UNIVERSIDADE FEDERAL DE PERNAMBUCO
DEPARTAMENTO DE FÍSICA – CCEN
PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA

PATRICK ALYSON GOUVEIA TORRES

Analysis of the transverse spatial image of beams transmitted and generated through a non-linear process in rubidium vapor

Recife
2023

PATRICK ALYSON GOUVEIA TORRES

Analysis of the transverse spatial image of beams transmitted and generated through a non-linear process in rubidium vapor

Dissertação apresentada ao Programa de Pós-Graduação em Física da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Física.

Área de Concentração: Óptica

Orientador (a): Profa. Sandra Sampaio Vianna

Recife
2023

Catalogação na fonte
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

T693a Torres, Patrick Alyson Gouveia
Analysis of the transverse spatial image of beams transmitted and generated through a non-linear process in rubidium vapor / Patrick Alyson Gouveia Torres. – 2023.
75 f.: il., fig.

Orientadora: Sandra Sampaio Vianna.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CCEN, Física, Recife, 2023.
Inclui referências e apêndice.

1. Óptica. 2. Mistura de quatro ondas. 3. Correlação espacial. 4. Imagem. 5. Vapor de rubídio I. Vianna, Sandra Sampaio (orientadora). II. Título.

535.2 CDD (23. ed.) UFPE- CCEN 2023 - 25

PATRICK ALYSON GOUVEIA TORRES

ANALYSIS OF THE TRANSVERSE SPATIAL IMAGE OF BEAMS TRANSMITTED AND GENERATED THROUGH A NON-LINEAR PROCESS IN RUBIDIUM VAPOR

Dissertação apresentada ao Programa de Pós-Graduação em Física da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Física.

Aprovada em: 17/01/2023.

BANCA EXAMINADORA

_____

Profa. Sandra Sampaio Vianna
Orientadora
Universidade Federal de Pernambuco

_____

Prof. Anderson Monteiro Amaral
Examinador Interno
Universidade Federal de Pernambuco

_____

Prof. Sebastião José Nascimento de Pádua
Examinador Externo
Universidade Federal de Minas Gerais

ACKNOWLEDGEMENTS

Agradeço a professora Sandra Vianna pela orientação. Foi muito prazeroso trabalhar com uma pessoa tão dedicada e entusiasmada com a física.

Aos colegas de laboratório, em especial Alexandre e Mateus, por toda a ajuda que me deram durante o mestrado e mesmo antes, na iniciação científica, durante minha graduação.

Por fim, agradeço a CAPES e ao CNPq pelo apoio financeiro.

ABSTRACT

In this work we describe an experimental investigation on the spatial intensity distribution of the two transmitted beams and the two beams generated in a four-wave mixing (FWM) process, using rubidium vapor as a nonlinear medium. The two FWM signals are detected in the $2\vec{k}_F - \vec{k}_P$ and $2\vec{k}_P - \vec{k}_F$ directions, where $\vec{k}_F$ and $\vec{k}_P$ are the incident beam directions. A diode laser, responsible by the two fields that driven the FWM process, is tuned to the $5S_{1/2}(F = 3) \longrightarrow 5P_{3/2}$ of the ^{85}Rb. The images of the four beams are obtained using a CMOS camera for a fixed atomic density and a fixed frequency of the input beams. From each image we can determine the spatial distribution of the intensity of the respective beam. We analyzed the spatial intensity distribution of the signals, as well as the spatial intensity fluctuation. The spatial intensity distribution of the transmitted beams shows a decrease of the beam waist for intensities close to the saturation intensity, while for high intensities the spatial profile is almost constant. This behavior indicates an autofocusing effect, related to the variation of the refractive index of the medium. The spatial distribution of the intensity fluctuation is obtained considering an average surface for each image, and is used to calculate the autocorrelation of the transmitted beams and the two generated FWM beams. Working with the intensity fluctuation, we can define a transverse spatial correlation length, which is used to compare the behavior of the four beams with that of a free beam. In this sense, the autocorrelations of intensity fluctuations of the transmitted beams show a decrease in the correlation length in the same region of intensities where we observe the self-focusing effect, and as we increase the intensity, the correlation length approaches the value obtained for the free beam. On the other hand, the correlation length for the autocorrelations of the signals generated by FWM increases as the intensity of the incident beams increases.

Keywords: four-wave mixing; spatial correlation; rubidium vapor; spatial image; non-linear process.

RESUMO

Neste trabalho descrevemos um estudo experimental sobre a distribuição de intensidade espacial dos dois feixes transmitidos e de dois feixes gerados em um processo de mistura de quatro ondas, tendo como meio não linear vapor de rubídio. Os dois sinais gerados nesse processo não linear são detectados nas direções $2\vec{k}_F - \vec{k}_P$ e $2\vec{k}_P - \vec{k}_F$, onde $\vec{k}_P$ e $\vec{k}_F$ são as direções dos feixes incidentes. Um laser de diodo, responsável pelos dois feixes utilizados no processo de mistura de quatro ondas, é sintonizado na transição $5S_{1/2}(F = 3) \longrightarrow 5P_{3/2}$ do ^{85}Rb. As imagens dos quatro feixes são obtidas utilizando uma câmera CMOS e mantendo fixa a densidade atômica e a frequência dos feixes de entrada. A distribuição espacial da intensidade de cada um dos feixes é determinada a partir de suas respectivas imagens. Foram analisadas tanto a distribuição espacial de intensidade dos feixes como a distribuição espacial das flutuações de intensidade. A distribuição espacial de intensidade dos feixes transmitidos apresenta uma diminuição da cintura do feixe para intensidades próximas à intensidade de saturação, enquanto que para altas intensidades o perfil espacial do feixe é aproximadamente constante. Esse comportamento indica um efeito de auto-focalização que está associado à variação do índice de refração do meio. A distribuição espacial das flutuações de intensidade é obtida a partir da subtração de uma superfície média de cada imagem, e é utilizada para calcular a autocorrelação das flutuações dos feixes transmitidos e dos feixes gerados. Na análise das flutuações de intensidade, nós definimos um comprimento de correlação espacial transversal, o qual é utilizado para comparar o comportamento dos quatro feixes com o de um feixe livre. Para os feixes transmitidos, observamos uma diminuição no comprimento de correlação na mesma região de intensidades onde verificamos o efeito de auto-focalização, sendo que, com o aumento da intensidade, o comprimento de correlação se aproxima do valor obtido para o feixe livre. No entanto, o comprimento de correlação dos sinais gerados no processo de mistura de quatro ondas aumenta conforme a intensidade dos feixes incidentes aumenta.

Palavras-chaves: mistura de quatro ondas; correlação espacial; imagem; vapor de rubídio; processo não linear.

LIST OF FIGURES

CONTENTS

1 INTRODUCTION

Nonlinear optics is the study of processes that occur as a consequence of the modification of the optical properties of a material system by the presence of light (1). The term 'nonlinear' is used in the sense that the response of the medium to the applied optical field depends in a nonlinear manner to the strength of the applied optical field (1). The discovery of the second harmonic generation (2) is often taken as the beginning of the field of nonlinear optics and since this discovery it has become a very active field. There is a great variety of nonlinear optical processes, but in this work are we only interested in one of them, the four-wave mixing (FWM). In a four-wave mixing (FWM) process a fourth field is generated as the result of the coherent combination of three electromagnetic fields interacting with a nonlinear sample, this process has been used extensively to investigate a variety of optical phenomena in atomic systems. In our laboratory some studies on the FWM process in atomic systems have already been developed (3) (4) (5), but always investigating the nonlinear process in the frequency domain. A first theoretical study of the spatial profile of these generated signals is described in the Master's thesis of Motta (6) and in (7), showing the influence of the ressonances of the medium. In the present work, we describe an experimental investigation of the images of these beams.

Today, almost all areas of science have been impacted by digital image processing. Visual observation plays an important role in the scientific process, in the beginning the only way to document the results of an experiment was by verbal description and manual drawings, but the invention of photography in the 19th century by Joseph Nicéphore Niépce (8) made possible to document these results in an objective way. One area of physics that has been most impacted by photography is astronomy. Photography has made possible to discover, record and classify celestial objects, celestial bodies and celestial events, astronomers are able to measure positions, magnitude, temperature and composition of these objects and events.

In this context, we describe an experimental investigation on the spatial intensity distribution of the two transmitted beams and the two beams generated in a four-wave mixing (FWM) process, using rubidium vapor as nonlinear medium. The spatial distribution of intensity of the beams is determined from the images of these beams.

In chapter 2, we present an introduction to digital image processing. We will present some basic concepts, definitions, operations and techniques of digital image processing. We will also present a more in-depth description of some of the algorithms that were implemented in this study.

In chapter 3, we describe our experimental setup and the procedures used to obtain our images of the beams for a later analysis.

It is in chapter 4 that we present our experimental results and the analysis carried out. This chapter is divided in two sections, one dedicated to obtain information from the transmitted beams and the other specific for the signals generated in the FWM process. Finally, we present

the conclusions and perspectives of this work.

2 FUNDAMENTALS OF IMAGE PROCESSING

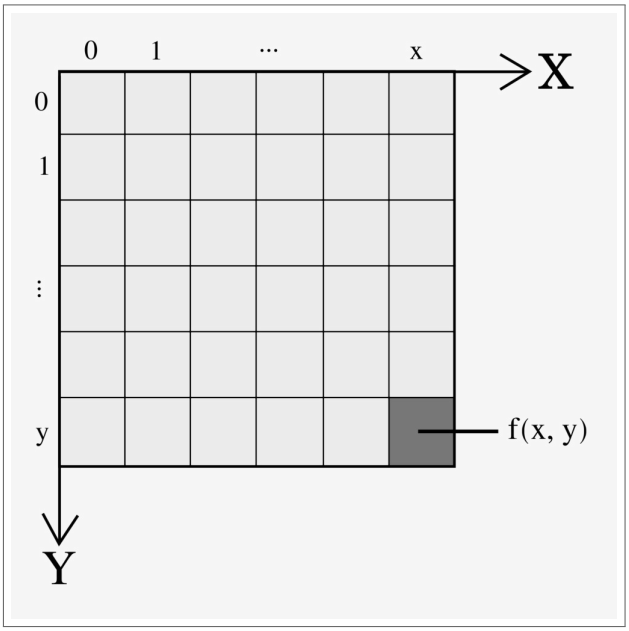
In this chapter we present an introduction to digital image processing. We will present some basic concepts, definitions, operations and techniques of digital image processing. We will also present a more in-depth description of some of the algorithms that were implemented in this study. This chapter was strongly inspired by reference (9).

2.1 BASIC PROPERTIES AND TYPES OF DIGITAL IMAGES

A digital image is defined as a two-dimensional function, $f(x, y)$, where x and y are finite and discrete spatial coordinates representing a picture element (*pixel*). The value of f at any given pair of coordinates (x, y) is a finite and discrete quantity called the *intensity* or *gray level* of the image at that point. A digital image can also be seen as a two-dimensional array of values representing light intensity, where each element within the array is considered a pixel (9) (10) (11).

By convention, the pixel with the coordinates $(0, 0)$ is located at the top, left corner of the image.

Figure 1 – Coventional coordinate system



Source: The author (2022)

A digital image has three basic properties.

Image resolution The resolution of an image is determined by its dimensions (height and width), the number of rows and columns in pixels. For example, an image with dimensions

1280 x 1024 (width x height) pixels means that the total number of pixels in the image is 1310720.

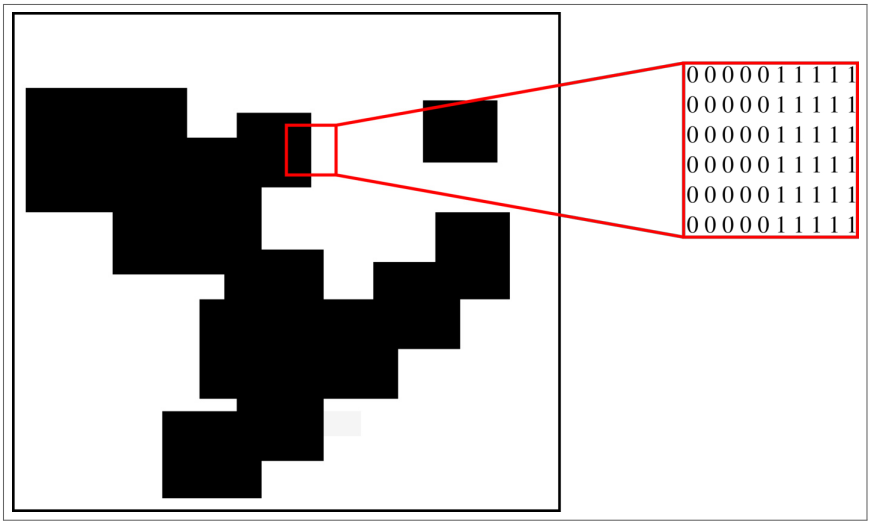
Image definition The definition of an image is indicated by the number of tones it can have (10). The pixels in an image can take on different values, and the number of bits used to encode them is called *bit depth*. An image with bit depth of *n* has a definition of 2^n, meaning that its pixels can assume 2^n different values. For example, an image with bit depth of 1 bit has pixels with two possible values (2^1): 0 or 1. An image with bit depth of 8 bits has pixels with 256 possible values ranging from 0 to 255. An image with bit depth of 16 bits has pixels with 65536 possible values ranging from 0 to 65535.

Number of channels The number of channels corresponds to the number of arrays that compose an image. A grayscale image has only one channel, while an RGB image has three channels: one for red component, one for green component and one for blue component.

There are four types of digital images (11):

Binary images For binary images each pixel can only assume one of two possible values 0 or 1, usually corresponding to black or white. An example is shown in figure 2.

Figure 2 – Binary image

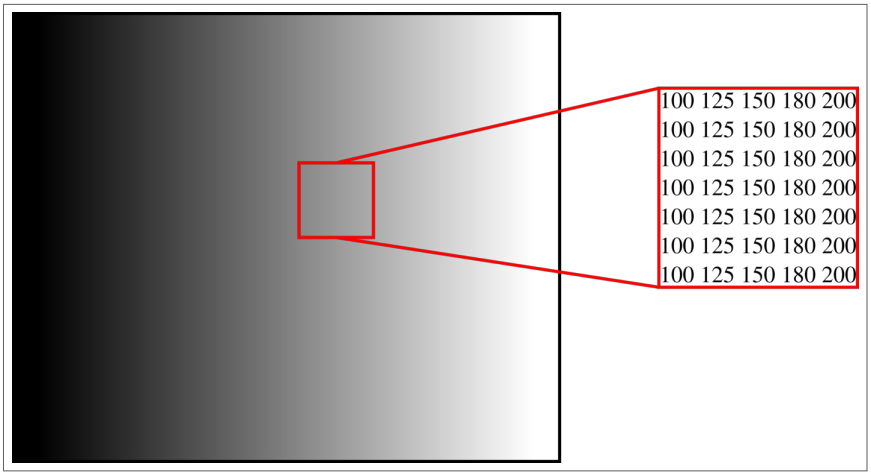


Source: The author (2022)

Grayscale images A grayscale image, sometimes referred to as an *intensity* image, has only one channel and usually each pixel has a bit depth equal to 8, but grayscale images with larger bit depth are also widely available. An example is shown in figure 3.

RGB images A RGB image, sometimes referred to as *truecolor* image, has three channels that define the red, green and blue color components for each individual pixel. RGB

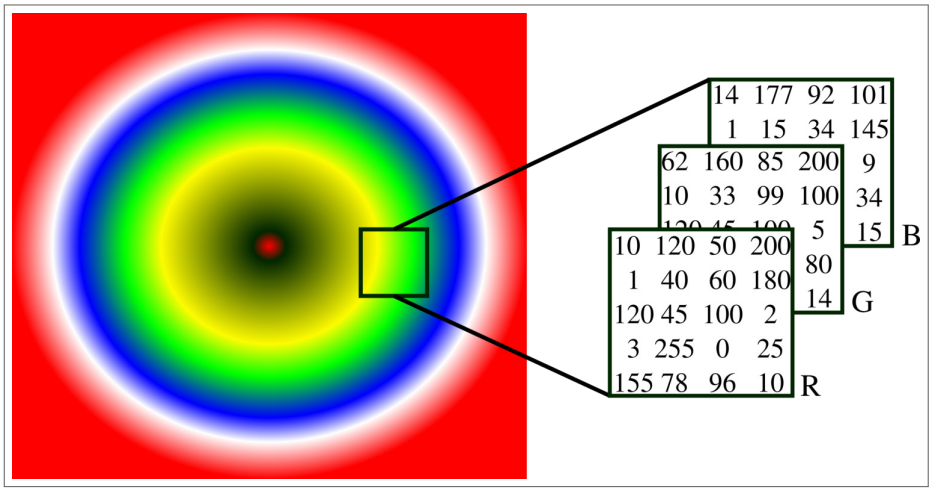
Figure 3 – Grayscale image



Source: The author (2022)

images are stored as 24-bit images, where each channel has a bit depth of 8 bits. An example is shown in figure 4.

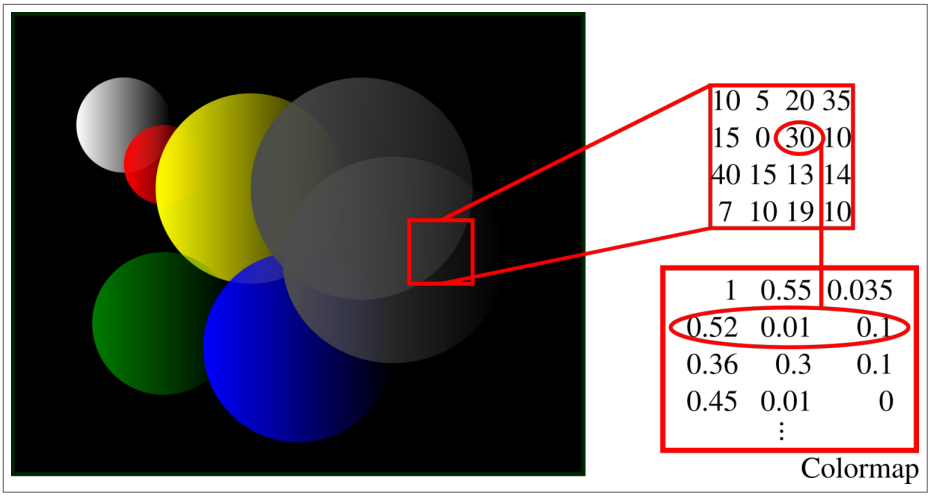
Figure 4 – RGB image



Source: The author (2022)

Indexed images An indexed image is a digital image that uses a color map, also known as a palette, to assign a specific color to each pixel in the image. The color information for each pixel is stored in the color map, rather than in the image data itself, which results in a smaller file size and faster loading times. Indexed images typically use a limited color palette, typically 256 colors, which can cause a lower image quality and less color variety than non-indexed images. An example is shown in 5.

Figure 5 – Indexed image



Source: The author (2022)

2.2 IMAGE FILE FORMATS AND BASIC ARITHMETIC OPERATIONS

Image file formats are standardized means of organizing and storing digital images (10). The formats bellow are some of the common image formats available:

- Portable Network Graphics (PNG)
- Tagged Image File Format (TIFF)
- Bitmap (BMP)
- Joint Photographic Experts Group (JPEG/JPG)
- Graphics Interchange Format (GIF)
- National Instruments Internal Image File Format (AIPD)

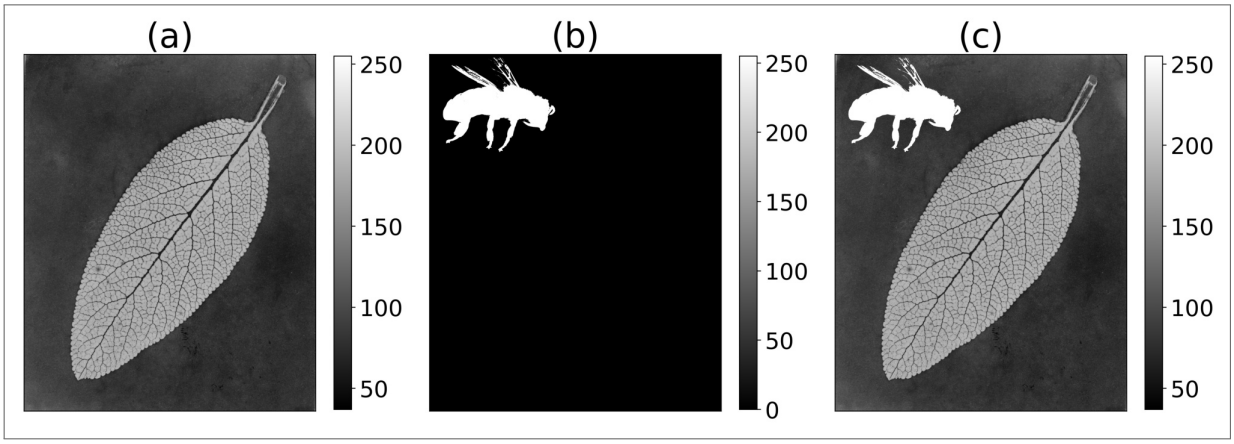
Standard 8-bit grayscale formats are PNG, JPEG, BMP, TIFF and AIDP. Standard 16-bit grayscale format are PNG and AIPD. Standard color file formats for RGB images are PNG, JPEG, BMP, TIFF and AIDP (10).

Images are composed of two-dimensional arrays of numbers. We can define simple mathematical operations on the elements of two arrays.

Addition The addition of two images, I_1 and I_2, of the same size is simply the element wise sum of the two images:

$$I_r(x,y) = I_1(x,y) + I_2(x,y), \qquad \forall (x,y). \tag{2.1}$$

Figure 6 – Adding images. (a) I_1. (b) I_2. (c) $I_1 + I_2$.



Source: The author (2022)

Subtraction The subtraction of two images is defined as:

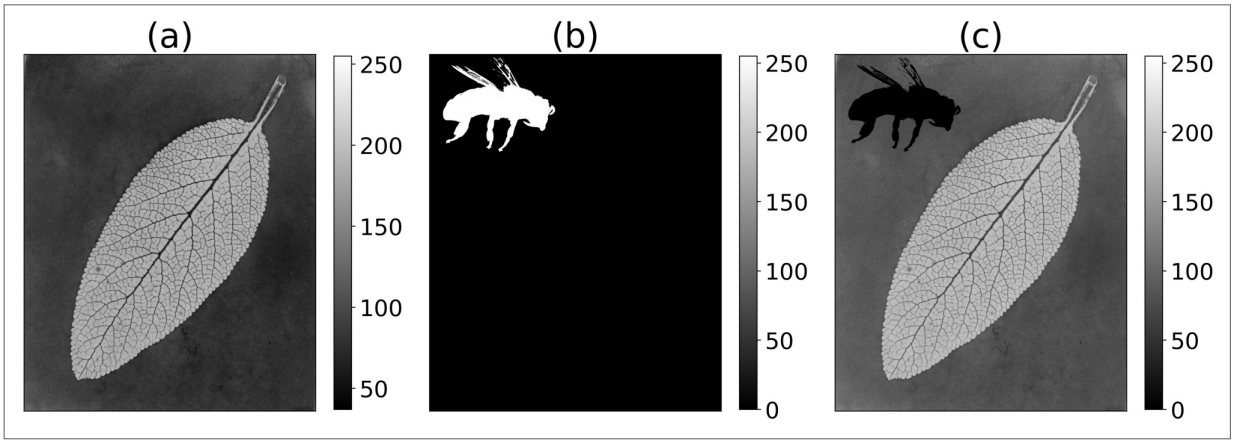
$$I_r(x,y) = I_1(x,y) - I_2(x,y), \qquad \forall(x,y) \tag{2.2}$$

or,

$$I_r(x,y) = |I_1(x,y) - I_2(x,y)|, \qquad \forall(x,y). \tag{2.3}$$

Both of these definitions are possible, here in this work we have used the second one. A subtraction of two images is shown in figure 7.

Figure 7 – Subtracting images. (a) I_1. (b) I_2. (c) $I_1 - I_2$.



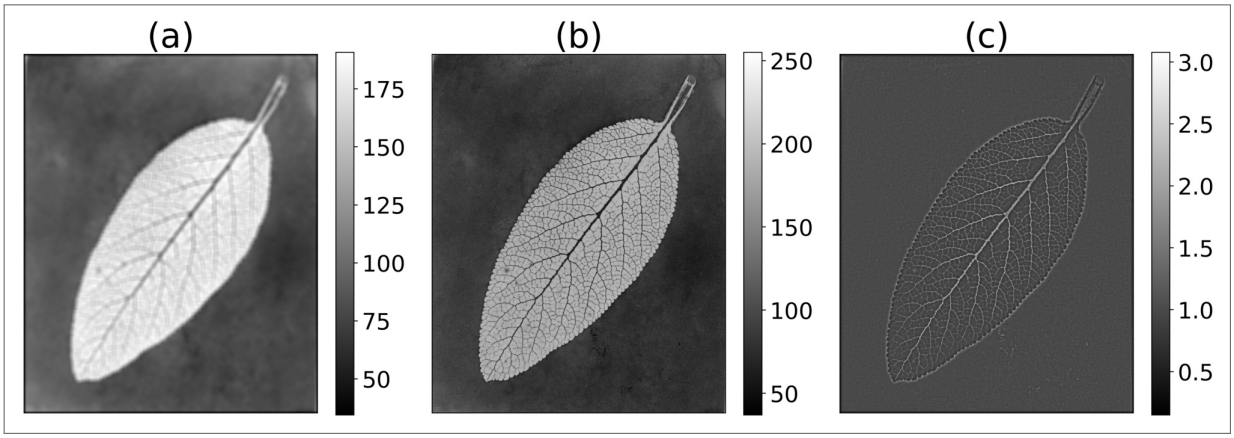
Source: The author (2022)

Division The division of two images is defined as:

$$I_r(x,y) = \frac{I_1(x,y)}{I_2(x,y)}, \qquad \forall(x,y), \tag{2.4}$$

where $I_2(x,y) \neq 0, \quad \forall(x,y)$. An example is shown in figure 8. In this example, we divided the blurred image by the original image, this process is a type of edge detection algorithm.

