ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

CONSTRUCTION OF HOLOGRAPHIC OPTICAL TWEEZERS SET-UP

M.Sc. THESIS

Nima BAVILI

Department of Physics Engineering

Physics Engineering Programme

MAY 2015

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

CONSTRUCTION OF HOLOGRAPHIC OPTICAL TWEEZERS SET-UP

M.Sc. THESIS

Nima BAVILI
(509121112)

Department of Physics Engineering

Physics Engineering Programme

Thesis Advisor: Prof. Dr. Günay BAŞAR

MAY 2015

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

HOLOGRAFİK OPTİK CIMBIZLAMA SETİ YAPIMI

YÜKSEK LİSANS TEZİ

Nima BAVILI
(509121112)

Fizik Mühendisliği Anabilim Dalı

Fizik Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Günay BAŞAR

MAYIS 2015

Nima BAVILI, a **M.Sc.** student of ITU **Graduate School of Science, Engineering and Technology** student ID **509121112**, successfully defended the **thesis** entitled "**CONSTRUCTION OF HOLOGRAPHIC OPTICAL TWEEZERS SET-UP**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Günay BAŞAR** Istanbul Technical University	....
Jury Members :	**Prof. Dr. Ayşe EROL** Istanbul University	....
	Asst. Prof. Dr. Alexandr JONAS Istanbul Technical University	....

Date of Submission : 30 April 2015
Date of Defense : 26 May 2015

To my family,

FOREWORD

I would like to express my sincere thanks to Professor Günay Başar for valuable guidance and thoughtful advice during my master's studies. I would also like to thank my best friend and laboratory fellow, Uğur Parlatan for the continuous encouragement and patience he showed in teaching several subjects**.**

Further thanks must go to all my friends in Istanbul technical university, especially Sare Akgöz, Peyman Beyranvand and Tuğçe Öztürk.

Undoubtedly most importantly, I would like to thank my family for unquestioning support during my whole life.

May 2015 Nima BAVILI

TABLE OF CONTENTS

Page

ABBREVIATIONS

SLM	: Spatial Light Modulator
HOT	: Holographic Optical Tweezers
LC	: Liquid Crystal
LC-SLM	: Liquid Crystal based Spatial Light Modulator
GUI	: Graphical User Interface
G&L	: Gratings and Lenses algorithm
DMD	: Digital Micromirror Device
RMET	: Random Mask Encoding Technique
GS	: Gerchberg-Saxton
EASLM	: Electrically Addressed Spatial Light Modulator
OASLM	: Optically Addressed Spatial Light Modulator
GL	: Gray Level
CCD	: Charge Coupled Device
MO	: Microscope Objective
LED	: Light Emitting Diode
LLF	: Laser Line Filter
FWHM	: Full Width at Half Maximum
REF	: Raman Edge Filter
CNLC	: Chiral Nematic Liquid Crystals
NLC	: Nematic Liquid Crystals
FF	: Fill Factor
DVI	: Digital Visual Interface
DE	**:** Diffraction Efficiency
AR	**:** Anti-Reflective

LIST OF TABLES

Page

LIST OF FIGURES

CONSTRUCTION OF HOLOGRAPHIC OPTICAL TWEEZERS SET-UP

SUMMARY

Since its discovery in mid-80s by Arthur Ashkin, optical tweezing has undergone an important breakthrough and expanded its horizons. Nowadays optical tweezers application in many fields, from biology to nanotechnology, is inevitable. Investigating mechanical properties of DNA, protein-DNA interactions, carbon nanotubes properties are among a few major examples of successful applications of optical tweezers.

The next milestone in optical tweezing was trapping several particles simultaneously. Advent of spatial light modulator (SLM) paved the way for this purpose in holographic optical tweezers (HOT). In fact, although many devices can be considered as SLM, liquid crystal (LC) devices play a significant role in HOT systems. LC-SLM enables creating and manipulating several traps simultaneously and independently. To put it another way, HOTs are like our fingers in microscale.

