light off. Images taken for measurements are corrected by subtracting the background image since the unwanted background light is independent of the atom trap operation. Next, the "atom" image, an image of the atoms in the science trap is taken. At last, the "no-atom" image is taken with the same optics and beam sequencing as the "atom" image but without the trapped atoms. After all three images are taken, the divided image can be calculated. For the pixel value $P(x,y)$ at the position (x,y) on the CCD, the same pixel value in the divided image is

$$P_{divided}(x,y) = \frac{P_{atom}(x,y) - P_{background}(x,y)}{P_{no-atom}(x,y) - P_{background}(x,y)}. \tag{3.3}$$

As you can see in Fig 3.2, there is some background noise in the original image. The noise within the field of view comes mainly from fluctuations in the intensity profile

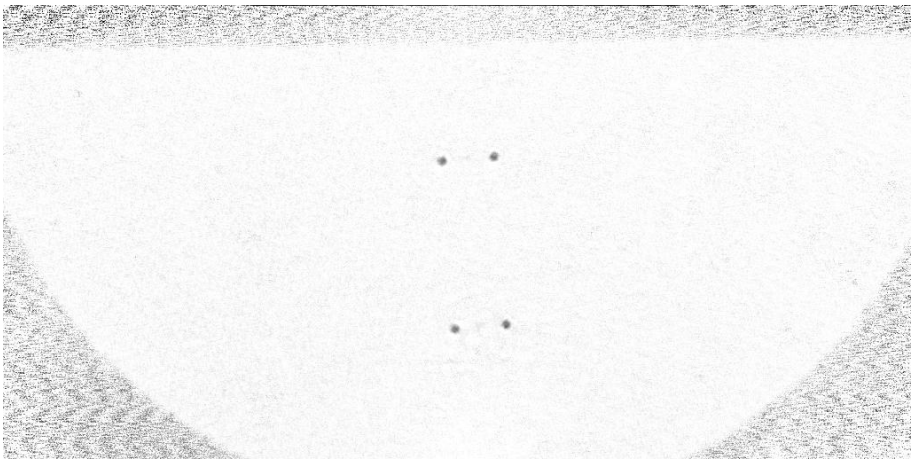


Figure 3.2: An absorption image after the second splitting

between the "atom" and "no-atom" images. The noise close to the top edge and two bottom corners have pixel value either 0 or 1. It is because there is no light in either "atom" or "no atom" image due to the finite size of the probe beam. When they are divided we get a 0/0 situation. The image processing of the camera makes the result either 0 or 1. The first thing to do is to denoise the image with an image filter.

There are many different kinds of image filters [37]. Some of the most commonly used filters are average filters and median filters [38]. Median filters tend to preserve edges well but they are not as effective as average filters in terms of blurring impulses. Since we are mainly interested in the center positions of the atom packets instead of their edges, we choose to use an average filter to denoise the images. We can use a three by three or five by five average filter to convolute with the original image. The averaging filter is to replace each pixel in the original image by the average pixel values around it within a three by three or five by five window. For example, Fig 3 shows the denoised Fig 2 using a five by five average filter. So the relationship between each pixel value in the filtered image $P^{'}(x, y)$ and original image $P(x, y)$ is

$$P(x, y) = \frac{1}{25} \sum_{m=-2}^{2} \sum_{n=-2}^{2} P(x + m, y + n) \tag{3.4}$$

Figure 3.3: A filtered image

The standard deviations of pixel values of the image before and after denoising are 0.1825 and 0.0786 respectively.

There is a boundary issue when the average filter convolutes with the pixels at the edges of the original image. In that case the image will be padded with zeros around it for this operation. As a result, the pixels near the boundary in the denoised image are close to zero. From Fig 3.3 we can see that the majority of pixels are relatively white. The pixels representing the atom packets are relatively dark. But there are still some background noise pixels left close to the boundary, which are not our interested signal. In order to get rid of the background noise completely, we would like to crop a smaller region including only the atom packets as dark pixels from Fig 3.3 and ignore the remaining pixels. Fig 3.5 shows the result.

To decide the size of the cropped image, given the splitting velocity v_B (mm/s), the trapping frequency ω (rad/s) and the image resolution of the camera r (pixel/mm), we know the distance (pixel) the atom packets can travel in the field of view is rv_B/ω. Then we crop from the center of the filtered image with height and width equal to $rv_B/\omega + 10$. The standard deviations of pixel values for the cropped image before and after denoising are 0.0350 and 0.0184 respectively.

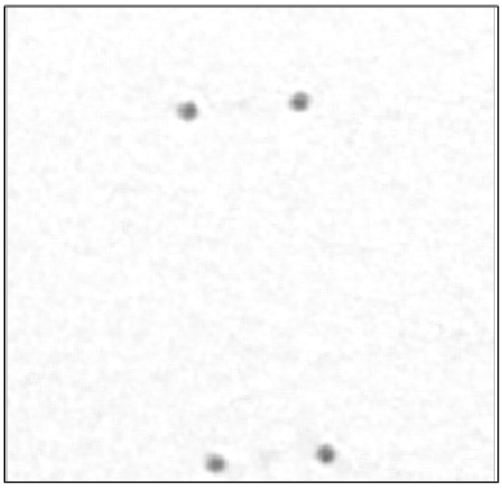


Figure 3.4: A cropped filtered image

Next, for the cropped image, we convert the grayscale image into a binarized image shown in Fig 5 so that each pixel value is either 0 or 1. The threshold value is chosen to minimize the sum of variances of pixels which are converted to black class and white class [39].

In this way, we can obtain the center positions of atom packets by calculating the center positions of black clusters in Fig 3.5. To do that, we extract the column and row numbers of the black pixels and make the plot in Fig 3.6.

The green circles are the black pixels in the binarized image. Then we use k-means clustering algorithm [40] to find the center position for each atom cluster. k is number of clusters to search for. This algorithm mainly includes four steps. First of all, given how many clusters we want to find, it initializes k random points as cluster centers within the data region. For our case, it starts with 4 cluster centers. Step

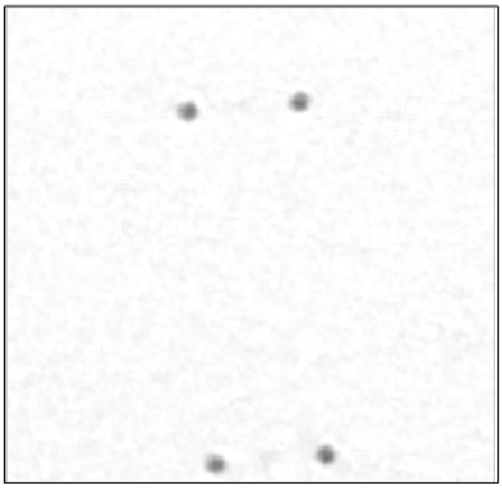


Figure 3.5: A cropped filtered image

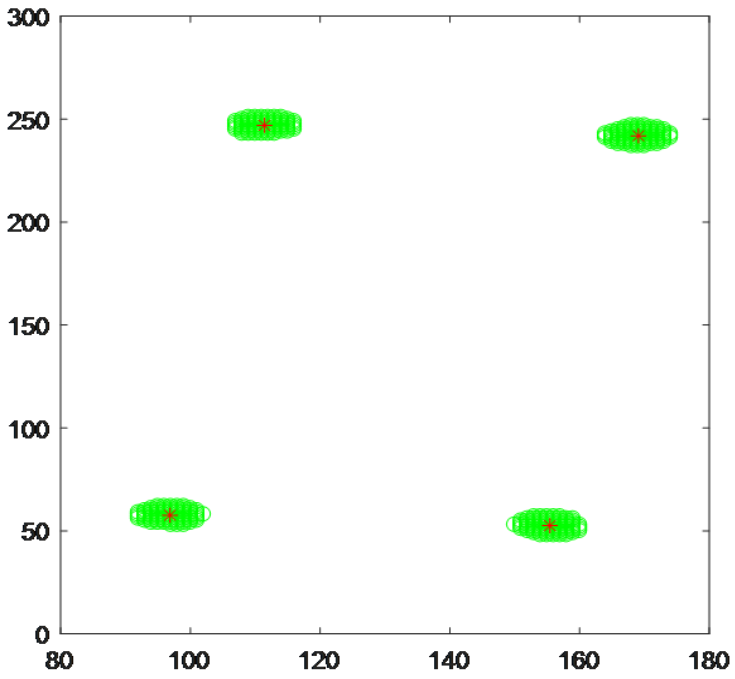


Figure 3.6: The column and row indexes of the black pixels in the binarized image

two, it assigns each data point to the closest center as in the same cluster. Step three, it calculates the average positions of data points in the same clusters as new cluster centers. Finally, it repeats the last two steps until the positions of cluster centers converge. Based on this method we can find the positions of atom packets from the image automatically instead of selecting a fitting area close to each packet one by one.

Then we go back to the original image and use the center positions to crop the BEC clouds from original images shown in Fig 3.7 and fit the original pixel values $P(x, y)$ and their coordinates (x, y) to a Gaussian function to get more accurate results for the center positions, widths and depths of BEC clouds.

$$P(x, y) = A + Be^{-(\frac{(x-x_0)^2}{w_x^2}+\frac{(y-y_0)^2}{w_y^2})} \tag{3.5}$$

w_x and w_y are the width of atom clouds in x and y direction, B represents the depth in z direction. x_0 and y_0 are the center position of the BEC cloud. And A is an offset.

We can use the reduced chi-square statistic to describe the goodness of the Gaussian fit. It is defined as chi-square per degree of freedom $\chi^2_\nu = \frac{\chi^2}{\nu}$ where ν is the degree of freedom, which is the number of pixels minus the number of fit parameters. The χ^2 is a weighted sum of squared deviations over all pixels used for the fitting $\sum \frac{\left(P_{prediction}-P_{original}\right)^2}{\sigma^2}$ where σ^2 is the variance of image pixels in a region without atoms. For example, when we use 13 $\times$ 13 px cropped image for each atom cloud in Fig 3.7, the average χ^2_ν for these four clouds is around 2. And we decide uncertainties for fitting parameters by varying each parameter individually to make chi-square doubled. The result shows that the uncertainty for each parameter in the unit of pixel is: A: 0.05, B: 0.09, w_x: 1.2, w_y: 1.1, x_0: 0.8, y_0: 0.8.

There are some cases when two BEC packets partly or entirely overlap and the

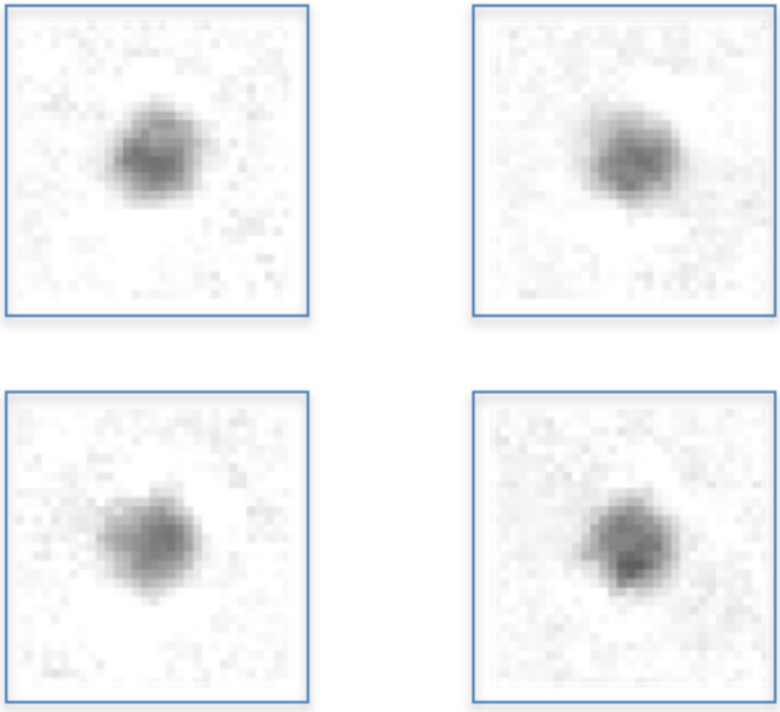


Figure 3.7: Cropped BEC packets from the original image

image only shows two clouds. In order to check these situations, we still initialize the algorithm with four clusters and then see if the distance between any two of four founded cluster centers is less than twice the size of the clouds. If so, that indicates there are overlapping BEC packets. We have tried two solutions for these situations. The first one is to rely users to manually select four positions and fit the whole cropped image to four gaussian functions to get more accurate results. The second one is that we redo the algorithm given two cluster centers to find. And for each one of two cropped images we fit with two gaussian functions so that we still get four positions from the original image. In order to make the method more general for future purposes, we decide to adopt the first solution.

In this way we can find the positions of all BEC clouds from each one in a sequence of images making up a trajectory. But it is not enough to only get positions of atom packets in each single image. We need to track each atom cloud between consecutive

images because the ultimate goal is to obtain the trajectory of each atom packet. However, due to the random initialization of the clustering algorithm, we do not know which cropped BEC cloud is which compared to the original image. The top left packet in Fig 3.7 is not necessarily the top left one in the original image Fig 3.2. The situation is similar between consecutive images. The top left cropped packet in one image is not necessarily the top left packet in the following image. To solve this problem, we need to label those atom packets. In order to do that, for the first three images in the image sequence, we group the packets based on whether their image coordinates are larger or smaller than the image coordinates of the center of four packets, namely which quadrant they are in respect to the image center. When there are a different number of atom packets, this label method is generalized: we sort the atom packets based on their polar angles respect to the image center at first. For those packets with the same polar angles, we sort them based on their polar radii. Starting from the fourth image, we use the latest three images to get a predicted position for each atom packet based on its previous positions using linear regression. Then we can find the actual position closest to my predicted position and save it for processing following images. Finally, we have the plot Fig 3.8 showing the labelled circular trajectory for four atom packets after the second splitting.

In order to make this BEC tracking method more general, we built an image analysis program with the user interface like Fig 3.9. It requires users' inputs for gaussian fitting parameters A, B, w_x and w_y and other parameters such as trapping frequencies and the camera resolution for calculating how far the atom clouds can travel within the field of view.

Users can select different numbers of atom packets to track with and the program can adapt to different interference trajectories. The program can automatically label the atom packets based on their polar angle and radii respect to the image center.

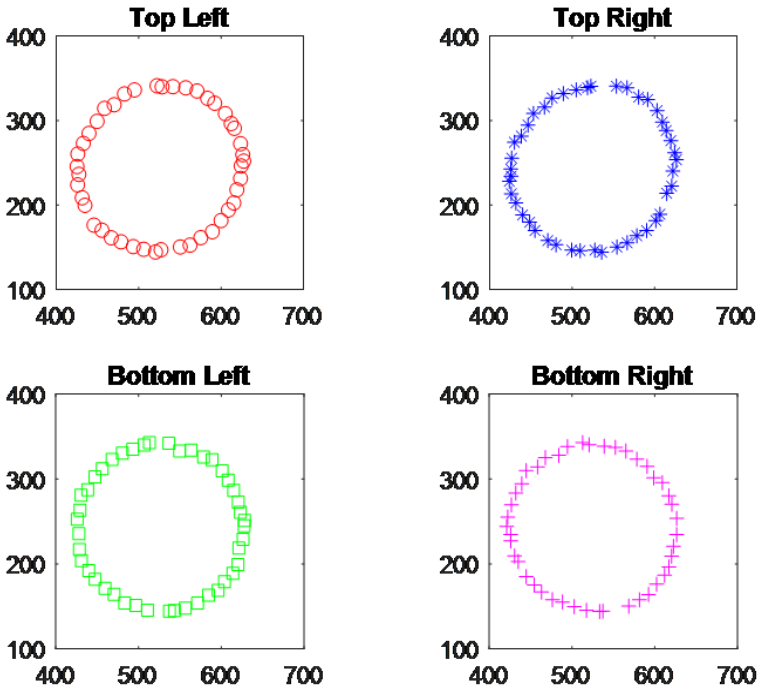


Figure 3.8: The trajectories of four atom packets after the second splitting

When the program finishes running, it put the label number on the top of each atom cloud in the first image. The trajectories, depth and width of atom clouds are saved in an excel file. The interface also shows the average positions of atom packets from all images. If the program runs normally, those positions are very close to each other. Otherwise there might be some abnormal situations. In most cases, the default pixel threshold derived from minimizing pixel variances works pretty well for the image binarization procedure. But in case the images are degraded, the program also allows the user to manually adjust the pixel threshold for image binarization. When atom packets are overlapping, it allows the user's manual selection for the locations of atom clouds. The user can also choose to skip an image with overlapping BEC clouds. Therefore, this program is more robust to image noises and degraded interference patterns. For our future atom interferometry experiments, we may want to kick those atom packets with a higher momentum to realize a splitting pattern