Figure 8 – Dividing images. (a) I_1. (b) I_2. (c) I_1/I_2.

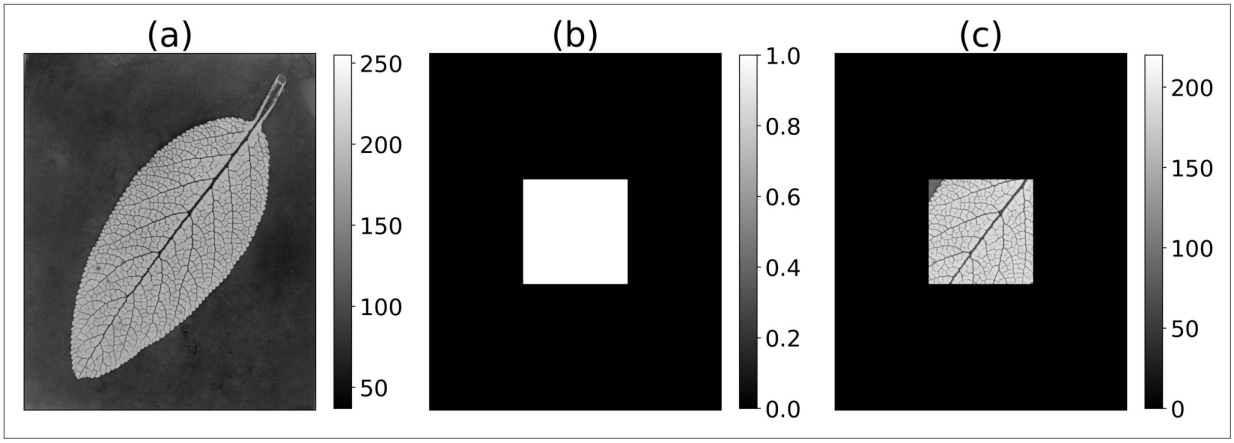


Source: The author (2022)

Multiplication The multiplication of two images (see in figure 9) is defined as:

$$I_r(x, y) = I_1(x, y) * I_2(x, y), \qquad \forall (x, y). \tag{2.5}$$

Figure 9 – Multiplying images. (a) I_1. (b) I_2. (c) $I_1 * I_2$.



Source: The author (2022)

In addition to the basic arithmetic operations, there are many more complex ones that can be performed on images, such as: resizing, rotating, cropping, splitting, merging, changing colors, etc. Some of these operations have great complexity of implementation, but fortunately, they are already implemented in many libraries and with different programming languages.

2.3 IMAGE ENHANCEMENT AND IMAGE RESTORATION

Image enhancement and *Image restoration* are the processes of manipulating an image so that the result is more appropriate than the original image for a specific task (9). These

processes of manipulation can be done in two broad approaches: *spatial domain methods* and *frequency domain methods*. Via these two approaches we can improve images in many ways, such as: sharpening the edges of objects, reducing random and periodic noise, correcting for unequal illumination, etc (12). Here we will focus on frequency domain methods, but first let us define *Fourier transform*, *convolution* and *correlation* for images.

2.3.1 Fourier transform

The Fourier transform, $F(u,v)$, of a two variables continuous function, $f(x,y)$, is defined by the equation

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}\,dx\,dy, \tag{2.6}$$

where $j = \sqrt{-1}$. We also define the *inverse* Fourier transform by

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{j2\pi(ux+vy)}\,du\,dv. \tag{2.7}$$

In the discrete case, the *discrete* Fourier transform(DFT) of an image, $f(x,y)$, of size *M x N* is given by the equation

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}, \tag{2.8}$$

for $u = 0,1,2,...,M-1$ and $v = 0,1,2,...,N-1$. Similarly, we define the inverse discrete Fourier transform by

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)}, \tag{2.9}$$

for $x = 0,1,2,...,M-1$ and $y = 0,1,2,...,N-1$. Equations 2.8 and 2.9 constitute the *two dimensional*, *discrete Fourier transform pair*. The variables x and y are known as the *spatial* or *image* variables, and u and v are known as the *transform* or *frequency* variables.

It is convenient sometimes to express $F(u,v)$ in polar coordinates

$$F(u,v) = |F(u,v)|e^{-j\phi(u,v)}, \tag{2.10}$$

where

$$|F(u,v)| = \sqrt{\Re^2[F(u,v)] + \Im^2[F(u,v)]}, \tag{2.11}$$

and

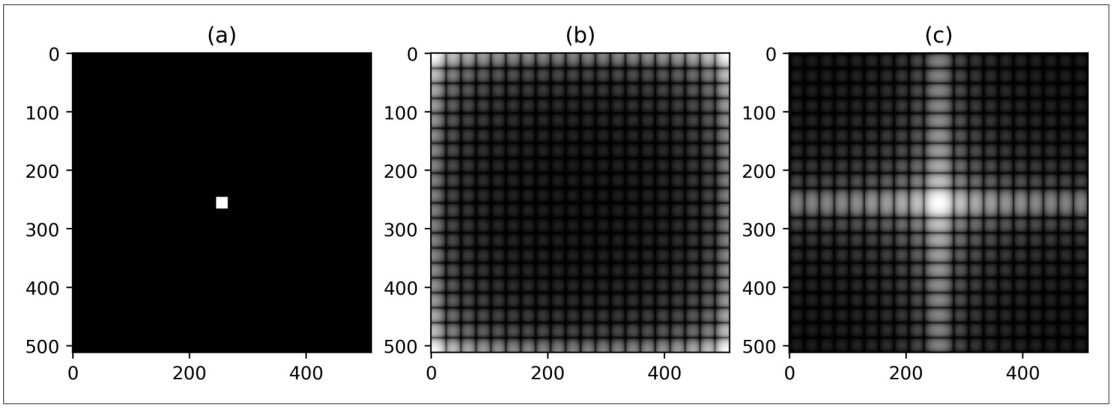
$$\phi(u,v) = \tan^{-1}\left[\frac{\Im[F(u,v)]}{\Re[F(u,v)]}\right] \tag{2.12}$$

are the *magnitude* or *spectrum* and *phase angle* or *phase spectrum* respectively. We also define another quantity called *power spectrum* as

$$P(u,v) = |F(u,v)|^2 = \Re^2[F(u,v)] + \Im^2[F(u,v)]. \tag{2.13}$$

Before computing the Fourier transform of an image it is common to multiply the image by the function $(-1)^{x+y}$, this shifts the zero-frequency components to the center of the spectrum, i.e., it centers the spectrum. The figure 10 shows a comparison between the Fourier transform of an image and the Fourier transform of the same image multiplied by $(-1)^{x+y}$.

Figure 10 – (a) Original image. (b) Logarithm of the power spectrum. (c) Logarithm of the centered power spectrum



Source: The author (2022)

Let us investigate some properties of the Fourier transform. Looking at the value of the Fourier transform at $(u, v) = (0, 0)$,

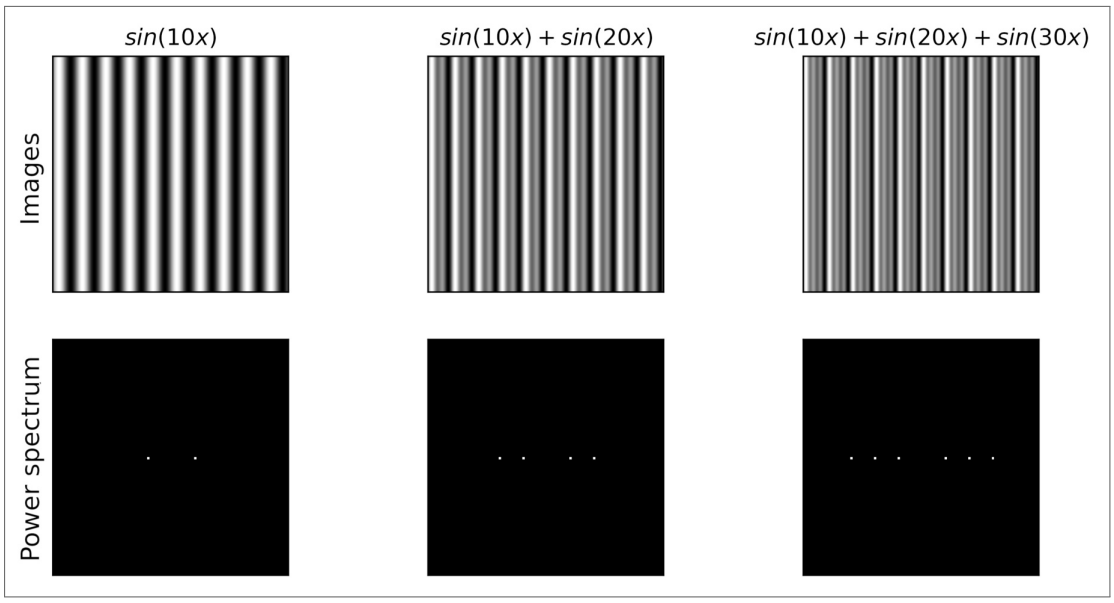
$$F(0,0) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) = MN\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) = MN\langle f(x,y)\rangle \tag{2.14}$$

where $\langle f(x,y)\rangle$ is the average value of $f(x,y)$. Because MN is usually large, $|F(0,0)|$ is typically the largest component of the Fourier spectrum by a factor of several orders of magnitude. $F(0,0)$ is called the *DC* component of the spectrum. Another important property is that, if $f(x,y)$ is real, then it can be shown that,

$$|F(u,v)| = |F(-u,-v)|, \tag{2.15}$$

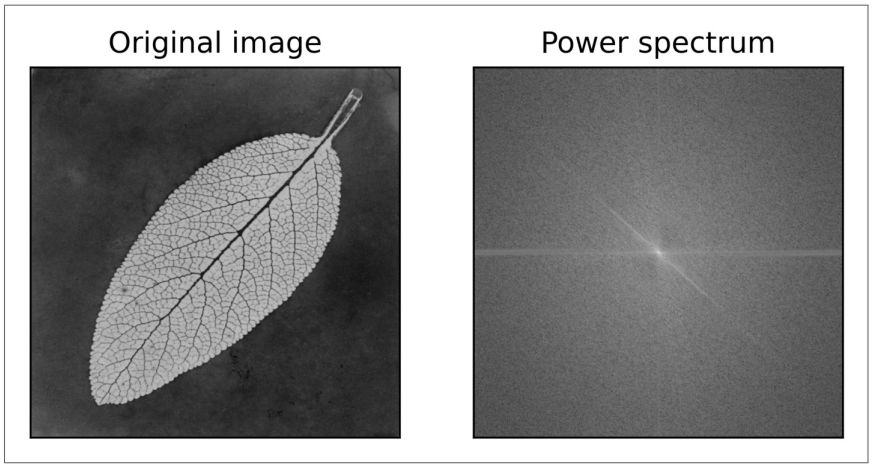
which implies that the spectrum of the Fourier transform is symmetric. Finally, the DFT of an image decomposes the image into a sum of sinusoids. The components of the spectrum determine the amplitude of these sinusoids. Figure 11 shows the power spectrum for sums of different sinusoids, for simple combinations of waves we have only a few bright spots. As discussed earlier we can see the symmetry of the power spectrum. The power spectrum of real images are much more complex than those of the sinusoids that we see in figure 11. Figure 12 is an example of the power spectrum of one such image.

Figure 11 – Power spectrums for diferent sinusoids



Source: The author (2022)

Figure 12 – Power spectrums of a real image



Source: The author (2022)

2.3.2 Convolution

In a 2D continuous space the convolution of two functions, $f(x,y)$ and $g(x,y)$, is defined as (13):

$$f(x,y)\bigstar g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(u,v)g(x-u,y-v)\,dx\,dy, \tag{2.16}$$

The 2D discrete convolution of a kernel, $w(x,y)$, of size *M x N* with an image $f(x,y)$ is defined as:

$$w(x,y)\bigstar f(x,y) = \sum_{m=-a}^{a}\sum_{n=-b}^{b} w(m,n)f(x-m,y-n), \tag{2.17}$$

where $a = (M-1)/2$ and $b = (N-1)/2$. The kernel $w(x,y)$ is also known as *filter*, *mask* or *point spread function* (FSD) depending on the situation.

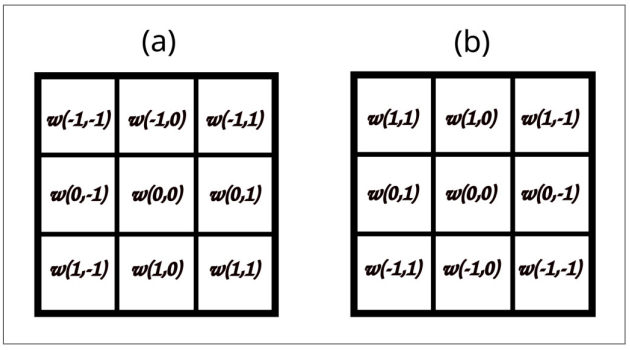
The convolution has the following properties:

- Commutativity: $w \star f = f \star w$.
- Associativity: $(w \star f) \star h = w \star (f \star h) = w \star f \star h$.
- Distributivity: $w \star (f + h) = (w \star f) + (w \star h)$.

The computation of the 2D discrete convolution can be understood intuitively as follow. Consider an image and a kernel, $f(x, y)$ and $w(x, y)$, respectively.

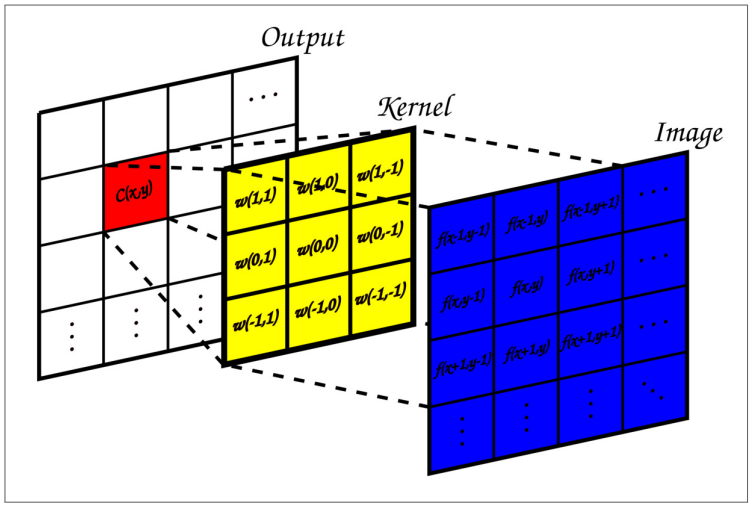
First, we flip the kernel in both horizontal and vertical directions, i.e., rotate by $180°$, figure 13. Then, we slide the flipped kernel across the image, $f(x, y)$, and at each location, we calculate the sum of the products of the kernel elements with the corresponding pixels directly under the kernel, as shown in figure 14.

Figure 13 – (a) Kernel. (b) Rotated kernel



Source: The author (2022)

Figure 14 – Output element of a convolution

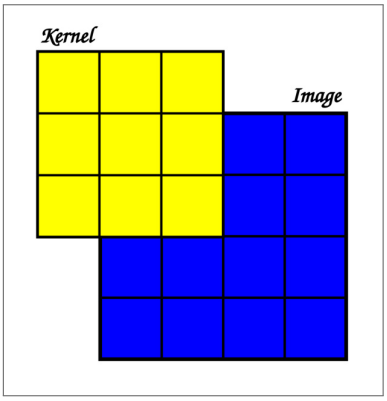


Source: The author (2022)

This calculation is performed for all values of the displacement variables x and y so that each element of w visits each pixel of f, this means that the correlation is a function of the

displacement variables x and y. The problem with the calculation as shown is that when we get close to the border of the image, the kernel extends over the edge of the image, as shown in figure 15. To solve this problem, we have to make some assumptions about pixels outside

Figure 15 – Kernel over the edge of the image



Source: The author (2022)

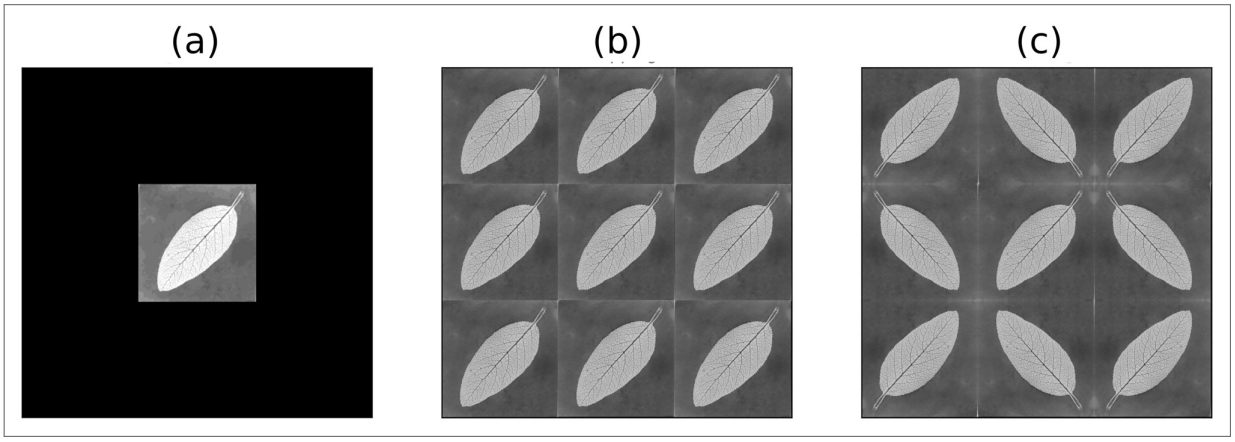
the image. There are many hypothesis that can be made, some of them shown in figure 16:

Filling with zeros We add a border area to the image and fill it with zeros. The result of this assumption is the *discrete linear convolution* (14).

Wrapping We add a border area to the image and fill it with copies of the image. The result of this assumption is known as a *cyclic convolution*, *circular convolution* or *wraparound* (14).

Mirroring We add a border area to the image and fill it with mirrored (horizontal, vertical or both depending on the direction) copies of itself.

Figure 16 – Different types of boundaries. (a) Filling with zeros. (b) Wrapping. (c) Mirroring.



Source: The author (2022)

It is important to note that no perfect boundary condition exists to handle the pixels close to the border of the image, all of them will introduce errors in the result (15).

Lastly, there is an import theorem known as *Convolution Theorem* that needs to be mentioned.

Theorem 1 (Convolution theorem) *Let $F(u,v)$ and $G(u,v)$ be the Fourier transform of two images, $f(x,y)$ and $g(x,y)$, of size M x N. Then,*

$$f(x,y)\bigstar g(x,y) = \mathcal{F}^{-1}[F(u,v)G(u,v)] \tag{2.18}$$

where $\mathcal{F}^{-1}$ indicates the inverse Fourier transform.

This means that convolution in the space domain corresponds to multiplication in the frequency domain (9).

The spatial domain filtering mentioned at the beginning of this section is performed by convoluting the input image with an appropriate filter kernel. Filter kernels are also known as filter masks, convolution masks and convolution kernels.

2.3.3 Correlation

The *2D discrete correlation* of a kernel, $w(x,y)$, of size *M x N* with an image $f(x,y)$ is defined by

$$w(x,y)\star f(x,y) = \sum_{m=-a}^{a}\sum_{n=-b}^{b} w(m,n)f(x+m,y+n) \tag{2.19}$$

where $a = (M-1)/2$ and $b = (N-1)/2$. The kernel $w(x,y)$ is also known as template. We note the similarities between the expressions for the discrete correlation given above and the discrete convolution given by equation 2.17. The difference between these two equations is the positive instead of negative signs in the second term of the summation. The implication of this difference is that the implementation of the correlation remains the same as the convolution with the exception that we do not flip the kernel in both directions, as shown in figure 17. The discussion we had in the convolution section about boundary conditions also applies here. The term *cross-correlation* is often used in place of *correlation* when $f(x,y) \neq w(x,y)$, images being correlated are different, and the term *autocorrelation* when $f(x,y) = w(x,y)$, or the images are the same.

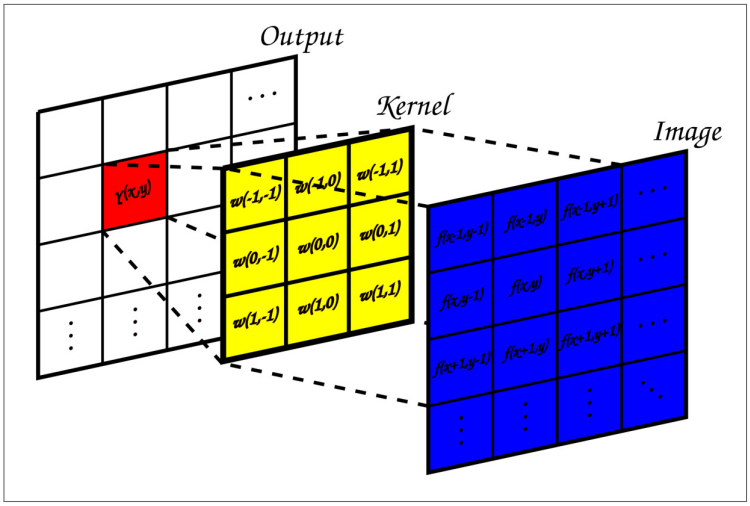
Another important theorem that needs to be mentioned is the *correlation theorem*.

Theorem 2 (Correlation theorem) *Let $F(u,v)$ and $G(u,v)$ be the Fourier transform of two images, $f(x,y)$ and $g(x,y)$, of size M x N. The correlation theorem states that (9),*

$$f(x,y)\star g(x,y) = \mathcal{F}^{-1}[F^*(u,v)G(u,v)] \tag{2.20}$$

where $\mathcal{F}^{-1}$ indicates the inverse Fourier transform and F^* indicates the complex conjugate of F.

Figure 17 – In the correlation the kernel is not rotated.



Source: The author (2022)

A special case of the correlation theorem occurs when $f = w$ and is known as *Wiener-Khinchin theorem* or *autocorrelation theorem*, which states that

$$f(x,y) \star f(x,y) = \mathcal{F}^{-1}\left[|F(u,v)|^2\right]. \tag{2.21}$$

In other words the autocorrelation of f is the inverse Fourier transform of its power spectrum.

Usually equation 2.19 as it was defined it is not very usefull for many applications because the result of the correlation is sensitive to scale changes in f and w (9). To overcome this problem, we defined the *Normalized Cross-Correlation*(NCC) by

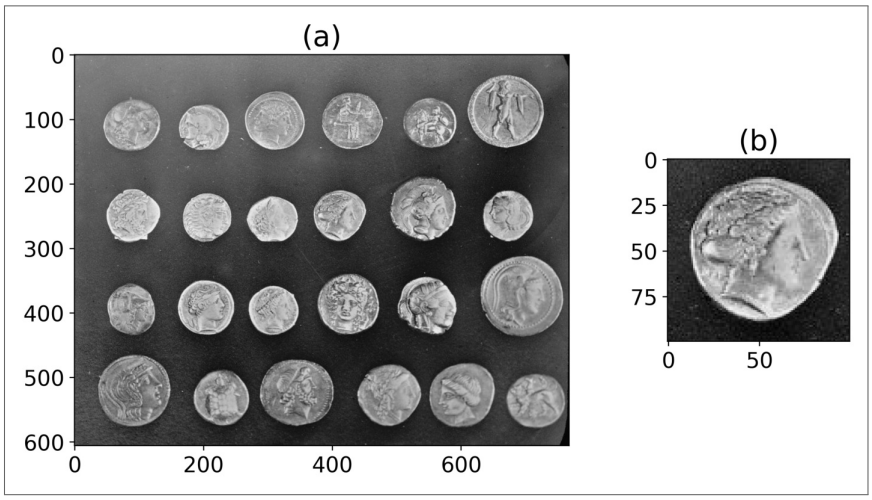
$$\gamma(x,y) = \frac{\sum_u \sum_v \left[w(u,v) - \overline{w}\right]\left[f(x+u,y+v) - \overline{f}_{xy}\right]}{\left\{\sum_u \sum_v \left[w(u,v) - \overline{w}\right]^2 \sum_u \sum_v \left[f(x+u,y+v) - \overline{f}_{xy}\right]^2\right\}^{\frac{1}{2}}}, \tag{2.22}$$

where $\overline{w}$ is the average value of w, $\overline{f}_{xy}$ is the mean value of f in the region where f and w overlap. The summation limits are taken over the region where f and w overlap. The values of $\gamma(x,y)$ are in the range $[-1,1]$.

The principal application of the correlation is *template matching*. In template matching we want to identify objects or regions in an input image that match a template image. The correlation of the input image and the template image will be maximum at the location where there is best correspondence. An example of template matching is shown bellow. The figure 18 (a) shows the input image, which is a set of old Greek coins. Figure 18 (b) is the template image, it is one of the coins of the input image, we will identify the best matching position of this coin in the input image.