In this thesis, I try to explain construction of holographic optical tweezer set-up in Istanbul technical university and major obstacles that we had to overcome, as several modification were done in single particle trapping set-up such as using different Keplerian telescopes etc.

One of the most important factors was calibration of the LC-SLM, since this device in default company configuration may function in unwanted phase shift interval.

I have also developed an interactive graphical user interface (GUI) in MATLAB and Simulink Student Suite 2014a in order to manipulate micron-sized particles in 3D. This program is able to get several traps positions from user and generate the desired trap holograms, in a way that, each trap can be manipulated in 3D independently. In this program, I have exploited the most straightforward and fastest method, which is gratings and lenses algorithm (G&L).

The program calculates the hologram after placing the spots in a region, which shows the sample plane by camera. Created hologram image in the object plane is transformed to image plane using Fourier transform via suitable number of lenses. For our future works, additional application were also implemented in this program in order to take a spot from initial coordinates to a desired coordinates with a speed and step numbers all defined by user.

In addition to these, several other measurements either in calibration or HOT set-up were done to achieve better adjustments.

HOLOGRAFİK OPTİK TUZAKLAMA SETİ YAPIMI

ÖZET

1980'lerden bugüne Arthur Ashkinin keşfettiği optik tuzaklama yöntemi ufuklarını genişletip, bir çok alanda başarılar elde etmiştir. Günümüzde optik tuzaklama yöntemi biyoloji ve nanoteknoloji gibi bir çok alanda kullanılmaktadır. Örneğin DNA moleküllerinin mekaniksel özellikleri, protein-DNA etkileşmesi ve karbon nanotüplerinin özelliklerinin incelenmesi optik tuzaklamanın çok önemli işlevleri arasında yer almaktadır.

Optik cımbızlama aslında sıkı odaklanmış laser ışını kullanarak mikro ve nano metre mertebesindeki parçacıkları tuzaklama ve inceleme yöntemidir.

Parçacıkların kullanılan dalga boyuyla karşılaştırılarak iki farklı rejimde incelenebilir. Eğer laserin dalgaboyu parçacıktan çok büyük ise Mie rejimi (veya geometrik optik) tuzaklamayı açıklamak için yeterlidir. Halbuki eğer parçacığın boyutu dalga boyundan çok küçük ise Rayleigh rejiminin kullanılması gerekmektedir. Yaptığımız deneylerde kullanılan parçacıkların boyutu genellikle mikrometre mertebesinde ve laserin dalga boyu 785 nm olduğundan dolayı ışın optiği çerçevesinde açıklanması mümkündür.

Kuşkusuz optik cımbızlamanın bir sonraki adımı, bir veya birkaç mikro metre veya nano metre mertebesindeki parçacıkları aynı anda 3 boyutta manipüle ve kontrol etmektir. Sıvı kristallerden yapılan uzaysal ışık modülatörlerinin (SLM) gelişi şüphesiz bu adımı mümkün kılmıştır. Günümüzde bir çok araştırma grubu dünyanın en önde gelen üniversitelerinde bu yöntemi geliştirmeye çalışmaktadır.

Aslında bir çok cihaz uzaysal ışık modülatörü sayılabilir; örneğin dijital mikro ayna cihazları vs. Ama sıvı kristallerden yapılan modülatörler sağladıkları esnekliklerden dolayı holografik optik tuzaklama (HOT) setlerinde daha çok tercih ediliyor. Bu tezde SLM ifadesi genellikle sıvı kristallerden yapılan modülatörler için kullanılmaktadır.

Sıvı kristaller ifadesi ilk başta anlamsız gözükmesine karşın sıvı ve katı fazları arasında bulunan ve egzotik özelliklere sahip olan bir ara fazın tanımıdır. Nematik ve Ferroelektrik sıvı kristaller SLM'de kullanılan en yaygın türlerdir. Bu iki türün elektrik alanda yönelimleri ve özelliklerinin değişimi SLM'in faz veya genlik modulasyonu için kullanılmasını mümkün kılıyor.

HOT yöntemi ile aynı anda çok sayıda parçacığı tuzaklayıp kontrol edebiliriz. Bir başka deyişle HOT bizim mikro dünyadaki parmaklarımızdır.