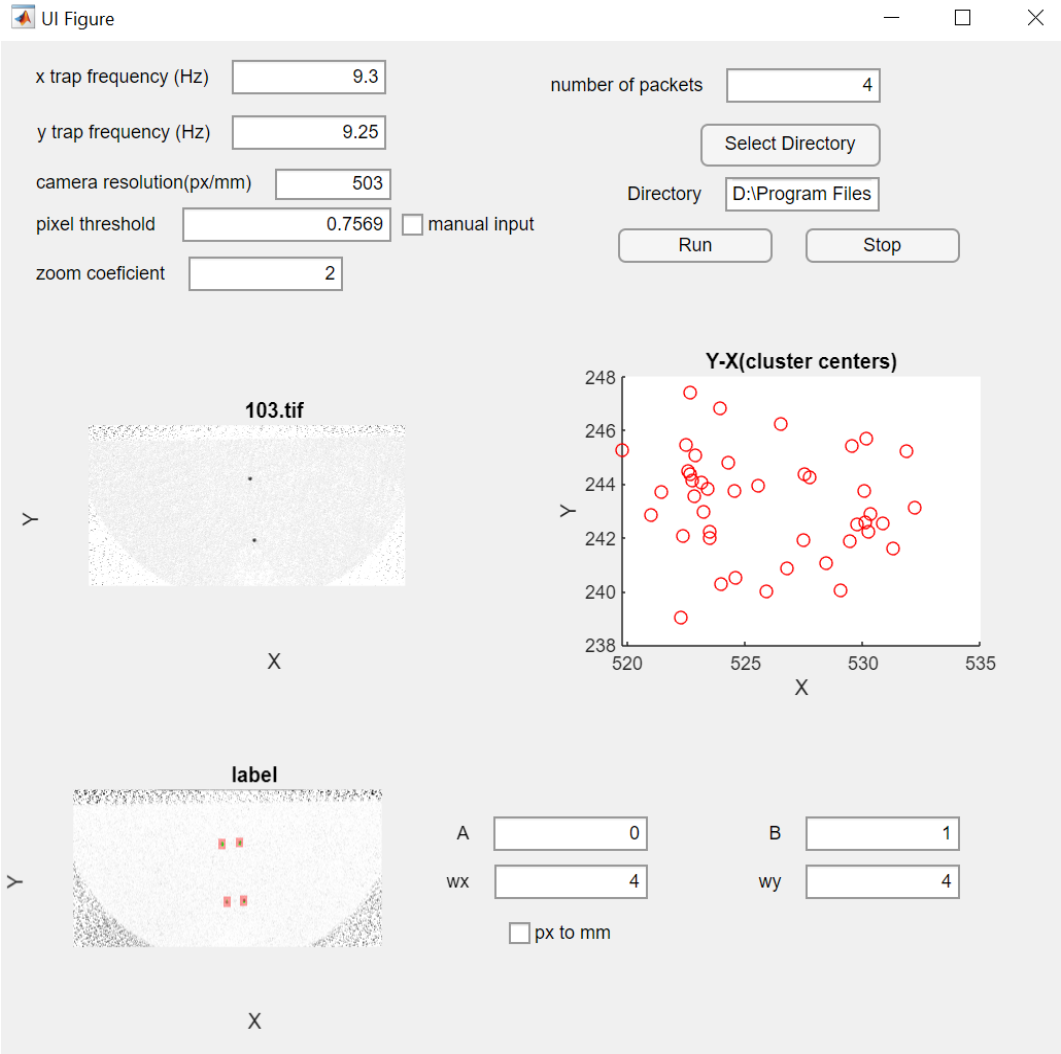


Figure 3.9: The user interface of BEC image analysis program

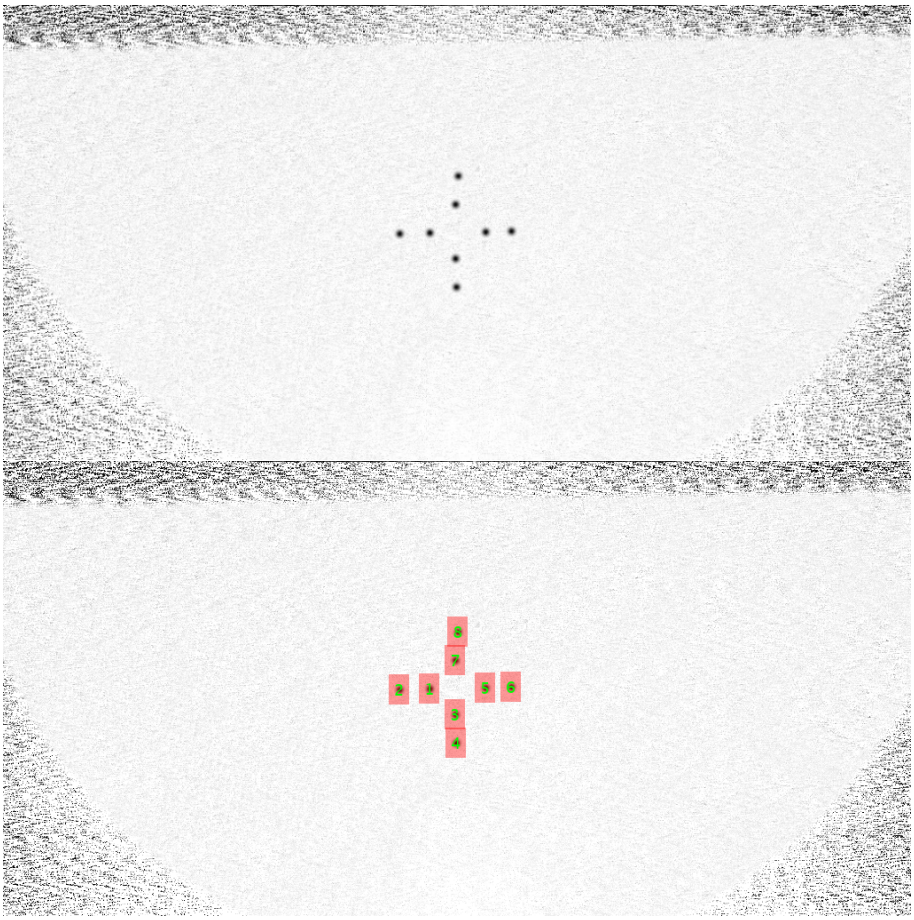


Figure 3.10: A splitting pattern for future interferometry experiments and the labeled result

shown in Fig 3.10. We test the image analysis program on this pattern, as you can see the BEC tracking method can adapt to the trajectories of BEC packets. Fig 3.11 is another example. Atoms with a higher momentum are splitting in 45 degrees directions and the program still works.

3.2 Admiral Program

In our lab, we use a homebuilt GUI called Twitch to compile all experiment scripts. Compiled routines are then run by Adwin computer, which controls a number of experimental hardware. Other experimental hardware including two cameras and the

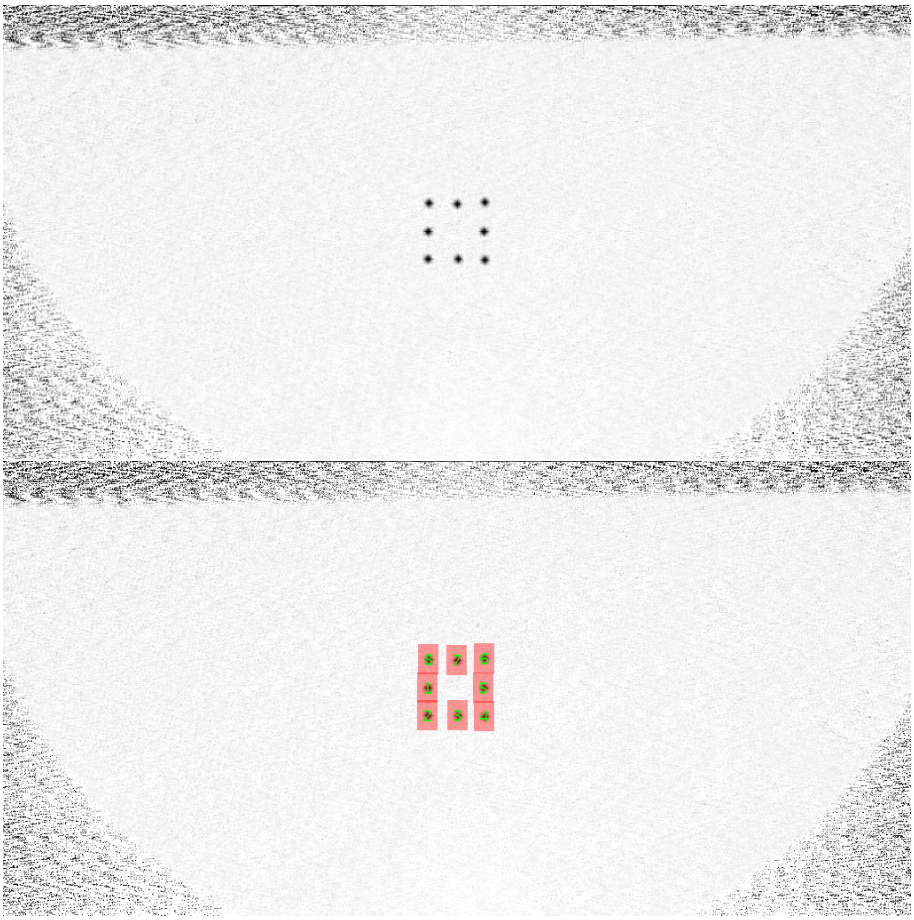


Figure 3.11: Another splitting pattern for future interferometry experiments and the labeled result

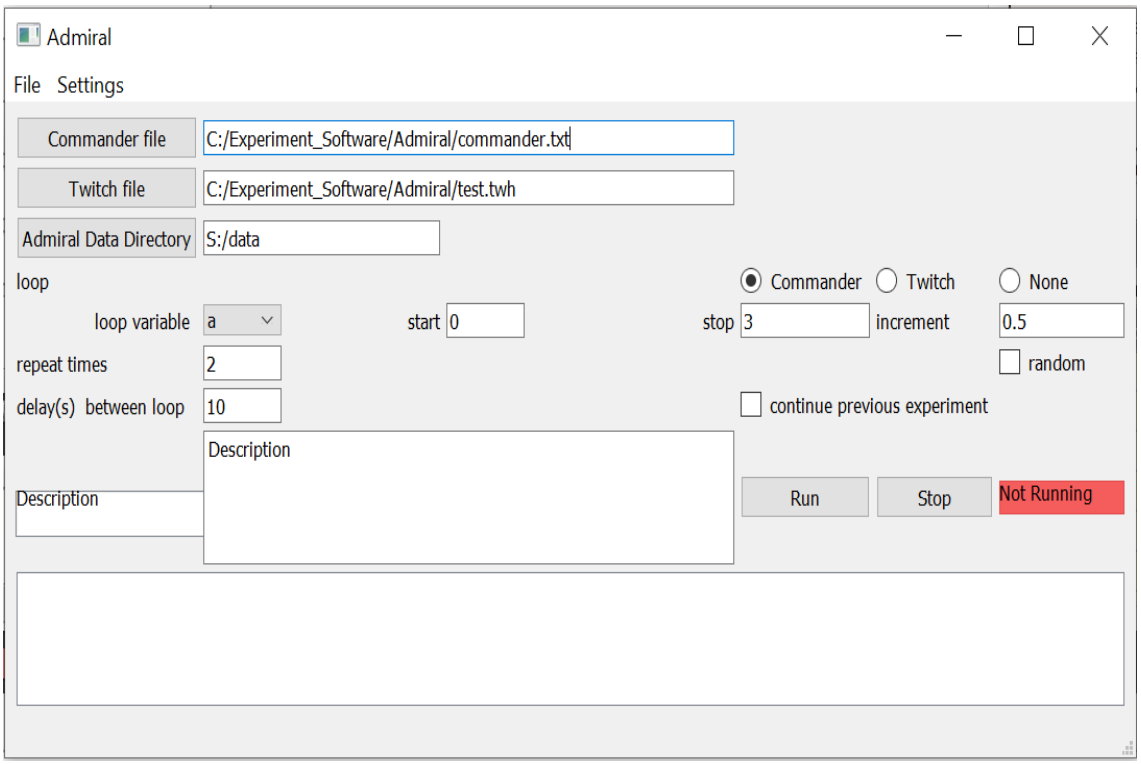


Figure 3.12: The user interface of Admiral program

RF function generator are controlled by a different program called Commander. For our future experiment we want to integrate these programs together and automate the experiment running process. Therefore, in addition to automating the image analysis procedure, we also build a program called Admiral Program to automate image generations and collections. This program is mainly used to iterate different experiment variables like laser intensity, trap strength, or propagation time and save images with experiment files. Its user interface shows in Fig 3.12.

This user interface requires inputs for commander file and twitch file, which are required for the experiment running. The user also needs to specify which directory is used for saving the images and experiment files in Admiral Data Directory, which variable is being iterated and which file is being iterated. For each variable, the start, end and increment values are needed for iterations. The user can choose how many times of iterations and whether the iterating variable are in increasing order or

random order. When the user chooses to continue previous experiment, the images and experiment files are saved in the same directory with the previous experiment. Otherwise, they are saved in a new directory with a new time stamp as the name of the directory. Besides, a log file including the iterated variable, its values and the description about experiments is also saved. In this way, the images are organized for conveniently running image analysis program. During experiments, the progress bar is keep updating the progress of iterations in percentage. After one iteration is finished, the user interface can show the most recent image at the bottom.

To sum up this chapter, an image processing program is developed to quickly extract positions and sizes of atom packets from their trajectories. And an experiment program is developed to automate image generations and collections

Chapter 4

Phase Caculations

The operation of atom interferometry experiments relies not only on the mechanical structure and software program, but also on understanding and controlling the interferometer phase shifts coming from the trap and Bragg laser imperfections. This chapter include the calculations that determine these phase shifts. The first section is about the derivation of the ideal signal phase. The second section focus on numerical computations caused by trap and laser imperfections.

4.1 Ideal Signal Phase

After the atom packets are recombined by the bragg laser beam along x direction (Fig 4.1), the ratio of numbers of atoms that return to rest for each interferometer $S_{\pm}$ depends on the signal phase between two interferometers, $\Delta\Phi$, according to: $S_{\pm} = C_{\pm} + A_{\pm}\sin(\Phi_N \pm \frac{\Delta\Phi}{2})$. $C_{\pm}$ define the center. $A_{\pm}$ are related to the visibility. Φ_N comes from noises. The differential phase $\Delta\Phi$ changes the eccentricity. To explore

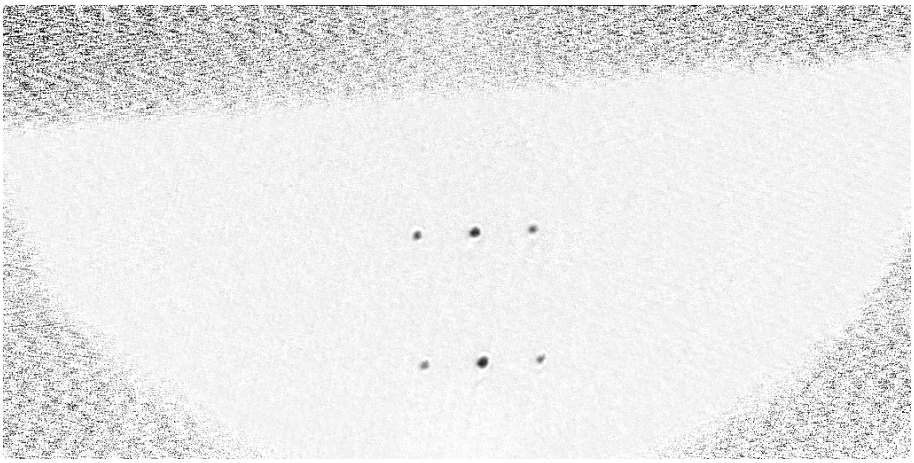


Figure 4.1: Atom packets after the recombination

the signal phase $\Delta\Phi$, we introduce an extra phase ϕ_β between x and y bias field.

$$B_{bias} = B_0 \begin{pmatrix} \sin(\Omega_1 t)\cos(\Omega_2 t) \\ \sin(\Omega_1 t)\sin(\Omega_2 t + \phi_\beta) \\ \cos(\Omega_1 t) \end{pmatrix} \tag{4.1}$$

We try to adjust the recombination time t_2 and the phase of bias fields ϕ_β in our experiment and see how the signal phase $\Delta\Phi$ change with them.

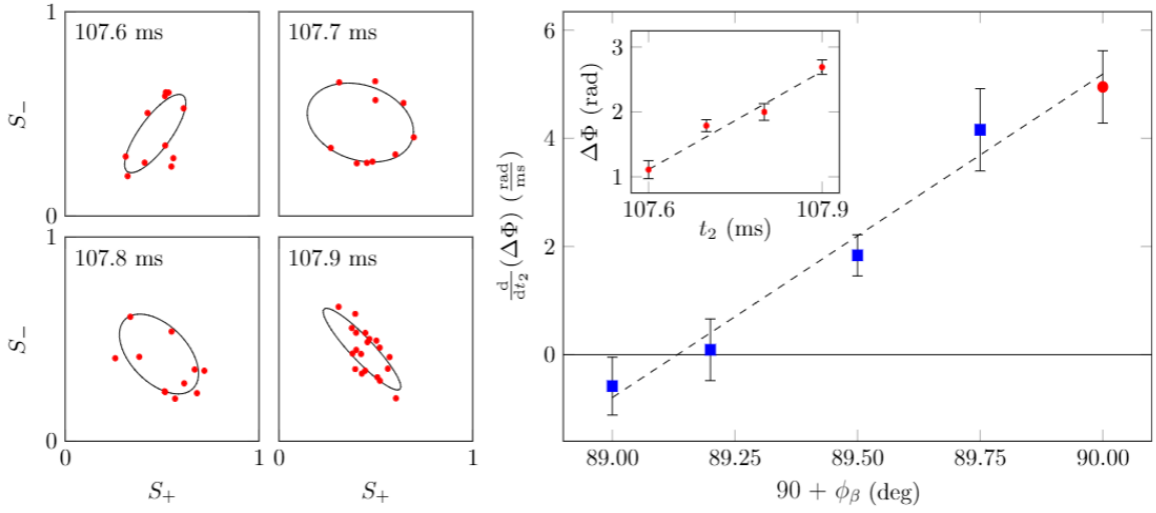


Figure 4.2: Bias field phase effect [41]

In Fig 4.2, we vary both ϕ_β and t_2 and look at how $\Delta\Phi$ changes. the inset on the right shows the linear variation with t_2 at a fixed ϕ_β and the signal phase $\Delta\Phi$ extracted from the $S_\pm$ plot on the left. The main graph on the right shows how the slope $d(\Delta\Phi)/dt_2$ changes as a function of ϕ_β. The result shows that $\partial^2\Delta\Phi/\partial t_2\partial\phi_\beta$ is about 6 rad/(ms deg), which is 3.44×10^5/s.

To explain the experiment result, we try to derive the analytical expression for the signal phase. We define x, y and z to be the "lab" coordinates determined by the nominal magnet coil axes and X, Y and Z to be the principal axes of the harmonic trapping potential . The Bragg beams have wavelength λ and wave numbers $k = 2\pi/\lambda$. They provide a nominal Bragg velocity $v_B = 2\hbar k/m$ to the atoms. At time $t = 0$, the condensate is split by applying the y Bragg beam. Define the packet moving in the +y direction to be on the right and the packet moving towards -y to be on the left. After time $t_1 \approx \pi/(2\omega_0)$, where ω_0 is the nominal horizontal oscillation frequency, the packets have positions and velocities $r_{1R} \approx R_0\hat{y}$, $r_{1L} \approx -R_0\hat{y}$, $v_{1R} \approx 0$ and $v_{1L} \approx 0$. The x Bragg beam is then applied, splitting the packets and causing them to move in approximately circular orbits of radius R_0. The four packets are then described by $r_{ij}(t)$ and $v_{ij}(t)$, with i = R or L and j = + for atoms receiving a kick of $+v_B\hat{x}$ and j = - for atoms receiving a kick $-v_B\hat{x}$. The packets propagate for time $t_2 \approx 2\pi/(\omega_0)$ before being recombined by another x Bragg pulse, which creates two interferometers on the left and right.