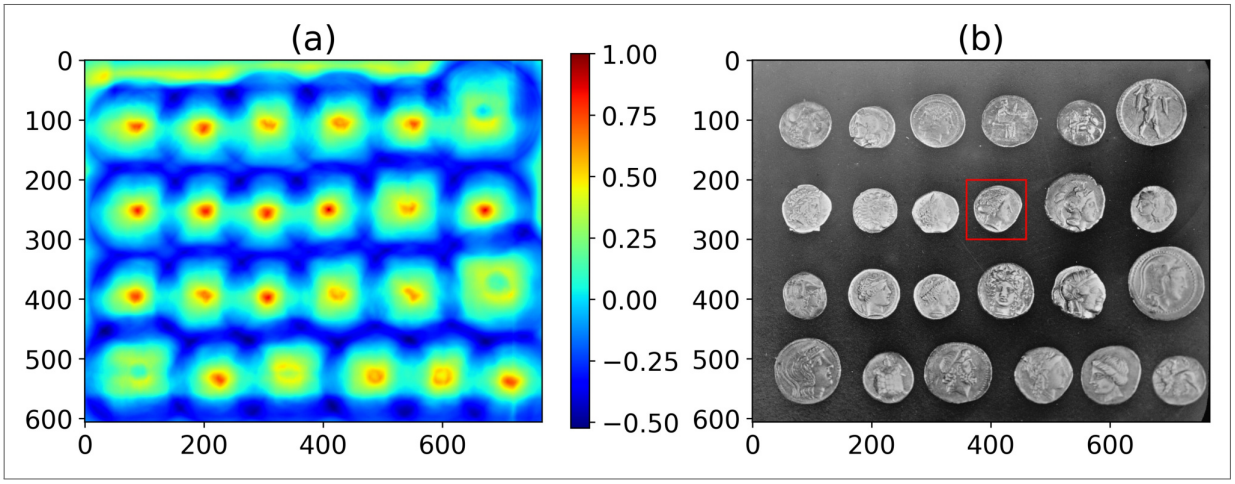
First we calculate the normalized cross-correlation between these images and then find the position of maximum correlation, this position has the best correspondence. Figure 19 (a)

Figure 18 – (a) Set of coins (input image). (b) One of the coins (template image).



Source: The author (2022)

Figure 19 – (a) Correlation of the images. (b) The position with best correspondence.



Source: The author (2022)

shows the correlation between the images and figure 19 (b) shows where the highest matching area is located in the input image, which is the exact position of the coin.

In figure 19 (a), the variables in the x and y axis are *displacements* and the value of each pixel is the value of the correlation at that specific pair (x,y).

2.3.4 Filtering in the frequency domain

Filtering is one of the most basic and common operations in image processing. We can summarize the process of filtering an image, $f(x, y)$, in the frequency domain, in the following three steps (9):

1. Calculate the centered Fourier transform, $F(u, v)$.

2. Multiply $F(u,v)$ by a filter function $H(u,v)$.

3. Compute the inverse Fourier transform and take the real part of the result.

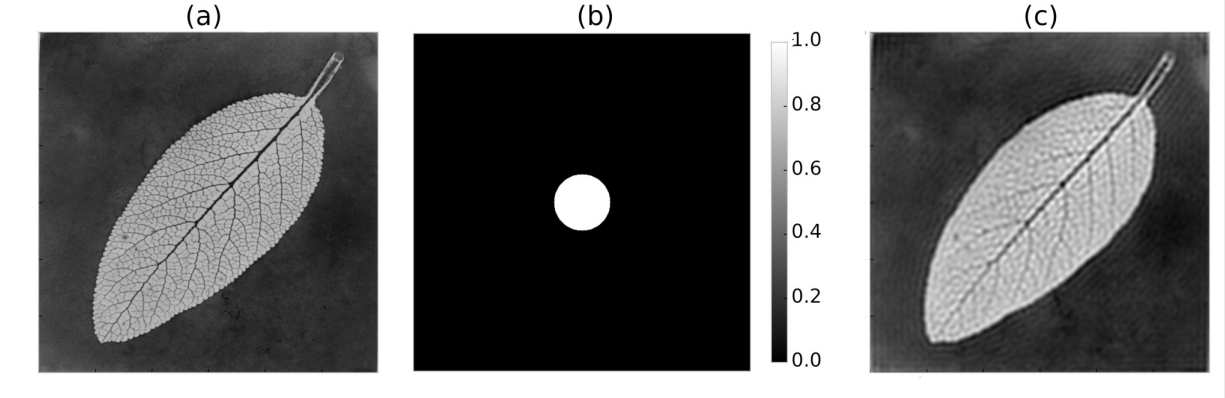
There are different types of filters, some of them are:

Ideal lowpass filter This filter attenuates all frequencies outside a circle of radius D_0, it is defined by the function

$$H(u,v) = \begin{cases} 1 & if\ D(u,v) \leq D_0 \\ 0 & if\ D(u,v) > D_0 \end{cases} \tag{2.23}$$

where $D(u,v)$ is the distance of the point (u,v) from the center. Figure 20 illustrates an example of ideal lowpass filtering. The black areas of the filter represent the regions that will be attenuated. Lowpass filters have the effect of blurring or smoothing the image, hence their alternate name 'blurring' or 'smoothing' filter."

Figure 20 – Ideal lowpass filtering. (a) Original image. (b) Ideal lowpass filter. (c) Filtered image.



Source: The author (2022)

Ideal highpass filter This filter is the opposite of the ideal lowpass filter, it attenuates all frequencies inside a circle of radius D_0. It is defined as,

$$H(u,v) = \begin{cases} 0 & if\ D(u,v) \leq D_0 \\ 1 & if\ D(u,v) > D_0 \end{cases}, \tag{2.24}$$

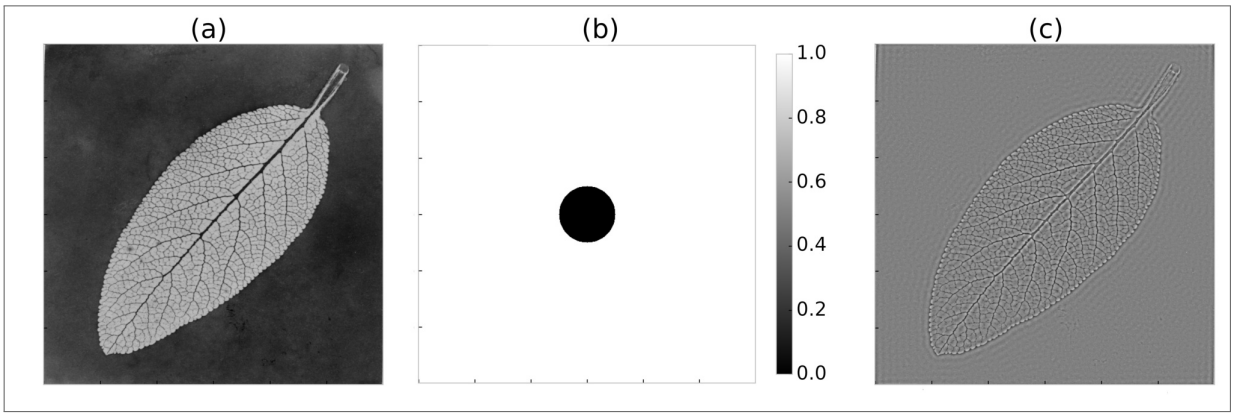
figure 21 shows an example of ideal highpass filtering. Highpass filters are commonly used in image processing to sharpen images and enhance edges.

Gaussian lowpass filter These filters have the form of a Gaussian function given by

$$H(u,v) = e^{D^2(u,v)/2D_0^2}, \tag{2.25}$$

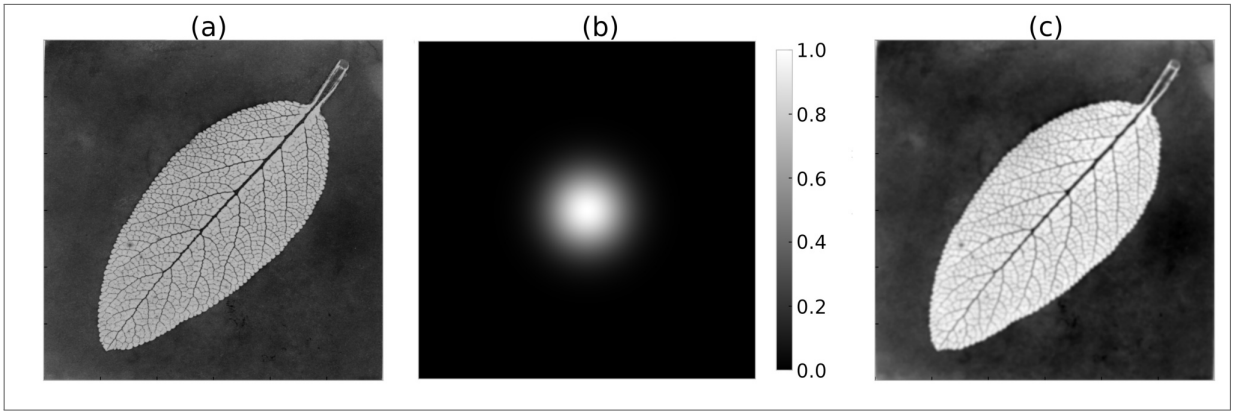
where D_0 is the cut off frequency. Gaussian lowpass filters are preferred over ideal lowpass filters. Figure 22 shows an example of Gaussian lowpass filtering. As explained before, a lowpass filter blurs the image.

Figure 21 – Ideal highpass filtering. (a) Original image. (b) Ideal highpass filter. (c) Filtered image.



Source: The author (2022)

Figure 22 – Gaussian lowpass filtering. (a) Original image. (b) Gaussian lowpass filter. (c) Filtered image.



Source: The author (2022)

Gaussian highpass filter This filter is the opposite of the Gaussian lowpass filter and has the form

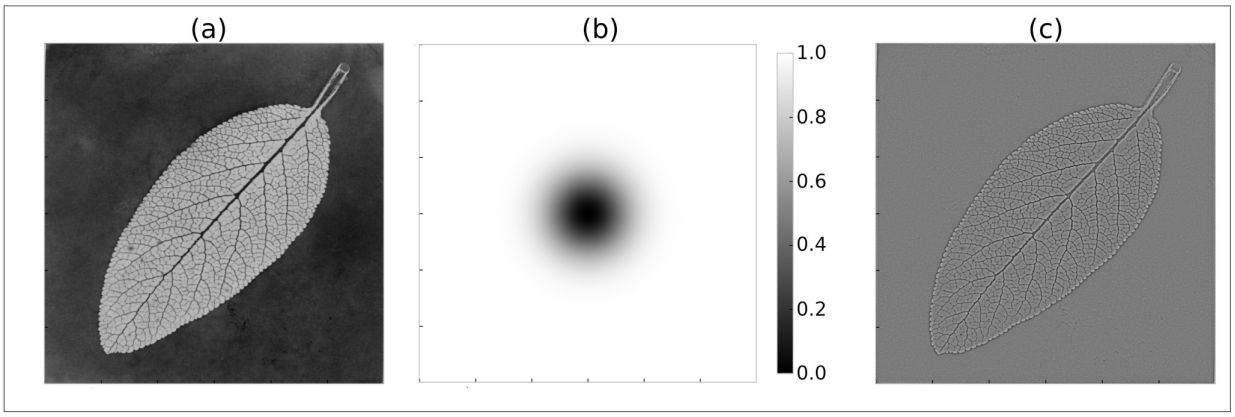
$$H(u,v) = 1 - e^{D^2(u,v)/2D_0^2}. \tag{2.26}$$

Gaussian highpass filters are preferred over ideal highpass filters .Figure 23 shows an example of Gaussian highpass filtering. As explained before, a highpass filter sharpens the image and enhance edges.

Notch filters The notch filter rejects (or passes) frequencies in predefined neighborhoods around a center frequency. The shape of the notch areas can also be arbitrary (e.g. rectangular or circular). These kind of filters are relevant because they can be used to remove periodic noise and interference from images. We made use of this kind of filter to remove the interference present in the laser beam images; more details will be given in the next section. In figure 24 we can see an example of interference filtering.

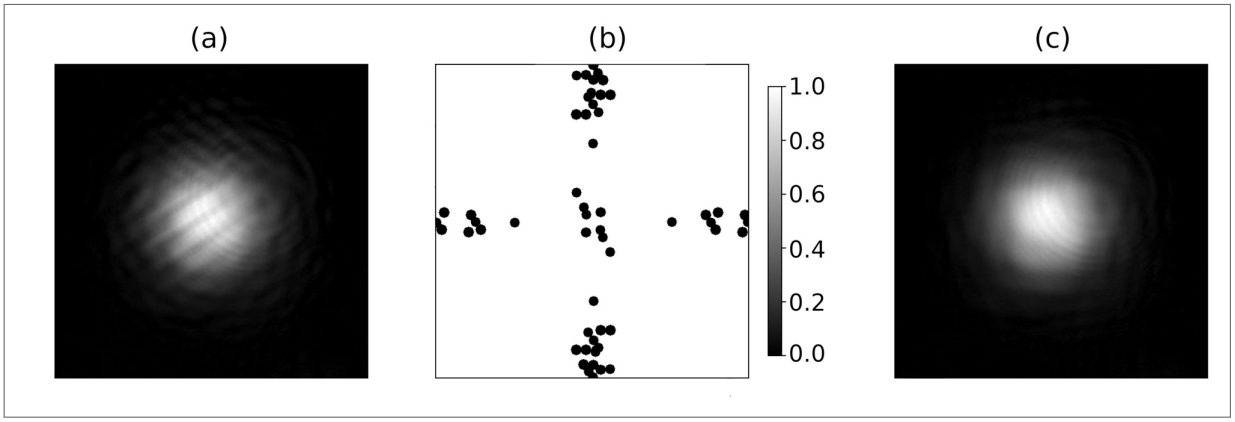
Ideal filters have a sharp cutoff that separates the frequencies that pass through the filter (the passband) from the frequencies that are blocked (the stopband). The transition between

Figure 23 – Gaussian highpass filtering. (a) Original image. (b) Gaussian highpass filter. (c) Filtered image.



Source: The author (2022)

Figure 24 – Notch filtering. (a) Original image. (b) Notch filter. (c) Filtered image.



Source: The author (2022)

these two regions is infinitely steep, resulting in a perfect distinction between the filtered and non-filtered frequencies. In contrast, Gaussian filters have a smooth cutoff that separates the passband and stopband. They are based on the Gaussian function, which results in a gradual transition between the filtered and non-filtered frequencies. In practice, Gaussian filters are preferred over ideal filters, as ideal filters are mostly used for educational purposes.

2.4 ALGORITHMS

In this study, various techniques of image processing will be employed to analyze laser beam images. In this section, we will describe the specific algorithms utilized in our analysis. To begin with, we will outline the programming language and software libraries employed in our research.

2.4.1 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It is a general-purpose programming-language, which means it can be used for almost anything. It is open-source, friendly, easy to learn and easy to read programming language. It is one of the most popular programming languages and it is widely used by the scientific community. All of our algorithms were implemented using python version 3.8.10. Because it is open-source it has thousands of third-party modules(libraries) available. In our analysis we made extensive use of the following libraries:

SciPy SciPy is an open-source library of numerical routines for scientific computing in Python. SciPy contains algorithms for linear algebra, optimization, integration, interpolation, special functions, Fourier transforms, image processing, differential equations and many other classes of problems (16).

NumPy NumPy is an open-source library of numerical routines for N-dimensional array processing in Python (17).

Matplotlib Matplotlib is an open-source library for creating static, animated, and interactive visualizations in Python (18).

OpenCV OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. It is available for C++, Python, Java and MATLAB (19).

pandas pandas is an open-source library for data analysis and data manipulation (20) (21).

Scikit-image Scikit-image is an open-source library for image processing (22).

Mayavi Mayavi is an open-source library for interactive scientific data visualization and 3D plotting in Python (23).

2.4.2 Beam diameter

In our study, we aimed to determine the diameter of laser beams from the images captured. We calculated the beam diameter in both the horizontal and vertical direction using the $1/e^2$ width definition. The $1/e^2$ width is defined as the distance between the two points on the marginal distribution that are 0.135 times the maximum value. For a Gaussian beam we have,

$$2w = \frac{\sqrt{2}\mathrm{FWHM}}{\sqrt{ln2}} = 1.699 * \mathrm{FWHM} \tag{2.27}$$

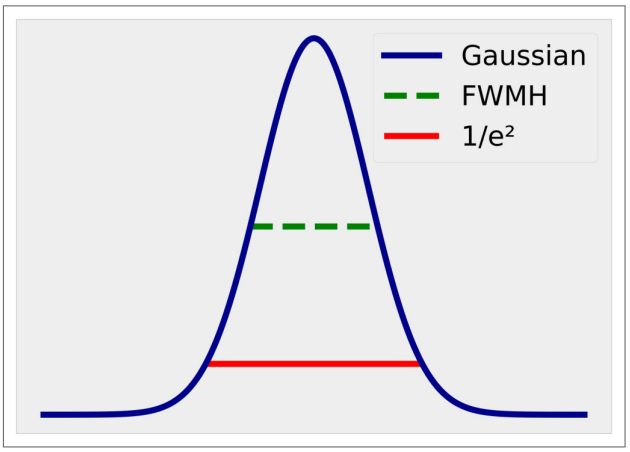
where $2w$ is the full $1/e^2$ width and FWHM is the *full width at half maximum* (24). Figure 25 shows the FWHM and the full $1/e^2$ width for a Gaussian curve.

The algorithm used to find the diameter from an image, has the following procedures:

1. We find two sums, the sum of all rows, S_x, and the sum of all columns, S_y.
2. We do a Gaussian fit on both of these sums.
3. From these Gaussian fits we get the $1/e^2$ width.
4. Multiply the $1/e^2$ widths by the pixel size of the camera. These values are the horizontal and vertical diameters of the beam.

From the sum of rows we get the horizontal diameter and from the sum of columns we get the vertical diameter. This procedure works well for beams with a circular profile. The figure 26 shows the sums of rows and columns and its respective Gaussian fits.

Figure 25 – FHMW and $1/e^2$ width



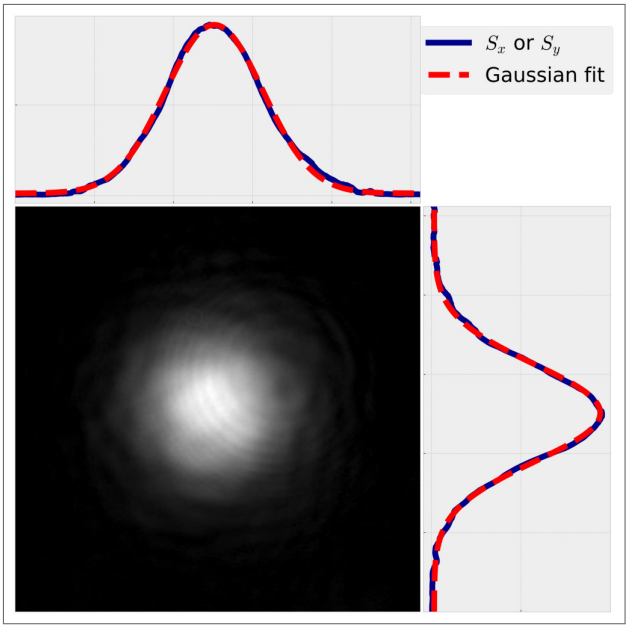
Source: The author (2022)

We can do all these previous steps using functions from the SciPy and NumPy libraries. To find the sum of the rows and columns we can use the function *sum* from NumPy and to find the Gaussian fit we can use the function *curve_fit* from the SciPy library.

2.4.3 Intensity fluctuations of the beams

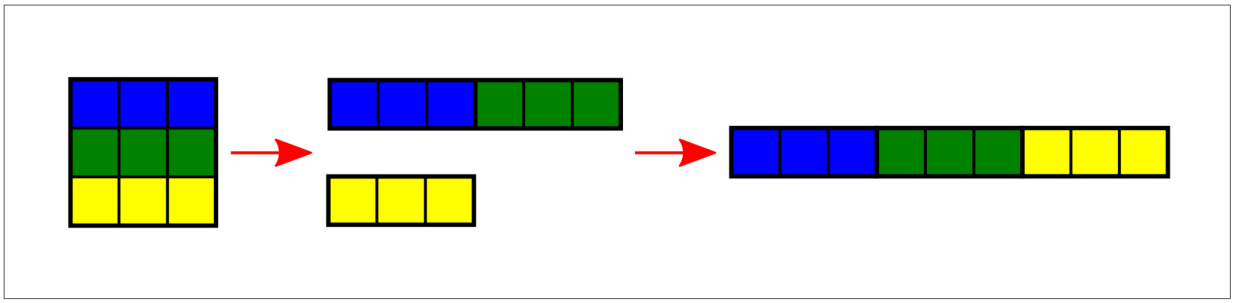
In our analysis, we had to find the intensity fluctuations of the laser beams. To accomplish this, we considered three different methods. In the first one, we take the image of the laser beam and fit a two dimensional Gaussian curve on it, to get the fluctuations, we subtract the Gaussian fit from the image. To find the two dimensional Gaussian fit, we first need to transform the input image, which is a two dimensional array, into a one dimensional array by using the NumPy method *ravel()*. What this method does is take each row and put it right at the end of the previous one, we say that it *flattened* the array. Figure 27 shows this process for a *3 x 3* array.

Figure 26 – Sums of rows and columns



Source: The author (2022)

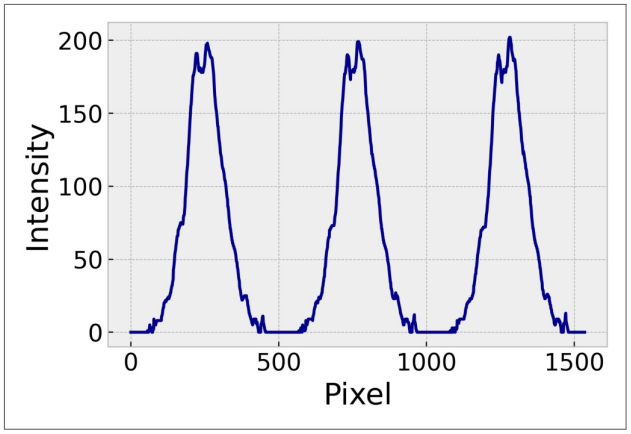
Figure 27 – Flattening of a *3 x 3* array



Source: The author (2022)

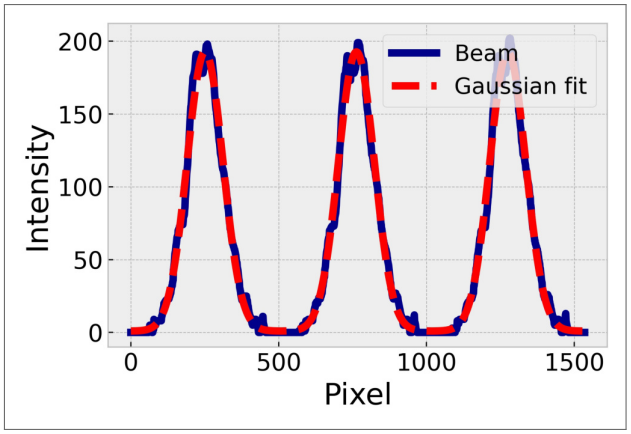
This one dimensional array is composed of the all the rows of the input image one after the other in sequence. For an image of a laser beam, if we plotted this one dimensional array what we would see is a sequence of curves resembling Gaussian curves. In figure 28 we can see the plot of a small section of one of these flattened laser beam images. The next step is to do a Gaussian fit on this vector using the SciPy function *curve_fit*, this will fit all the Gaussian curves contained in the flattened array. For this process, we define a function that generates a 2D Gaussian and returns it flattened. It is this function that we use for the fitting. Figure 29 shows a small section of one of these flattened laser beam images and its Gaussian fit.
Finally, we use the method *reshape* from the the NumPy library to revert back the flattened array to its original dimensions and use the function *subtract* from the OpenCv library to subtract the Gaussian fit from the laser beam image. We use the function *subtract* to guarantee that both images have the same type and depth, if this condition is not satisfied, then an error is thrown by the function. Figure 30 shows on the first row, the beam image, the Gaussian fit

Figure 28 – Section of a flattened image of a laser beam



Source: The author (2022)

Figure 29 – Section of a flattened image of a laser beam and its gaussian fit



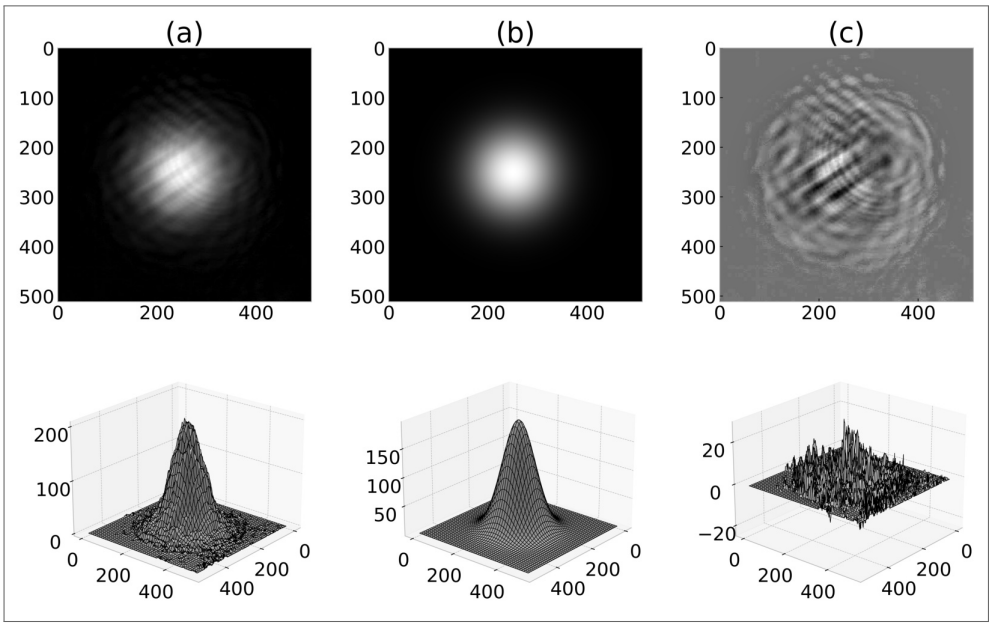
Source: The author (2022)

and the fluctuations; On the second row we have a 3D view of these images.