Bu tezde ilk başta İstanbul Teknik Üniversitesi'nde tek parçacık için kurulan optik tuzaklama seti anlatılıyor. Bu seti değiştirmek için kullandığımız, farklı Kepler teleskoplarının denenmesi gibi birçok yöntem ve yapılandırma sonraki bölümlerde detaylı bir biçimde anlatılıyor.

Bir sonraki adımda ise seçilen modülatörün özellikleri ve seçilme sebepleri anlatılmaktadır. Işık modülatörlerleri çok az sayıda firma tarafından üretiliyor ve

genellikle iki farklı adreslenme yöntemine sahiplerdir: elektrik ve optik. Seçilen SLM, yansıtıcı olup, ve sıvı kristal hücreleri elektrik adreslenme yöntemi ile çalışmaktadır.

HOT setinin kurulmasının başlangıç aşaması ışık modülatörünün kalibre edilmesidir. Bu aşamada amacımız cihazın 256 Gri tonlama (Gray Level) adımında 2π faz değişimini sağlamasıdır. Normalde firmanın yapılandırması farklı faz değişimine sebep olabilir. Farklı faz değişiminin sebebi ise üretici firmanın varsayılan dalgaboyu, kullanılan dalga boyundan farklı olmasıdır.

Kalibrasyonun yapılması için daha önceden bir çok yöntem girişim desenlerinin analizi üzerine kurulmuştur. Seçtiğimiz yöntemde laser ışını çift yarık kullanılarak iki farklı ışına ayrılıyor. Her bir ışın SLM'in farklı yarısına gelmesi sağlanıyor. SLM'in bir tarafı deney boyunca siyah (tonu 1) ve sabitken diğer yarısı 1'den 256'ya kadar tonlari birer birer tarıyor. Bu işlem, kamerada görüntülenen girişim desenlerinde bir kayma meydana getirir. Bu kaymaların incelemesi faz değisimini ortaya koymaktadır.

Kalibrasyon yöntemi ve yapılandırması MATLAB'da tasarlanan programın yanı sıra detaylı biçimde 3. bölümde anlatılıyor.

Işık modülatörleri faz modülasyonun yanı sıra genlik modulasyonu içinde kullanılıyorlar. LCD televizyonlarının çalışma prensibi genlik modulasyonuna dayalıdır.

Kullandığımız SLM, faz modülatörü olarak tasarlandığı için laserin genliği deney boyunca aşırı değişikliklere maruz kalmıyor. Yapılan deneylerin sonuçlarıda bu ifadeyi desteklemekte.

Kalibrasyondan sonra bir örnek olarak Laguerre-Gaussian ışık profilini ona ait olan hologramı görüntüleyerek elde edildi. Laguerre-Gaussian helis şeklinde dalga düzlemine sahiptir ve donut şeklini görüntülememizi sağlıyor.

HOT setinde parçacıkları üç boyutta manipüle ve kontrol etmek için SLM üzerinde hologramlar oluşturmak gerekiyor. En yaygın ve en hızlı yöntem *prisms and lenses* algoritmasıdır. Bu yöntem aslında basit optik elemanlarının, yani prizma ve lens özelliklerine dayanarak çalışıyor. Prizma ve lenslerden oluşan kombinasyon hologramlarını SLM üzerinde görüntüleyerek çok sayıda parçacığı birbirinden bağımsız olarak üç boyutta kontrol edebiliriz. Bu yöntem detaylı bir şekilde 4. bölümde anlatılıyor.

Anlatılan algoritmanın yanı sıra birkaç farklı hologram oluşturma yöntemi vardır. Yaygın şekilde kullanılanlar: Rastgele maske kodlaması (Random mask encoding) ve Gerchberg-Saxton algoritmasıdır. Rastgele maske kodlamasında SLM pikselleri rastgele bölgelere ayrılıyor ve her bölge farklı bir tuzak noktası için kullanılıyor. Gerchberg-Saxton algoritmasında iki boyutta iyi sonuçlar verirken üç boyutta ise çok yavaştır ve o yüzden sadece iki boyuttaki uygulamalarda tercih edilmektedir.