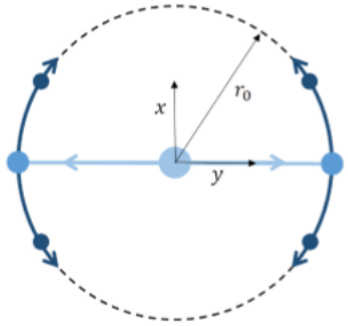


Figure 4.3: Trajectory of atoms in the interferometer

We assume the x and y Bragg beams are incident along the directions

$$\hat{k}_x = \cos\psi_x \hat{X} + \sin\psi_x \cos\xi_x \hat{Y} + \sin\psi_x \sin\xi_x \hat{Z} \tag{4.2}$$

and

$$\hat{k}_y = \sin\psi_y \cos\xi_y \hat{X} + \cos\psi_y \hat{Y} + + \sin\psi_y \sin\xi_y \hat{Z} \tag{4.3}$$

where ψ_x and ψ_y are nominally zero. We assume that the Bragg beams are retro-reflected with no error. In order to derive the ideal phase signal, we want to calculate the interferometer phase in the case of a harmonic potential at first. Assuming the atoms are trapped in a harmonic potential $U = \frac{1}{2}m(\omega_x X^2 + \omega_y Y^2 + \omega_z Z^2)$, where X, Y, Z are the principal axes of the trap and ω_x, ω_y, ω_z are the corresponding trap frequencies, we can express the ideal trajectory of atom packets in one dimension by sine functions. For example, in X direction the trajectory is

$$X_a \cos\omega_x t + \frac{v_a}{\omega_x}\sin\omega_x t \tag{4.4}$$

X_a and v_a are the initial position and velocity. In this case, the phase a wavepacket develops in one dimension is given by the action over $\hbar$ [42].

$$\phi_{xa} = \frac{1}{\hbar}\int_0^t (T-V)dt = \frac{m}{4\hbar\omega_x}[(v_a^2 - \omega_x^2 X_a^2)\sin 2\omega_x t + 2v_a X_a \omega_x(\cos 2\omega_x t - 1)] \tag{4.5}$$

Where T and V are the kinetic and potential energy. The total phase separates as $\phi_{xa}+\phi_{ya}+\phi_{za}$. We can express the trajectory in X direction after the second splitting by

$$X_{L\pm}(t) = X_{L0}\cos\omega_x t + (v_{xL0} \pm v_B \cos\psi_x)\frac{\sin\omega_x t}{\omega_x} \tag{4.6}$$

$$X_{R\pm}(t) = X_{R0}\cos\omega_x t + (v_{xR0} \pm v_B\cos\psi_x)\frac{\sin\omega_x t}{\omega_x} \tag{4.7}$$

X_{L0} and X_{R0} are the initial positions of the second splitting for two interferometers and therefore the final positions of the first splitting. v_{xL0} and v_{xR0} are the final velocities of the first splitting and therefore the derivatives of X_{L0} and X_{R0} respect to time

$$X_{L0} = X_0\cos\omega_x t_1 + (v_{x0} - v_B\sin\psi_y\cos\xi_y)\frac{\sin\omega_x t_1}{\omega_x} \tag{4.8}$$

$$X_{R0} = X_0\cos\omega_x t_1 + (v_{x0} + v_B\sin\psi_y\cos\xi_y)\frac{\sin\omega_x t_1}{\omega_x} \tag{4.9}$$

X_0 and v_{x0} are the initial position and velocity before the first splitting. We substitute Eq (4.6) and (4.7) into (4.5) to obtain the phase accumulated by each interferometer after the second splitting are

$$\phi_{xRa} = \frac{mv_B\cos\psi_x}{\hbar\omega_x}[v_{xR0}\sin 2\omega_x t_2 + X_{R0}\omega_x(\cos 2\omega_x t_2 - 1)] \tag{4.10}$$

$$\phi_{xLa} = \frac{mv_B\cos\psi_x}{\hbar\omega_x}[v_{xL0}\sin 2\omega_x t_2 + X_{L0}\omega_x(\cos 2\omega_x t_2 - 1)] \tag{4.11}$$

Then from Eq (4.8) to (4.9) the action integration part of the signal phase is the difference between ϕ_{xRa} and ϕ_{xLa}

$$\Delta\Phi_{xa} = \phi_{xRa} - \phi_{xLa} = \frac{2mv_B^2\cos\psi_x\sin\psi_y\cos\xi_y}{\hbar\omega_x}[\sin\omega_x(2t_2 + t_1) - \sin\omega_x t_1] \tag{4.12}$$

In the ideal case, $\Delta\Phi_{xa}$ is zero. Based on similar derivations we can also obtain the signal phase in Y and Z direction

$$\Delta\Phi_{ya} = -\frac{2mv_B^2\cos\psi_y\sin\psi_x\cos\xi_x}{\hbar\omega_y}[\sin\omega_y(2t_2 + t_1) - \sin\omega_y t_1] \tag{4.13}$$

$$\Delta\Phi_{za} = \frac{2mv_B^2\sin\psi_x\sin\psi_y\sin\xi_x\sin\xi_y}{\hbar\omega_z}[\sin\omega_z(2t_2 + t_1) - \sin\omega_z t_1] \tag{4.14}$$

The total action integration part of the signal phase is the sum of $\Delta\Phi_{xa}$, $\Delta\Phi_{ya}$ and $\Delta\Phi_{za}$.

There are additional contributions to the signal phase because the two packets do not end up in the same location with the same velocity. Before the recombination, we can express the wave function of atoms in one dimension (I generalize later to three dimensions) by

$$\psi = \frac{1}{\sqrt{2}}\left(e^{i\phi_+}e^{ik_+(x-x_+)} + e^{i\phi_-}e^{ik_-(x-x_-)}\right) \tag{4.15}$$

where

$$\phi_\pm = \frac{1}{\hbar}\int L_\pm dt \tag{4.16}$$

is the evolution phase. Eq (4.12) - (4.14) give the contribution from each direction. $x_\pm$ are the final positions of the classical trajectories, and $k_\pm = (m/\hbar)v_\pm$, for final classical velocities $v_\pm$.

The recombination pulse provides velocity kicks $\mp v_B = \mp\hbar k_B/m$, where $k_B = 2k$ is twice the laser wave number. After recombination the state is

$$\psi = \frac{1}{2}\left(e^{i\phi_+}e^{ik_+(x-x_+)}e^{-ik_Bx} + e^{i\phi_-}e^{ik_-(x-x_-)}e^{ik_Bx}\right) \tag{4.17}$$

We have omitted the fast-moving terms and keep only those brought nearly to rest. To simplify, take:

$$k_\pm = \pm k_B + k_0 \pm \frac{\delta k}{2} \tag{4.18}$$

and

$$x_\pm = x_0 \pm \frac{\delta x}{2} \tag{4.19}$$

Here k_0 and x_0 represent the mean final velocity and position of the packets, while

δk and δx are the differences between the packets. In this notation we obtain

$$\psi = \frac{1}{2}\left\{\exp\left[i\left(\phi_+ + k_B x - k_B x_0 - \frac{k_B \delta x}{2} + k_0 x - k_0 x_0 - \frac{k_0 \delta x}{2} + \frac{x\delta k}{2} - \frac{x_0 \delta k}{2} - \frac{\delta x \delta k}{4} - k_B x\right)\right]\right.$$
$$\left. + \exp\left[i\left(\phi_- - k_B x + k_B x_0 - \frac{k_B \delta x}{2} + k_0 x - k_0 x_0 + \frac{k_0 \delta x}{2} - \frac{x\delta k}{2} + \frac{x_0 \delta k}{2} - \frac{\delta x \delta k}{4} + k_B x\right)\right]\right. \tag{4.20}$$

Simplifying and removing common phases leaves

$$\psi = \frac{1}{2}\left\{\exp\left[i\left(\phi_+ - k_B x_0 - \frac{k_0 \delta x}{2} + \frac{x\delta k}{2} - \frac{x_0 \delta k}{2}\right)\right] + \exp\left[i\left(\phi_- + k_B x_0 + \frac{k_0 \delta x}{2} - \frac{x\delta k}{2} + \frac{x_0 \delta k}{2}\right)\right]\right\} \tag{4.21}$$

This function still depends on x, through the $x\delta k/2$ terms, so the phase is not uniform across the wave packet. We assume that δk is sufficiently small that the phase variation can be neglected. Then we evaluate x at the mean packet position x_0, causing the two δk terms to cancel. We are left with

$$\psi = \frac{1}{2}\left[e^{i\left(\phi_+ - k_B x_0 - \frac{k_0 \delta x}{2}\right)} + e^{i\left(\phi_- + k_B x_0 + \frac{k_0 \delta x}{2}\right)}\right] \tag{4.22}$$

The measured phase difference is therefore

$$\phi = \phi_+ - \phi_- - 2k_B x_0 - k_0 \delta x \tag{4.23}$$

Now use $x_0 = (x_+ + x_-)/2$, $\delta x = x_+ - x_-$, and $k_0 = (m/2\hbar)(v_+ + v_-)$. This gives

$$\phi = \phi_+ - \phi_- - \frac{m v_B}{\hbar}(x_+ + x_-) - \frac{m}{2\hbar}(v_+ + v_-)(x_+ - x_-) \tag{4.24}$$

In three dimensions the phase measured in one interferometer (left or right) is given by

$$\phi = \frac{1}{\hbar}\int_{t_1}^{t_2}(L_+ - L_-)dt - \frac{m}{\hbar}\mathbf{v_B}\cdot(\mathbf{r}_+ + \mathbf{r}_-) - \frac{m}{2\hbar}(\mathbf{v}_+ + \mathbf{v}_-)\cdot(\mathbf{r}_+ - \mathbf{r}_-) \tag{4.25}$$

where $L = T - V$ is the classical Lagrangian, evaluated at (r_+,v_+) or (r_-,v_-). The kinetic energy is $T = \frac{1}{2}mv^2$. The integral term can be evaluated analytically or

numerically using the packet trajectories. The differential phase between two interferometers is the signal of interest, with $\Delta\Phi = \phi_R - \phi_L$. So then we consider the $-\frac{m}{2\hbar}(v_+ + v_-)(x_+ - x_-)$ term in Eq (4.25). Based on Eq (4.6) and (4.7)

$$X_{R+}(t_2) - X_{R-}(t_2) = X_{L+}(t_2) - X_{L-}(t_2) = 2v_B \cos\psi_x \frac{\sin\omega_x t_2}{\omega_x} \tag{4.26}$$

$$v_{xR+}(t_2) + v_{xR-}(t_2) = -2\omega_x X_{R0} \sin\omega_x t_2 + 2v_{xR0} \cos\omega_x t_2 \tag{4.27}$$

$$v_{xL+}(t_2) + v_{xL-}(t_2) = -2\omega_x X_{L0} \sin\omega_x t_2 + 2v_{xL0} \cos\omega_x t_2 \tag{4.28}$$

From Eq (4.8) and (4.9) we can obtain the contribution of the $-\frac{m}{2\hbar}(v_+ + v_-)(x_+ - x_-)$ term for the signal phase in x direction

$$\begin{aligned}\Delta\Phi_{xd} &= -\frac{m}{2\hbar} 2v_B \cos\psi_x \frac{\sin\omega_x t_2}{\omega_x} \left[2(v_{xR0} - v_{xL0})\cos\omega_x t_2 - 2\omega_x(x_{R0} - x_{L0})\sin\omega_x t_2\right] \\ &= -\frac{2mv_B^2 \cos\psi_x \sin\psi_y \cos\xi_y}{\hbar\omega_x}[\sin\omega_x(2t_2 + t_1) - \sin\omega_x t_1]\end{aligned} \tag{4.29}$$

This is the exact opposite of the action integration in Eq (4.12). Based on similar derivations we can obtain the same conclusion for y and z direction. So the only contribution for the signal phase is the $-\frac{mv_B}{\hbar}(x_+ + x_-)$ term in Eq (4.25). Based on Eq (4.6) - (4.9), we can obtain its contribution for the signal phase in x direction

$$\Delta\Phi_x = -\frac{4mv_B^2 \cos\psi_x \sin\psi_y \cos\xi_y}{\hbar\omega_x} \sin\omega_x(t_1 + t_2) \tag{4.30}$$

The contributions to the signal phase from y and z direction are

$$\Delta\Phi_y = -\frac{4mv_B^2 \cos\psi_y \sin\psi_x \cos\xi_x}{\hbar\omega_y} \sin\omega_y(t_1 + t_2) \tag{4.31}$$

$$\Delta\Phi_z = -\frac{4mv_B^2 \sin\psi_x \sin\psi_y \sin\xi_x \sin\xi_y}{\hbar\omega_z} \sin\omega_z(t_1 + t_2) \tag{4.32}$$

Then we use the same expression for quadruple field in Eq (1.11) and calculate the magnitude of the superposition of these two fields. And we assume $\sin(\Omega_2 t + \phi_\beta) = \sin(\Omega_2 t) + \phi_\beta \cos(\Omega_2 t)$ when $\phi_\beta \ll 1$. It turns out that ϕ_β causes a cross potential term in the trap potential

$$\Delta U = -\frac{B_1^{'2}\mu_B \phi_\beta}{64 B_0} xy \tag{4.33}$$

This cross term will cause the principal axes of the trap rotate by an angle θ_p and change the trapping frequencies as well.

$$U = \frac{1}{2}m(\omega_z^2 z^2 + a_x^2 x^2 + a_y^2 y^2 + 2Kxy) \rightarrow \frac{1}{2}m(\omega_z^2 Z^2 + \omega_x^2 X^2 + \omega_y^2 Y^2) \tag{4.34}$$

where

$$mK = -\frac{\mu_B B_1^{'2}}{64 B_0}\phi_\beta \tag{4.35}$$

$$\tan(2\theta_p) = \frac{2K}{a_x^2 - a_y^2} \tag{4.36}$$

$$\omega_x^2 = \frac{1}{2}(a_x^2 + a_y^2 + \sqrt{(a_x^2 - a_y^2)^2 + 4K^2}) \tag{4.37}$$

$$\omega_y^2 = \frac{1}{2}(a_x^2 + a_y^2 - \sqrt{(a_x^2 - a_y^2)^2 + 4K^2}) \tag{4.38}$$

We assume the Bragg beams to be perfectly aligned and allow the ϕ_β term to rotate trap axes. So we set $\psi_x = \theta_p$, $\psi_y = -\theta_p$ and $\xi_x = \xi_y = 0$(so that $\Delta\Phi_z$ can be ignored) in Eq (4.30) - (4.31) to get the magnitude for the signal phase

$$\Delta\Phi = \frac{2mv_B^2 \sin 2\theta_p}{\hbar\omega_x}\sin\omega_x(t_2 + t_1) - \frac{2mv_B^2 \sin 2\theta_p}{\hbar\omega_y}\sin\omega_y(t_2 + t_1) \tag{4.39}$$

$$\frac{d(\Delta\Phi)}{dt_2} = \frac{2mv_B^2 \sin 2\theta_p}{\hbar}(\cos\omega_x(t_2 + t_1)) - \frac{2mv_B^2 \sin 2\theta_p}{\hbar}(\cos\omega_y(t_2 + t_1)) \tag{4.40}$$

We substitute θ_p, ω_x and ω_y by expressions in Eq (4.36)-(4.38)

$$\frac{d(\Delta\Phi)}{dt_2} = \frac{2mv_B^2}{\hbar}\frac{2K}{\sqrt{4K^2+(a_x^2-a_y^2)^2}}(\cos\sqrt{\frac{1}{2}(a_x^2+a_y^2+\sqrt{(a_x^2-a_y^2)^2+4K^2})}(t_2+t_1))- \\ \frac{2mv_B^2}{\hbar}\frac{2b}{\sqrt{4K^2+(a_x^2-a_y^2)^2}}(\cos\sqrt{\frac{1}{2}(a_x^2+a_y^2-\sqrt{(a_x^2-a_y^2)^2+4K^2})}(t_2+t_1)) \tag{4.41}$$

We are focused on the ϕ_β effect here. So we set $a_x = a_y = A$ as we did in the experiment. And we take the derivative of $d(\Delta\Phi)/dt_2$ respect to K

$$\frac{d(\Delta\Phi)}{dt_2} = \frac{2mv_B^2}{\hbar}(\cos\sqrt{A^2+K}(t_2+t_1)) - \frac{2mv_B^2}{\hbar}(\cos\sqrt{A^2-K}(t_2+t_1)) \tag{4.42}$$

$$\frac{\partial^2(\Delta\Phi)}{\partial t_2\partial K} = -\frac{2mv_B^2}{\hbar}(t_2+t_1)\left(\frac{\sin\sqrt{A^2+K}(t_2+t_1)}{2\sqrt{A^2+K}} + \frac{\sin\sqrt{A^2-K}(t_2+t_1)}{2\sqrt{A^2-K}}\right) \tag{4.43}$$

When $K = 0$, Based on Eq (1.16), Eq (4.35) and Eq (4.43)

$$\frac{\partial^2(\Delta\Phi)}{\partial t_2\partial\phi_\beta} = \frac{\partial^2(\Delta\Phi)}{\partial t_2\partial K}\frac{\partial K}{\partial\phi_\beta} = \frac{5\pi mv_B^2}{7\hbar} \approx 4.20\times10^5/\text{s} \tag{4.44}$$

The theory prediction is in the same order with the experiment result 3.44×10^5/s. The difference could be caused by the unharmonicity of the trap.

4.2 Numerical Phase Calculation

In the last section we considered just two sources of imperfection: the ϕ_β phase and t_2 errors. Now we want to consider the more general problem of all the different types of errors that could occur. The calculation will now be numerical. Besides the Bragg laser beams, we also incorporate some small parameters in the expression for

the potential energy of the trap

$$\begin{aligned} V(r) =& \frac{1}{2}m\omega_0^2[(1+\alpha)x^2 + (1-\alpha)y^2 + (1+\beta)z^2 + 2\gamma_1 xy + 2\gamma_2 xz + 2\gamma_3 yz] \\ &+ m\omega_0^2(\frac{1}{3}az^3 + \frac{1}{4}b\rho^4 + \frac{1}{2}c\rho^2 z) \end{aligned} \tag{4.45}$$

where the potential at the coordinate origin is defined to be zero, ω_0 is the nominal horizontal oscillation frequency, and $\rho^2 = x^2 + y^2$. The parameters α, β and γ_i are dimensionless, a and c have units of 1/(length), and b has units 1/(length)^2. A suitable length scale is the nominal orbit radius $R_0 = v_B/\omega_0$. There are of course other possible anharmonic terms, but let us focus on these three at first. This is motivated by the expansion of ideal time-orbiting trap potential. We ignore any linear gradient due to a mismatch between gravity and the trap levitation force.

Assuming that the atom packets move according to Newton's law,

$$m\frac{dv}{dt} = \mathbf{F} = -\nabla V \tag{4.46}$$

with here

$$F_x = -m\omega_0^2[(1+\alpha)x + \gamma_1 y + \gamma_2 z + b\rho^2 x + cxz] \tag{4.47}$$

$$F_y = -m\omega_0^2[(1-\alpha)y + \gamma_1 x + \gamma_3 z + b\rho^2 y + cyz] \tag{4.48}$$

$$F_z = -m\omega_0^2[(1+\beta)z + \gamma_2 x + \gamma_3 y + az^2 + \frac{1}{2}c\rho^2] \tag{4.49}$$

We use the above three equations to numerically compute the trajectories for each of the four packets. Note that the mass drops out of the equations.

Interpreting the phase effects will be easiest if they can be categorized in terms of an expansion around small dimensionless parameters, as in Table 4.1. We take the initial condensate position to be $\mathbf{r}_0$ and the initial velocity to be $\mathbf{v}_0$. These parameters drop out in a purely harmonic trap, but can matter when combined with

α	xy asymmetry
β	z asymmetry
$\gamma_1, \gamma_2, \gamma_3$	harmonic cross terms
$\psi_x^{'}$	x Bragg alignment term $\sin\psi_x \cos\xi_x$
$\psi_x^{''}$	x Bragg alignment term $\sin\psi_x \sin\xi_x$
$\psi_y^{'}$	y Bragg alignment term $\sin\psi_y \cos\xi_y$
$\psi_y^{''}$	y Bragg alignment term $\sin\psi_y \sin\xi_y$
a	anharmonic term
b	anharmonic term
c	anharmonic term
$\mathbf{r}_0$	initial position (vector)
$\mathbf{v}_0$	initial velocity (vector)
δ_1	timing error of splitting pulse
δ_2	timing error of recombination pulse

Table 4.1: Small parameters for laser and trap imperfections

anharmonicity. In a harmonic trap we could define $\delta_1 = \omega_0 t_1 - \pi/2$ and $\delta_2 = \omega_0 t_2 - 2\pi$. However, the γ and b terms change the atom oscillation frequency.