The second method we used to find the fluctuations was by calculating the mean image of the input image and then subtracting it from the input image. The mean image is obtained by linear averaging filtering of the input image. This kind of filter replaces the value of every pixel in an image by the average value of neighboring pixels, including itself. The number of pixels used in the calculation of the mean value depends on the dimensions of the filter mask. This is a type of filtering in the spatial domain, where the filter mask is a two dimensional array divided by the number of elements. In this case, all elements of this two dimentional array are equal to 1. Figure 31 shows an example of a *3 x 3* averaging kernel, this kernel has 9 elements which means that 9 pixels will be used for calculating the average value. Figure 32 shows the beam image, average image, fluctuations and the 3D view of these quantities.

The final method for finding intensity fluctuations is to subtract an average of all video frames of the image. Each video consists of a series of images, then we take the average of this set of images and subtract this average from the images to obtain the fluctuations of

Figure 30 – (a) Laser beam. (b) Gaussian fit. (c) Fluctuations



Source: The author (2022)

Figure 31 – 3 x 3 smoothing averaging kernel

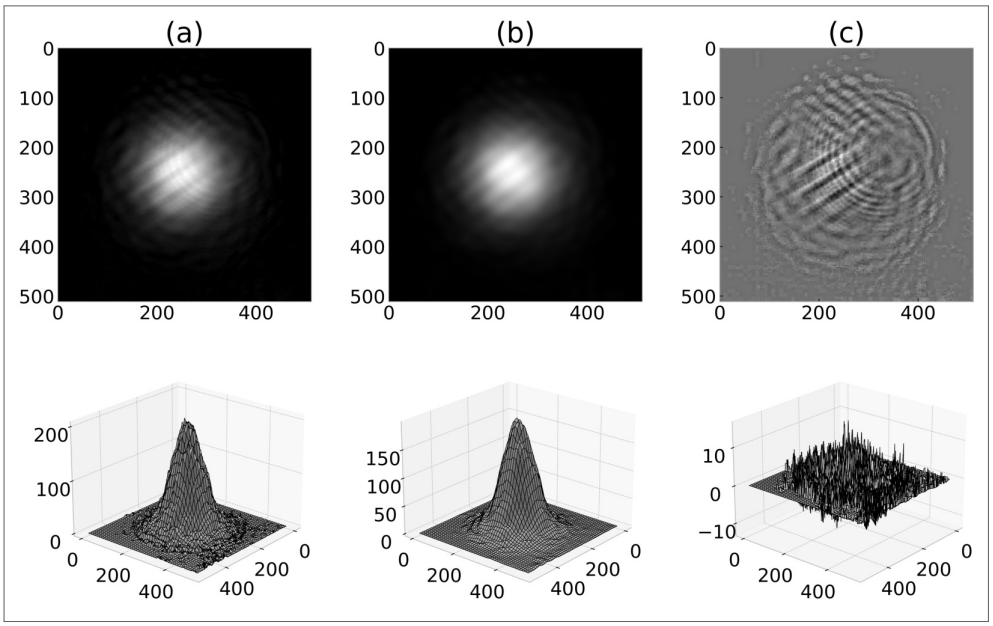
$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Source: The author (2022)

intensity. This method has some problems. First, averaging must be done on a set with a large number of images, however, most of our videos contain a maximum of 10 frames. We are not frequency locking our laser, so each frame of the video is in a slightly different condition. Thus, we are averaging images that are not in the same condition. This is much more evident in the images of the FWM signals where we can visually observe differences in the frames.

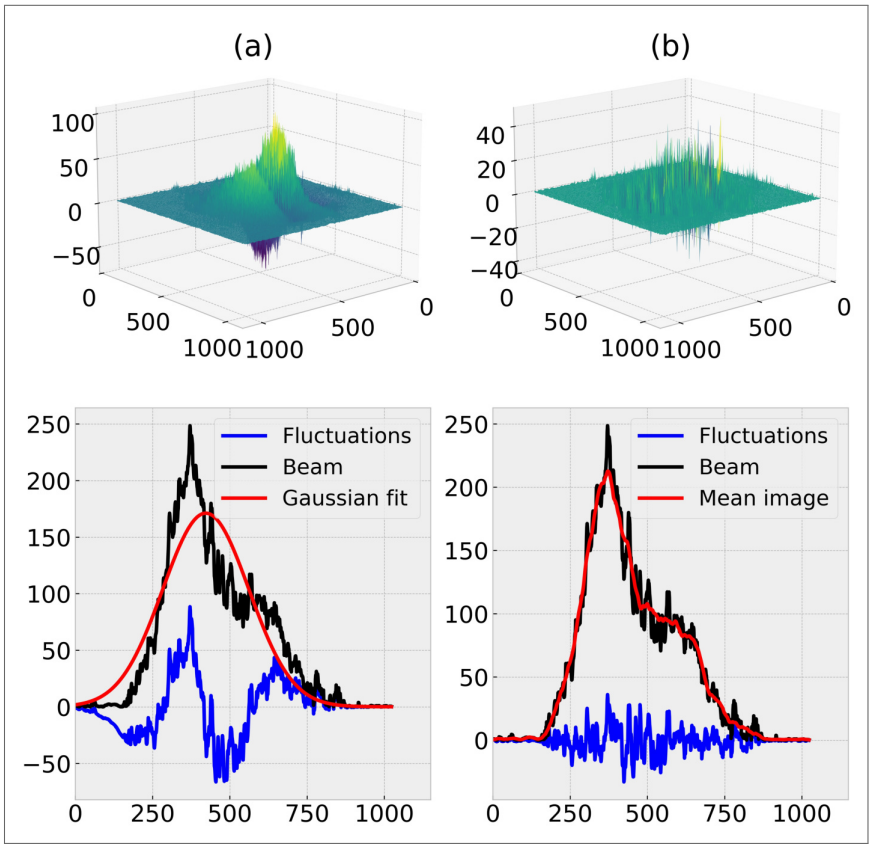
In figure 33 we have a comparison between the first and second methods. On the first row we have a 3D view of these intensity fluctuations and in the second row we have plotted a line of the original beam, a line of the Gaussian fit in (a)(average image in (b)), and a line of the resulting intensity fluctuation. Our beam does not have a perfect Gaussian profile, this indicates that subtracting a Gaussian fit to find the intensity fluctuations may not be the best choice. Subtracting the average image gives us a much better result. Thus, from now on when we talk about intensity fluctuations we are referring to the intensity fluctuation that we get

Figure 32 – (a) Laser beam. (b) Average image. (c) Fluctuations



Source: The author (2022)

Figure 33 – Intensity fluctuations. (a) Subtracting Gaussian fit. (b) Subtracting mean image.



Source: The author (2022)

from subtracting the average image from the intensity profile image.

2.4.4 Cross-correlation and Autocorrelation

The normalized cross-correlation(NCC) can be calculated directly by the equation 2.22, however, it has a high computational cost which means it is very time consuming. For the image f and template w of sizes *M x N* and *m x n* respectively, it can be shown that the correlation has computational complexity of *O(m x n x M x N)* (25). An alternative way of calculating the NCC is by taking advantage of the implementations similarities between the correlation and convolution. In subsections 2.3.2 and 2.3.3 we discussed how both of these implementations are the same with the exception that for the convolution we have to flip the kernel in both directions(rotate by $180°$), meaning that if we already have the convolution implemented we don't need to implement the correlation directly, we may use the convolution to find the correlation. In order to find the correlation between two images using a pre-implemented convolution function, we must first rotate the template by 180 degrees. This is because the function first rotates the template before performing the necessary computations. By passing the two images to the function with the template already rotated, the function will rotate the template again by 180 degrees, resulting in a total rotation of 360 degrees. This means that the calculations that follow will be done with the original images, thus calculating the correlation. However, this method only gives us the numerator of the Normalized Cross-Correlation (NCC) (equation 2.22), additional computations are needed to find the denominator. The problematic quantity in the denominator is $\sum_u \sum_v \big[f(x+u,y+v) - \overline{f}_{xy}\big]^2$. Suppose f has size *M x M* and w has size *N x N*, then

$$\sum_u \sum_v \big[f(x+u,y+v) - \overline{f}_{xy}\big]^2 = \sum_u \sum_v \big[f(x+u,y+v)\big]^2 - 2\sum_u \sum_v f(x+u,y+v)\overline{f}_{xy} + \sum_u \sum_v \big[\overline{f}_{xy}\big]^2,$$

the summation limits are taken over the region where f and w overlap. The terms $\overline{f}_{xy}$ can be precomputed, so they can leave the double summation.

$$\sum_u \sum_v \big[f(x+u,y+v) - \overline{f}_{xy}\big]^2 = \sum_u \sum_v \big[f(x+u,y+v)\big]^2 - 2\overline{f}_{xy}\sum_u \sum_v f(x+u,y+v) + N^2 \overline{f}_{xy}{}^2,$$

We used the fact that $\sum_u \sum_v 1 = N^2$. By definition $\overline{f}_{xy} = \frac{1}{N^2}\sum_u \sum_v f(x+u,y+v)$, so

$$\begin{aligned}\sum_u \sum_v \big[f(x+u,y+v) - \overline{f}_{xy}\big]^2 &= \sum_u \sum_v \big[f(x+u,y+v)\big]^2 - \frac{2}{N^2}\left[\sum_u \sum_v f(x+u,y+v)\right]^2 \\ &\quad + N^2 \frac{1}{N^4}\left[\sum_u \sum_v f(x+u,y+v)\right]^2 \\ &= \sum_u \sum_v \big[f(x+u,y+v)\big]^2 - \frac{1}{N^2}\left[\sum_u \sum_v f(x+u,y+v)\right]^2,\end{aligned}$$

both of these terms can be calculated via the convolution. For a given (x, y), the first term, $\sum_u \sum_v \big[f(x+u, y+v)\big]^2$, is the element (x, y) of the convolution of f^2 with a two dimensional array of ones with the same size as the template. For the second term, $\sum_u \sum_v f(x+u, y+v)$, we convolve f with a two dimensional array of ones with same size as the template. By calculating these two last quantities we can find the denominator in the NCC. We can summarize the process of calculating the NCC in the following steps:

1. Remove the mean value of the image and template.

2. Make the image convolution with the template rotated by $180°$.

3. Use the convolution to calculate the normalization terms.

In python, we can use the function *fftconvolve* from SciPy to calculate the convolution and the normalizing terms, and use the function *rot90* from NumPy to rotate the template. In python code we have:

```
# Based on the Octave implementation by Benjamin Eltzner, 2014 <b.eltzner@gmx.de>
# Step 1
template = template - np.mean(template)
image = image - np.mean(image)

# Step 2
template_ones = np.ones(np.shape(template))
template_rot = np.rot90(template, 2)
output = fftconvolve(image, np.conjugate(template_rot), mode='same')

# Step 3
normalization_terms = fftconvolve(np.square(image), template_ones, mode='same') -
     np.square(fftconvolve(image, template_ones, mode=mode))/(np.prod(np.shape(
    template)))

# Result
sum_squares_template = np.sum(np.square(template))
output = output/np.sqrt(sum_squares_template * normalization_terms)
```

2.4.5 Radial profile

Given an image, $\mathcal{I}$, we want to find the radial distribution of intensity (average intensity as a function of the radius) or radial profile of that image. Here we will describe in detail the algorithm that we use to find this radial profile.

The first step is to find a matrix of distances, $\mathcal{D}$, where for a given pixel in the image $\mathcal{I}$ at position (x,y), we find its distance to a specified center and store this value in the matrix of distances $\mathcal{D}$ at the position (x,y). In other words, we find the distance of all pixels in the image $\mathcal{I}$ with respect to a specified center and store it in a matrix $\mathcal{D}$ at the exact same position of the corresponding pixel in the image $\mathcal{I}$. This can be achieved in many ways. One possible

way is to use *for loops* to visit each pixel of the image $\mathcal{I}$ and calculate the distance. Another method is by using the function *indices* from the Numpy library, this function returns an array representing the indices of a grid. We pass to this function the dimensions of the image $\mathcal{I}$. This array of indices represent the positions x and y of the pixels in the image, so we can use this array to find the distances. In both methods we will find the distances as decimal numbers (floating point numbers), but we are dealing with distances in pixels, so it is necessary to convert these values to integers. Here we used the method *astype* from the Numpy library on the matrix $\mathcal{D}$ to change from decimal numbers to integers. As an example consider the image $\mathcal{I}$ as the 5x5 matrix shown in figure 34. By using any of the two methods mentioned above

Figure 34 – Image $\mathcal{I}$

1	1	2	1	1
1	4	7	4	1
2	7	11	7	2
1	4	7	4	1
1	1	2	1	1

Source: The author (2022)

we find the matrix $\mathcal{D}$, shown in figure 35. For a 5x5 matrix we have possible radii of 0, 1 and 2.

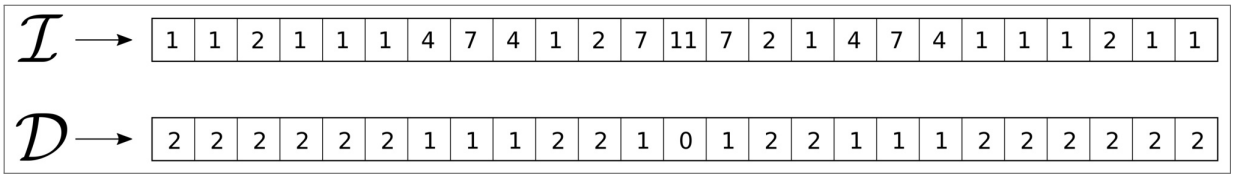
Figure 35 – Matrix of distances $\mathcal{D}$

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Source: The author (2022)

The next step consists of finding the sums of all pixels for a given radius. To do this we use the function *bincount* from the Numpy library. We pass to this function the image $\mathcal{I}$ and the matrix $\mathcal{D}$, however this function only works for one dimensional arrays, so we flatten both the image $\mathcal{I}$ and the matrix $\mathcal{D}$ before passing them on to the function. The function *bincount* is usually used for counting the number of occurrences of each value in an array of non-negative integers, but when in addition to the first array, an array of weights is given, this function will perform sums over variable-size chunks of the first array. Let us make this operation clearer using the example image $\mathcal{I}$ in figure 34 and the corresponding matrix $\mathcal{D}$ in figure 35. In figure 36 we see both of these arrays flattened. In this situation the weights are the flattened array $\mathcal{I}$. The function *bincount* will associate the first value of the flattened array $\mathcal{I}$ with the first

Figure 36 – Image $\mathcal{I}$ and matrix $\mathcal{D}$ flattened



Source: The author (2022)

value of the flattened array $\mathcal{D}$, the second value of the flattened array $\mathcal{I}$ and the second value of the array $\mathcal{D}$ and so on, i.e., it will associate each pixel in $\mathcal{I}$ with its respective distance in $\mathcal{D}$. Then, for each possible radius, *bincount* will store in a new array the sum of all elements that have the same radius associated. We show this calculation bellow.

$$
\begin{aligned}
Radius \quad 0: \quad & 11 \\
Radius \quad 1: \quad & 4+7+4+7+7+4+7+4 = 44 \\
Radius \quad 2: \quad & 1+1+2+1+1+1+1+2+2+1+1+1+1+2+1+1 = 20
\end{aligned}
$$

The returned array of sums is [11, 44, 20] and the elements are the sums of the intensities of the pixels at the radius of 0, 1 and 2 respectively. The last step consists of finding the number of pixels at each radius and then dividing each element of the returned array of sums by its respective number of pixels. The number of pixels at each radius is easy to find, we use again the function *bincount*, but this time we pass on to it only the flattened array $\mathcal{D}$. As a result we obtain the array [1, 8, 16]. Carrying out the division of each element of the array of sums is [11, 44, 20] by its respective number of pixels, we obtain:

$$\text{radial profile} = [11/1, 44/8, 20/16] = [11, 5.5, 1.25]$$

We can summarize this process in the following steps:

1. Calculate the distance of each pixel in the image with respect to a specified center and store them in a matrix.

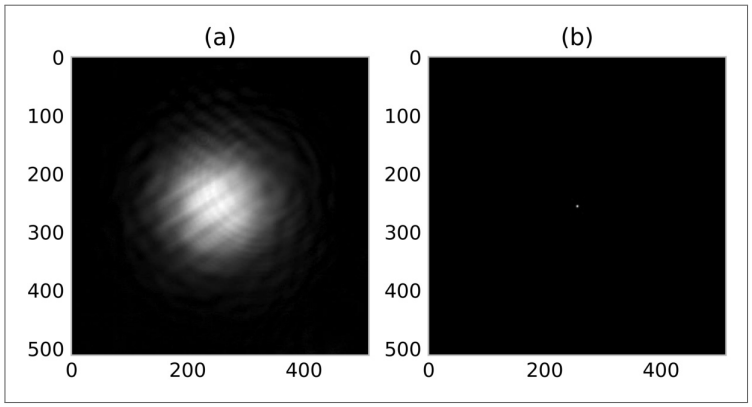
2. Use the function *indices* from the Numpy library to find the radial sums.
3. Use again the function *indices* to find the number of pixels at each possible radius.
4. Divide each element of the radial sums' array by the respective number of pixels at that radius.

2.4.6 Pattern removal filter

In our laser beam images, we consistently observed interference patterns. These regular patterns can interfere with the results of our analysis and potentially lead to incorrect conclusions. To mitigate this issue, there are various techniques for eliminating these patterns. Most of these methods involve identifying and eliminating the frequencies associated with the patterns in the Fourier transform of the images. However, a drawback of many of these methods is that they require a significant amount of experimentation to identify the related frequencies, making it challenging to automate the removal of these patterns (26). Fortunately, there are a few methods that enable this automation process (26) (27) (28). We used the method explained in *Background pattern removal by power spectral filtering* by *Michael Cannon, Alex Lehar, and Fred Preston* (26).

As it was discussed in subsection 2.3.1, the Fourier transform breaks down the image into a sum of sinusoids that constitute the image and these sinusoids appear as bright spots on the power spectrum. Hence, the periodic interference that we want to remove also appears as bright spots in the power spectrum. We need to identify these Fourier components so that we can attenuate them. However, as it was discussed in the same subsection, the DC component of the Fourier transform is the largest component of the Fourier spectrum by a factor of several orders of magnitude, therefore, only the center of the power spectrum can be seen in the power spectrum. This can be observed in figure 37, which displays an image of the laser beam and its corresponding power spectrum, where only the DC component is visible.

Figure 37 – (a) Image. (b) Power spectrum



Source: The author (2022)

In order to detect the other components in the power spectrum, we apply the *averaging method of Welch* (26) (29) to obtain a modified power spectrum that allows us to identify these components. The averaging method of Welch consists of dividing the image into overlapping subsections, performing a modified periodogram on each subsection, which is the squared magnitude of the Fourier transform multiplied by a 2D hanning window, and then averaging all these modified periodograms. We can summarize this process in the following steps,

1. Divide the image into a number N of subsections, $I_i(x, y)$, that overlap.
2. Multiply each subsection by a 2D Hanning window, $w(x, y)$, to reduce edge effects.
3. Find the magnitude squared of the Fourier transform for each of these products $I_i(x, y) * w(x, y)$.
4. Average all the results of step 3.

The result of all the steps above return the modified power spectrum,

$$\phi_m = \frac{1}{N} \sum_i \left| \mathcal{F} \left[I_i(x, y) \cdot w(x, y) \right] \right|^2, \tag{2.28}$$

where $\mathcal{F}$ represents the Fourier transform operator.

In step 2 we multiply the image subsections by a 2D Hanning window, also known as *Hann window*. A Hann window is a type of window function that is commonly used in signal processing to reduce spectral leakage in the Fourier transform. In one dimension, a Hanning window is defined as

$$w(x) = 0.5 - 0.5 \cos\left(\frac{2\pi x}{M-1}\right) \quad 0 \le x \le M-1. \tag{2.29}$$

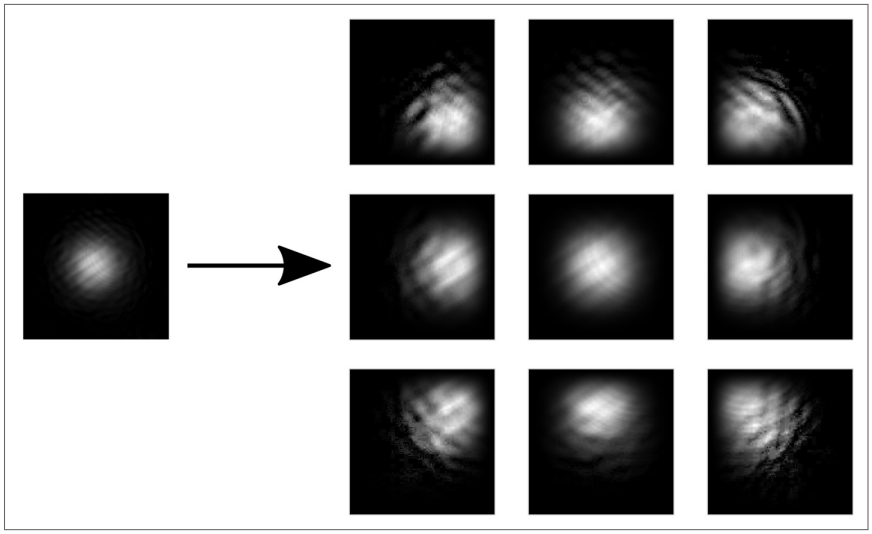
To obtain a 2D Hanning window, we compute the outer product of two 1D Hanning windows,

$$w(x, y) = w_1(x) w_2(y). \tag{2.30}$$

The number of subsections, N, of the image can vary based on the amount of overlap between the subsections. A higher overlap results in more subsections, while less overlap leads to fewer subsections. The typical approach is to have the subsections overlap by half of their size. Figure 38 shows an image and some of its subsections multiplied by the 2D Hanning window. Figure 39, we display the original image, the power spectrum, and the modified power spectrum. The modified power spectrum allows us to clearly observe the other components of the Fourier transform.

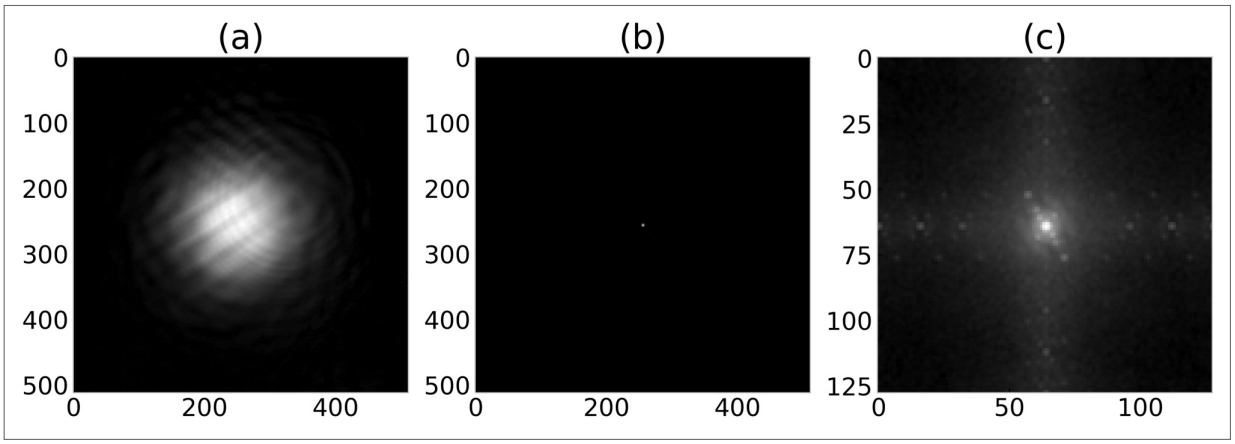
The next step is to remove the DC peak of the modified power spectrum without disturbing the other peaks. This is achieved in two steps. Firstly, we calculate a radial average of the modified power spectrum, which means we find the average value of the pixels for each radius. Figure 40 shows the radial average of the modified power spectrum. We then subtract this

Figure 38 – Subsections of an image



Source: The author (2022)

Figure 39 – (a) Image. (b) Power spectrum. (c) Modified power spectrum

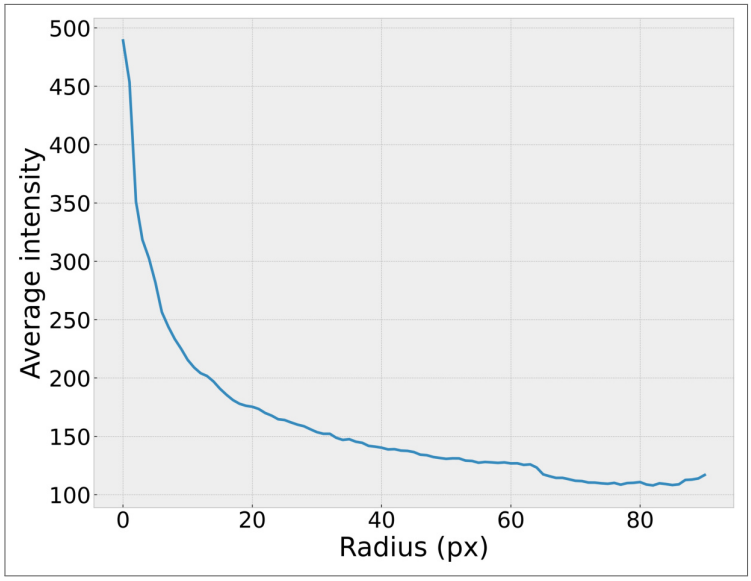


Source: The author (2022)

radial average from the modified power spectrum. However, since the radial average is a one-dimensional curve, we can not subtract it directly from the two-dimensional modified power spectrum. To make this subtraction possible, we rotate the radial average around one of the axes, resulting in a two-dimensional curve. Figure 41 shows the modified power spectrum, the rotated radial average, and a comparison between the center row of both the modified power spectrum and the rotated radial average. Figure 42 shows the result of the subtraction of the rotated radial average from the modified power spectrum.