Parçacık tuzaklarını oluşturmak ve kontrol etmek için bir Grafiksel Kullanıcı Arayüzü (GUI) geliştirildi. Bu program daha önce anlatıdığı gibi en yaygın metodu yani *prisms and lenses* kullanarak hologramları oluşturuyor.

Kullanıcı dostu olan bu program tuzak sayıları ve konumlarını kullanıcıdan alarak tuzakların oluşmasını sağlayan hologramı hesaplayıp görüntülüyor. Ayrıca tuzak noktalarının konumu değiştikce hologram tekrar hesaplanıp güncelleniyor.

Gelecek araştırmalarımızda kullanılmak üzere farklı bir fonksiyon da bu arayüze yerleştirildi. Bu fonksiyon spot noktasını alıp istenilen hızda ve adım sayısında istenilen noktaya taşıyabiliyor.

Daha önce de belirtildiği gibi HOT setinin kurulması ve en iyi sonuçlar elde edilmesi için farklı yapılandırmalar denendi.

Uygun teleskop sistemleri en iyi görüntü ve en iyi tuzaklamayı elde etmek için büyük rol oynamakta. Teleskop kullanmadan SLM de modüle edilen ışınlar ayna veya ters mikroskop girişini taşıp tuzaklamayı imkansız hale getirebiliyor. Bu sebepten dolayı genellikle iki farklı Kepler teleskopu birincisi kullanılan dikroik ayna öncesi ikincisi ise ters mikroskopun girişi öncesinde yerleştiriliyor. Kamera öncesinde yerleştirilen Kepler teleskopunda kullanılan lenslerin AR (anti-reflection coating) kaplamaya sahip olmaları büyük önem taşımakta. Yansıtmayı düşüren bu kaplamalı lensleri kullanmadan yansıyan ışınlar ters mikroskoptaki örneklerin görüntülemesini engelleyebiliyor.

Kameranın konum kalibrasyonu, bir başka deyişle parçacıkların tasarlanan arayüzdeki konum değişkliliği birimine karşılık gelen gerçek koordinasyon değişiminin hesaplanması şüphesiz en önemli diğer adımlardan birisidir. Bunu yapabilmek için polistiren parçacıkları test parçacıkları olarak kullanılabilir.

Gelecek çalışmalarda kurulan setin iyileştirimesi ve otomatik parçacık ayrıştırmak için kullanılması tasarlanan arayüze bahsedilen diğer algoritmaların eklenmesi öngörülmektedir.

1. INTRODUCTION

1.1 Optical Trapping

As wave-particle duality is not a new concept, the idea that light carries momentum has been suggested and modified by several scientists throughout the history. The phenomenon that now we know as “radiation pressure” probably dates back to J. Kepler (1571-1630) who tried to explain the visible atmosphere of comets. He argued that comet particles are pushed away by light pressure of sun, shaping its tail. Many other modification and experiments were done by several scientists. In 1847, James Clerk Maxwell, the founder of electromagnetics theory, predicted radiation pressure as an outcome of his theory. Two years later, A. Bartoli, Italian physicist, conducted an experiment showing the existence of radiation pressure through thermodynamics [1, 2]. That is why it is also called Maxwell-Bartoli pressure.

Two other experiments in early 20th century were an important milestone. P. Lebedev, in 1901, and E. F. Nichols and G. F. Hull in 1903 experimentally proved and published what Maxwell had predicted before. Lebedev observed moving the winglet of a radiometer in a vacuum bulb when it was illuminated by an arc lamp. Nichols and colleagues quantitatively demonstrated the same experiment with different gases in different pressures [3, 4].