Empirically, we set t_1 and t_2 in the actual trap, so it seems best to define them relative to the real trajectories. We therefore propose to define the nominal value of t_1 as the time when the y-component of the motion reaches its maximum. Specifically, numerically determine t_{1o} for a given set of trap parameters as the time when $y_{1R} - y_{1L}$ reaches a local maximum. Then define $\delta_1 = \omega_0(t_1 - t_{1o})$. Similarly, we define t_{2o} as the time when the recombining packets are closest, such that $\mid r_{R+} - r_R \mid^2 + \mid r_{L+} - r_L \mid^2$ has a local minimum. Then define $\delta_2 = \omega_0(t_2 - t_{2o})$.

In the original experiment, the β parameter was not small. However, we hope to achieve nominal spherical symmetry with the new system. Anticipating this, we consider β as a small parameter. And we assume that the x Bragg directions are the same for both interferometers.

The final phase $\Delta\Phi$ will be a function of the small parameters, collectively ϵ_i. We would like to express this function as a Taylor expansion up to third order. The way

to achieve this is as follows: Start with all the parameters at zero. Then determine the one-parameter contributions by making small variations in each parameter in turn, and fitting the response of $\Delta\Phi$ up to 3rd order polynomial functions. Next, determine the two-parameter response by varying each two parameters combination and fitting polynomial functions. If any one-parameter contribution was found for that parameter, that variation can be subtracted from $\Delta\Phi$ to simplify finding the two-parameter dependence. It is then necessary to vary the parameters in pairs (ϵ_1, ϵ_2) and look for a dependence proportional to $\epsilon_1\epsilon_2$ or $\epsilon_1\epsilon_2^2$. Finally, consider the three-parameter response $\epsilon_1\epsilon_2\epsilon_3$. There are about 1500 different combinations to consider. We vary each three parameters combination and fit to 3rd order polynomial functions. Similarly, if any one-parameter contribution or two-parameter contribution was found, that variation can be subtracted from $\Delta\Phi$ to simplify finding the three-parameter dependence.

Table 4.2 shows the sensitive parameter terms and corresponding polynomial coefficients for a 2 Hz trap. The wave number k and the radius of the circular trajectory R are used to make these coefficients dimensionless. $k = 2\pi/\lambda$ where λ is 780.2 nm. $R = v_B/\omega$ where v_B is 11.8 mm/s and ω is $2\pi \times 2$ Hz. Among these polynomial functions, the most important terms with the largest dimensionless coefficients are: $\gamma_1\delta_1$, $\gamma_1\delta_2$, $\gamma_2\psi_y''\delta_1$, $\gamma_3\psi_x''\delta_2$, $\alpha\psi_x'\delta_1$ and $\alpha\psi_x'\delta_2$. We can tell from them that the most important factors related to the signal phase are: 1. Timing errors 2. Harmonic cross terms 3. Bragg alignments 4. xy asymmetry.

Table 4.2: Sensitive parameter terms and corresponding polynomial coefficients.

Terms	**Coefficients**	**95% confidence bounds**
$\gamma_1 kR$	3.9422093	(3.9422091, 3.9422094)
$\psi_x' kR$	-7.8842969	(-7.8842973, -7.8842965)
		Continued on next page

Table 4.2 – continued from previous page

Terms	Coefficients	95% confidence bounds
$\psi_y^{'}kR$	-7.8842969	(-7.8842973, -7.8842967)
$\gamma_1^3 kR$	-58.329	(-58.502, -58.155)
$\gamma_1 \alpha kR$	-72.965	(-72.991, -72.940)
$\alpha\psi_x^{'}kR$	-3.942176	(-3.942189,-3.942163)
$\alpha^2\psi_x^{'}kR$	152.649	(152.634, 152.665)
$\alpha\psi_y^{'}kR$	3.942146	(3.942142, 3.942149)
$\alpha^2\psi_y^{'}kR$	6.77596	(6.77218, 6.77975)
$\gamma_1\gamma_3^2 kR$	-38.875310	(-38.875318, -38.875303)
$\gamma_1^2\psi_x^{'}kR$	-155.8244	(-179.3387, -132.3100)
$\gamma_1^2\psi_y^{'}kR$	57.836242	(57.836238, 57.836246)
$\gamma_1{\psi_y^{'}}^2 kR$	-1.97108424	(-1.97108843, -1.97108005)
$\gamma_1{\psi_x^{''}}^2 kR$	-1.97108533	(-1.97108836, -1.97108230)
$\gamma_1{\psi_y^{''}}^2 kR$	-1.97108533	(-1.97108836, -1.97108230)
$\gamma_1 bkR^3$	-4.17045	(-4.19525, -4.14565)
$\gamma_1 b^2 kR^5$	43.7006	(10.9107 76.4200)
$\gamma_1 c^2 kR^3$	-47.150316	(-47.150328, -47.150305)
$\gamma_1\delta_1$	234070	(233919, 234212)
$\gamma_1\delta_2$	234070	(233919, 234212)
$\gamma_2\gamma_3 kR$	57.836228	(57.836225, 57.836231)
$\gamma_2^2\psi_y^{'}kR$	57.836252	(57.836250, 57.836255)
$\gamma_2\psi_y^{''}kR$	3.942141	(3.942139, 3.942144)
$\gamma_3^2\psi_x^{'}kR$	57.836256	(57.836252, 57.836260)
$\psi_x^{'}bkR^3$	1.9710772	(1.9710723, 1.9710821)
$\psi_x^{'}b^2kR^5$	-1.7295	(-1.7359, -1.7230)

Continued on next page

Table 4.2 – continued from previous page

Terms	Coefficients	95% confidence bounds
$\psi_y^{'} bkR^3$	1.9710772	(1.9710723, 1.9710821)
$\psi_y^{'} b^2kR^5$	-1.7295	(-1.7359, -1.7230)
$\psi_x^{'2}\psi_y^{'}kR$	3.942154	(3.942151, 3.942156)
$\psi_x^{'}\psi_y^{'2}kR$	3.942155	(3.942152, 3.942158)
$\alpha\gamma_1 bkR^3$	220.3365	(154.7229, 285.95)
$\alpha\gamma_2\gamma_3 kR$	-1.2299	(-1.5292, -0.9306)
$\alpha\gamma_2\psi_y^{''}kR$	23.7728	(23.7272, 23.8185)
$\alpha\gamma_3\psi_x^{''}kR$	-136.1754	(-136.2015, -136.1493)
$\alpha\psi_x^{'} bkR^3$	-33.9246	(-34.2722, -33.5769)
$\alpha\psi_x^{'}\delta_1$	-374347.1766	(-401063.8494, -347630.5038)
$\alpha\psi_x^{'}\delta_2$	-374347.1625	(-401063.8142, -347630.5109)
$\alpha\psi_y^{'} bkR^3$	-42.362	(-42.5038, -42.2201)
$\alpha\psi_y^{'}\delta_1$	93587.40111	(67026.79513, 120148.0071)
$\alpha\psi_y^{'}\delta_2$	93587.41167	(67026.8084, 120148.0149)
$\beta\gamma_2\gamma_3 kR$	-38.8752	(-39.021, -38.7293)
$\beta\gamma_2\psi_y^{''}kR$	18.9288	(18.8744, 18.9831)
$\beta\gamma_3\psi_x^{''}kR$	58.3611	(58.3437, 58.3785)
$\gamma_1\gamma_2\psi_x^{''}kR$	-233.4434	(-268.2453, -198.6416)
$\gamma_1\gamma_3\psi_y^{''}kR$	18.9288	(18.9284, 18.9292)
$\gamma_1 cz_0 kR$	15.8229	(15.8223, 15.8235)
$\gamma_2\gamma_3 bkR^3$	-90.8384	(-90.8657, -90.8111)
$\gamma_2\gamma_3\delta_1$	-58492.20629	(-59563.34815, -57421.06443)
$\gamma_2\gamma_3\delta_2$	35094.95208	(34037.08741, 36152.81676)
$\gamma_2\psi_y^{''}bkR^3$	-18.6951	(-18.7126, -18.6776)

Continued on next page

Table 4.2 – continued from previous page

Terms	Coefficients	95% confidence bounds
$\gamma_2\psi_y^{''}\delta_1$	233967.935	(233701.4564, 234234.4137)
$\gamma_2\psi_y^{''}\delta_2$	46793.4018	(46500.35527, 47086.44833)
$\gamma_3\psi_x^{'}\psi_y^{''}kR$	3.9421	(0.4190, 7.4653)
$\gamma_3\psi_x^{''}bkR^3$	-58.3379	(-58.4393, -58.2366)
$\gamma_3\psi_x^{''}\delta_1$	187174.4293	(187042.8875, 187305.971)
$\gamma_3\psi_x^{''}\delta_2$	233968.0066	(233809.8879, 234126.1253)
$\gamma_3 cx_0kR$	16.2609	(16.2609, 16.2609)
$\psi_x^{'}b\delta_1R^2$	93603.99346	(63386.37602, 123821.6109)
$\psi_x^{'}b\delta_2R^2$	93604.00615	(63386.18248, 123821.8298)
$\psi_y^{'}b\delta_1R^2$	93604.02385	(63386.42145, 123821.6262)
$\psi_y^{'}b\delta_2R^2$	93604.00608	(63386.16807, 123821.8441)

To sum up this chapter, we explore the imperfections of Bragg laser beams and trapping potential. We introduce some small parameters into the directions of laser beams and derive the ideal signal phase expression at first. Then we incorporate some anharmonic terms into the magnetic trap potential and see how the signal phase varies according to small parameters.

Chapter 5

Conclusion

This thesis presents some research works in preparation for future atom interferometry experiments in our lab.

We designed and constructed a new atom chip based on double layer spiral copper wires with different chirality. This chip is mainly used to provide pure magnetic gradient field for BEC production and interferometry experiments, which simplifies the operation and analysis. And it reduces the required currents and power consumption. A supporting chamber for testing the atom chip is also designed and used to adjust trapping frequencies and allow laser beams coming through. The chamber will overcome the thermal variation problem our current experiment apparatus suffers from.

Besides, we developed an image analysis program to quickly obtain the trajectories of all atom packets from an image sequence. The time it takes is negligible compared to manually dealing with each atom packet in an image sequence. An experiment control program is also built to automate image generations and collections. It will help to decrease the experiment running time from two minutes to ten seconds.

Program Metrics	Requirements
System Volume	0.5 liter (sum of components)
Power	10 W (sum of components)
Allen deviation	< 30 mrad*s$^{(1/2)}$ for 1-3600 s
Phase repeatability	10 mrads 2/24/2 hr on/off/on
Acceleration invariance	1 mrad (3 tests: 0°, 90°, 180°)
Temperature stability	10 ppb/ °C of interferometer area
Mechanical vibration	measurement 0.01 g^2/Hz (10 Hz to 1000 Hz)

Table 5.1: A-PhI Gyroscope Program Metrics

At last, in order to improve the understanding and controlling of the interferometer phase shifts, we derived the analytical expression for the ideal signal phase at first and then numerically fit signal phase to polynomial functions of small parameters related to the trap and Bragg laser imperfections. The fitting results will be useful when we use the interferometer for rotation sensing. The signal phase caused by rotations can be obtained by substracting contributions caused by other sources from the total interferometer phase shifts.

With these progresses we are getting closer to realize a compact and portable microchip-based atom gyroscope for rotation sensing and inertial navigation, namely the A-PhI program introduced in Chapter 1. This program includes two phases. The initial phase of the program shall be the demonstration of the BIGTOP interferometer in a compact integrated system. The second phase of the program shall be improvement of the BIGTOP performance, suppression of environmental effects, and implementation in a miniaturized system. The requirements for the second phase are listed in Table 5.1. My work will be critical for meeting all of the second phase requirements:

The volume of the testing chamber we designed is less than 0.4 liter, which satisfies the system volume requirement.

When the bias field is 20 Gauss, the total power consumption of the chip and bias

coils is 7.59 W, which is less than 10 W and therefore satisfies the power requirement.

In order to meet the Allen deviation requirement, the chip design will allow the experiment running time decreased from 2 minutes to 10 seconds. Since the experiment rate is increased, the uncertainty of phase measurement is decreased when we average measurement results during a given amount of time.

When we run same measurements during consecutive days, we hope the interferometer phase shifts vary within 10 mrads. The phase calculation result improves our understanding of what factors contribute to the phase drift. The good thermal stability of the chip will make trap parameters less fluctuated. These progresses will help to meet the phase repeatability requirement.

While we flip the system by 90° and 180° for same measurements, we hope the the phase drifts within 1 mrad. Our phase analysis can tell how well we have to control the atom potential as we rotate the system to realize the acceleration invariance. It also tells us how well we will have to stabilize the system to achieve the temperature stability.

The low inductance of the chip design permits rapid changes of the gradient field to compensate for mechanical vibrations applied on the system.

Up to the point where this thesis is finished, we are in the initial phase of the program. We have reached milestones such as the delivery of current driver,the delivery of prototype cell system and 10% interference visibility. And we are working on performance improvement tasks including realizing 10 mm^2 Sagnac area, the delivery of compact system and BEC in compact system.

In order to practically apply our approach in a gyroscope in a vehicle, there are vibration and temperature compensation systems need to be built by our collabora-

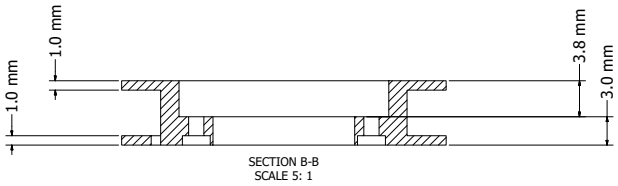
tors. For the vibration compensation system, a compact conventional accelerometer will be mounted to the vacuum cell including magnetic and optics structure. The acceleration values will be read by a digital control system, which will determine how the trap currents must be adjusted to keep the trapping potential fixed in an inertial frame. For the temperature compensation system, we need sensors to measure temperature variations which are then compensated by adjusting the magnet currents. And the entire vacuum system will be mounted on passive isolation dampers in order to prevent the system from experiencing vibrations that could damage the vacuum system or optics.

Appendices

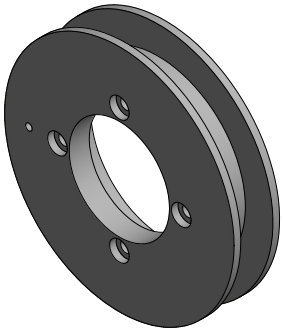
Appendix A

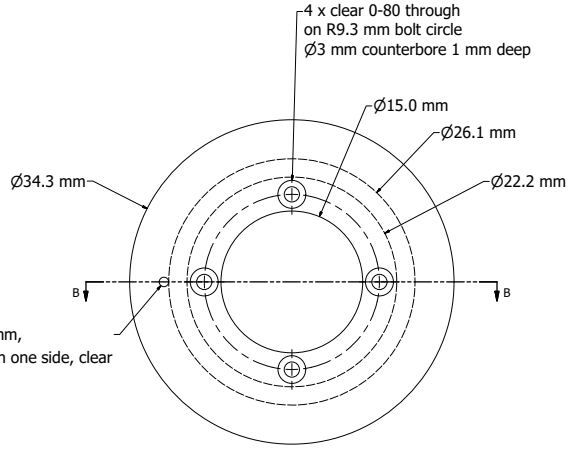
AutoCAD Design for testing chamber

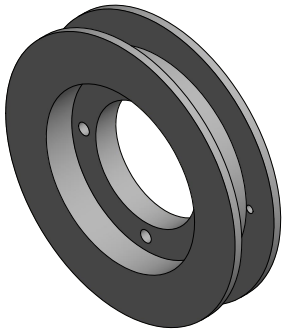
This chapter include AutoCAD files of the testing chamber. It includes top z coil holder, bottom z coil holder, x and y bias coil holder, base, top base and chip holder



SECTION B-B
SCALE 5: 1

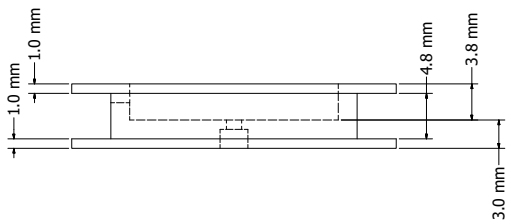


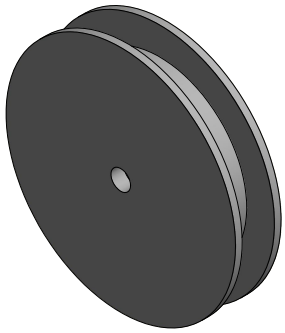


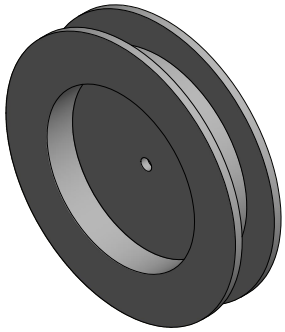


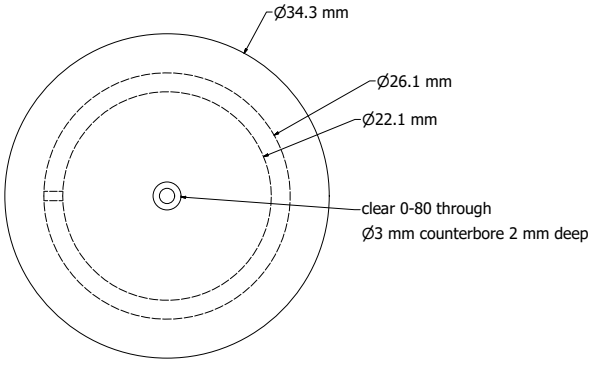
Shapal Hi-M soft
1 piece

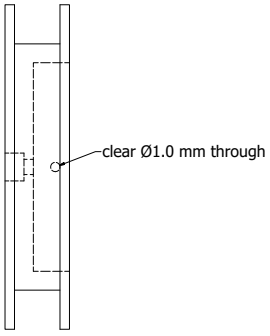
DRAWN luozh	3/7/2019	
CHECKED		
QA		TITLE: Top z coil
MFG		
APPROVED		SIZE: D; DWG NO: 5; REV
		SCALE: 5: 1; SHEET 1 OF 1