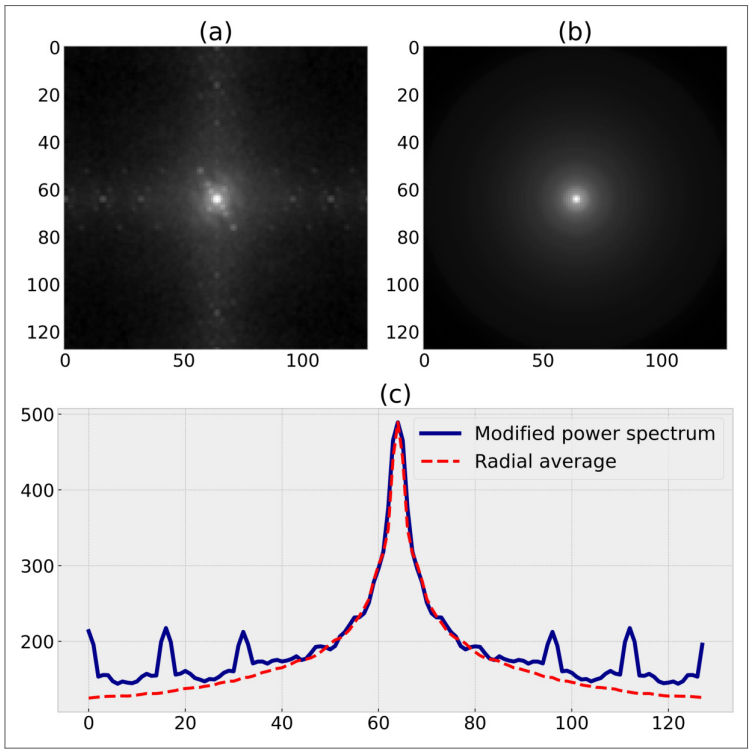
The next step is to isolate the peaks in the Fourier transform that are related to the periodic interference, these peaks are much larger than the other components of the Fourier transform. To accomplish this, we establish a threshold, and set anything below this threshold to zero. Afterwards, we resize the result to match the input image. Figure 43 shows the modified power spectrum after these modifications. Finally, we want to eliminate these resulting peaks from

Figure 40 – Radial average



Source: The author (2022)

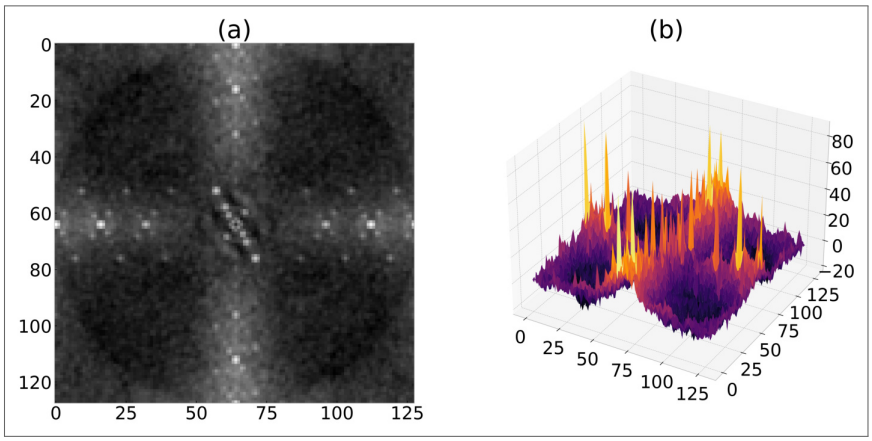
Figure 41 – (a) Modified power spectrum. (b) Radial average rotated. (c) Comparison between the center row of (a) and (b)



Source: The author (2022)

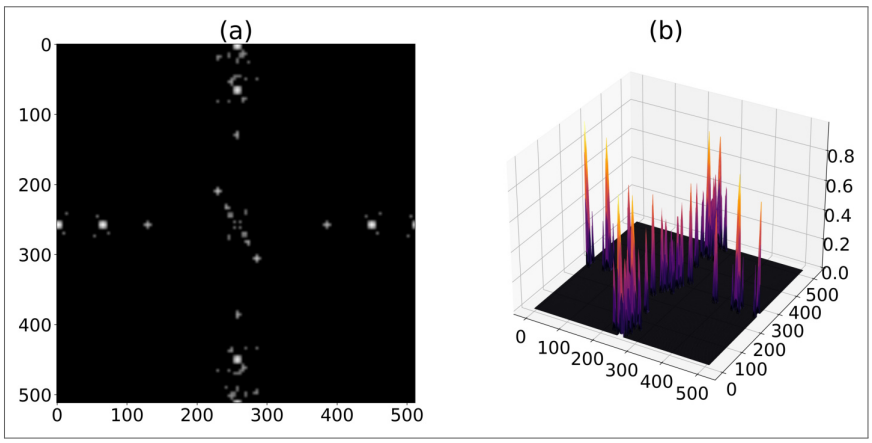
the Fourier transform. To do this, we need to locate these peaks and then create a notch filter that eliminates neighborhoods centered on these positions. We choose a circular shape for the notch areas. To filter the image, we multiply the notch filter with the Fourier transform of the input image, take the inverse Fourier transform, and then take the real part. The result of this

Figure 42 – (a) Result of the subtraction. (b) 3D view.



Source: The author (2022)

Figure 43 – (a) Smaller coefficients set to zero and resized. (b) 3D view.



Source: The author (2022)

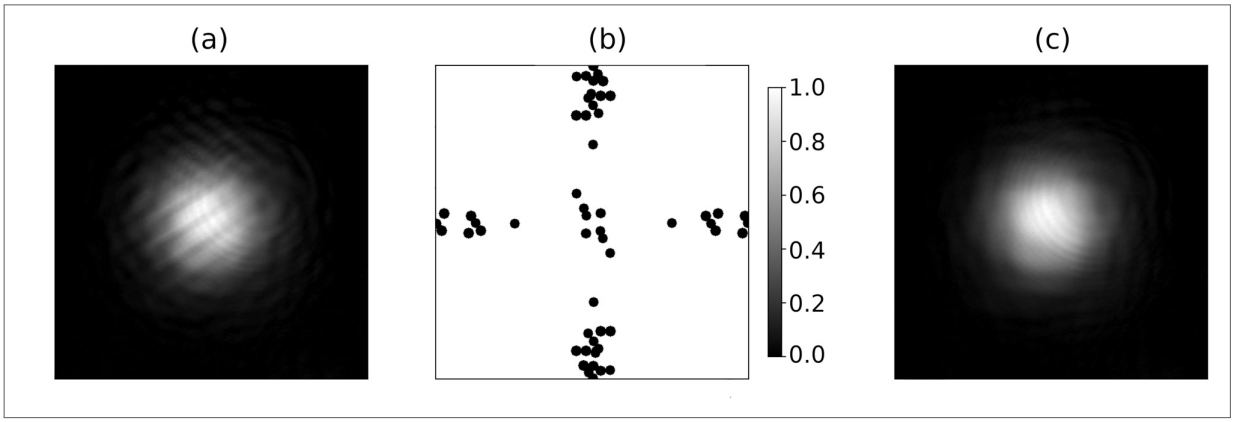
process is the filtered image. We can summarize this process in the following steps:

1. Find the modified power spectrum of the input image.
2. Remove the DC peak.
3. Set small Fourier components to zero and resize the modified power spectrum.
4. Find the peaks and create a notch filter.
5. Multiply the notch filter by the Fourier transform of the input image.
6. Calculate the inverse Fourier transform and take the real part.

Figure 44 shows the input image, the notch filter and the filtered image.

This algorithm can be implemented as a function with four adjustable parameters to improve the final filtered image. The first parameter is the size of the image segments, which are $n \times n$ in size, where n is the first parameter. Care should be taken when choosing n as it greatly

Figure 44 – Notch filtering. (a) Original image. (b) Notch filter. (c) Filtered image.



Source: The author (2022)

impacts the performance of the program. For our images, $n = 128$ was found to be suitable. It is important to avoid attenuating any frequencies that are too close to the DC peak (center of the spectrum) as this can cause distortion in the filtered image. To achieve this, the second parameter is an exclusion radius from the center of the modified power spectrum, any peak within this radius will be ignored. The third parameter is a threshold used to set small Fourier components to zero. Any value below this threshold will be set to zero. The fourth parameter is the radius of the circular notch areas. Finally, it is important to note that due to a resizing of the modified power spectrum in step 3 of the algorithm, it is desirable for the input image to be square shaped.

3 EXPERIMENT

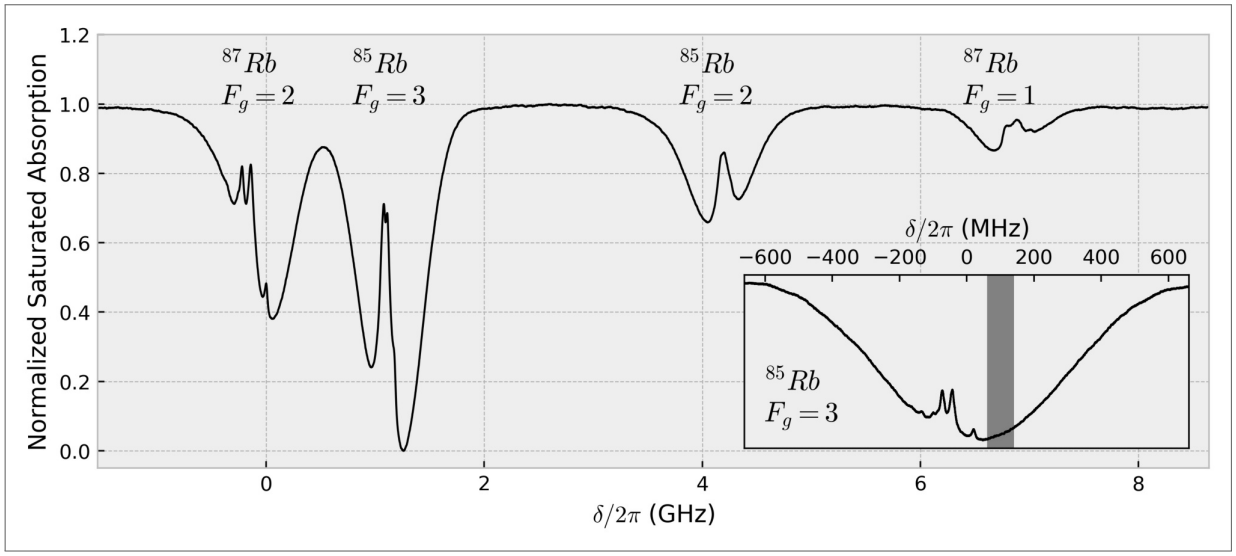
In this chapter we discuss our experimental setup and the methods used to obtain the images of the intensity profile of the beams.

3.1 EXPERIMENTAL SETUP

For the FWM experiment we used a diode laser from Sanyo, model DL7140-201S with a homemade electronic box for temperature and current control.

For the purpose of controlling the laser frequency and having a frequency reference, we use a saturated absorption (SA) experiment. In the saturated absorption we have two counter-propagating beams of same frequency, a strong one called *pump* and a weak one called *probe*, and they interact with atoms of rubidium inside a vapor cell in our system. When the laser frequency does no match the hyperfine levels, both beams interact with different group of atoms resulting in an absorption spectra equal to the ordinary absorption. When the frequency is equal to one of the hyperfine levels, both beams interact with the same group of atoms, therefore the probe beam is less absorbed resulting in a dip in the absorption spectra. Figure 45 shows the absorption spectra of rubidium. We see the four Doppler lines corresponding to the

Figure 45 – Saturated absorption of Rb. The inset graph is a zoom of the transition $5S_{1/2}, F_g = 3 \rightarrow 5P_{3/2}$ of ^{85}Rb. The region in gray is the region of frequency where all of our experiments were performed.

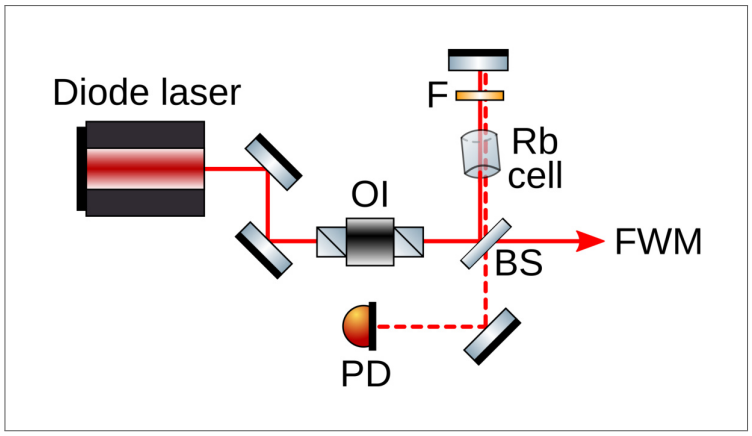


Source: The author (2022)

two isotopes of Rubidium: ^{85}Rb e ^{87}Rb. In each Doppler line there are many dips induced by the saturation absorption, but only half of them correspond to hyperfine levels. When the laser frequency is in the middle of two hyperfine transitions, an additional peak, called crossover, will appear between the hyperfine peaks in the absorption spectrum. For our measurements the laser frequency was tuned to $5S_{1/2}(F = 3) \longrightarrow 5P_{3/2}$ of ^{85}Rb.

Our experimental saturated absorption experiment setup is shown in figure 46. A small portion of the laser beam is reflected by a beam-splitter (BS), it passes through a Rb cell, then it is attenuated by a filter (F) and is reflected back through the Rb cell to a photo-detector (PD). The attenuated beam that goes to the photo-detector is the probe beam and the beam that first entered the cell is the pump beam. We use a 60 dB optical isolator (OI) from Isowave to prevent reflected light from reaching the diode.

Figure 46 – Scheme of a saturated absorption experiment



Source: The author (2022)

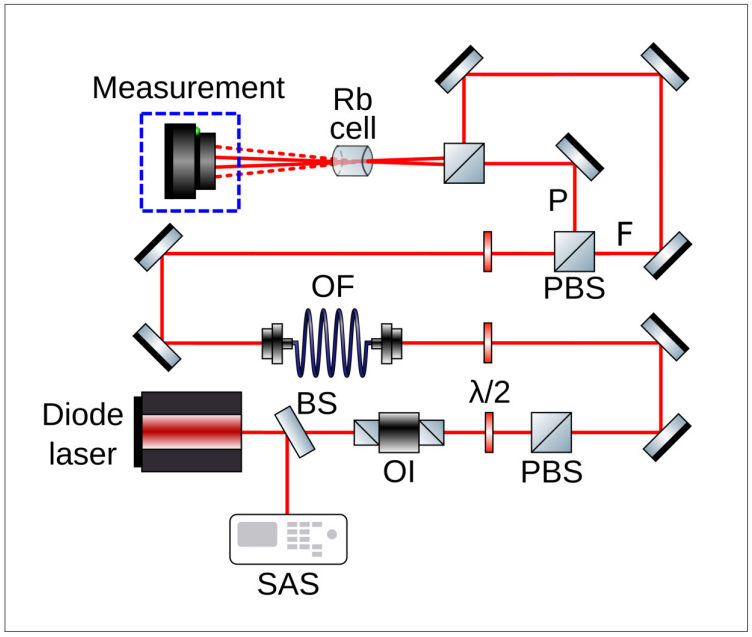
The frequency locking of the laser was achieved by manually finding the desired region in the saturated absorption spectrum and then turning off the amplitude of the ramp used to scan the frequency of the diode laser. The region we used for our experiments is approximately 100 ± 40 MHz above the $5S_{1/2}, F_g = 3 \rightarrow 5P_{3/2}, F_e = 4$ transition of ^{85}Rb, represented the the gray region on the inset graph in figure 45.

3.2 FOUR-WAVE MIXING SETUP

Only a fraction of the laser beam power is necessary for the saturated absorption, the remaining power goes to the FWM experiment. In figure 47 we have the simplified scheme of our setup. This beam is guided by a set of optical components until it reaches a single-mode optical fiber (OF in the diagram). We use an optical fiber in order to have an almost Gaussian profile of the beam at the fiber's exit. Figure 48 shows a comparison between the laser beam profile before and after going through the fiber.

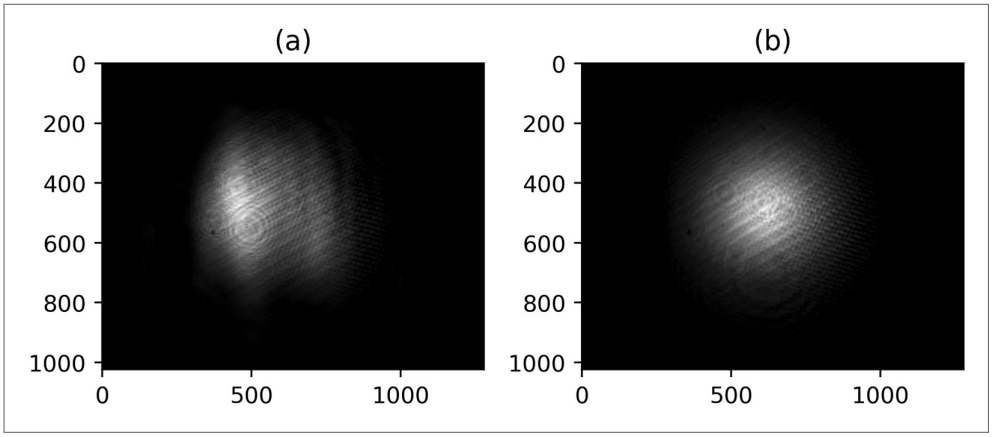
After the optical fiber, the laser beam is then divided into two, one that is reflected at the PBS, T_P, and the other that is transmitted, T_F. During the experiments we tried to mantain both beams, T_P and T_F, with almost the same intensity. The transmitted beams, T_P and T_F, with wave-vectors $\vec{k}_P$ and $\vec{k}_F$, respectively, and with an orthogonal linear polarization, cross paths inside a cell containing a natural concentration of rubidium atoms. At room temperature, $\sim 24^oC$, the atomic density is low, about $10^{10} at/cm^3$. In order to increase the density of

Figure 47 – Simplified scheme of the FWM experimental setup



Source: The author (2022)

Figure 48 – Comparison of the beam profile before and after the optical fiber. (a) Before. (b) After



Source: The author (2022)

rubidium atoms, we heated the cell to $\sim 72^oC$. We can determine the atomic density of rubidium at different temperatures via the expression (30),

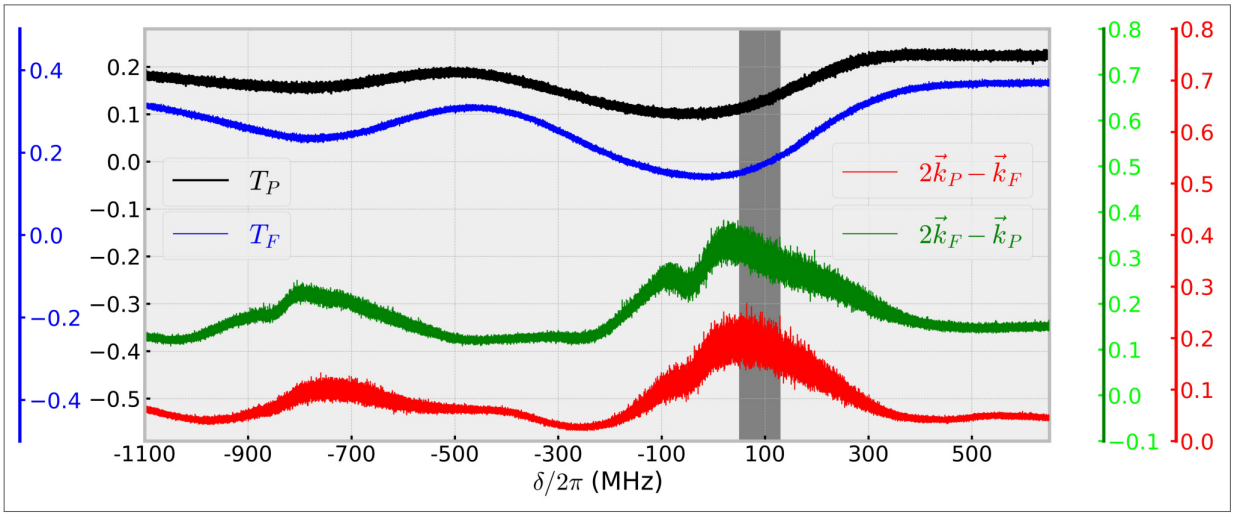
$$\log_{10}(N) = -\frac{4032}{T} - 2.5\log_{10}(T) + 30.707, \tag{3.1}$$

where N is the atomic density in at/cm^3 and T is the temperature in Kelvin. For a temperature of $\sim 72^oC$ we have atomic density of about $8 \times 10^{11} at/cm^3$.

The two beams, T_P and T_F, overlap in the middle of the cell, and due to the interaction with the atomic medium, two beams at directions $2\vec{k}_F - \vec{k}_P$ and $2\vec{k}_P - \vec{k}_F$ are generated. In figure 49 we show the Doppler absorption curve of the two transmissions and the intensity of the two FWM signals as a function of the laser detuning $(\delta/2\pi)$. The gray region represents

where we perform our experiments, it is in this region where we have the strongest FWM signals.

Figure 49 – FWM signals and transmissions



Source: The author (2022)

3.3 MEASUREMENTS

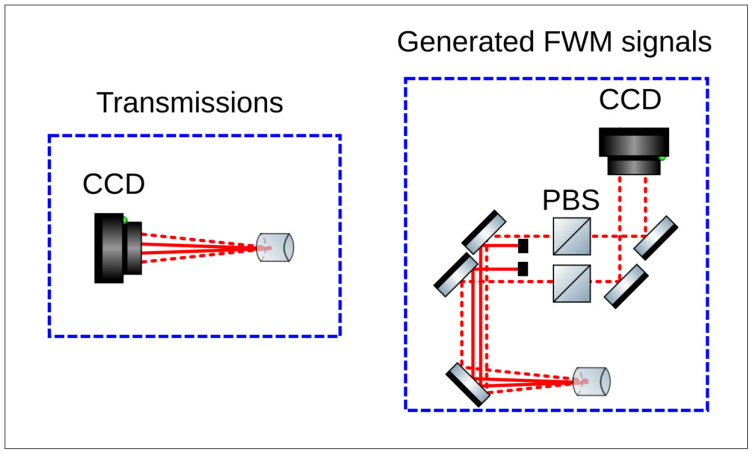
We have four beams coming out of the cell, the two transmitted beams, T_P and T_F, and two FWM signals at directions $2\vec{k}_P - \vec{k}_F$ and $2\vec{k}_F - \vec{k}_P$. These generated beams have linear orthogonal polarizations due to the fact that the incident beams also have linear orthogonal polarizations. The nonlinear signals are very weak and there are a lot of scattered light from the other beams. The linear orthogonal polarization allows us to clean part of the scattered light, using PBS on each path.

To capture images of the intensity profile of these beams we use a CCD camera model DCC1545M - GL - CMOS with 1280 x 1024 pixels of resolution, monochrome sensor from Thorlabs. We had only one of these CCD, hence we could only take images either of the transmitted beams or the generated FWM signals at one time. At the beginning of our measurements, we took pictures of the beams, but then we started recording videos, each video with 10 frames. This change was made in order to average the results of our analysis over all frames of each video. Most of our experiments were done with the beam diameters at the entrance of the cell of 1.4 mm, but a few were done with the diameters of order of 2.2 mm.

For the transmitted beams measurements, we put the CCD right after the cell. For each set of parameters, we took videos of both transmitted beams at the same time, and separately each at once. In this way, for each measurement we took three videos. For the generated FWM signals we had to put the CCD further from the cell because the signals are very close to the transmitted beams. To separate the nonlinear signals from the transmitted beams and the

scattered lights, we use mirrors, PBS and irises. These irises also helped in the alignment of the CCD. In figure 50 we show a simplified scheme for the measurements of the transmitted beams and FWM generated signals.

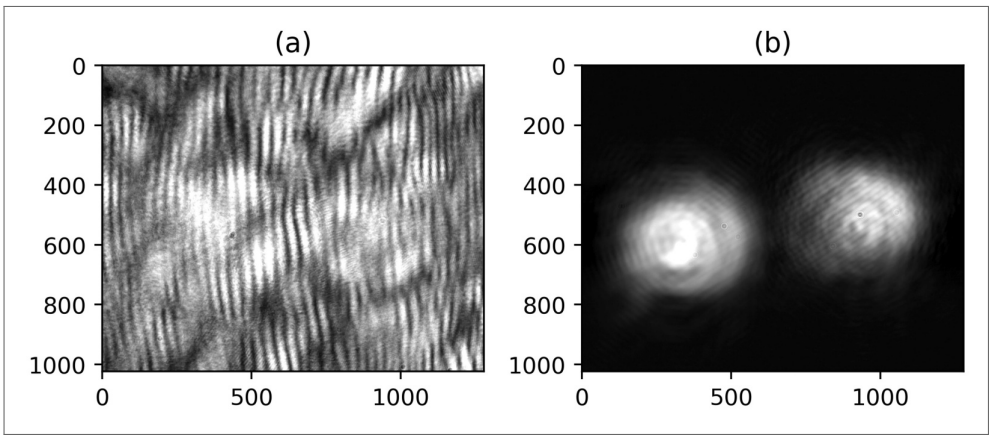
Figure 50 – Simplified scheme for taking images of the transmissions and generated FWM signals



Source: The author (2022)

The use of irises and PBSs to clean the FWM signals was crucial, without it would not be possible to do any analysis on the resulting images. Figure 51 shows the intensity profile of the FWM signal without these irises and PBSs and with these irises and PBSs. We can still

Figure 51 – (a) Without the irises and PBSs. (b) With the irises and PBSs



Source: The author (2022)

see some interference patterns in figure 51(b), to remove them we use the interference filter developed in subsection 2.4.6.