Now it is clear that light carries linear and angular momentum, so it can exert force and torque when it interacts with matter. However, for ordinary light sources, this force is insensible. Arthur Ashkin founded the basis of revolutionary phenomenon that now is known as Optical Trapping or Optical Tweezers. In his first paper published in 1970, he reported accelerating micron-sized particles levitated in a medium by laser, a great success in manipulating small particles. An important obstacle that Ashkin could overcome in this experiment was thermal forces, or in other words, radiometric forces. Generally, these forces mask the radiation pressure so we cannot detect radiation pressure of light beam. He used relatively transparent medium to avoid these effects. (Transparent micron-sized latex beads accelerated in water [5].)

Another relevant experiment that Ashkin reported was trapping the particle using two opposing beams. A particle was trapped between the beam waists of the two beams, in a location where their radiation pressure just cancel each other. By blocking one of the beams, the particle would be accelerated in the opposite direction. Ashkin noticed that in the process of accelerating, particles tended to move to the region of the higher intensity of the beam due to the gradient force [6]. This paved the way to optical trapping by single beam in 1986.

In the following years, Ashkin and colleages did several experiments using radiation pressure. In one of them, a single particle was trapped using an upward laser beam, so the gravitational force and radiation pressure could balance each other. However, in 1986, they showed and explained trapping of particles as small as 25 nm to several µm, just using a single laser beam and high numerical-aperture-microscope-objective[6].

Nowadays, optical trapping either alone or combined with different spectroscopic methods such as Raman spectroscopy is quite common and popular in many fields especially in biology and nanotechnology and enormous variety of application have been introduced. Single cell, single DNA studies, Carbon nanotubes and nanoparticles manipulation are among prominent examples [7-11].

1.2 Holographic Optical Trapping (HOT)

The other important step is to trap and manipulate several particles simultaneously, which is undeniably important in optical sorting. Without reservation, advent of Spatial Light Modulators (SLM) paved the way for multiple particle manipulation.

Many devices can be named as Spatial light modulator, for example, adaptive devices, which include deformable mirrors, and digital micromirror devices (DMDs). However, liquid crystal-based SLMs (LC-SLM) play a major role in optical manipulation nowadays. First of all because of its high resolution, secondly because of its ability to dynamically and simulataneously manipulate tens of traps. As we mainly focus on LC-SLM in this thesis, the word SLM mainly refers to liquid crystal spatial light modulators.

Generally, phase-mostly SLMs are used in holographic optical tweezers set-ups. Various configurations and algorithms have been proposed by several groups around the world. The most straightforward non-iterative algorithm undoubtedly is grating and lenses (G&L) algorithm that is based on physical properties of these simple optical devices, which can be generated by modulating the wavefront of light incident on SLM. We are able to create and manipulate several traps using this method. The other prominent algorithm is random mask encoding technique (RMET). Simply, in this method the whole pixelated screen of SLM is divided into subdomains. Each domain is then assigned to an independent trap. In this method, we are also able to create and manipulate multiple traps simultaneously. Gerchberg-Saxton (GS) is an iterative method that is used in HOT systems. It is generally based on back and forward propagation of the complex field and optimizing intensity distribution in Fourier plane by replacing phase shift values. This method is too slow relative to other above-mentioned algorithms in 3D. For instance, it may take days to calculate desired hologram. It is necessary to indicate that all these algorithms are based on Fourier transformation [12, 13].

1.3 The Structure of Thesis

In chapter 2, optical trapping principles as well as Mie and Rayleigh regime of optical trapping will be discussed. In addition, single particle optical tweezers set-up in Istanbul technical university and previous works will be explained briefly.

In chapter 3, different kinds of liquid crystals and their characteristics as well as the structure of LC based SLM will be discussed. Electrically addressed SLM (EASLM) and optically addressed SLM (OASLM) are two major categories that are manufactured. HOT set-ups generally use phase-mostly devices, so in this chapter we will also present the basis of phase modulation as well as a configuration for amplitude modulation. The most important factor in HOT design is calibration of the SLM in order to obtain optimized linear relation between gray levels (GL) and phase shift. An optimized calibration is defined when the phase shift is 2π between two dark or bright fringes in the interference pattern of a double slit. Toward the end of this chapter, we will provide several existing calibration methods, and our calibration experiment, set-up and results.