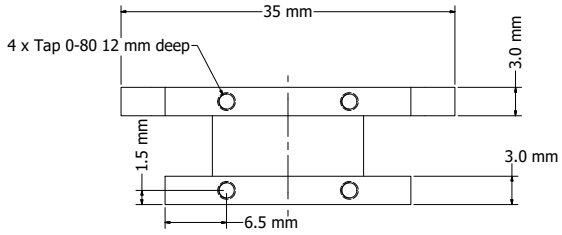


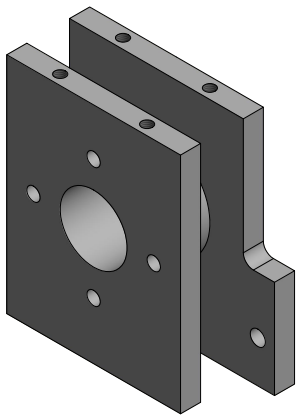


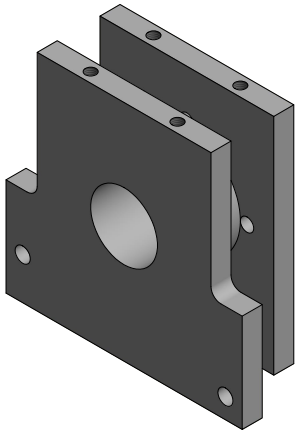


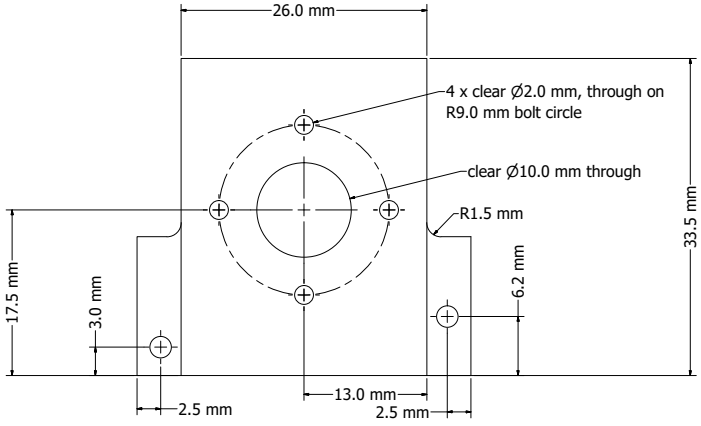
Shapal Hi-M soft
1 piece

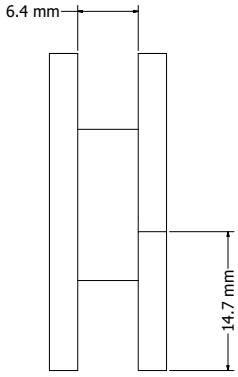
DRAWN luozh	3/24/2019	
CHECKED		
QA		TITLE: Bottom z coil
MFG		
APPROVED		SIZE: D; DWG NO: 6; REV
		SCALE: 5 : 1; SHEET 1 OF 1





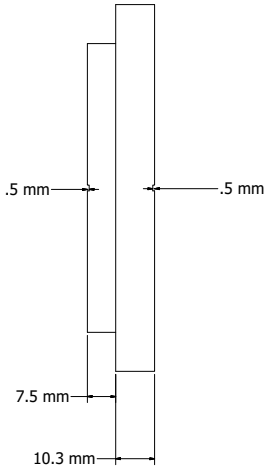


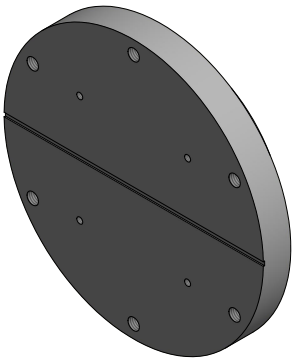


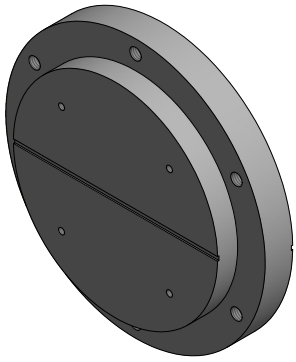


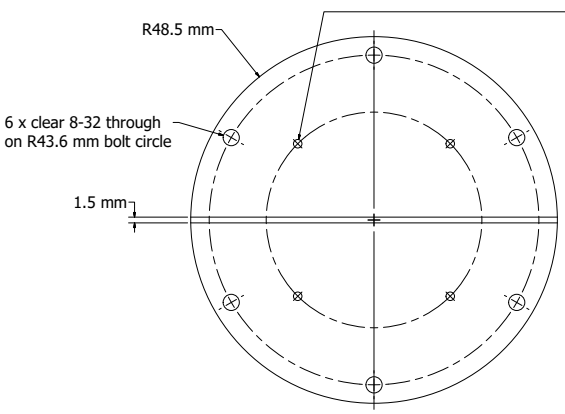
Shapal Hi-M soft
4 pieces

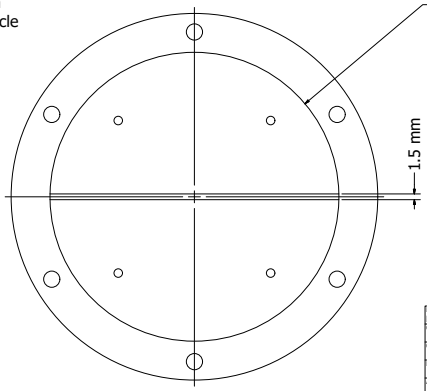
DRAWN luozh	4/30/2019		
CHECKED			
QA		TITLE	
MFG		Bias Holder	
APPROVED			
		SIZE D	DWG NO 2
		SCALE 5 : 1	SHEET 1 OF 1





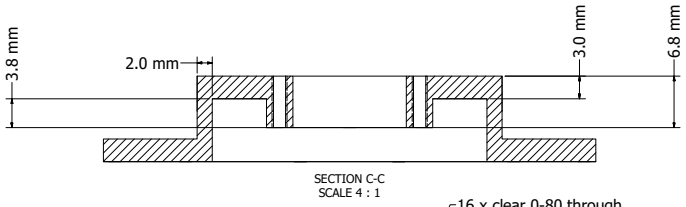




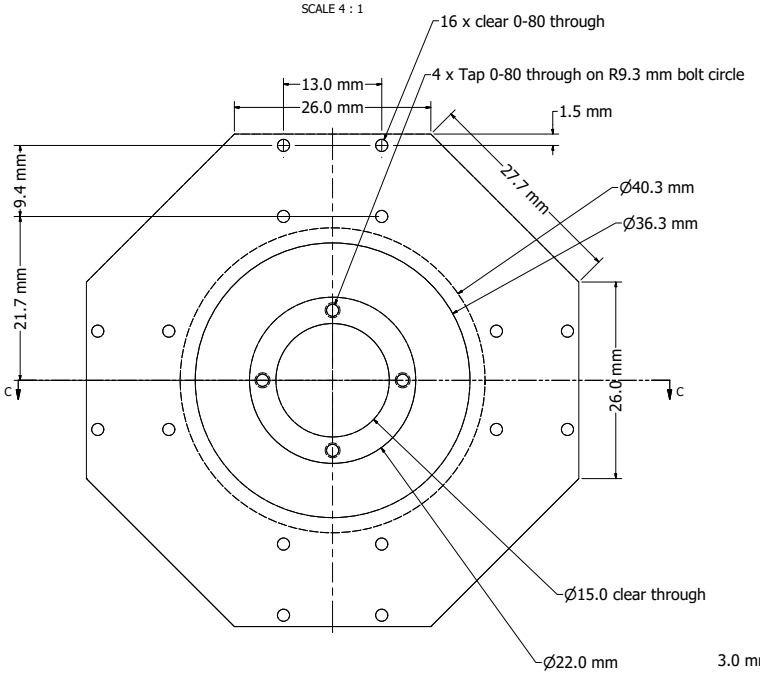


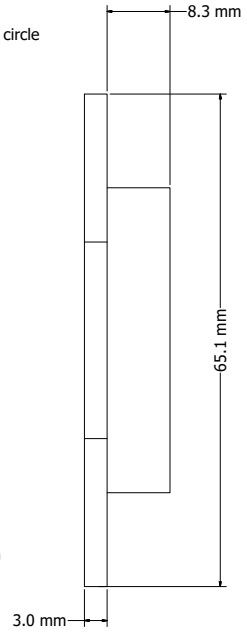
Shapal Hi-M soft
1 piece

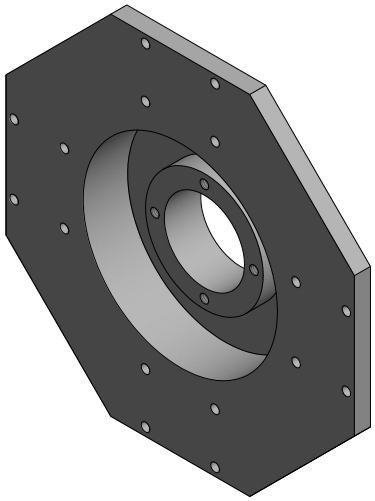
DRAWN luozh	3/11/2019	
CHECKED		
QA		TITLE: Base
MFG		
APPROVED		SIZE: D; DWG NO: 4; REV
		SCALE: 2 : 1; SHEET 1 OF 1



SECTION C-C
SCALE 4 : 1

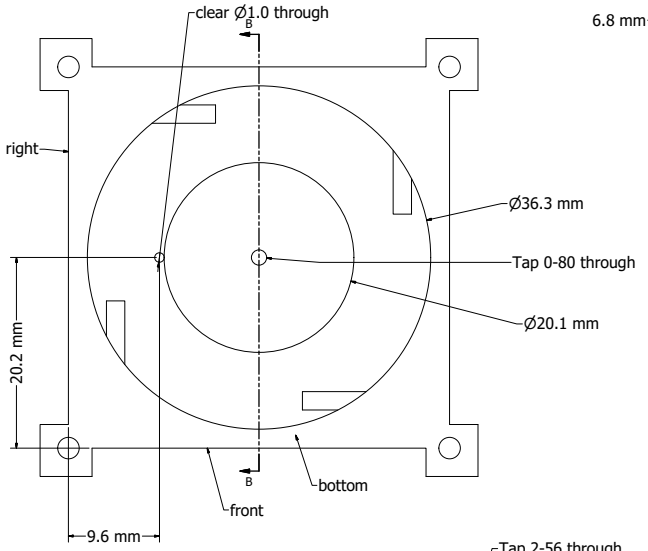


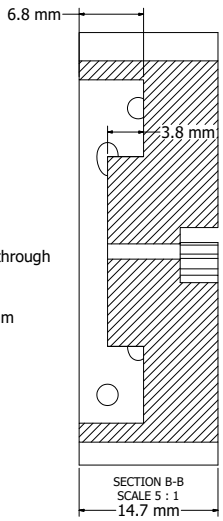


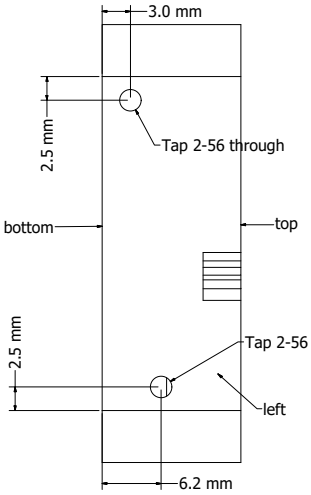


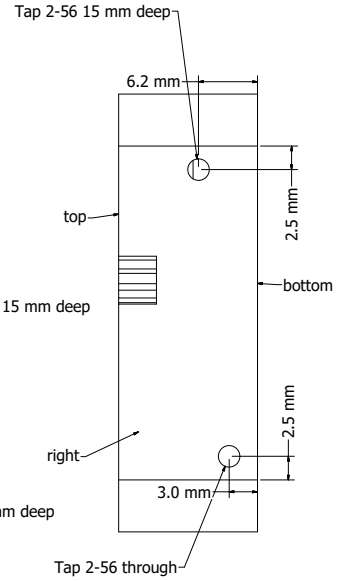
Shapal Hi-M soft
1 piece

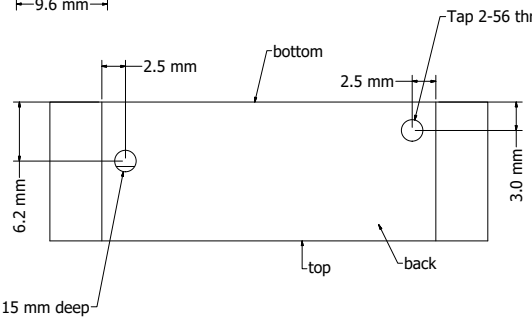
DRAWN luozh	3/24/2019		
CHECKED		TITLE	
QA		Top Base	
MFG			
APPROVED		SIZE D	DWG NO top base_2 REV
		SCALE 4 : 1	SHEET 1 OF 1

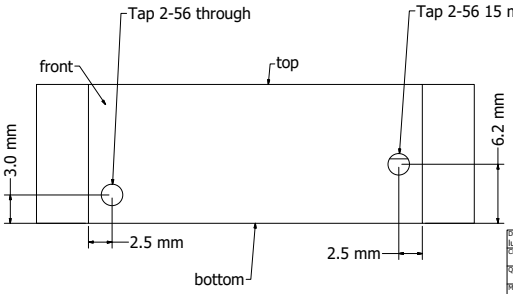






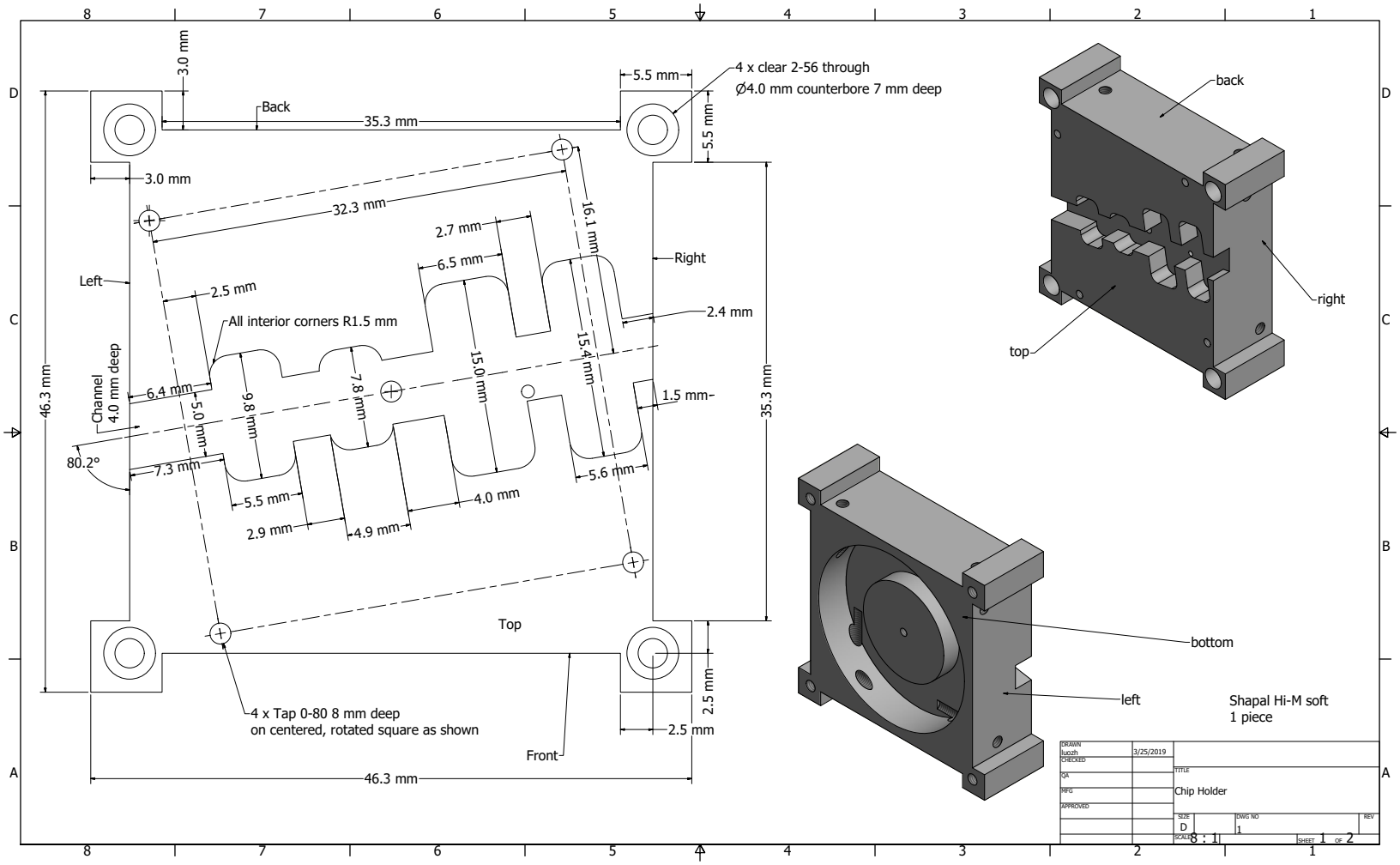






Shapal Hi-M soft
1 piece

DRAWN luozh	3/25/2019	
CHECKED		TITLE: Chip Holder
QA		
MFG		
APPROVED		SIZE D · DWG NO 1 · REV
		SCALE 5 : 1 · SHEET 2 OF 2


3.0 mm
Back
35.3 mm
5.5 mm
4 x clear 2-56 through
Ø4.0 mm counterbore 7 mm deep
5.5 mm
3.0 mm
32.3 mm
16.1 mm
2.7 mm
6.5 mm
Right
Left
2.5 mm
All interior corners R1.5 mm
2.4 mm
15.4 mm
15.0 mm
7.8 mm
9.8 mm
5.0 mm
6.4 mm
Channel
4.0 mm deep
46.3 mm
35.3 mm
1.5 mm
80.2°
7.3 mm
5.5 mm
2.9 mm
4.9 mm
4.0 mm
5.6 mm
Top
4 x Tap 0-80 8 mm deep
on centered, rotated square as shown
Front
2.5 mm
2.5 mm
46.3 mm
back
right
top
bottom
left
Shapal Hi-M soft
1 piece
DRAWN
luozh
3/25/2019
CHECKED
QA
MFG
APPROVED
TITLE
Chip Holder
SIZE
D
DWG NO
1
REV
SCALE
8 : 1
SHEET 1 OF 2

Appendix B

Codes for image analysis program

This chapter includes MATLAB code for image analysis program

In addition to the tracking purpose introduced in the main chapter, we also want to use the program for measuring interferometer outputs. These will have a set of six packets (three on each side) for the interferometer outputs B.1.

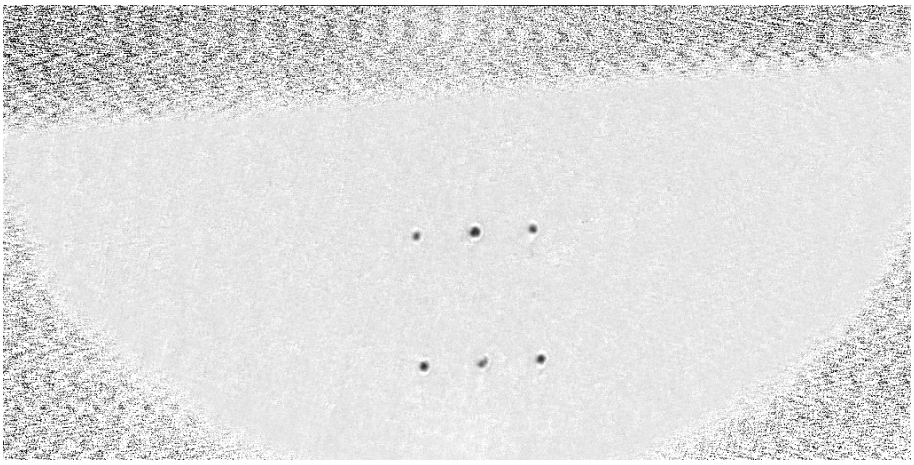


Figure B.1: Interferometer outputs after recombination

The packets will always be in the same places but their amplitudes will vary. We hope to use the program to analyze a set of images like B.1. To switch from tracking

atoms to measuring interferometer outputs, the user need to click a checkbox called "no position change" in the UI.