4 RESULTS AND DISCUSSION

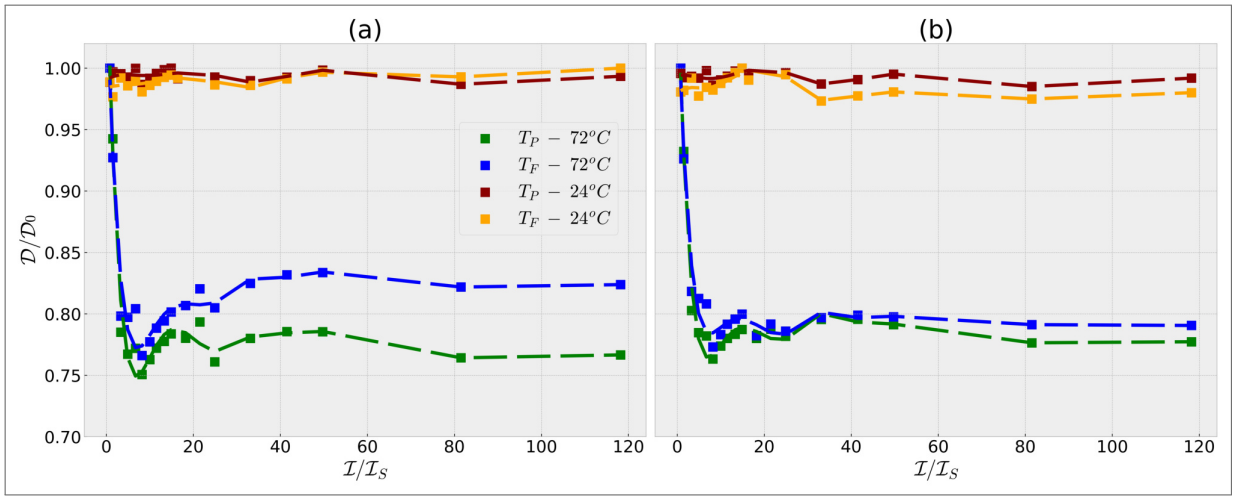
In this chapter we present and discuss our main results and how we analyze them. We also discuss some effects that can be associated with our observations. The chapter was divided in two sections. In the first one we discuss only the results associated to the transmitted beams; and in the second part we extend our analysis to the images for the FWM signals.

4.1 TRANSMISSIONS

4.1.1 Diameters

The first analysis comprised the investigation of the variation of the diameters of the transmitted beams. We looked at the diameters of the transmited beams with the cell's temperature at 24^oC and 72^oC for different intensities. Figure 52 shows the behavior for both horizontal and vertical directions. Each point in this figure is an average of multiple images

Figure 52 – Diameters comparison of the transmitted beams at 24^oC and 72^oC for varying intensities. (a) Diameters in the horizontal direction. (b) Diameters in the vertical direction.



Source: The author (2022)

and the error bars, only shown when they are bigger than the point marker, are the standard deviation. The dashed lines serve as visual aids to demonstrate the general trend of the data. On this graph the Y-axis is $\mathcal{D}/\mathcal{D}_0$ where $\mathcal{D}$ is the diameter in μm and $\mathcal{D}_0$ is the diameter of the transmitted beam with smallest intensity also in μm, this means that we are looking at changes in diameter instead of the absolute value. On the X-axis we have $\mathcal{I}/\mathcal{I}_S$ where $\mathcal{I}$ is the beam intensity before entering the Rb cell in mW/cm^2 and $\mathcal{I}_S$ is the saturation intensity also in mW/cm^2, given by

$$\mathcal{I}_S = \frac{c\epsilon_0\Gamma^2\hbar^2}{4|\hat{\epsilon}\cdot\hat{d}|^2}, \tag{4.1}$$

where Γ is the natural decay rate of the excited state, $\hat{\epsilon}$ is the unit polarization vector of the light field and $\hat{d}$ is the atomic dipole moment. We use the saturation intensity of isotropic light at the cycling transition which has value $\sim 3.9\ mW/cm^2$ (31). Ideally, $\mathcal{I}$ should be the intensity at the center of the cell where the transmitted beams cross paths, but we were unable to calculate the intensity at that position. In theory, the intensity of light at position z inside the cell can be found by the *Beer-Lambert Law* (32),

$$\mathcal{I}(z) = \mathcal{I}_0 e^{\alpha(\nu,T)z}, \tag{4.2}$$

where $\mathcal{I}_0$ is the intensity at the entrance and α is the absorption coefficient which is dependent on the frequency, ν, of the incident light and the temperature, T, of the medium. The absorption coefficient can be written as,

$$\alpha = \mathcal{N}\sigma, \tag{4.3}$$

where $\mathcal{N}$ is the number density of the atomic gas and σ is the microscopic atomic absorption cross-section. Finally, σ can be written as,

$$\sigma = \frac{\sigma_0}{1 + 4(\Delta/\Gamma)^2 + (\mathcal{I}_0/\mathcal{I}_S)}, \tag{4.4}$$

where Δ is the detuning of the laser, Γ is the natural decay rate of the excited state and σ_0 is the on-resonance scattering cross-section given by

$$\sigma_0 = \frac{\hbar\omega\Gamma}{2\mathcal{I}_S}.$$

The absorption coefficient has dependencies on temperature, frequency and density which makes determining it very challenging, in our case, all our attempts were unsuccessful.

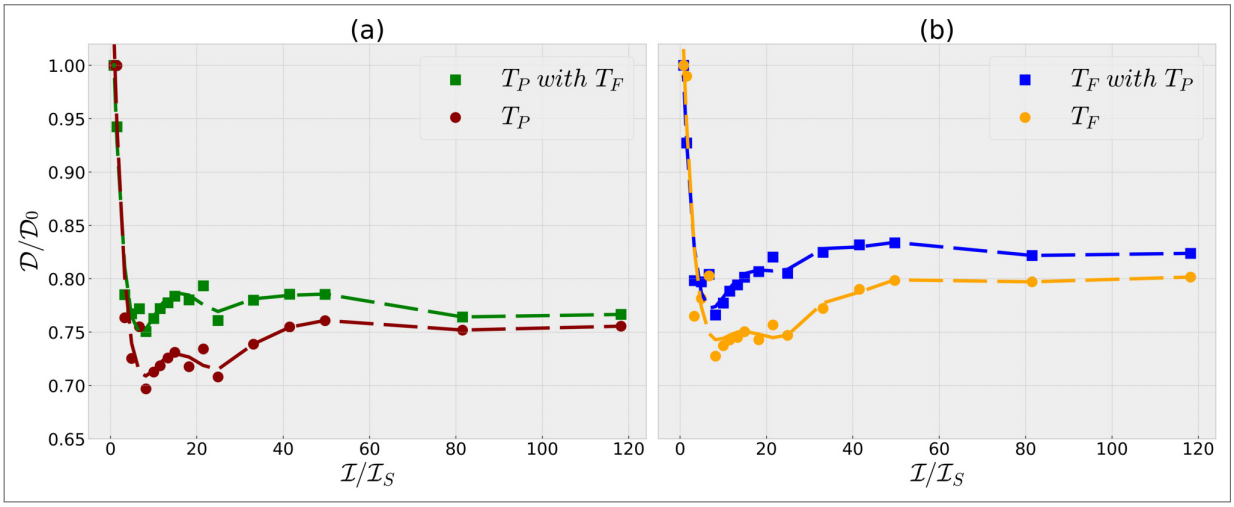
We can clearly see in figure 52 that at 24^oC the diameters are almost constant in all range of intensities for both horizontal and vertical directions. On the other hand, at 72^oC we can see a great variation in the diameter values for small intensities, a decrease of about 20 to 25% in the diameter value. As the intensity increases the diameters become approximately constant. In other words, for small intensities we have a self-focusing effect due to changes in the refractive index of the medium. Due to the electric field of the light wave, changes in the refractive index occur. For high intensity light beams, the refractive index changes according to

$$\Delta n = n_2 I, \tag{4.5}$$

where n_2 is the *nonlinear refractive index* and I is the intensity. The nonlinear refractive index is related to the third order susceptibility $\chi^{(3)}$ and the effect on the beam depends on whether $\chi^{(3)}$ is positive or negative. Positive $\chi^{(3)}$ causes the *self-focusing* of the beam and negative $\chi^{(3)}$ causes the *self-defocusing* (33).

We also investigated how the presence of one the transmitted beams affects the diameter of the other. For this we took videos when: (i) both transmitted beams crossing the cell at the same time, and (ii) each beam crosses the cell singly.

Figure 53 – Comparison of the transmitted beams when they are isolated and when they are together. (a) Diameters in the horizontal direction. (b) Diameters in the vertical direction.



Source: The author (2022)

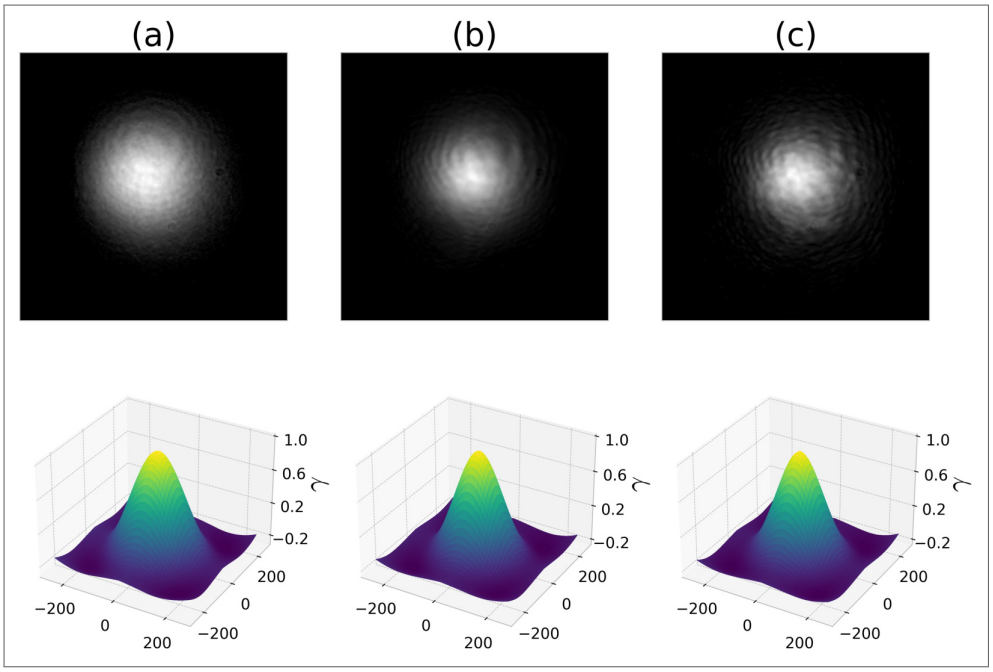
The results are shown in figure 53, where we can clearly see that the decrease in diameter is larger for the isolated beams. For example, in figure 53(a), the isolated beam has a maximum diameter variation of approximately 30%, while the maximum diameter variation when both beams are present is approximately 25%. Again, the dashed lines serve as visual aids to demonstrate the general trend of the data. We can understand this result if we consider that when both beams are present the self-focusing effect appears to be distributed for both of them.

In summary, we observe a decrease in the diameters of the transmitted beams for smaller intensities and as the intensity increases the diameters becomes approximately constant. This effect is greater when the beams are isolated.

4.1.2 Autocorrelation/Cross-correlation of the images

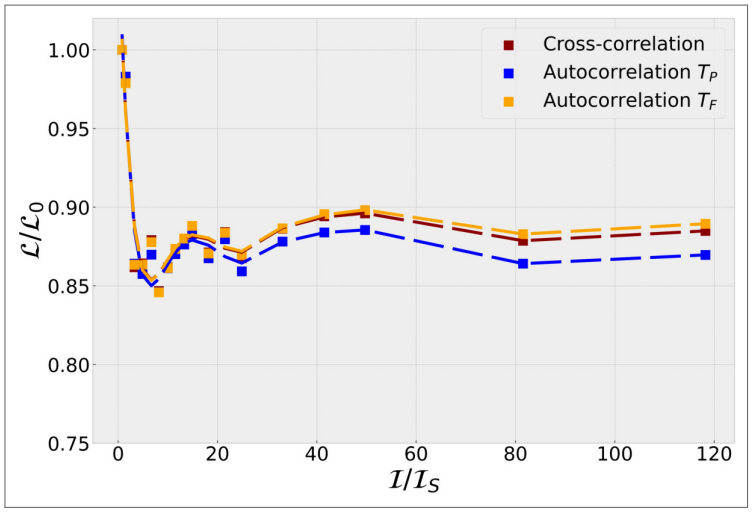
We also calculated the spatial cross-correlation and autocorrelation of the intensity profile images of the transmitted beams. Figure 54 shows some examples of intensity profile images of the transmitted beams and its respective autocorrelations in a 3D view, for three different intensity: (a) 119 $\mathcal{I}_S$, (b) 22 $\mathcal{I}_S$ and (c) 1 $\mathcal{I}_S$. Although, the pictures seem to indicate some change in the intensity distribution as the intensity increases, the autocorrelations are very similar, with a miximum equal to 1 at the center of the image and with a shape similar to a Gaussian. To compare these autocorrelations/cross-correlations, we take the full width at half maximum of the autocorrelations and cross-correlations for different intensities, we see that the widths present the same behavior as the obtained for the diameters, shown in figure 55. The dashed lines serve as visual aids to demonstrate the general trend of the data. For small intensities we have a decrease in the FWHM and as the intensity increases the FWHM

Figure 54 – Intensity profile of the transmissions and its spatial autocorrelations. (a) 1 $\mathcal{I}_S$. (b) 22 $\mathcal{I}_S$. (c) 119 $\mathcal{I}_S$.



Source: The author (2022)

Figure 55 – FWHM of the cross-correlations and autocorrelations



Source: The author (2022)

becomes approximately constant. We note that the variation here in the FHWM is about 15%, smaller than the variation observed in the diameters, approximately 25%.

We understand that the variation in the FWHM of the autocorrelation/cross-correlation and in the diameters of the images are manifestations of the nonlinear behavior of the refractive index for high intensity fields, usually denominated as Kerr effect.

4.1.3 Autocorrelation of the intensity fluctuations

In the last decades, there has been an interest in studying light fluctuations when it interacts with matter. In particular, to obtain spectroscopic information using the intensity fluctuations of a laser beam interacting with atomic vapor. The root of this phenomenon is the conversion of phase-noise to amplitude-noise due to the light-matter interaction, with interesting results such as the study of correlations and anti-correlations in electromagnetically induced transparency, controlling intensity noise correlations and squeezing of four-wave mixing processes via polarization (34) (35) (36), and the generation of correlated and anticorrelated fields via atomic spin coherence (37).

In this direction some results were obtained with our experimental system, by Alexandre Almeida, measuring the intensity fluctuations in time. A high correlation was observed for the transmitted beams, and also for the two generated nonlinear signals. Preliminar results are described in (38).

In the work described here, we detected the images using a CCD camera that has a long integration time. This means that this type of fluctuation is blurred and no information due to conversion phase-noise to amplitude-noise is observed. However, we hope to see same contribution due to macroscopic fluctuations of the refraction index.

In this sense, we try to see if we can get some information about the atom-field interaction, or about the nonlinear process, looking for the autocorrelation of the intensity fluctuations.

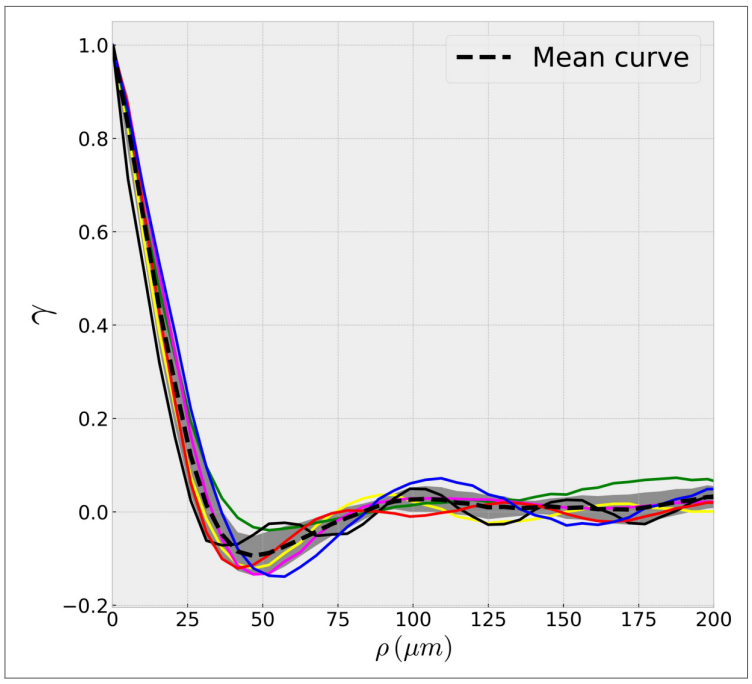
The first thing we need to do is to understand how the autocorrelation of the intensity fluctuations behave when the beam does not propagate through the medium, we will use this result as a base for comparison. We took several images of the beam without the cell and we calculated the autocorrelation of the intensity fluctuations for each image. The intensity fluctuations were obtained using the method described in subsection 2.4.3. In figure 56 we plot the autocorrelation, γ, obtained for each measurement as a function of the radial parameter, ρ. To convert pixel measurements to micrometers, we multiply by the size of each pixel, which is 5.2 μm. We see that the autocorrelation decreases rapidly and then oscillates around zero. We find the mean curve of this set of curves, which we represent by a dashed line with the gray region that surrounds it as the uncertainty.

We define a spatially transverse correlation length as the width, ρ_c, of the autocorrelation of intensity fluctuations such as

$$\gamma(\rho_c) = 1/2.$$

Simply put, it is the width of the curve at $1/2$. When the beam does not propagate through the medium we have a correlation length of $14 \pm 3\ \mu m$. To calculate the uncertainty in the transverse correlation length, we take into account the fact that each point in the autocorrelation of intensity fluctuations is the average of ten values, and each of these ten values has its own uncertainty. This is because each curve represents the average of ten images. We assume that each value has an uncertainty of $\pm 2.5\ \mu m$. The total uncertainty for each average point is

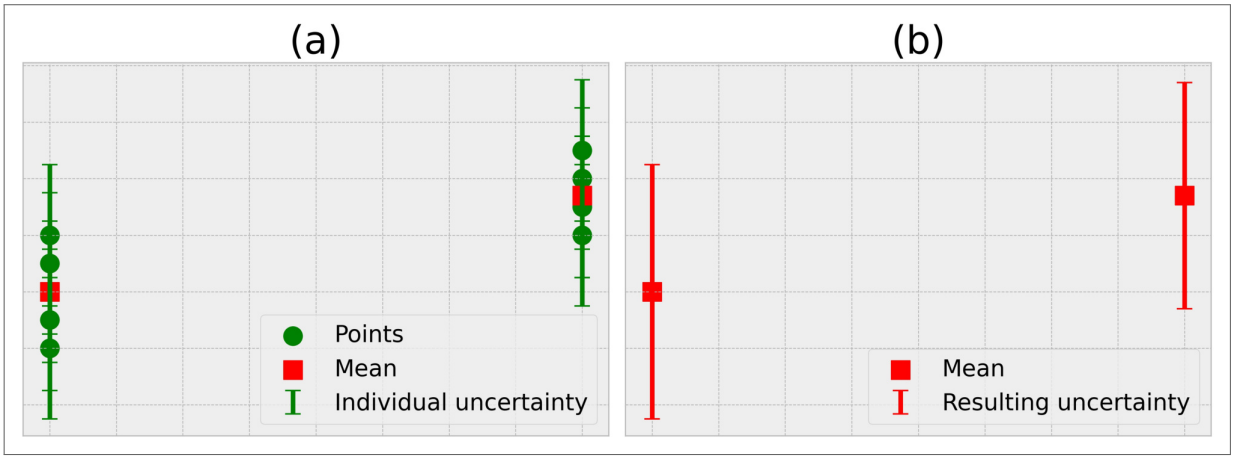
Figure 56 – Autocorrelation of the intensity fluctuations of a beam without a cell.



Source: The author (2022)

then taken as the distance from the top of the uncertainty of the highest point to the bottom of the uncertainty of the lowest point of the set of ten points used to calculate the average. In figure 57 we have a visual representation of this process. In figure 57 (a) we present the points, its uncertainties and the mean values. In figure 57 (b) we present the mean values and the resulting uncertainties.

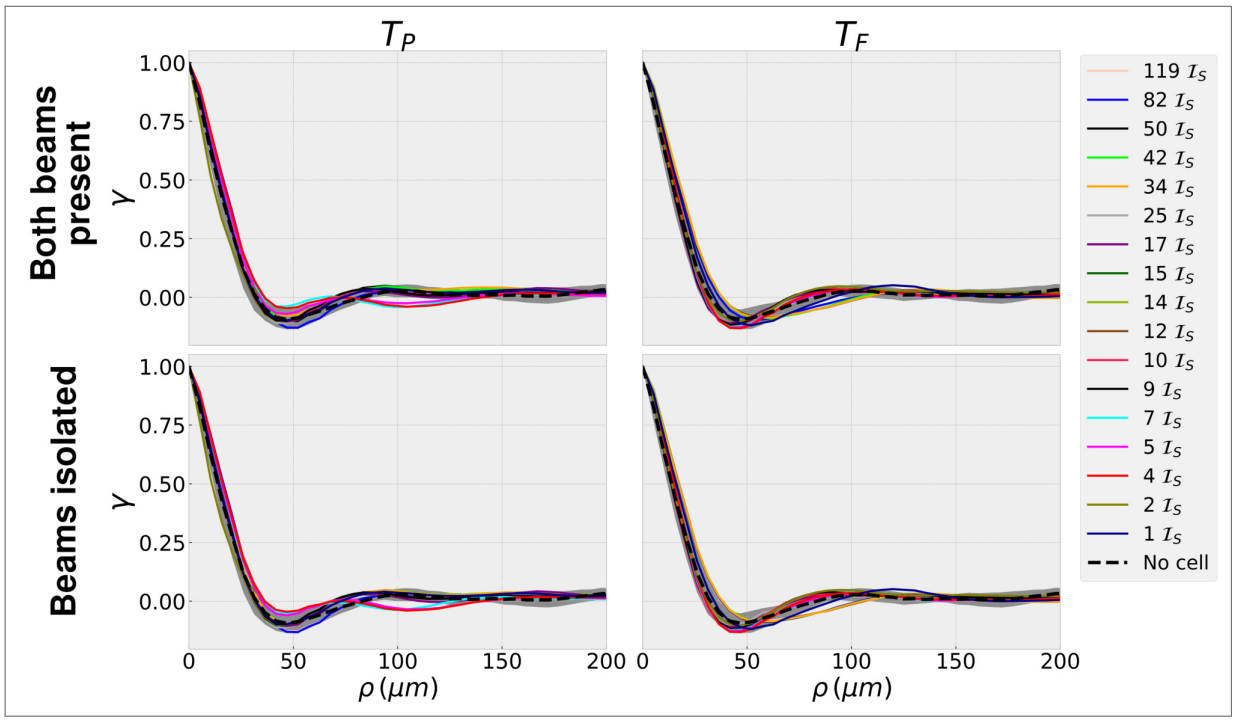
Figure 57 – Uncertainties



Source: The author (2022)

Let us start our analysis by looking at how the autocorrelations of the fluctuations behave when the cell is at a temperature of 24^oC for different intensities. We calculated the autocorrelation of the intensity fluctuations for two different settings. First, when both beams are

Figure 58 – Autocorrelations of the fluctuations for the transmissions at cell temperature 24^oC. On the first row both transmitted beams are present. Second row the transmitted beams are isolated.