In chapter 4, the most prominent method in creating holograms (gratings and lenses algorithm), several other methods, and their details will be provided. In addition designed graphical user interface program as well as HOT set-up will be presented.

2. OPTICAL TWEEZERS

2.1 Single Particle Trapping Principle

For thorough depiction of trapping particles in highly focused light beam, we should use Lorentz force equation; the force per unit volume for an unknown charge distribution is:

$$\vec{f} = \rho\vec{E} + \vec{J} \times \vec{B} \tag{2.1}$$

However, in many cases, solving this equation is somehow tedious because of arbitrary illumination and unsymmetrical shapes and needs the use of the T-matrix formulation. Fortunately, we are able to use two limiting conditions, ray optics treatment when the particle size is much greater than wavelength $r \geq 5\lambda$ and electrical dipole approximation (Rayleigh regime) when the particle size is much smaller than the wavelength $r \ll \lambda$.

In the Rayleigh regime, the particle is considered as a dipole in an inhomogeneous electromagnetic field. As we know, although an atom may be neutral electrically, it consists of negative electron cloud around positive nucleus. In an external electrical field $\vec{E}$, these charged parts of atoms tend to move to opposite directions with regard to the external field and reach an equilibirium state by balancing internal and external electric forces. As a result, atom will have a dipole moment with the same direction as the external field. In the linear optics regime, this induced dipole moment is proportional to $\vec{E}$.

$$\vec{p} = \alpha\vec{E} \tag{2.2}$$

In which α is electronic polarizibility. By applying the Lorentz equation to each of the charges and using Maxwell equations one can find the total force exerted.

As the two charges, q forming an electric dipole have opposite signs, we can write:

$$\vec{F} = q[\overrightarrow{E_1}(x,y,z) - \overrightarrow{E_2}(x,y,z) + \frac{d(\overrightarrow{x_1} - \overrightarrow{x_2})}{dt} \times \vec{B}] \qquad \textbf{(2.3)}$$

$$= q[\overrightarrow{E_1}(x,y,z) - \left((\overrightarrow{x_1} - \overrightarrow{x_2}).\nabla\right)\vec{E} - \overrightarrow{E_1}(x,y,z) + \frac{d(\overrightarrow{x_1} - \overrightarrow{x_2})}{dt} \times \vec{B}] \qquad \textbf{(2.4)}$$

An electric dipole is defined as $p = q.d$, in which d is equal to the distance between charges d = x_1-x_2. By replacing it in the equation above, we have:

$$\vec{F} = \left(\vec{p}.\vec{\nabla}\right)\vec{E} + \frac{d\vec{p}}{dt} \times \vec{B} \qquad \textbf{(2.5)}$$

By replacing $p = \alpha E$:

$$\vec{F} = \alpha[\left(\vec{E}.\vec{\nabla}\right)\vec{E} + \frac{d\vec{E}}{dt} \times \vec{B}] \qquad \textbf{(2.6)}$$

In the next step, we have to consider two other equations, first from vector analysis:

$$\left(\vec{E}.\vec{\nabla}\right)\vec{E} = \vec{\nabla}\left(\frac{1}{2}\vec{E}^2\right) - \vec{E} \times (\vec{\nabla} \times \vec{E}) \qquad \textbf{(2.7)}$$

Second one from Maxwell equations:

$$\vec{\nabla} \times \vec{E} = -\frac{d\vec{B}}{dt} \qquad \textbf{(2.8)}$$

By inserting the vectorial analysis equation (2.7) in to (2.6), we have:

$$\vec{F} = \alpha[\frac{1}{2}\,\vec{\nabla}\,\vec{E}^2 - \vec{E} \times \left(\vec{\nabla} \times \vec{E}\right) + \frac{d\vec{E}}{dt} \times \vec{B}] \qquad \textbf{(2.9)}$$

Then we will substitute from Maxwell equation (2.8):

$$\vec{F} = \alpha[\frac{1}{2}\,\vec{\nabla}\,\vec{E}^2 - \vec{E} \times \left(-\frac{d\vec{B}}{dt}\right) + \frac{d\vec{E}}{dt} \times \vec{B}] \qquad \textbf{(2.10)}$$

$$\vec{F} = \alpha[\frac{1}{2}\vec{\nabla}\,\vec{E}^2 + \frac{d}{dt}\big(\vec{E}\times\vec{B}\big)] \quad \textbf{(2.11)}$$