There are some changes made to image processing steps for measuring interferometer outputs compared to tracking purposes. At first, a Gaussian image filter is applied to denoise the image. It is a weighed averaging filter. The closer a pixel is to the target pixel, the more weight it is assigned to. Therefore, it is less possible to wipe out weak atom clouds with small B values. Besides, connected component algorithm is applied to count and label atom clouds. In a binarized image, this algorithm will count how many connected white pixel clusters in a black background. Compared to k-means clustering algorithm, we do not need to specify how many clusters we want to find in advance. The program will require manual selection whenever the number of clusters does not equal to 6. The situations include whole black images, whole white images, images with different numbers of clusters depending on whether there are residual clouds and whether the atom clouds are fully combined. In addition, before we apply connected component algorithm on the binarized image, we use a vertical line structural element to erode the binarized image. Fully combined circle shape clusters can usually stand through the image erosion while the partially combined atom clouds may be split into two clouds and therefore cause users' manual selections.

During experiments we also want to use the program to monitor each new image. To satisfy the requirement, we add a checkbox in the user interface called "Monitor". If the checkbox is clicked, the program will check the selected directory in every second and retrieve the latest image file. If it is a new image, the program will find the atom clouds using connected component algorithm and insert the fit results(widths and depths of clouds) as texts on the image. The program will stop monitoring if the checkbox is unchecked.