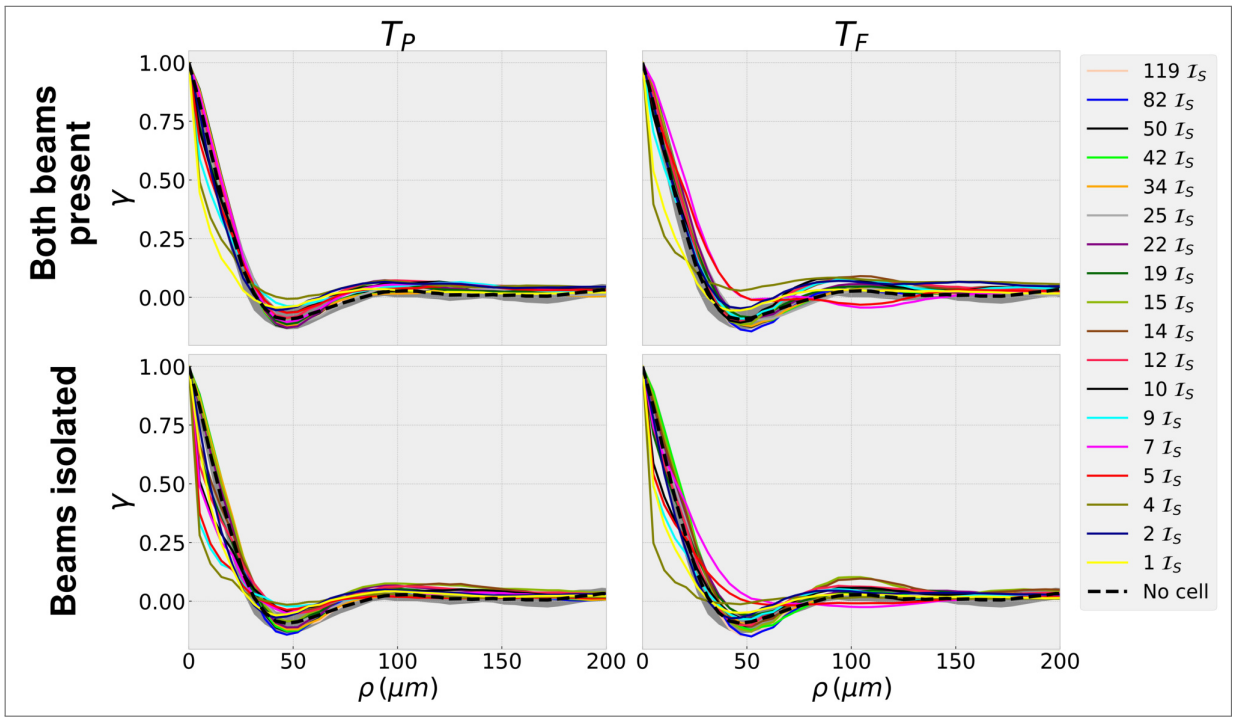


Source: The author (2022)

present, i.e., both transmitted beams crossing the cell at the same time; Second, the transmitted beams cross the cell individually. The purpose of having these two settings is to observe how one the transmitted beams affects the intensity fluctuations of the other. In the first row of figure 58, we have both beams present and in the second row we have the beams isolated. On the X-axis of each one of these plots we have the radius, ρ, in μm and on the Y-axis we have the autocorrelation, γ. In each plot we have multiple curves being plotted, each curve corresponds to a different intensity. Firstly, we can note that all the curves are inside the gray region which is the uncertainty of the 'No cell' curve, indicating that the introduction of the medium at this temperature seems to not have affected the intensity fluctuations. Another thing that we can point out, by looking at figure 58's first and second row, is that the presence of the other beam seems to have no effect on the fluctuations of the other at this temperature.

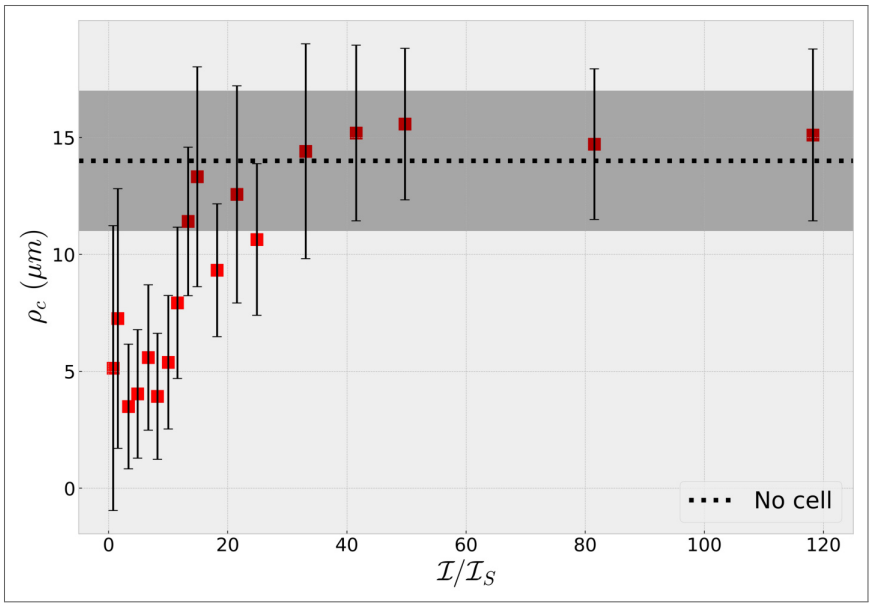
Now, we repeat the analysis with the cell's temperature at 72^oC, shown in figure 59. At this temperature we notice that for smaller intensities we have a decrease in correlation length and as we increase the intensity the curves behave as the 'No cell' curve. We can also notice that the intensity fluctuations of one beam is affected by the presence of the other one, when the beams are isolated the decrease in correlation length seems to be larger. A better way to visualize this decrease in correlation length is to look at the value of the correlation length as a function of intensity, shown in figure 60. In this figure the dotted line represents the correlation length for the beam when it does not propagate through the medium and the gray region represents the uncertainty of this value. We can clearly see the decrease in the

Figure 59 – Autocorrelations of the fluctuations for the transmissions at cell temperature 72^oC. On the first row both transmitted beams are present. Second row the transmitted beams are isolated.



Source: The author (2022)

Figure 60 – Correlation length for the transmitted beams



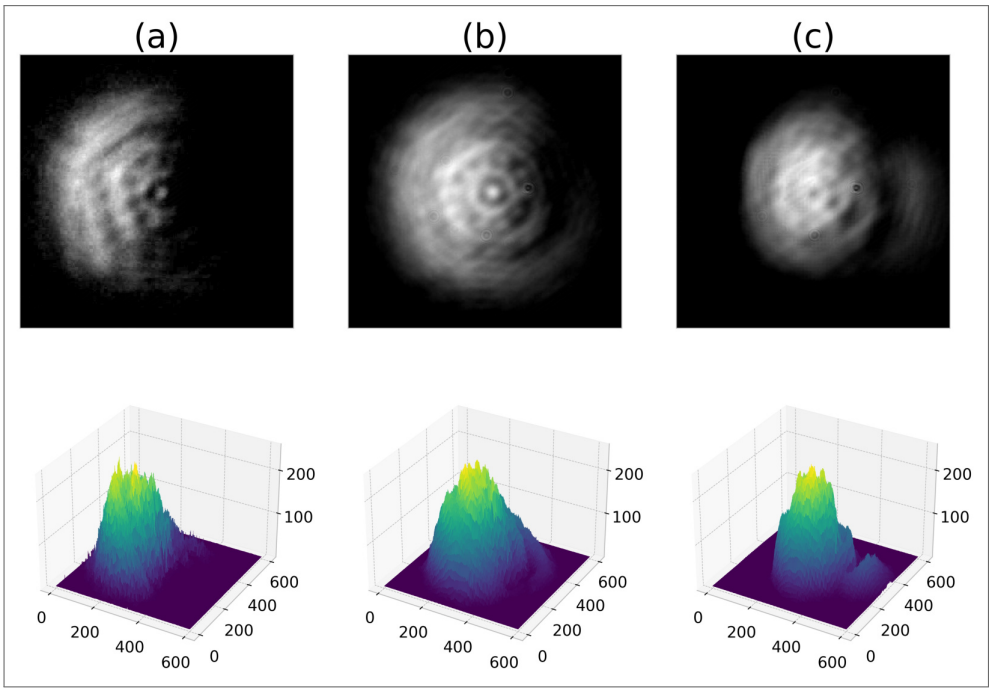
Source: The author (2022)

correlation as the intensity decreases.

4.2 FWM SIGNALS

Now, we will analyse the FWM signals. The images of these nonlinear signals at the camera and in 3D for three intensities are shown in figure 61. At first, we need to point out that we

Figure 61 – FWM generated signals. On the first row we have the images and on the second one we have these images in 3D. (a) 11 $\mathcal{I}_S$. (b) 24 $\mathcal{I}_S$. (c) 109 $\mathcal{I}_S$.



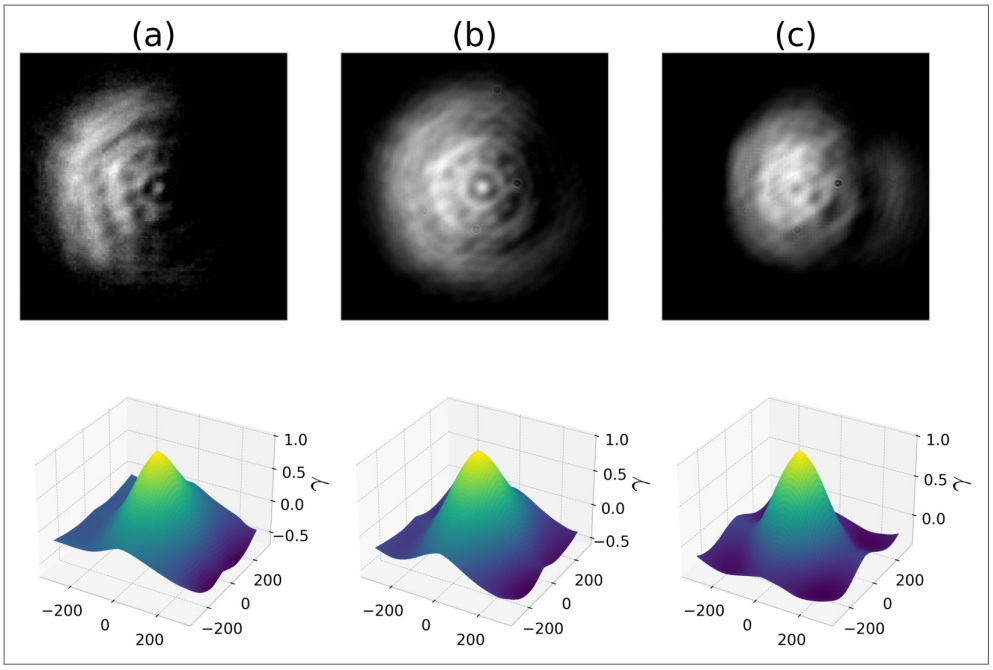
Source: The author (2022)

can only see these FWM signals for high intensities, we can see in the first image that the signal is very weak (only half of the signal can be seen). Another thing to point out is that the spatial profile of these beams is distorted. First, we study the correlation/autocorrelation of these images.

4.2.1 Correlation of the images

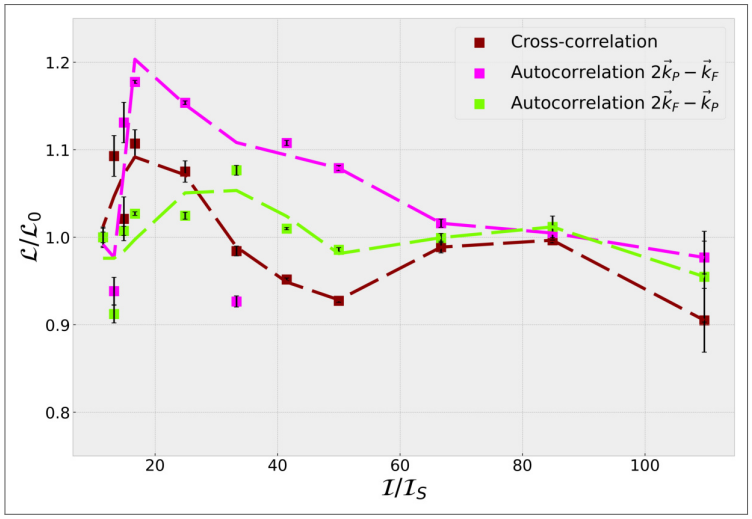
In the first row of figure 62 we have the same intensity profiles of the FWM signals showed in figure 61, and in the second row its respective autocorrelations. We will repeat the analysis that we performed for the transmitted beams. Figure 63 shows the full width at half maximum of the autocorrelation and cross-correlation for different intensities. The dashed lines serve as visual aids to demonstrate the general trend of the data. We can see a 20% variation in the FHWM for smaller intensities, but no clear pattern. This variation occurs in the region of intensities where the signal is very weak and therefore with great uncertainty. In this way, it is difficult to get some conclusion from these results. Instead, we are going to investigate the fluctuations of intensity of the FWM signals.

Figure 62 – Intensity profile of the FWM signals and its spatial autocorrelations. (a) 11 $\mathcal{I}_S$. (b) 24 $\mathcal{I}_S$. (c) 109 $\mathcal{I}_S$.



Source: The author (2022)

Figure 63 – FWHM of the cross-correlations and autocorrelations

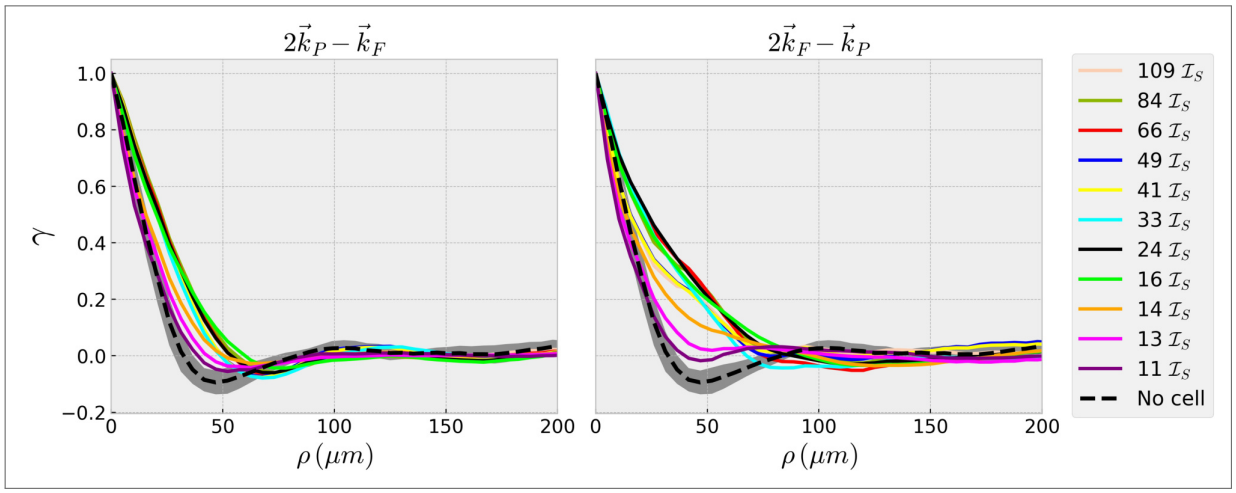


Source: The author (2022)

4.2.2 Autocorrelation of the intensity fluctuations

In figure 64 we plot the autocorrelations for the intensity fluctuations of the FWM signals for a large range of intensities, (a) $2\vec{k}_P - \vec{k}_F$ and (b) $2\vec{k}_F - \vec{k}_P$. We also present a dotted line that correspond to the 'No cell' curve, together the uncertainty, gray region. It is clear that we have an increase in the correlation length for higher intensities and as we decrease the intensity, the curves behave like the 'No cell' curve. A better way to visualize this increase in

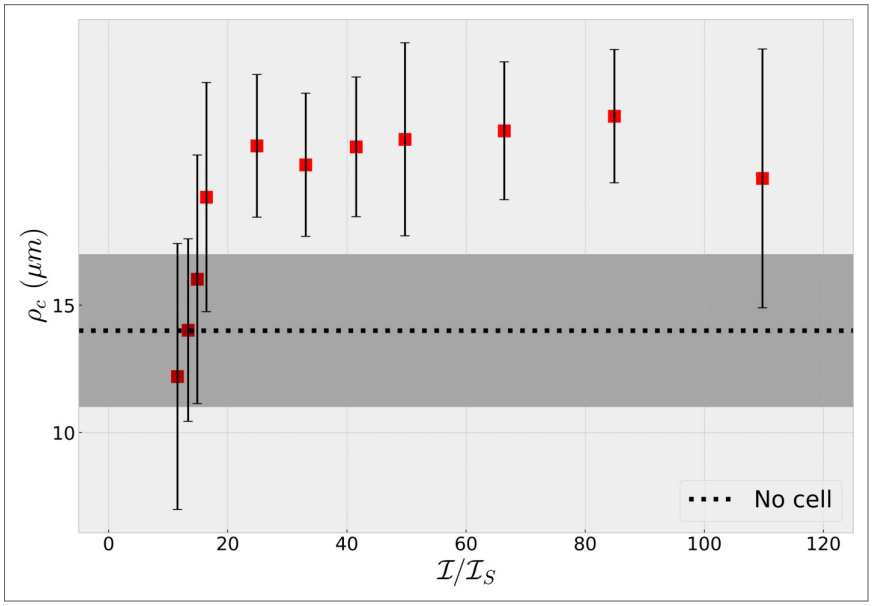
Figure 64 – Autocorrelation of intensity fluctuations for the FWM signals. (a) $2\vec{k}_P - \vec{k}_F$. (b) $2\vec{k}_F - \vec{k}_P$.



Source: The author (2022)

correlation length is by plotting the correlation length as a function of the intensity as shown in figure 65. In this figure the dotted line represents the correlation length for the beam without

Figure 65 – Correlation length of the FWM signals.



Source: The author (2022)

the medium and the gray region represents the uncertainty. We can clearly see the increase in the correlation as the intensity increases.

4.2.3 A simple theoretical analysis

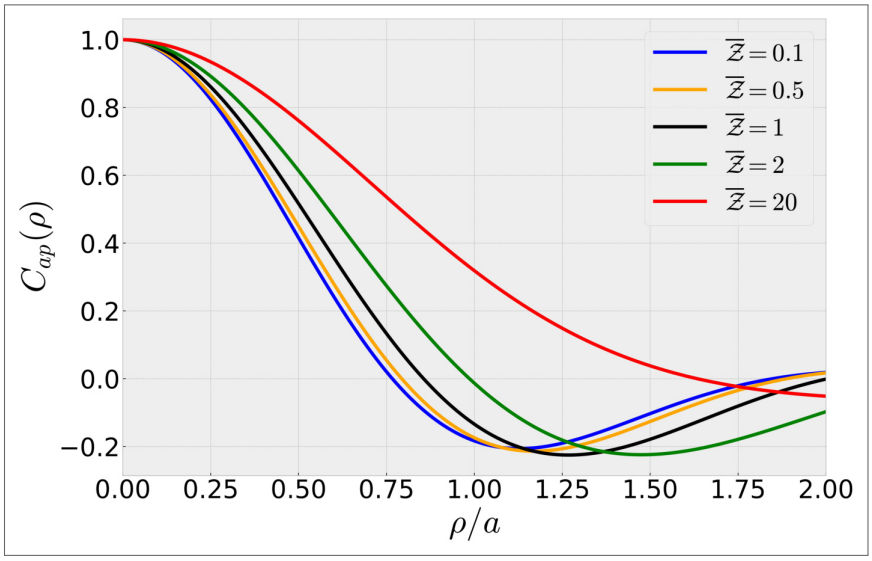
Looking at the literature on autocorrelation of intensity fluctuations, we found an interesting discussion on the autocorrelation of intensity fluctuations of a beam propagating in a

medium that fluctuates randomly in space only, not in time, very weakly and very slowly compared with the wavelength of light (39). These autocorrelations have different forms depending on whether work with a plane wave or a spherical wave. For a plane wave, the authors find a compact expression for the autocorrelation, that can be written as:

$$C_{ap}(\rho) = \frac{\overline{\mathcal{Z}} exp(-\rho^2/a^2) - \mathrm{Im} E_i\left(-\frac{\rho^2/a^2}{1-j\overline{\mathcal{Z}}}\right)}{\overline{\mathcal{Z}} - tan^{-1}\overline{\mathcal{Z}}}, \tag{4.6}$$

where $\overline{\mathcal{Z}}$ is called the wave parameter, a is the correlation length and E_i is the integral exponential function. Figure 66 shows the autocorrelation curves obtained by equation 4.6 for different values of $\overline{\mathcal{Z}}$ and fixed $a = 20\mu m$. All these curves have a Gaussian like decay at

Figure 66 – Autocorrelation of intensity fluctuations, $C_{ap}(\rho)$, for different values of $\overline{\mathcal{Z}}$ and $a = 20\mu m$.

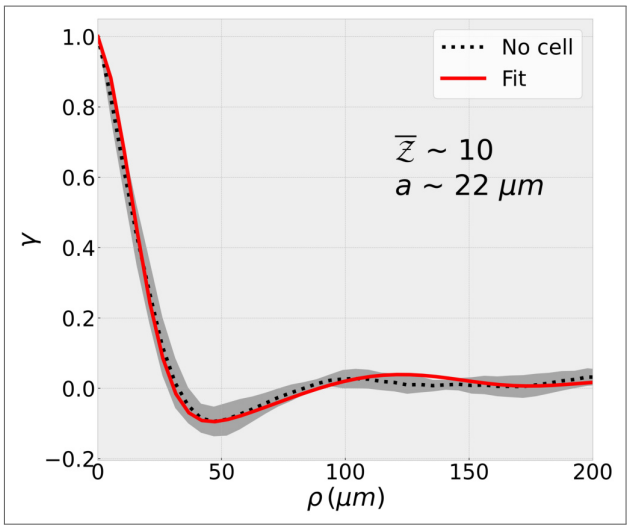


Source: The author (2022)

the beginning and then some oscilations around zero. These curves closely resemble what we found experimentally, so we will use expression 4.6 to try to fit our experimental curves.

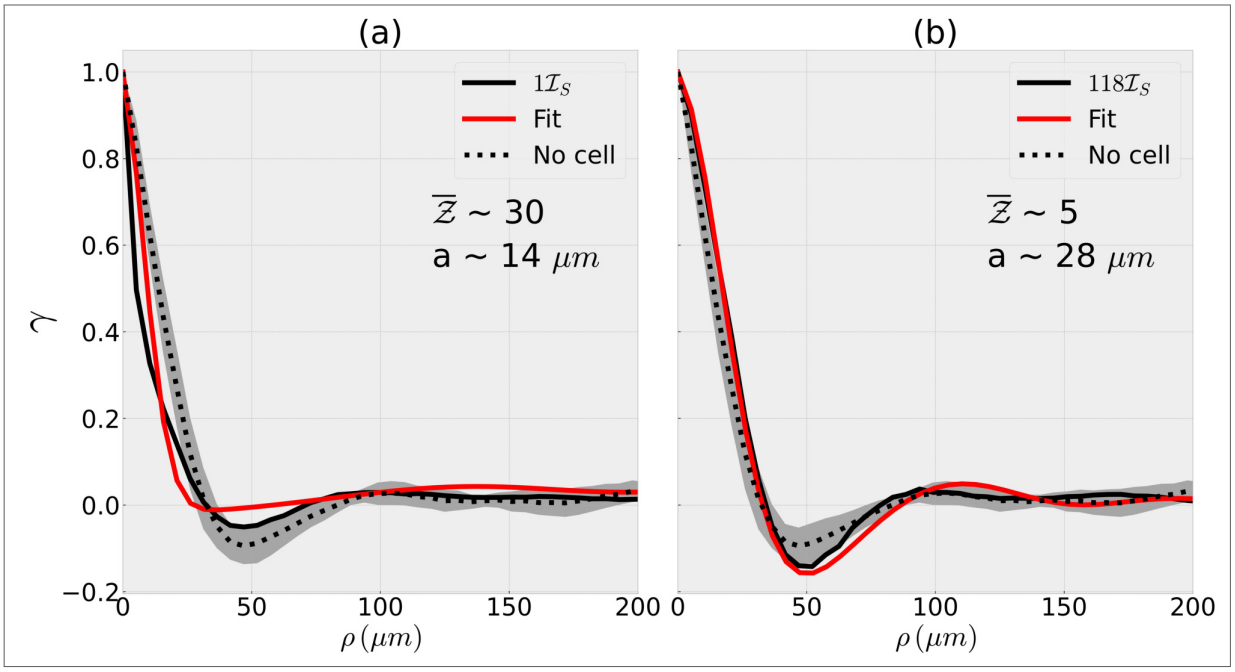
First, let us fit the autocorrelation of intensity fluctuations for the beam when it does not propagate through the medium, shown in figure 67. We see that expression 4.6 describes very well our experimental curve, both the initial decay and the oscilations. It is important to note that the parameter a is approximately equal to $2\rho_c$, where ρ_c is the correlation length that we defined in subsection 4.1.3. Now, let us try to fit the autocorrelation curves for transmitted beams when the medium is at 72^oC. For low intensities, figure 68(a), we see that the adjustment is not very precise. Expression 4.6 cannot adjust for initial steep drop of the experimental curve nor for the oscilations that follow. For the high intensity, figure 68 (b), a good agreement is obtained with our experimental curve. This is expected because the experimental curves for higher intensities are very similar to the curve of the beam when it does not propagate through the medium.

Figure 67 – Fit of the autocorrelation of intensity fluctuations for the beam when it does not propagate through the medium.



Source: The author (2022)

Figure 68 – Fit of the transmitted beams. (a) $1\mathcal{I}_S$. (b) $118\mathcal{I}_S$.



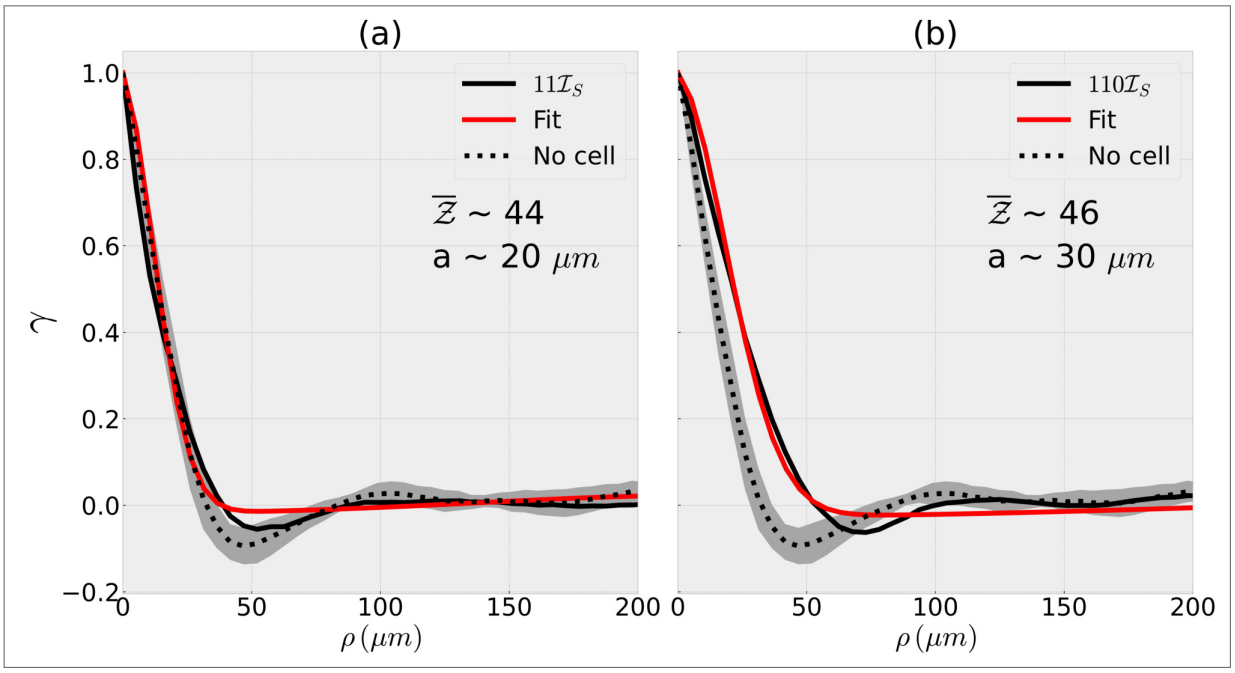
Source: The author (2022)

For the FWM signals, the fit of autocorrelation curves are shown in figure 69. We observe that the adjustment is not very good, mainly in the region of oscilations that follow the steep drop at the beginning.