The second term in the equation above is similar to Poynting vector but with a multiplicative constant difference. Poynting vector is defined as power per unit area that passes through surface. Second term will be equal to zero if we consider that laser power is constant. As a result:

$$F_{grad} = \frac{1}{2}\alpha\nabla E^2 \quad \textbf{(2.12)}$$

By replacing $\alpha = 4\pi n_0^2\epsilon_0 a^3\,(m^2-1)/(m^2+2)$, which is the polarizibility of spherical particle we obtain:

$$F_{grad} = \frac{2\pi n_0 a^3}{c}\left(\frac{m^2-1}{m^2+2}\right)\nabla I(r) \quad \textbf{(2.13)}$$

a and n_0 are the radius of the sphere and refractive index of medium respectively. m also is the relative index, n_1/n_0, of the particle. Formula (2.13) provides so-called gradient optical force which is function of gradient of the square of electrical field magnitude, or in other words the intensity of the beam as a function of position [14]. We can also compute the scattering force on a particle by evaluating vectorial difference of momentum incident on the particle and momentum reradiated from it:

$$F_{scat} = \frac{8\pi n_{med} k^4 r^6}{3c}\left(\frac{m^2-1}{m^2+2}\right)^2 I \quad \textbf{(2.14)}$$

Where k is wave number. The intensity gradient, which leads to the gradient force (2.13), is the causative factor in attracting particle to the highest intensities, what Ashkin observed experimentally for the first time.

On the other hand, for understanding the ray optics approximation or Mie regime it is necessary to consider the direction and the intensity of the beam before and after encountering the particle. First, let us discuss two rays of light, which belong to the incident light beam passing through a transparent bead with refractive index higher than the refractive index of the surrounding medium. Consider that the particle is slightly displaced from the axis of the beam. As it is clear from figure 2.1, both rays will undergo a momentum change, which according to Newton's second law, corresponds to the force acting on the rays. According to Newton's third law a force with the same magnitude but in the opposite direction will accompany this light momentum change. As the intensity of beam is higher at the center of the beam, the momentum contained in the two considered rays is also different and the bead will feel a net force toward the center of the light beam.

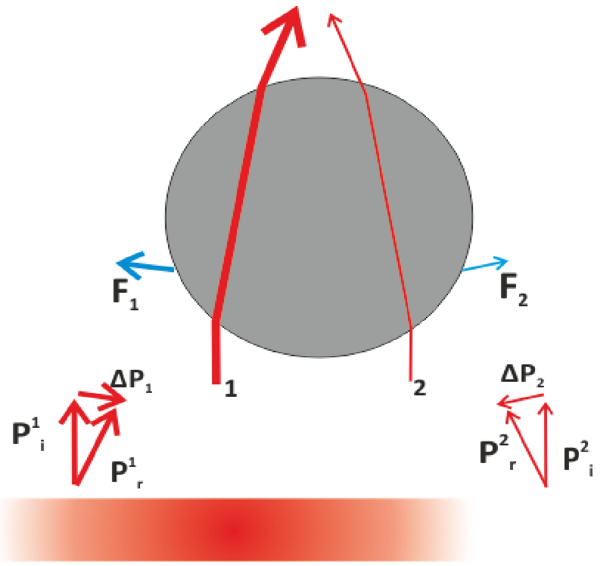


Figure 2.1 : Schematic view of a transparent spherical particle in a Gaussian beam. P_i and P_r are incident and refracted momentum, respectively. ΔP is momentum change and F is the exerted force.

Now consider another case shown in figure 2.2, in which the particle is slightly below or above the waist of focused Gaussian beam propagating upward.

a) When the particle is located before the beam focus, it increases the divergence of the beam; the net momentum flux of the beam along the optical axis decreases after passing the bead.

b) In contrast, when the particle is located after the focus, the beam is convergent and the momentum flux along the optical axis increases after passing the particle. By applying second and third law of Newton, we can see the direction of force exerted on the particle in these two cases, too.