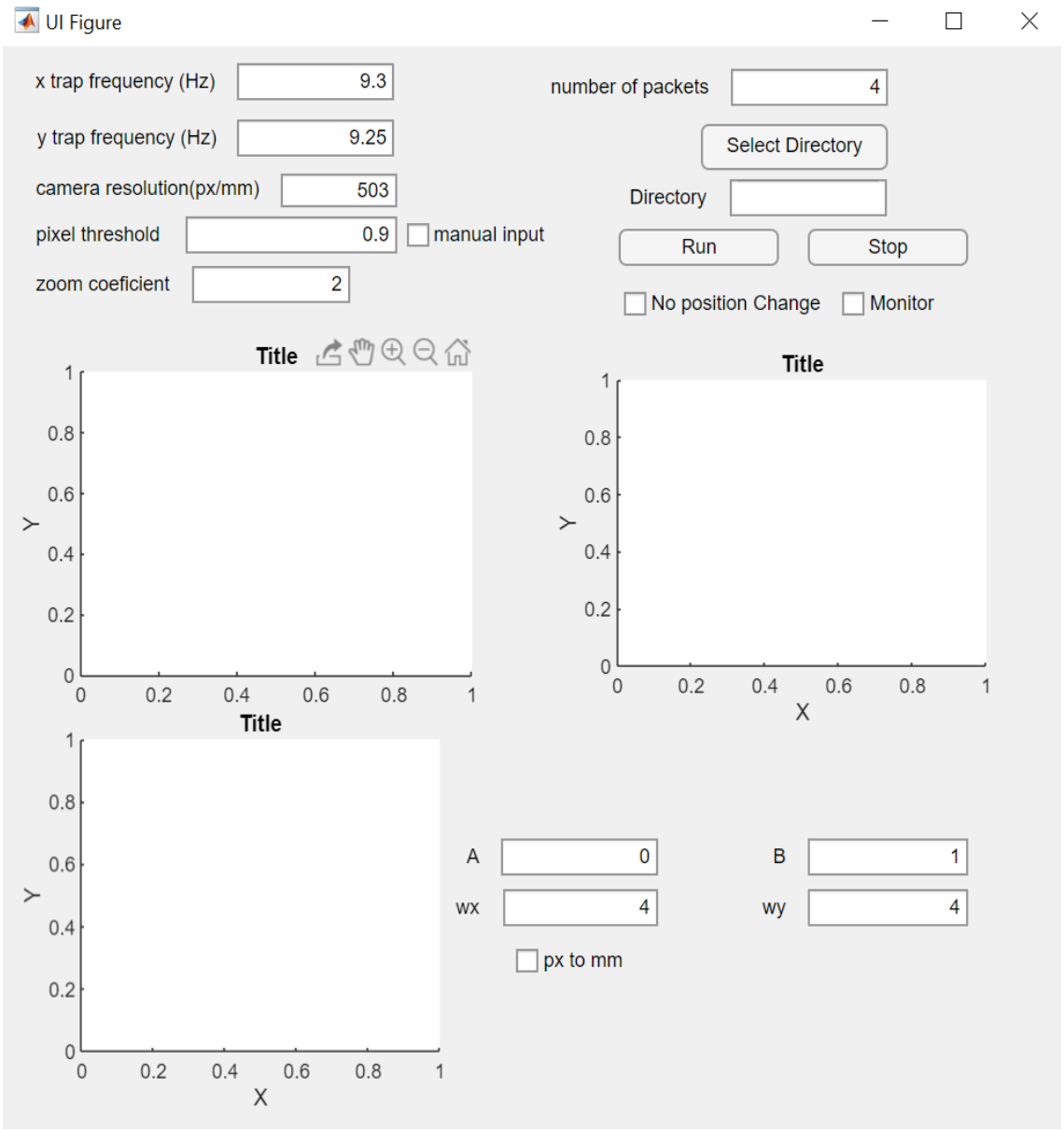


Figure B.2: The updated image analysis program user interface

```
classdef BEC_tracking_7_exported3 < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        SelectDirectoryButton           matlab.ui.control.Button
        UIAxes                          matlab.ui.control.UIAxes
        xtrapfrequencyHzEditFieldLabel  matlab.ui.control.Label
        xtrapfrequencyHzEditField       matlab.ui.control.NumericEditField
        pixelthresholdEditFieldLabel    matlab.ui.control.Label
        pixelthresholdEditField         matlab.ui.control.NumericEditField
        DirectoryEditFieldLabel         matlab.ui.control.Label
        DirectoryEditField              matlab.ui.control.EditField
        RunButton                       matlab.ui.control.Button
        ytrapfrequencyHzEditFieldLabel  matlab.ui.control.Label
        ytrapfrequencyHzEditField       matlab.ui.control.NumericEditField
        UIAxes_2                        matlab.ui.control.UIAxes
        cameraresolutionpxmmEditFieldLabel  matlab.ui.control.Label
        cameraresolutionpxmmEditField   matlab.ui.control.NumericEditField
        UIAxes2                         matlab.ui.control.UIAxes
        AEditFieldLabel                 matlab.ui.control.Label
        AEditField                      matlab.ui.control.NumericEditField
        BEditFieldLabel                 matlab.ui.control.Label
        BEditField                      matlab.ui.control.NumericEditField
        wxEditFieldLabel                matlab.ui.control.Label
        wxEditField                     matlab.ui.control.NumericEditField
        wyEditFieldLabel                matlab.ui.control.Label
        wyEditField                     matlab.ui.control.NumericEditField
        numberofpacketsEditFieldLabel   matlab.ui.control.Label
        numberofpacketsEditField        matlab.ui.control.NumericEditField
        zoomcoeficientEditFieldLabel    matlab.ui.control.Label
        zoomcoeficientEditField         matlab.ui.control.NumericEditField
        manualinputCheckBox             matlab.ui.control.CheckBox
        StopButton                      matlab.ui.control.Button
        pxtommCheckBox                  matlab.ui.control.CheckBox
        NopositionChangeCheckBox        matlab.ui.control.CheckBox
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: SelectDirectoryButton
        function SelectDirectoryButtonPushed(app, event)
            clc
            close all
            selpath = uigetdir;
            app.UIFigure.Visible = 'off';
            app.UIFigure.Visible = 'on';
            oldFolder = cd(selpath);
            app.DirectoryEditField.Value = selpath;
```

```
            Show1st = dir(fullfile(app.DirectoryEditField.Value,'*.tif'));
            imshow( Show1st(1).name,'Parent',app.UIAxes);
%             app.Image.ImageSource = Show1st(1).name;
        end

        % Button pushed function: RunButton
        function RunButtonPushed(app, event)
        if app.NopositionChangeCheckBox.Value
            close all
            S = dir(fullfile(app.DirectoryEditField.Value,'*d.tif'));%read images in the working directory
            cd(app.DirectoryEditField.Value)%switch to the working directory
            Ni = app.numberofpacketsEditField.Value;% the number of atom clouds
            wx = app.wxEditField.Value;% fitting parameter: the width of clouds in x direction
            wy = app.wyEditField.Value;% fitting parameter: the width of clouds in y direction
            AA = app.AEditField.Value;% fitting parameter: A
            B = app.BEditField.Value;% fitting parameter: B
            tr = 512;% total number of rows for the image
            se = strel('rectangle',[6 1]);% a structiral element for image erosion, it is a vertical line
            n = length(S);% the number of images in the directory
            P = zeros(n,2*Ni+1);%A matrix for saving time, x and y position
            P(:,1) = -1;% initialize the first column to be -1 for the time stamp
            W = zeros(n,2*Ni+1);% A matrix for saving fitting parameter wx and wy
            D = zeros(n,2*Ni+1);%A matrix for saving fitting parameter A and B
            for i = 1:n
                S(i).time = str2double(S(i).name(1:end-5));%read the time from file name for each image
            end
            SS = struct2table(S); % convert the struct array to a table
            sortedSS = sortrows(SS, 'time'); % sort the table by time stamps
            S = table2struct(sortedSS); % change it back to struct array if necessary
            button = 1;% a flag indicating whether the mouse is right or left clicked
            Sel = 0;% a flag indicating whether the center of atom clouds has been selected or not
            for k=1:n%iterating image dataset
                if Sel == 0% if the center of atom clouds has not been selected
                    A = imread(S(k).name);% read a image
                    imshow(A)
                    title('select the center of atom clouds or right click to skip')
                    hold on
                    [selx,sely,button] = ginput(1);
                    if button == 3% if the mouse is right clicked, then skip the image
                        continue
                    else
                        Sel = 1;% otherwise the center has been selected, then break from the loop
                        startidx = k;% this image will be the starting one for the
```

```
following processing
                        break
                    end
                end
            end
            xmin = selx-2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
xtrapfrequencyHzEditField.Value)-5;% calculating the left x coordinatas for cropping↙
images
            ymin = sely-2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
ytrapfrequencyHzEditField.Value)-5;% calculating the top y coordinatas for cropping↙
images
            width = 2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
xtrapfrequencyHzEditField.Value)*2+10;% the width of the cropped image
            height = 2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
ytrapfrequencyHzEditField.Value)*2+10;% the height of the cropped image
            for k = startidx:n% iterating the image dataset
                A = imread(S(k).name);% read images
                Ag = imgaussfilt(A,2.5);% filter the original image with a gaussian↙
filter
                AC = imcrop(Ag,[xmin,ymin,width,height]);% crop the image based on the↙
selected cloud center and calculated width and height
                Ab = imbinarize(AC);% turn the crop image to a binarized image, the↙
atom clouds are expressed by black pixels, the background is white
                Ac = imcomplement(Ab);% turn the binarized image to its complement,↙
then atom clouds are expressed by white pixes, the background is black
                Ae = imerode(Ac,se);% erode the image to separate a partially↙
recombined atom cloud into two clouds
                CC = bwconncomp(Ae);% count the number of white regions in the↙
binarized image
                if CC.NumObjects ~= app.numberofpacketsEditField.Value || mean(A,'all')↙
< 0.5 %if the number of clouds is not equal to the number we want to find or the↙
average pixel value of the original image is below 0.5
                    figure
                    imshow(A)% show the image and require manual selection for atom↙
cloud positions or right click to skip
                    title(strcat('select',{' '},num2str(Ni),' positions (1. top to↙
bottom, 2. left to right) or right click to skip'))
                    hold on
                    P(k,1) = str2double(S(k).name(1:end-5));
                    for i=1:Ni
                        [x,y,button] = ginput(1);
                        if button == 3
                            break
                        end
                        P(k,2*i) = y;%save y position
                        P(k,2*i+1) = x;%save x position
                        title(strcat(num2str(Ni-i),' positions left'))%remind how many↙
left to choose
                        plot(x,y,'ro', 'MarkerSize', 10);% mark the selected positions↙
with red circles
```

```
                    end
                    close all
                else% when the number of atom clouds is what we expect
                    figure
                    imshow(A)
                    title(S(k).name)
                    hold on
                    c = regionprops(CC,'Centroid');%calculate the center position for each atom cloud
                    P(k,1) = str2double(S(k).name(1:end-5));% save the time stamp in the first column
                    for i=1:Ni
                        P(k,2*i) = c(i).Centroid(2)+ymin-1;%save the y coordinate in the original image
                        P(k,2*i+1) = c(i).Centroid(1)+xmin-1;%save the x coordinate in the original image
                        plot(P(k,2*i+1),P(k,2*i),'ro', 'MarkerSize', 10);% mark the founded atom clouds with red circles
                    end
                    pause(0.5)% pause some time for viewing purpose
                    close
                end
                if button == 3 % if users want to skip a image, then continue
                    button = 1;
                    P(k,1) = -1;
                    continue
                end
                N = 4*(wx+wy);% crop atom clouds from the original image and fit to gaussians to get more accurate result
                I = zeros(2*N+1,2*N+1,Ni);% A maxtrix for saving the cropped atom clouds
                for i = 1:Ni
                    I(:,:,i) =  imcrop(A,[P(k,2*i+1)-N,P(k,2*i)-N,2*N,2*N]);
                end
                x0 = [AA + 1,B,wy,N+1,wx,N+1];% savethe initial fitting parameter
                x = zeros(Ni,6);% save the final fitting result
                for i = 1:Ni
                    fun = @(x)D2GaussFunction1(x,I(:,:,i),N);
                    x(i,:) = fminsearch(fun,x0);% fit to gaussians using smallest mean square method
                    P(k,2*i:2*i+1) = [tr-(P(k,2*i)-N+x(i,4)-1) P(k,2*i+1)-N-1+x(i,6)];% save y and x position, the y coordinates originally start from the top, here we substract it from the total row number so that it starts from the bottom
                    W(k,2*i:2*i+1) = [x(i,3) x(i,5)];%save wy and wx
                    D(k,2*i:2*i+1) = [x(i,1)-1 x(i,2)];%save A and B
                end
                W(k,1) = P(k,1);%save the time stamp
                D(k,1) = P(k,1);%save the time stamp
            end
            W = W(find(P(:,1)>=0),:);%keep images which have not been skipped
```

```
            D = D(find(P(:,1)>=0),:);%keep images which have not been skipped
            P = P(find(P(:,1)>=0),:);%keep images which have not been skipped
            n = length(P(:,1));
            Pave3 = mean(P(1:3,2:end));%calculate the average positions using the first
three images
            for i = 4:n% iterate the rest of images and sort out positions for same
atom clouds in same columns
                TP = zeros(1,2*Ni+1);%a vector for saving postions temporarily
                TD = zeros(1,2*Ni+1);%a vector for saving A and B temporarily
                TW = zeros(1,2*Ni+1);%a vector for saving widths temporily
                for j = 1:Ni%for each image, compare the average position for each
cloud with positions in the current image and find the closest one
                    pd = inf;% the distance between the average position derive from
the first three images and current position
                    mdx = 0;% the index corresponding to the minimum distance
                    for k = 1:Ni
                        if pdist([Pave3(2*j-1) Pave3(2*j);P(i,2*k) P(i,2*k+1)]) < pd
                            pd = pdist([Pave3(2*j-1) Pave3(2*j);P(i,2*k) P(i,2*k+1)]);
                            mdx = k;
                        end
                    end
                    TP(2*j:2*j+1) = [P(i,2*mdx) P(i,2*mdx+1)];
                    TD(2*j:2*j+1) = [D(i,2*mdx) D(i,2*mdx+1)];
                    TW(2*j:2*j+1) = [W(i,2*mdx) W(i,2*mdx+1)];
                end
                P(i,2:2*Ni+1) = TP(2:2*Ni+1);
                D(i,2:2*Ni+1) = TD(2:2*Ni+1);
                W(i,2:2*Ni+1) = TW(2:2*Ni+1);
            end
            ImageLabel = zeros(Ni,3);%save the label number and positions
            A = imread(strcat(num2str(P(1,1)),'d.tif'));% mark the label number on the
first remaining image
            for i = 1:Ni
                ImageLabel(i,1) = i;%laber number
                ImageLabel(i,2) = P(1,2*i+1);%column number
                ImageLabel(i,3) = tr- P(1,2*i);%row number
            end
            RGB = insertText(A,ImageLabel(:,2:3),ImageLabel(:,1),'FontSize',
18,'BoxColor','red','BoxOpacity',0.4,'TextColor','green','AnchorPoint','Center');
            imwrite(RGB,'label.png');%write the label image in the same directory and
name it as "label"
            T = array2table(P);%convert the array to a table for saving in excel file
            Tname = cell(1,2*Ni+1);% a cell save the name for each column in T
            Tname(1) = cellstr('time');% the first column is time
            for i = 1:Ni
                Tname(2*i) = cellstr(strcat('y',num2str(i)));%columns named by y+label
number
                Tname(2*i+1) = cellstr(strcat('x',num2str(i)));%columns named by
x+label number
            end
```

```
            T.Properties.VariableNames = Tname;%assign names to the table T
            writetable(T,strcat(cd,'\data.xls'),'Sheet','positions');%write table into↙
excel file
            for i = 1:Ni%write wx,wy,A and B for each atom cloud into excel file
                TB = array2table([W(:,1) W(:,2*i:2*i+1) D(:,2*i:2*i+1)]);
                TB.Properties.VariableNames = {'Time' 'wy' 'wx' 'A' 'B'};
                writetable(TB,strcat(cd,'\data.xls'),'Sheet',num2str(i))
            end
            Parameter = table(app.xtrapfrequencyHzEditField.Value,app.↙
ytrapfrequencyHzEditField.Value,app.cameraresolutionpxmmEditField.Value,app.AEditField.↙
Value,app.BEditField.Value,app.wxEditField.Value,app.wyEditField.Value);
            Parameter.Properties.VariableNames = {'fx' 'fy' 'cameraresolution' 'A' 'B'↙
'wx' 'wy'};
            writetable(Parameter,'data.xls','Sheet','parameters');% write fitting↙
parameters, trap frequencies and cameraresolution into excel file
        else
            close all
            S = dir(fullfile(app.DirectoryEditField.Value,'*.tif'));
            n = length(S);
            for i = 1:n
                S(i).time = str2double(S(i).name(1:end-4));
            end
            SS = struct2table(S); % convert the struct array to a table
            sortedSS = sortrows(SS, 'time'); % sort the table by 'DOB'
            S = table2struct(sortedSS); % change it back to struct array if necessary
            t0 = sortedSS.time;%time stamp
            NC = app.numberofpacketsEditField.Value;%number of clusters to find
            zoomcoefficient = app.zoomcoeficientEditField.Value;
            Ni = NC;
            P = zeros(n,2*Ni+1);%save y and x position
            W = zeros(n,2*Ni+1);%save wx wy
            D = zeros(n,2*Ni+1);%save A B
            tr = 512;%total row number
            ts = [];%skipped time
            skpidx = [];%skipped time index
            tidx = [];
            tn = [];
            for k = 1:n
                A = imread(S(k).name);
                imshow(S(k).name,'Parent',app.UIAxes);
                app.UIAxes.Title.String = S(k).name;
                A = im2single(A);
                P(k,1) = str2double(S(k).name(1:end-4));
                W(k,1) = str2double(S(k).name(1:end-4));
                D(k,1) = str2double(S(k).name(1:end-4));
                filter = ones(5)/25;
                Ac = imfilter(A,filter);%replace each pixel with the average value↙
around it
                xmin = 512-2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
xtrapfrequencyHzEditField.Value)-5;
```

```
                ymin = 256-2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
ytrapfrequencyHzEditField.Value)-5;
                width = 2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
xtrapfrequencyHzEditField.Value)*2+10;
                height = 2*(app.cameraresolutionpxmmEditField.Value)*11.8/2/pi/(app.↙
ytrapfrequencyHzEditField.Value)*2+10;
                AC = imcrop(Ac,[xmin,ymin,width,height]);
                M = mean(AC,'all');
                AC = imadjust(AC,[0,M-0.09]);
                if app.manualinputCheckBox.Value% users can choose using default pixel↙
threshold or not
                    ACb = imbinarize(AC,app.pixelthresholdEditField.Value);
                else
                    ACb = imbinarize(AC);
                    app.pixelthresholdEditField.Value = graythresh(AC);
                end
                [row,col] = find(imcomplement(ACb));% extract the row and columns↙
indexes for pixels representing atom clouds
                ROI = [col,row];
                [idx,C]=kmeans(ROI,NC);%find the center of atom clouds using kmeans↙
clustering algorithm
                sp = 0;% calculate the sum of pixel values in the cloud centers.↙
Normally it should be zero
                for i = 1:NC
                    sp = sp + ACb(round(C(i,2)),round(C(i,1)));
                end
                while sp ~= 0 || min(pdist(C)) <= 6
                    if sp ~= 0 % if the sum is not zero, it means the actual number of↙
clouds is larger, so we need to increase the number
                        NC = NC + 1;
                        [idx,C]=kmeans(ROI,NC);
                        sp = 0;
                        for i = 1:NC
                            sp = sp + ACb(round(C(i,2)),round(C(i,1)));
                        end
                    else
                        NC = NC - 1;% if the distance between cluster centers is too↙
small, then the actuall number of clouds is smaller
                        [idx,C]=kmeans(ROI,NC);
                        sp = 0;
                        for i = 1:NC
                            sp = sp + ACb(round(C(i,2)),round(C(i,1)));
                        end
                    end
                end
                if NC < Ni%if there are clouds overlapping, it requires user mannual↙
selection for positions
                    figure(k)
                    imshow(A)
                    title(strcat('select',{' '},num2str(Ni),' positions or right click↙
```

```
to skip'))
                    zoom(zoomcoefficient)
                    skpidx = [skpidx k];
                    ts = [ts t0(k)];
                    hold on
                    tPx = [];
                    tPy = [];
                    for i=1:Ni
                        [x,y,button] = ginput(1);
                        if button == 3
                            break
                        end
                        tPx = [tPx x];
                        tPy = [tPy y];
                        P(k,2*i:2*i+1) = [tr-y x];%save the y and x coordinates
                        title(strcat(num2str(Ni-i),' positions left'))
                        plot(x,y,'ro', 'MarkerSize', 10);
                    end
                    if button == 3
                        P(k,1) = -1;
                        close(k)
                        continue
                    end
                    Ymin = min(tPy);%the minimum y position among cloud centers
                    Ymax = max(tPy);%the maximum y position among cloud centers
                    Xmin = min(tPx);%the minimum x position among cloud centers
                    Xmax = max(tPx);%the maximum x position among cloud centers
                    Ac = imcrop(A,[Xmin-10,Ymin-10,Xmax-Xmin+20,Ymax-Ymin+20]);%crop↙
all atom clouds from the original image
                    [r c] = size(Ac);
                    x0 = zeros(1,1+Ni*3);% a vector saving the fitting parameter↙
include A, B wy and wx for all clouds
                    x0(1) = app.AEditField.Value+1;% the first value is A+1
                    if app.pxtommCheckBox.Value
                        for i = 1:Ni
                            x0(3*i-1) = app.BEditField.Value;
                            x0(3*i) = app.wyEditField.Value*app.↙
cameraresolutionpxmmEditField.Value;
                            x0(3*i+1) = app.wxEditField.Value*app.↙
cameraresolutionpxmmEditField.Value;
                        end
                    else
                        for i = 1:Ni
                        x0(3*i-1) = app.BEditField.Value;
                        x0(3*i) = app.wyEditField.Value;
                        x0(3*i+1) = app.wxEditField.Value;
                        end
                    end
                    fun = @(x)D2GaussFunctionNP(x,Ac,r,c,Ni,P(k,:),Xmin,Ymin,tr);
                    xf = fminsearch(fun,x0);%fit pixel values to gaussian functions
```

```
                    for i = 1:Ni
                        W(k,2*i:2*i+1) = [xf(3*i) xf(3*i+1)];%save the fit result wy↙
and wx
                        D(k,2*i:2*i+1) = [xf(1)-1 xf(3*i-1)];%save the fit result A B
                    end
                    close(k)
                else
                    %   If there are more clusters than we want to find,
                    %   then we only consider the N darkest clouds
                    tidx = [tidx k];
                    tn = [tn t0(k)];
                    while min(pdist(C)) <= 6
                        [idx,C]=kmeans(ROI,NC);
                    end
                    avp = zeros(NC,1);%average pixel values
                    wd = 0.5*(app.wxEditField.Value+app.wyEditField.Value)-1;
                    for i =1:NC%calculate the averaged pixel value for each cluster and↙
then sort them based on the pixel value
                        avp(i) = mean(ACb(round(C(i,2))-wd:round(C(i,2))+wd,round(C(i,↙
1))-wd:round(C(i,1))+wd),'all');
                    end
                    [avp, Idx]=sort(avp);
                    C = C(Idx,:);
                    C = C(1:Ni,:);%keep only N darkest clouds
                    plot(app.UIAxes2,ROI(:,1),ROI(:,2),'go',C(:,1),C(:,2),'r*')
                    pbaspect([1 1 1])
                    app.UIAxes2.Title.String = S(k).name;
                    pause(0.1)
                    N = 8;
                    I = zeros(2*N+1,2*N+1,length(C));%a matrix save cropped image for↙
each cloud from the original image
                    for i = 1:length(C)
                        I(:,:,i) =  imcrop(A,[C(i,1)+xmin-1-N,C(i,2)+ymin-1-N,2*N,↙
2*N]);
                    end
                    if app.pxtommCheckBox.Value%x0 saves the initial fitting parameter
                        x0 = [app.AEditField.Value + 1,app.BEditField.Value,app.↙
wyEditField.Value*app.cameraresolutionpxmmEditField.Value,N+1,app.wxEditField.↙
Value*app.cameraresolutionpxmmEditField.Value,N+1];
                    else
                        x0 = [app.AEditField.Value + 1,app.BEditField.Value,app.↙
wyEditField.Value,N+1,app.wxEditField.Value,N+1];
                    end
                    x = zeros(Ni,6);%x saves the final fitting result
                    for i = 1:Ni
                        fun = @(x)D2GaussFunction1(x,I(:,:,i),N);
                        x(i,:) = fminsearch(fun,x0);%fit pixel value to gaussian↙
function
                        P(k,2*i:2*i+1) = [tr-(C(i,2)+ymin-1-(N+1)+x(i,4)) C(i,1)+xmin-↙
1-(N+1)+x(i,6)];
```

```
                    W(k,2*i:2*i+1) = [x(i,3) x(i,5)];%wy wx
                    D(k,2*i:2*i+1) = [x(i,1)-1 x(i,2)];%A B
                end
            end
            if k == 1% label the atom clouds based on the polar angles and radii
                xc = mean(P(1,3:2:2*Ni+1))-xmin+1;% calculate the center of atom↙
clouds
                yc = tr-mean(P(1,2:2:2*Ni))-ymin+1;
                nROIx = ROI(:,1)-xc;% make the pixel coordinates centralized
                nROIy = -ROI(:,2)+yc;
                PolarAngle = atan2(nROIy,nROIx)*180/pi;%calculate the polar angles
                PolarRad = zeros(length(nROIx),1);% calculate the polar radii
                for i = 1:length(nROIx)
                    PolarRad(i) = pdist([nROIy(i),nROIx(i);0,0]);
                end
                close (1)
                h = histogram(PolarAngle,360);%plot the histogram for polar angles↙
of all black pixel values
                figure(2)
                findpeaks(h.Values,linspace(h.BinLimits(1),h.BinLimits(2),↙
360),'MinPeakDistance',10,'MinPeakHeight',3.5);%MinPeakProminence',4
                [pks,locs] = findpeaks(h.Values,linspace(h.BinLimits(1),h.BinLimits↙
(2),360),'MinPeakDistance',10,'MinPeakHeight',3.5);
                nl = length(locs);%find how many peaks there are in the histogram
                angledist = zeros(nl-1,1);%calculate the distance between the peaks↙
in the histogram
                for i=1:nl-1
                    angledist(i) = locs(i+1)-locs(i);
                end
                [~,Iangle] = min(angledist);% find the smallest distance between↙
peaks
                phi0 = 0.5*(locs(Iangle)+locs(Iangle+1));% the average polar angle↙
between the peaks with the smallest distance