These results indicate that we can have a good adjustment of the experimental autocorrelations of intensity fluctuations, when the experimental results are very similar to what we have obtained without the medium.

For the transmitted beams, we see that in the intensity region where the greatest refractive

Figure 69 – Fit of the FWM signals. (a) $11\mathcal{I}_S$. (b) $110\mathcal{I}_S$.



Source: The author (2022)

index variation should occur, when the diameters decreases, it corresponds to the region where the fit is worse. This seems to indicate that this simple theoretical model cannot describe the strong fluctuations in the refractive index caused by the high intensity of the incident beams. Our results indicate that these strong fluctuation in the refractive index, probably are responsible by the observed decrease in the correlation length. What is not clear for us is why in the FWM process, when the interaction light-matter is essential, the correlation length increase.

5 CONCLUSIONS

In this work we conducted an experimental study on the spatial intensity distribution of the two transmitted beams and the two beams generated in a four-wave mixing (FWM) process, using rubidium vapor as nonlinear medium.

From the analysis of the diameters of the transmitted beams, we observed a self-focusing effect due to changes in the refractive index of the medium for intensities close to the saturation intensity. As the intensity increases, the intensity profile of the transmitted beams do not change. We also calculated the cross-correlation and autocorrelation of the images of the transmitted beams for different intensities. We observe that the full width at half maximum of these autocorrelations and cross-correlations present the same behavior as the diameters. The same analysis for the nonlinear signals was performed, but we could not see any clear pattern.

We also analyzed the spatial intensity fluctuations of the transmitted and generated FWM signals. We defined a transverse spatial correlation length that we used to compare the behaviour of our beams with that of a free beam. For the transmitted beams, we observe a decrease in the transverse spatial correlation length in the same region of intensities where we observe the self-focusing effect. While for the generated FWM signals, we observe an increase in the transverse spatial correlation length as the intensity of the incident beams increases. These results indicate that the variations in the refractive index, which cause the self-focusing effect in the transmitted beams are also responsible for decrease/increase in the transverse spatial correlation length for the fluctuations of intensity, although it is not clear how this happens.

Finally, we looked at a simple theoretical model for the autocorrelation of the fluctuations of intensity found in reference (39). We use a closed expression given by the authors for the autocorrelation of intensity fluctuations to fit our experimental curves. A good adjustment of the experimental autocorrelations of intensity fluctuations was observed, when the experimental results were very similar to those obtained without the nonlinear medium. The fit gets worse in the regions of intensity where the fluctuations of intensity shows a significant increase/decrease in the transverse spatial correlation length.

A more realistic theoretical model need to be developed, taking into account the spatial profile - Gaussian - of the incident beams and the strong fluctuations of the refractive index induced by the high intensity beams.

REFERENCES

1 BOYD, R. W. Chapter 1 - the nonlinear optical susceptibility. In: BOYD, R. W. (Ed.). *Nonlinear Optics (Fourth Edition)*. Fourth edition. Academic Press, 2020. p. 1–64. ISBN 978-0-12-811002-7. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780128110027000102>.

2 FRANKEN, P. A.; HILL, A. E.; PETERS, C. W.; WEINREICH, G. Generation of optical harmonics. *Phys. Rev. Lett.*, American Physical Society, v. 7, p. 118–119, Aug 1961. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevLett.7.118>.

3 ALVAREZ, A. S.; ALMEIDA, A. A. C. de; VIANNA, S. S. Two symmetric four-wave mixing signals generated in a medium with anomalous refractive index. *Journal of Physics B: Atomic, Molecular and Optical Physics*, IOP Publishing, v. 54, n. 4, p. 045403, feb 2021. Disponível em: <https://dx.doi.org/10.1088/1361-6455/abe178>.

4 ALMEIDA, A. A. C. de. *SPECTRAL ANALYSIS OF THE FOUR-WAVE MIXING IN A COLD ATOMIC SAMPLE USING A SINGLE CW LASER*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2019.

5 GARCíA-WONG, A.; de Almeida, A.; de Melo, N.; VIANNA, S. Velocity-selective spectroscopy of rb vapor with a train of short pulses and a diode laser. *Optics Communications*, v. 427, p. 462–468, 2018. ISSN 0030-4018. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0030401818306047>.

6 MOTTA, M. R. L. da. *Two symmetric four-wave mixing signals induced by beams with nonuniform distributions*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2021.

7 MOTTA, M. R. L. da; ALMEIDA, A. A. C. de; VIANNA, S. S. Spatial distribution of two symmetric four-wave-mixing signals induced by gaussian beams. *Phys. Rev. A*, American Physical Society, v. 106, p. 053502, Nov 2022. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevA.106.053502>.

8 BAATZ, W. *Photography*. 1st. ed. [S.l.]: Barron's, 1997.

9 GONZALEZ, R.; WOODS, R. *Digital Image Processing*. Pearson/Prentice Hall, 2008. ISBN 9780131687288. Disponível em: <https://books.google.com.br/books?id=8uGOnjRGEzoC>.

10 IMAQ vision : IMAQ vision concepts manual. [S.l.]: National Instruments Corp, 2003.

11 ABDULRAHMAN, A.; VAROL, S. A review of image segmentation using matlab environment. In: *2020 8th International Symposium on Digital Forensics and Security (ISDFS)*. [S.l.: s.n.], 2020. p. 1–5.

12 SMITH, S. W. *The scientist and engineer's guide to digital signal processing*. [S.l.]: California Technical Pub, 1997.

13 YOUNG, I. T.; GERBRANDS, J. J.; VLIET, L. J. V. Fundamentals of image processing. Delft University of Technology, 1998.

14 BOVIK, A. C. 2.3 - basic tools for image fourier analysis. In: BOVIK, A. (Ed.). *Handbook of Image and Video Processing (Second Edition)*. Second edition. Burlington: Academic Press, 2005, (Communications, Networking and Multimedia). p. 57–72. ISBN 978-0-12-119792-6. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780121197926500681>.

15 JäHNE, B. *Digital Image processing*. 6th revised and extended ed.. ed. [S.l.]: Springer, 1997. ISBN 9783540240358; 3540240357.

16 VIRTANEN, P.; GOMMERS, R.; OLIPHANT, T. E.; HABERLAND, M.; REDDY, T.; COURNAPEAU, D.; BUROVSKI, E.; PETERSON, P.; WECKESSER, W.; BRIGHT, J.; van der Walt, S. J.; BRETT, M.; WILSON, J.; MILLMAN, K. J.; MAYOROV, N.; NELSON, A. R. J.; JONES, E.; KERN, R.; LARSON, E.; CAREY, C. J.; POLAT, İ.; FENG, Y.; MOORE, E. W.; VanderPlas, J.; LAXALDE, D.; PERKTOLD, J.; CIMRMAN, R.; HENRIKSEN, I.; QUINTERO, E. A.; HARRIS, C. R.; ARCHIBALD, A. M.; RIBEIRO, A. H.; PEDREGOSA, F.; van Mulbregt, P.; SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, v. 17, p. 261–272, 2020.

17 HARRIS, C. R.; MILLMAN, K. J.; WALT, S. J. van der; GOMMERS, R.; VIRTANEN, P.; COURNAPEAU, D.; WIESER, E.; TAYLOR, J.; BERG, S.; SMITH, N. J.; KERN, R.; PICUS, M.; HOYER, S.; KERKWIJK, M. H. van; BRETT, M.; HALDANE, A.; RÍO, J. F. del; WIEBE, M.; PETERSON, P.; GÉRARD-MARCHANT, P.; SHEPPARD, K.; REDDY, T.; WECKESSER, W.; ABBASI, H.; GOHLKE, C.; OLIPHANT, T. E. Array programming with NumPy. *Nature*, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020. Disponível em: <https://doi.org/10.1038/s41586-020-2649-2>.

18 HUNTER, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.

19 BRADSKI, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

20 REBACK, J.; JBROCKMENDEL; MCKINNEY, W.; BOSSCHE, J. V. den; AUGSPURGER, T.; CLOUD, P.; HAWKINS, S.; ROESCHKE, M.; GFYOUNG; SINHRKS; KLEIN, A.; HOEFLER, P.; PETERSEN, T.; TRATNER, J.; SHE, C.; AYD, W.; NAVEH, S.; DARBYSHIRE, J.; GARCIA, M.; SHADRACH, R.; SCHENDEL, J.; HAYDEN, A.; SAXTON, D.; GORELLI, M. E.; LI, F.; ZEITLIN, M.; JANCAUSKAS, V.; MCMASTER, A.; BATTISTON, P.; SEABOLD, S. *pandas-dev/pandas: Pandas 1.4.1*. Zenodo, 2022. Disponível em: <https://doi.org/10.5281/zenodo.6053272>.

21 MCKINNEY Wes. Data Structures for Statistical Computing in Python. In: WALT Stéfan van der; MILLMAN Jarrod (Ed.). *Proceedings of the 9th Python in Science Conference*. [S.l.: s.n.], 2010. p. 56 – 61.

22 WALT, S. van der; SCHÖNBERGER, J. L.; Nunez-Iglesias, J.; BOULOGNE, F.; WARNER, J. D.; YAGER, N.; GOUILLART, E.; YU, T.; CONTRIBUTORS the scikit-image. scikit-image: image processing in Python. *PeerJ*, v. 2, p. e453, 6 2014. ISSN 2167-8359. Disponível em: <https://doi.org/10.7717/peerj.453>.

23 RAMACHANDRAN, P.; VAROQUAUX, G. Mayavi: 3D Visualization of Scientific Data. *Computing in Science & Engineering*, IEEE, v. 13, n. 2, p. 40–51, 2011. ISSN 1521-9615.

24 Wikipedia contributors. *Beam diameter — Wikipedia, The Free Encyclopedia*. 2022. [Online; accessed 18-July-2022]. Disponível em: <https://en.wikipedia.org/w/index.php?title=Beam_diameter&oldid=1069692735>.

25 KHALIL, M.; IBRAHIM, A. Quick techniques for template matching by normalized cross-correlation method. *British Journal of Mathematics and Computer Science*, v. 11, p. 1–9, 01 2015.

26 CANNON, M.; LEHAR, A.; PRESTON, F. Background pattern removal by power spectral filtering. *Appl. Opt.*, Optica Publishing Group, v. 22, n. 6, p. 777–779, Mar 1983. Disponível em: <http://opg.optica.org/ao/abstract.cfm?URI=ao-22-6-777>.

27 SUR, F.; GREDIAC, M. Automated removal of quasiperiodic noise using frequency domain statistics. *Journal of Electronic Imaging*, SPIE, v. 24, n. 1, p. 1 – 19, 2015. Disponível em: <https://doi.org/10.1117/1.JEI.24.1.013003>.

28 AIZENBERG, I. N.; BUTAKOFF, C. Frequency domain medianlike filter for periodic and quasi-periodic noise removal. In: DOUGHERTY, E. R.; ASTOLA, J. T.; EGIAZARIAN, K. O. (Ed.). *Image Processing: Algorithms and Systems*. SPIE, 2002. v. 4667, p. 181 – 191. Disponível em: <https://doi.org/10.1117/12.467980>.

29 WELCH, P. The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, v. 15, n. 2, p. 70–73, 1967.

30 GALLAGHER, A.; LEWIS, E. L. Determination of the vapor pressure of rubidium by optical absorption. *J. Opt. Soc. Am.*, Optica Publishing Group, v. 63, n. 7, p. 864–869, Jul 1973. Disponível em: <http://opg.optica.org/abstract.cfm?URI=josa-63-7-864>.

31 STECK, D. A. Rubidium 87 d line data. 09 2001. Disponível em: <http://steck.us/alkalidata>.

32 SIDDONS, P.; ADAMS, C. S.; GE, C.; HUGHES, I. G. Absolute absorption on rubidium d lines: comparison between theory and experiment. *Journal of Physics B: Atomic, Molecular and Optical Physics*, IOP Publishing, v. 41, n. 15, p. 155004, jul 2008. Disponível em: <https://doi.org/10.1088/0953-4075/41/15/155004>.

33 PASCHOTTA, R. *Encyclopedia of Laser Physics and Technology*. Wiley, 2008. ISBN 9783527408283. Disponível em: <https://books.google.com.br/books?id=hdkJ5ASTFjcC>.

34 MARTINELLI, M.; VALENTE, P.; FAILACHE, H.; FELINTO, D.; CRUZ, L. S.; NUSSENZVEIG, P.; LEZAMA, A. Noise spectroscopy of nonlinear magneto-optical resonances in rb vapor. *Phys. Rev. A*, American Physical Society, v. 69, p. 043809, Apr 2004. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevA.69.043809>.

35 CRUZ, L.; FELINTO, D.; GOMEZ, J.; MARTINELLI, M.; VALENTE, P.; LEZAMA, A.; NUSSENZVEIG, P. Laser-noise-induced correlations in electromagnetically induced transparency. *Conference on Quantum Electronics and Laser Science (QELS) - Technical Digest Series*, v. 41, p. 531–539, 03 2007.

36 XIAO, Y.; WANG, T.; BARYAKHTAR, M.; CAMP, M. V.; CRESCIMANNO, M.; HOHENSEE, M.; JIANG, L.; PHILLIPS, D. F.; LUKIN, M. D.; YELIN, S. F.; WALSWORTH, R. L. Electromagnetically induced transparency with noisy lasers.

Phys. Rev. A, American Physical Society, v. 80, p. 041805, Oct 2009. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevA.80.041805>.

37 YANG, X.; SHENG, J.; KHADKA, U.; XIAO, M. Generation of correlated and anticorrelated multiple fields via atomic spin coherence. *Phys. Rev. A*, American Physical Society, v. 85, p. 013824, Jan 2012. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevA.85.013824>.

38 ALMEIDA, A.; MOTTA, M.; VIANNA, S. Intensity correlations in the forward four-wave mixing driven by a single pump. Disponível em: <http://arxiv.org/abs/2212.01491>.

39 KINOSHITA, Y.; ASAKURA, T.; SUZUKI, M. Autocorrelation of gaussian-beam fluctuation caused by a random medium. *J. Opt. Soc. Am.*, Optica Publishing Group, v. 58, n. 8, p. 1040–1047, Aug 1968. Disponível em: <https://opg.optica.org/abstract.cfm?URI=josa-58-8-1040>.

APPENDIX A – PATTERN REMOVAL FILTER

```
import numpy as np
import matplotlib.pyplot as plt
import cv2
from scipy import interpolate
from scipy.ndimage.filters import maximum_filter
from scipy.ndimage.morphology import generate_binary_structure,
binary_erosion


def radial_profile(data, center):
    y, x = np.indices((data.shape))
    r = np.sqrt((x - center[0])**2 + (y - center[1])**2)
    r = r.astype(np.int)

    tbin = np.bincount(r.ravel(), data.ravel())
    nr = np.bincount(r.ravel())
    radialprofile = tbin / nr
    return radialprofile


def DFT_2D(image):
    dft = cv2.dft(np.float32(image), flags=cv2.DFT_COMPLEX_OUTPUT)
    dft_shift = np.fft.fftshift(dft)
    magnitude_spectrum = 20 * np.log((cv2.magnitude(dft_shift[:, :, 0],
                                                    dft_shift[:, :, 1]))+1)
    return dft_shift, magnitude_spectrum


def find_patches(image, size_patches):
    rows, cols = image.shape
    han = np.hanning(size_patches)
    han_2d = np.outer(han, han)
    han_patches = []
    step_patches = size_patches//4

    for i in range(0, rows-size_patches, step_patches):
        for j in range(0, cols-size_patches, step_patches):
            han_patches.append(
                image[i:i+size_patches, j:j+size_patches] * han_2d)
    return han_patches


def log_power_spectral_estimate(patches):
    power_spectral_estimate = np.zeros(patches[0].shape)
    for patch in patches:
        fft = np.fft.fftshift(
            cv2.dft(np.float32(patch), flags=cv2.DFT_COMPLEX_OUTPUT))
        mag = np.power(cv2.magnitude(fft[:, :, 0], fft[:, :, 1]), 2)
        power_spectral_estimate = power_spectral_estimate + mag

    power_spectral_estimate = power_spectral_estimate/len(patches)
    return 20*np.log(power_spectral_estimate)


def angular_average(spectral_estimate):

    nrow, ncol = spectral_estimate.shape
    radial_average = radial_profile(spectral_estimate, (nrow//2, ncol//2))

```

```
59
60      x = np.linspace(0, len(radial_average), len(radial_average))
61      f = interpolate.interp1d(x, radial_average, kind='linear')
62      x_new = np.linspace(0, len(radial_average), 8*len(radial_average))
63      radial_average_interpolated = f(x_new)
64
65      return radial_average, radial_average_interpolated
66
67
68  def angular_average_2D(ang_average, size_patches, radius):
69
70      rows, cols = len(ang_average)*2, len(ang_average)*2
71      crow, ccol = int(rows/2), int(cols/2)
72      mask = np.ones((rows, cols))
73      mask = mask * np.min(ang_average)
74      center = [crow, ccol]
75      x, y = np.ogrid[:rows, :cols]
76      mask_area = np.sqrt((x - center[0]) ** 2 + (y - center[1]) ** 2)
77      list_of_radius = np.linspace(0, radius, len(ang_average))
78
79      tolerance = 0.2
80      indice = 0
81      for r in list_of_radius:
82          res = np.where(np.abs(mask_area - r) <= tolerance)
83          mask[res] = ang_average[indice]
84          indice += 1
85
86      mask[crow, ccol] = np.max(ang_average)
87      angular_average = mask[crow - (size_patches//2):crow + (size_patches//2),
88                             ccol - (size_patches//2):ccol + (size_patches//2)]
89      return angular_average
90
91
92  def distance(p1, p2):
93      return np.sqrt((p1[0]-p2[0])**2 + (p1[1]-p2[1])**2)
94
95
96  def image_back(fft):
97      res = np.fft.ifftshift(fft)
98      res = cv2.idft(res)
99      return res[:, :, 0]*255/np.max(res[:, :, 0])
100
101
102 def detect_peaks(image):
103
104     neighborhood = generate_binary_structure(2, 2)
105     local_max = maximum_filter(image, footprint=neighborhood) == image
106     background = (image == 0)
107     eroded_background = binary_erosion(
108         background, structure=neighborhood, border_value=1)
109     detected_peaks = local_max ^ eroded_background
110
111     return detected_peaks
112
113
114 def pattern_filter(image, size_patches=128, threshold=0.5, notch_radius=5,
    exclusion_radius=10):
115
116     patches = find_patches(
117         image, size_patches=size_patches)
```

```
    log_pow_estimate = log_power_spectral_estimate(patches)


    radial_average, radial_average_interpolated = angular_average(
        log_pow_estimate)

    angular_average_itself = angular_average_2D(
        radial_average_interpolated, size_patches=size_patches,
radius=len(radial_average))

    modified_log_power_inter = cv2.subtract(log_pow_estimate.astype(
        np.float64), angular_average_itself.astype(np.float64))

    modified_log_power_inter = modified_log_power_inter / \
        np.max(modified_log_power_inter)
    smaller = np.where(np.abs(modified_log_power_inter) < threshold)
    modified_log_power_inter[smaller] = 0
    smaller = np.where((modified_log_power_inter) < 0)
    modified_log_power_inter[smaller] = 0

    modified_log_power_r = cv2.resize(
        modified_log_power_inter, image.shape[::-1])

    peaks = detect_peaks(modified_log_power_r)
    peaks = np.where(peaks == True)

    rows, cols = modified_log_power_r.shape
    filter = np.ones((rows, cols, 2))
    x, y = np.ogrid[:rows, :cols]
    img_center = (rows//2, cols//2)
    for peak in zip(peaks[0], peaks[1]):
        if distance(img_center, peak) < exclusion_radius:
            continue
        else:
            mask_area = (x - peak[0]) ** 2 + (y - peak[1]
                                              ) ** 2 <=
np.power(notch_radius, 2)
            filter[mask_area] = 0

    fft, img_mag = DFT_2D(image)
    fshift = fft * filter
    result = image_back(fshift)

    return abs(result)

```

APPENDIX B – DIAMETERS

```python
import cv2
import numpy as np
from scipy.optimize import curve_fit
import matplotlib.pyplot as plt
from sys import argv, exit
from os.path import join


def gaussian(x, y0, xc, A, w):
    return y0 + (A*np.exp(-(1.0/2.0)*np.power(((x-xc)/w), 2)))

video = int(argv[1])
diametro_h = []
diametro_v = []

for foto in range(1, 10):
    path = f'/path/{video}/{foto}.jpg'
    imagem = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
    ################################################
    feixe_1, _ = imagem[300:300+512, 100:100+512]

    ############### Horizontal ####################
    dados_h = np.sum(feixe_1, axis=0)
    dados_x = np.linspace(0, len(dados_h), len(dados_h))
    p0 = (1, 400, 10000, 150)
    popt, pcov = curve_fit(gaussian, dados_x, dados_h, p0=p0)
    diametro_h.append(abs(round(popt[3]*4*5.2)))
    del imagem, dados_h, dados_x, popt, pcov

    ############### Vertical ####################
    dados_v = np.sum(feixe_1, axis=1)
    dados_x = np.linspace(0, len(dados_v), len(dados_v))
    p0 = (1, 400, 10000, 150)
    popt, pcov = curve_fit(gaussian, dados_x, dados_v, p0=p0)
    diametro_v.append(abs(round(popt[3]*4*5.2)))
    del dados_v, dados_x, popt, pcov


path_save = 'Path to save'
path = join(path_save, 'file_name.txt')
with open(path, 'a') as fo:
    fo.write(f"{video},{np.mean(diametro_h)},{np.std(diametro_h, ddof=1)}
{np.mean(diametro_v)},\
{np.std(diametro_v, ddof=1)}\n")

```

APPENDIX C – RADIAL PROFILE

```python
import numpy as np

def radial_profile(data, center):
  y, x = np.indices((data.shape))
  r = np.sqrt((x - center[0])**2 + (y - center[1])**2)
  r = r.astype(np.int)

  tbin = np.bincount(r.ravel(), data.ravel())
  nr = np.bincount(r.ravel())
  radialprofile = tbin / nr
  return radialprofile
```

APPENDIX D – AUTO-CORRELATION OF INTENSITY FLUCTUATIONS

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
from estatistica_feixes import radial_profile, normxcorr2
from pattern_removal_filter import pattern_filter
from scipy.signal import fftconvolve
from sys import argv
from os.path import join
import os
from scipy.signal import peak_widths


def average_kernel(order):
    kernel = (1/(order**2))*np.ones((order, order))
    return kernel


P = figura[220:220+600, 50:50+600]
feixe_1_f = pattern_filter(P, threshold=0.2,
                           notch_radius=5, exclusion_radius=9)

F = figura[220:220+600, 680:680+600]
feixe_2_f = pattern_filter(F, threshold=0.2,
                           notch_radius=5, exclusion_radius=9)

# Creating the average kernel
kernel = average_kernel(20)

# Mean filtering the image
mean_image_1 = fftconvolve(feixe_1_f, kernel, mode=mode)
mean_image_2 = fftconvolve(feixe_2_f, kernel, mode=mode)

# Flutuactions
diff_1 = cv2.subtract(feixe_1_f, mean_image_1)
diff_2 = cv2.subtract(feixe_2_f, mean_image_2)

# Autocorrelations
auto_1 = normxcorr2(diff_1, diff_1, mode=mode)
auto_2 = normxcorr2(diff_2, diff_2, mode=mode)

max_auto1 = np.unravel_index(
    np.argmax(auto_1, axis=None), auto_1.shape)[::-1]
max_auto2 = np.unravel_index(
    np.argmax(auto_2, axis=None), auto_2.shape)[::-1]

# Radial
rad_auto1 = radial_profile(auto_1, max_auto1)[0:40]
rad_auto2 = radial_profile(auto_2, max_auto2)[0:40]
rad_auto1 = np.concatenate((rad_auto1[::-1], rad_auto1[1:]))
rad_auto2 = np.concatenate((rad_auto2[::-1], rad_auto2[1:]))
pos_1 = np.where(rad_auto1 >= 0)
pos_2 = np.where(rad_auto2 >= 0)
rad_auto1 = rad_auto1[pos_1]
rad_auto2 = rad_auto2[pos_2]

# Finding maximum
max_auto1_radial = np.where(abs(rad_auto1) == np.max(rad_auto1))
max_auto2_radial = np.where(abs(rad_auto2) == np.max(rad_auto2))

```

```
60 # Finding widths
61 width_auto1_radial = peak_widths(
62     rad_auto1, max_auto1_radial[0], rel_height=0.5)
63 width_auto2_radial = peak_widths(
64     rad_auto2, max_auto2_radial[0], rel_height=0.5)
65
```