                for i = 1:length(nROIx)% shift the polar angles smaller than phi0↙
by 360 degrees
                    if PolarAngle(i)< phi0
                        PolarAngle(i) = PolarAngle(i)+360;
                    end
                end
                close all
                h = histogram(PolarAngle,360);%plot the new histogram for polar↙
angles of all black pixel values
                figure(2)
                findpeaks(h.Values,linspace(h.BinLimits(1),h.BinLimits(2),↙
360),'MinPeakDistance',10,'MinPeakHeight',3.5);%MinPeakProminence',4
                [pks,locs] = findpeaks(h.Values,linspace(h.BinLimits(1),h.BinLimits↙
(2),360),'MinPeakDistance',10,'MinPeakHeight',3.5);
                nl = length(locs);%find how many peaks there are
                sector = linspace(phi0,phi0+360,nl+1);%split all polar angles into↙
nl+1 sectors
```

```
                    for i = 2:nl
                        sector(i) = 0.5*(locs(i-1)+locs(i));
                    end
                    CPx = mean(P(1,3:2:2*Ni+1));%avearge x position of all packet↙
centers
                    CPy = mean(P(1,2:2:2*Ni));%avearge y position of all packets↙
centers
                    AP = zeros(1,2*Ni+1);%centered packets positions
                    for i = 1:Ni
                        AP(2*i:2*i+1) = [P(1,2*i)-CPy P(1,2*i+1)-CPx];
                    end
                    polarCP = zeros(1,2*Ni+1);%polar angle and radius
                    for i = 1:Ni
                        polarCP(2*i:2*i+1) = [atan2(AP(2*i),AP(2*i+1))*180/pi pdist([AP↙
(2*i),AP(2*i+1);0,0])];
                        if polarCP(2*i) < phi0
                            polarCP(2*i) = polarCP(2*i)+360;
                        end
                    end
                    distPackets = zeros(nl,Ni);%sort the cluster centers based on their↙
polar angles
                    for i = 1:Ni
                        for j = 1:nl
                            if polarCP(2*i) > sector(j) && polarCP(2*i) < sector(j+1)
                                distPackets(j,i) = 1;
                            end
                        end
                    end
                    nPidx = zeros(1,Ni);%save the sorted idx of P based on polar angle↙
and radius
                    prev = 0;
                    for i = 1:nl
                        [~,sidx]=sort(polarCP(2*find(distPackets(i,:))+1));%for those↙
clusters within the same polar angle sector, sort them based on polar radii
                        Find = find(distPackets(i,:));
                        nPidx(prev+1:prev+sum(distPackets(i,:))) = Find(sidx);
                        prev = prev + sum(distPackets(i,:));
                    end
                    TP = zeros(1,2*Ni+1);
                    TD = zeros(1,2*Ni+1);
                    TW = zeros(1,2*Ni+1);
                    for i = 1:Ni
                        TP(2*i:2*i+1) = P(1,2*nPidx(i):2*nPidx(i)+1);
                        TD(2*i:2*i+1) = D(1,2*nPidx(i):2*nPidx(i)+1);
                        TW(2*i:2*i+1) = W(1,2*nPidx(i):2*nPidx(i)+1);
                    end
                    P(1,2:2*Ni+1) = TP(2:2*Ni+1);%y at first than x
                    D(1,2:2*Ni+1) = TD(2:2*Ni+1);%y at first than x
                    W(1,2:2*Ni+1) = TW(2:2*Ni+1);%y at first than x
                    ImageLabel = zeros(Ni,3);%save the label number and positions
```

```
                    for i = 1:Ni
                        ImageLabel(i,1) = i;
                        ImageLabel(i,2) = P(1,2*i+1);
                        ImageLabel(i,3) = tr-P(1,2*i);
                    end
                    RGB = insertText(A,ImageLabel(:,2:3),ImageLabel(:,1),'FontSize',↙
18,'BoxColor','red','BoxOpacity',0.4,'TextColor','green','AnchorPoint','Center');
                    close all
                end
            end
            t0 = t0(find(P(:,1)>=0));
            W = W(find(P(:,1)>=0),:);%save wx wy
            D = D(find(P(:,1)>=0),:);%save A B
            P = P(find(P(:,1)>=0),:);% keep images which have not been skipped
            n = length(P(:,1));
            pc = zeros(n,3);%central position of 8 packets of each image
            for i=1:n
                pc(i,1) = P(i,1);%time
                pc(i,2) = mean(P(i,2:2:2*Ni));%average y position
                pc(i,3) = mean(P(i,3:2:2*Ni+1));%average x position
            end
            plot(app.UIAxes_2,pc(:,3),pc(:,2),'ro')
            app.UIAxes_2.Title.String = 'Y-X(cluster centers)';
            vpc = var(pc(:,2:3))%variance of central positions
            CP = zeros(n,2*Ni+1);%centralized positions, x(column) at first, y(row) at↙
last,
            for i = 1:n
                CP(i,1) = P(i,1);
                CP(i,2:2:2*Ni) = P(i,3:2:2*Ni+1)-pc(i,3);
                CP(i,3:2:2*Ni+1) = P(i,2:2:2*Ni)-pc(i,2);
            end
            for k = 2:3% for the first three images in the dataset, group clusters↙
based on polar angles and radii
                polarCP = zeros(1,2*Ni+1);%polar angle and radius
                for i = 1:Ni
                    polarCP(2*i:2*i+1) = [atan2(CP(k,2*i+1),CP(k,2*i))*180/pi pdist([CP↙
(k,2*i),CP(k,2*i+1);0,0])];
                    if polarCP(2*i) < phi0
                        polarCP(2*i) = polarCP(2*i)+360;
                    end
                end
                distPackets = zeros(nl,Ni);
                for i = 1:Ni
                    for j = 1:nl
                        if polarCP(2*i) > sector(j) && polarCP(2*i) < sector(j+1)
                            distPackets(j,i) = 1;
                        end
                    end
                end
                nPidx = zeros(1,Ni);%save the sorted idx of P based on polar angle and↙
```

```
radius
                prev = 0;
                for i = 1:nl
                    [~,sidx]=sort(polarCP(2*find(distPackets(i,:))+1));
                    Find = find(distPackets(i,:));
                    nPidx(prev+1:prev+sum(distPackets(i,:))) = Find(sidx);
                    prev = prev + sum(distPackets(i,:));
                end
                TP = zeros(1,2*Ni+1);
                TCP = zeros(1,2*Ni+1);
                TD = zeros(1,2*Ni+1);
                TW = zeros(1,2*Ni+1);
                for i = 1:Ni
                    TP(2*i:2*i+1) = P(k,2*nPidx(i):2*nPidx(i)+1);
                    TCP(2*i:2*i+1) = CP(k,2*nPidx(i):2*nPidx(i)+1);
                    TD(2*i:2*i+1) = D(k,2*nPidx(i):2*nPidx(i)+1);
                    TW(2*i:2*i+1) = W(k,2*nPidx(i):2*nPidx(i)+1);
                end
                P(k,2:2*Ni+1) = TP(2:2*Ni+1);%y at first than x
                CP(k,2:2*Ni+1) = TCP(2:2*Ni+1);%x at first than y
                D(k,2:2*Ni+1) = TD(2:2*Ni+1);%y at first than x
                W(k,2:2*Ni+1) = TW(2:2*Ni+1);%y at first than x
            end
            NCP =  CP;
            Nt0 =  t0;
            ND =  D;
            NW = W;
            for i = 4:n% starting from the fourth image, using the latest three images↙
to get a predited position based on linear regression
                TNCP = zeros(1,2*Ni+1);
                TND = zeros(1,2*Ni+1);
                TNW = zeros(1,2*Ni+1);
                for j = 1:Ni
                    prdx = predict(fitlm(Nt0(i-3:i-1),NCP(i-3:i-1,2*j)),Nt0(i));
                    prdy = predict(fitlm(Nt0(i-3:i-1),NCP(i-3:i-1,2*j+1)),Nt0(i));
                    pd = inf;
                    mdx = 0;
                    for k = 1:Ni
                        if pdist([prdx prdy;NCP(i,2*k) NCP(i,2*k+1)]) < pd
                            pd = pdist([prdx prdy;NCP(i,2*k) NCP(i,2*k+1)]);
                            mdx = k;
                        end
                    end
                    TNCP(2*j:2*j+1) = [NCP(i,2*mdx) NCP(i,2*mdx+1)];
                    TND(2*j:2*j+1) = [ND(i,2*mdx) ND(i,2*mdx+1)];
                    TNW(2*j:2*j+1) = [NW(i,2*mdx) NW(i,2*mdx+1)];
                end
                NCP(i,2:2*Ni+1) = TNCP(2:2*Ni+1);
                ND(i,2:2*Ni+1) = TND(2:2*Ni+1);
                NW(i,2:2*Ni+1) = TNW(2:2*Ni+1);
```

```
            end
            CP = NCP;
            W = NW;
            D = ND;
            for i = 1:n
                CP(i,2:2:2*Ni) = CP(i,2:2:2*Ni)+pc(i,3);
                CP(i,3:2:2*Ni+1) = CP(i,3:2:2*Ni+1)+pc(i,2);
            end
            P = CP;
            T = array2table(P);
            Tname = cell(1,2*Ni+1);
            Tname(1) = cellstr('time');
            for i = 1:Ni
                Tname(2*i) = cellstr(strcat('x',num2str(i)));
                Tname(2*i+1) = cellstr(strcat('y',num2str(i)));
            end
            T.Properties.VariableNames = Tname;
            writetable(T,strcat(cd,'\data.xls'),'Sheet','positions');
            for i = 1:Ni
                TB = array2table([W(:,1) W(:,2*i:2*i+1) D(:,2*i:2*i+1)]);
                TB.Properties.VariableNames = {'Time' 'wy' 'wx' 'A' 'B'};
                writetable(TB,strcat(cd,'\data.xls'),'Sheet',num2str(i))
            end
            close all
            imshow(RGB,'Parent',app.UIAxes2)
            app.UIAxes2.Title.String = 'label';
            imwrite(RGB,'label.png');
            Parameter = table(app.xtrapfrequencyHzEditField.Value,app.↙
ytrapfrequencyHzEditField.Value,app.cameraresolutionpxmmEditField.Value,app.AEditField.↙
Value,app.BEditField.Value,app.wxEditField.Value,app.wyEditField.Value);
            Parameter.Properties.VariableNames = {'fx' 'fy' 'cameraresolution' 'A' 'B'↙
'wx' 'wy'};
            writetable(Parameter,'data.xls','Sheet','parameters');
        end
            function F = D2GaussFunction1(x,a,N)
                F =0;
                for l=1:(2*N+1)
                    for q=1:(2*N+1)
                        F = F + (x(1)-x(2)*exp(   -((l-x(4))^2/(x(3)^2) + (q-x(6))^2/(x↙
(5)^2) )    )-a(l,q))^2;
                    end
                end
            end

            function F = D2GaussFunctionNP(x,a,r,c,N,P,Xmin,Ymin,tr)
                F =0;
                for e=1:r
                    for f=1:c
                        est = x(1);
                        for g = 1:N
```

```
                            est = est - x(3*g-1)*exp(-((e-(tr-P(2*g)-(Ymin-10)+1))^2/(x↙
(3*g)^2) + (f-(P(2*g+1)-(Xmin-10)+1))^2/(x(3*g+1)^2)));
                        end
                        F = F + (est-a(e,f))^2;
                    end
                end
            end

        end

        % Button pushed function: StopButton
        function StopButtonPushed(app, event)
            error('stopped')
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 693 775];
            app.UIFigure.Name = 'UI Figure';

            % Create SelectDirectoryButton
            app.SelectDirectoryButton = uibutton(app.UIFigure, 'push');
            app.SelectDirectoryButton.ButtonPushedFcn = createCallbackFcn(app,↙
@SelectDirectoryButtonPushed, true);
            app.SelectDirectoryButton.Position = [454 685 117 27];
            app.SelectDirectoryButton.Text = 'Select Directory';

            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Title')
            xlabel(app.UIAxes, 'X')
            ylabel(app.UIAxes, 'Y')
            app.UIAxes.PlotBoxAspectRatio = [1.35532994923858 1 1];
            app.UIAxes.Position = [12 301 300 270];

            % Create xtrapfrequencyHzEditFieldLabel
            app.xtrapfrequencyHzEditFieldLabel = uilabel(app.UIFigure);
            app.xtrapfrequencyHzEditFieldLabel.HorizontalAlignment = 'right';
            app.xtrapfrequencyHzEditFieldLabel.Position = [18 737 118 22];
            app.xtrapfrequencyHzEditFieldLabel.Text = 'x trap frequency (Hz)';

            % Create xtrapfrequencyHzEditField
            app.xtrapfrequencyHzEditField = uieditfield(app.UIFigure, 'numeric');
```

```
            app.xtrapfrequencyHzEditField.Position = [151 737 100 22];
            app.xtrapfrequencyHzEditField.Value = 9.3;

            % Create pixelthresholdEditFieldLabel
            app.pixelthresholdEditFieldLabel = uilabel(app.UIFigure);
            app.pixelthresholdEditFieldLabel.HorizontalAlignment = 'right';
            app.pixelthresholdEditFieldLabel.Position = [19 633 86 22];
            app.pixelthresholdEditFieldLabel.Text = 'pixel threshold ';

            % Create pixelthresholdEditField
            app.pixelthresholdEditField = uieditfield(app.UIFigure, 'numeric');
            app.pixelthresholdEditField.Position = [119 633 135 22];
            app.pixelthresholdEditField.Value = 0.9;

            % Create DirectoryEditFieldLabel
            app.DirectoryEditFieldLabel = uilabel(app.UIFigure);
            app.DirectoryEditFieldLabel.HorizontalAlignment = 'right';
            app.DirectoryEditFieldLabel.Position = [401 654 54 22];
            app.DirectoryEditFieldLabel.Text = 'Directory';

            % Create DirectoryEditField
            app.DirectoryEditField = uieditfield(app.UIFigure, 'text');
            app.DirectoryEditField.Position = [470 654 100 22];

            % Create RunButton
            app.RunButton = uibutton(app.UIFigure, 'push');
            app.RunButton.ButtonPushedFcn = createCallbackFcn(app, @RunButtonPushed,↙
true);
            app.RunButton.Position = [401 619 100 22];
            app.RunButton.Text = 'Run';

            % Create ytrapfrequencyHzEditFieldLabel
            app.ytrapfrequencyHzEditFieldLabel = uilabel(app.UIFigure);
            app.ytrapfrequencyHzEditFieldLabel.HorizontalAlignment = 'right';
            app.ytrapfrequencyHzEditFieldLabel.Position = [19 698 118 22];
            app.ytrapfrequencyHzEditFieldLabel.Text = 'y trap frequency (Hz)';

            % Create ytrapfrequencyHzEditField
            app.ytrapfrequencyHzEditField = uieditfield(app.UIFigure, 'numeric');
            app.ytrapfrequencyHzEditField.Position = [151 698 100 22];
            app.ytrapfrequencyHzEditField.Value = 9.25;

            % Create UIAxes_2
            app.UIAxes_2 = uiaxes(app.UIFigure);
            title(app.UIAxes_2, 'Title')
            xlabel(app.UIAxes_2, 'X')
            ylabel(app.UIAxes_2, 'Y')
            app.UIAxes_2.PlotBoxAspectRatio = [1.35532994923858 1 1];
            app.UIAxes_2.Position = [360 315 286 243];
```

```
% Create cameraresolutionpxmmEditFieldLabel
app.cameraresolutionpxmmEditFieldLabel = uilabel(app.UIFigure);
app.cameraresolutionpxmmEditFieldLabel.HorizontalAlignment = 'right';
app.cameraresolutionpxmmEditFieldLabel.Position = [19 662 145 22];
app.cameraresolutionpxmmEditFieldLabel.Text = 'camera resolution(px/mm)';

% Create cameraresolutionpxmmEditField
app.cameraresolutionpxmmEditField = uieditfield(app.UIFigure, 'numeric');
app.cameraresolutionpxmmEditField.Position = [179 662 75 20];
app.cameraresolutionpxmmEditField.Value = 503;

% Create UIAxes2
app.UIAxes2 = uiaxes(app.UIFigure);
title(app.UIAxes2, 'Title')
xlabel(app.UIAxes2, 'X')
ylabel(app.UIAxes2, 'Y')
app.UIAxes2.PlotBoxAspectRatio = [1.21134020618557 1 1];
app.UIAxes2.Position = [2 59 294 262];

% Create AEditFieldLabel
app.AEditFieldLabel = uilabel(app.UIFigure);
app.AEditFieldLabel.HorizontalAlignment = 'right';
app.AEditFieldLabel.Position = [280 211 25 22];
app.AEditFieldLabel.Text = 'A';

% Create AEditField
app.AEditField = uieditfield(app.UIFigure, 'numeric');
app.AEditField.Position = [320 211 100 22];

% Create BEditFieldLabel
app.BEditFieldLabel = uilabel(app.UIFigure);
app.BEditFieldLabel.HorizontalAlignment = 'right';
app.BEditFieldLabel.Position = [482 211 25 22];
app.BEditFieldLabel.Text = 'B';

% Create BEditField
app.BEditField = uieditfield(app.UIFigure, 'numeric');
app.BEditField.Position = [522 211 100 22];
app.BEditField.Value = 1;

% Create wxEditFieldLabel
app.wxEditFieldLabel = uilabel(app.UIFigure);
app.wxEditFieldLabel.HorizontalAlignment = 'right';
app.wxEditFieldLabel.Position = [280 179 25 22];
app.wxEditFieldLabel.Text = 'wx';

% Create wxEditField
app.wxEditField = uieditfield(app.UIFigure, 'numeric');
app.wxEditField.Position = [321 179 99 22];
app.wxEditField.Value = 4;
```

```
            % Create wyEditFieldLabel
            app.wyEditFieldLabel = uilabel(app.UIFigure);
            app.wyEditFieldLabel.HorizontalAlignment = 'right';
            app.wyEditFieldLabel.Position = [482 179 25 22];
            app.wyEditFieldLabel.Text = 'wy';

            % Create wyEditField
            app.wyEditField = uieditfield(app.UIFigure, 'numeric');
            app.wyEditField.Position = [522 179 100 22];
            app.wyEditField.Value = 4;

            % Create numberofpacketsEditFieldLabel
            app.numberofpacketsEditFieldLabel = uilabel(app.UIFigure);
            app.numberofpacketsEditFieldLabel.HorizontalAlignment = 'right';
            app.numberofpacketsEditFieldLabel.Position = [352 728 104 22];
            app.numberofpacketsEditFieldLabel.Text = 'number of packets';

            % Create numberofpacketsEditField
            app.numberofpacketsEditField = uieditfield(app.UIFigure, 'numeric');
            app.numberofpacketsEditField.Position = [471 728 100 22];
            app.numberofpacketsEditField.Value = 4;

            % Create zoomcoeficientEditFieldLabel
            app.zoomcoeficientEditFieldLabel = uilabel(app.UIFigure);
            app.zoomcoeficientEditFieldLabel.HorizontalAlignment = 'right';
            app.zoomcoeficientEditFieldLabel.Position = [19 598 89 22];
            app.zoomcoeficientEditFieldLabel.Text = 'zoom coeficient';

            % Create zoomcoeficientEditField
            app.zoomcoeficientEditField = uieditfield(app.UIFigure, 'numeric');
            app.zoomcoeficientEditField.Position = [123 598 100 22];
            app.zoomcoeficientEditField.Value = 2;

            % Create manualinputCheckBox
            app.manualinputCheckBox = uicheckbox(app.UIFigure);
            app.manualinputCheckBox.Text = 'manual input';
            app.manualinputCheckBox.Position = [261 633 91 22];

            % Create StopButton
            app.StopButton = uibutton(app.UIFigure, 'push');
            app.StopButton.ButtonPushedFcn = createCallbackFcn(app, @StopButtonPushed,↙
true);
            app.StopButton.Position = [522 619 100 22];
            app.StopButton.Text = 'Stop';

            % Create pxtommCheckBox
            app.pxtommCheckBox = uicheckbox(app.UIFigure);
            app.pxtommCheckBox.Text = 'px to mm';
            app.pxtommCheckBox.Position = [330 145 72 22];
```

```
            % Create NopositionChangeCheckBox
            app.NopositionChangeCheckBox = uicheckbox(app.UIFigure);
            app.NopositionChangeCheckBox.Text = 'No position Change';
            app.NopositionChangeCheckBox.Position = [409 577 128 22];

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = BEC_tracking_7_exported3

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

Bibliography

[1] Pippa Storey and Claude Cohen-Tannoudji. The Feynman path integral approach to atomic interferometry: A tutorial. *J. Phys. II*, 4(11):1999–2027, 1994.

[2] Hervé C Lefère. *The Fiber-Optic Gyroscope*. Artech House, Norwood, MA, 2nd edition, 2014.

[3] M. Y. Jeon, H. J. Jeong, and B. Y. Kim. Mode-locked fiber laser gyroscope. *Opt. Lett.*, 18(4):320–322, Feb ts , url = http://ol.osa.org/abstract.cfm?URI=ol-18-4-320, doi = 10.1364/OL.18.000320,.

[4] V. S. Bagnato, K. M. F. Magalhães, J. A. Seman, E. A. L. Henn, and E. R. F. Ramos. Introduction to the basic-concepts of bose-einstein condensation. *AIP Conference Proceedings*, 994(1):79–97, 2008.

[5] R. K. Pathria. *Statistical Mechanics*. Elsevier, Burlington, MA, 2nd edition, 1996.

[6] C. J. Pethick and H. Smith. *Bose–Einstein Condensation in Dilute Gases*. Cambridge University Press, 2 edition, 2008.

[7] M. R. Andrews, C. G. Townsend, H.-J. Miesner, D. S. Durfee, D. M. Kurn, and W. Ketterle. Observation of interference between two bose condensates. *Science*, 275(5300):637–641, 1997.

[8] N. Hinkley, J. A. Sherman, N. B. Phillips, M. Schioppo, N. D. Lemke, K. Beloy, M. Pizzocaro, C. W. Oates, and A. D. Ludlow. An atomic clock with 10–18 instability. *Science*, 341(6151):1215–1218, 2013.

[9] K. J. Hughes, J. H. T. Burke, and C. A. Sackett. Suspension of atoms using optical pulses, and application to gravimetry. *Phys. Rev. Lett.*, 102:150403, Apr 2009.

[10] Mark Kasevich and Steven Chu. Atomic interferometry using stimulated raman transitions. *Phys. Rev. Lett.*, 67:181–184, Jul 1991.

[11] Holger Müller, Sheng-wey Chiow, Quan Long, Sven Herrmann, and Steven Chu. Atom interferometry with up to 24-photon-momentum-transfer beam splitters. *Phys. Rev. Lett.*, 100:180405, May 2008.

[12] Robert Horne. *A Cylindrically Symmetric, Magnetic Trap for Bose-Einstein Condensate Atom Interferometry Applications*. PhD thesis, University of Virginia, 2015.

[13] T. L. Gustavson, P. Bouyer, and M. A. Kasevich. Precision rotation measurements with an atom interferometer gyroscope. *Phys. Rev. Lett.*, 78:2046–2049, Mar 1997.

[14] Lu Qi, Zhaohui Hu, Tristan Valenzuela, Yuchi Zhang, Yueyang Zhai, Wei Quan, Nick Waltham, and Jiancheng Fang. Magnetically guided cesium interferometer for inertial sensing. *Applied Physics Letters*, 110(15):153502, Apr 2017.

[15] Peter J. Martin, Bruce G. Oldaker, Andrew H. Miklich, and David E. Pritchard. Bragg scattering of atoms from a standing light wave. *Phys. Rev. Lett.*, 60:515–518, Feb 1988.

[16] ER Moan, RA Horne, T Arpornthip, Z Luo, AJ Fallon, SJ Berl, and CA Sackett. Quantum rotation sensing with dual sagnac interferometers in an atom-optical waveguide. *Physical Review Letters*, 124(12):120403, 2020.

[17] Tanwa Arpornthip. *Characterizing the potential energy of an atom trap through tomographic fluorescence imaging*. PhD thesis, University of Virginia, 2016.

[18] Harold J. Metcalf and Peter van der Straten. *Laser Cooling and Trapping*. Springer, New York, 1 edition, 1999.

[19] Pierre Meystre. *Atom Optics*. Springer, New York, 2001.

[20] C. J. Foot. *Atomic Physics*. New York: Oxford University Press, 2010.

[21] C. Adams and E. Riis. Laser cooling and trapping of neutral atoms. *Progress in Quantum Electronics*, 21(1):1–79, 1997.

[22] Henry I. Smith. A review of submicron lithography. *Superlattices and Microstructures*, 2(2):129 – 142, 1986.

[23] R. A. Horne and C. A. Sackett. A cylindrically symmetric magnetic trap for compact bose-einstein condensate atom interferometer gyroscopes. *Review of Scientific Instruments*, 88(1):013102, 2017.

[24] József Fortágh and Claus Zimmermann. Magnetic microtraps for ultracold atoms. *Rev. Mod. Phys.*, 79:235–289, Feb 2007.

[25] Daniel M. Farkas, Kai M. Hudek, Evan A. Salim, Stephen R. Segal, Matthew B. Squires, and Dana Z. Anderson. A compact, transportable, microchip-based system for high repetition rate production of bose–einstein condensates. *Applied Physics Letters*, 96(9):093102, 2010.

[26] A. Günther, H. Bender, A. Stibor, J. Fortágh, and C. Zimmermann. Observing quantum gases in real time: Single-atom detection on a chip. *Phys. Rev. A*, 80:011604, Jul 2009.

[27] Jörg Schmiedmayer, Ron Folman, and Tommaso Calarco. Quantum information processing with neutral atoms on an atom chip. *Journal of Modern Optics*, 49(8):1375–1388, 2002.

[28] Weiping Zhang, Ewan M. Wright, Han Pu, and Pierre Meystre. Fundamental limit for integrated atom optics with bose-einstein condensates. *Phys. Rev. A*, 68:023605, Aug 2003.

[29] R. Folman, P. Kruger, J. Schmiedmayer, J. Denschlag, and C. Henkel. Microscopic atom optics: from wires to an atom chip, 2008.

[30] S. Abend, M. Gebbe, M. Gersemann, H. Ahlers, H. Müntinga, E. Giese, N. Gaaloul, C. Schubert, C. Lämmerzahl, W. Ertmer, W. P. Schleich, and E. M. Rasel. Atom-chip fountain gravimeter. *Phys. Rev. Lett.*, 117:203003, Nov 2016.

[31] G.-B. Jo, J.-H. Choi, C. A. Christensen, Y.-R. Lee, T. A. Pasquini, W. Ketterle, and D. E. Pritchard. Matter-wave interferometry with phase fluctuating bose-einstein condensates. *Phys. Rev. Lett.*, 99:240406, Dec 2007.

[32] E. Moan, Z. Luo, and C. A. Sackett. A large-area Sagnac interferometer using atoms in a time-orbiting potential. In , volume 10934 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 109341X, March 2019.

[33] Cass Sackett. personal communication.

[34] Franco Dalfovo, Stefano Giorgini, Lev P. Pitaevskii, and Sandro Stringari. Theory of bose-einstein condensation in trapped gases. *Rev. Mod. Phys.*, 71:463–512, Apr 1999.

[35] Jonathan Mitschele. Beer-lambert law. *Journal of Chemical Education*, 73(11):A260, 1996.

[36] Tadas Pyragius. Developing and building an absorption imaging system for ultracold atoms, 2012.

[37] Rafael C. Gonzalez and Richard E. Woods. *Digital image processing*. Prentice Hall; 2nd edition, 2002.

[38] I. Pitas and A. Venetsanopoulos. Nonlinear mean filters in image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(3):573–584, 1986.

[39] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[40] Robert A. Beezer, T. Hastie, R. Tibshirani, and J. Friedman Springer. The elements of statistical learning: Data mining, inference and prediction. by, 2002.

[41] Edward Moan. *Rotation Sensing Using Atom Interferometry in a Magnetic Trap*. PhD thesis, University of Virginia, 2020.

[42] L. D. Landau and E. M. Lifshitz. *Mechanics*. New York: Pergamon Press, 1982.