The significant factor is that gradient force must exceed scattering force for trapping particle [15].

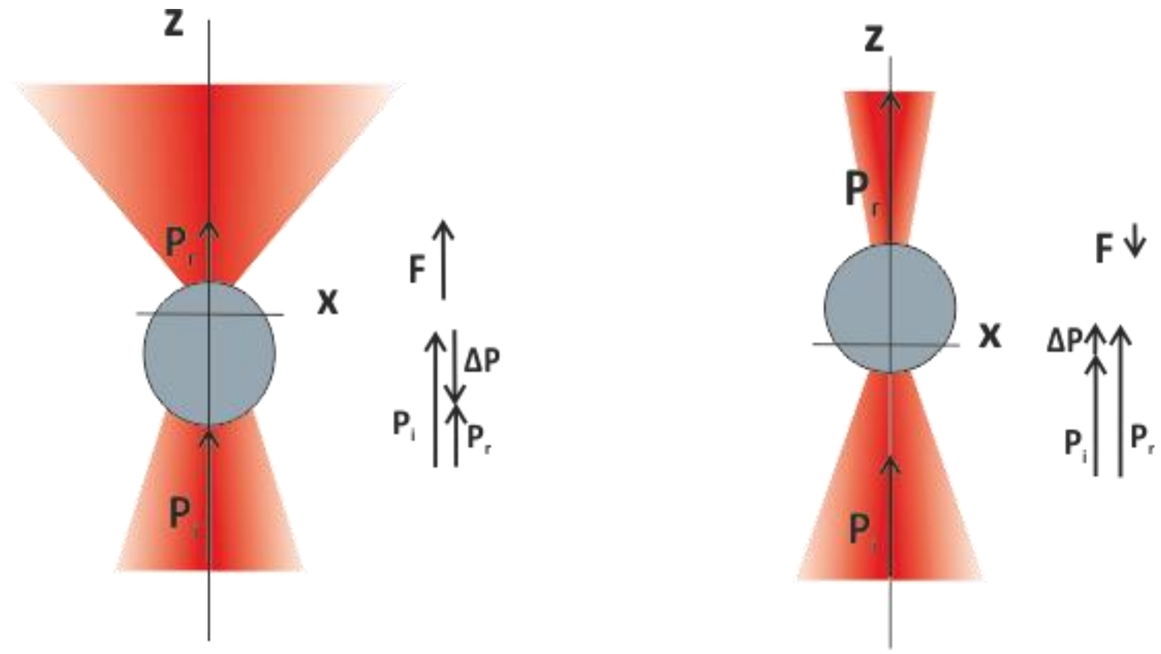


Figure 2.2 : Schematic view of a transparent particle in Gaussian beam slightly below and above the beam waist. P_i and P_r are incident and refracted momentum, respectively. ΔP is momentum change and F is the exerted force.

2.2 Single Particle Trapping Set-up

Single particle trapping set-up of ITU laser and spectroscopy laboratory is shown in figure 2.3. It is designed in order to get Raman spectra of a trapped particle. As this set-up is used for HOT in further experiments, its specification will be discussed in this section. This experimental configuration is on a 1.2 m x 2.4 m optical table. Devices and their functions are as follows:

Diode Laser: a diode laser with 785 nm wavelength and TEM_{00} mode is used in order to measure Raman spectrum of trapped particles in 200-1900 cm^{-1} region. The output power is around 100 mW.

Spectrograph: Set-up also consists of a spectrograph that is equipped with a 216 x 1024-pixellated CCD camera in order to measure Raman scattering from the trapped sample.

Inverted microscope: Optical trapping takes place in an inverted microscope. It includes three parts (figure 2.4). The lowest part is equipped with a dielectric mirror in order to guide the beam to the microcope platform with the angle of 90 degrees. This dielectric mirror has the reflectivity around 99% for 600-1000 nm wavelengths.

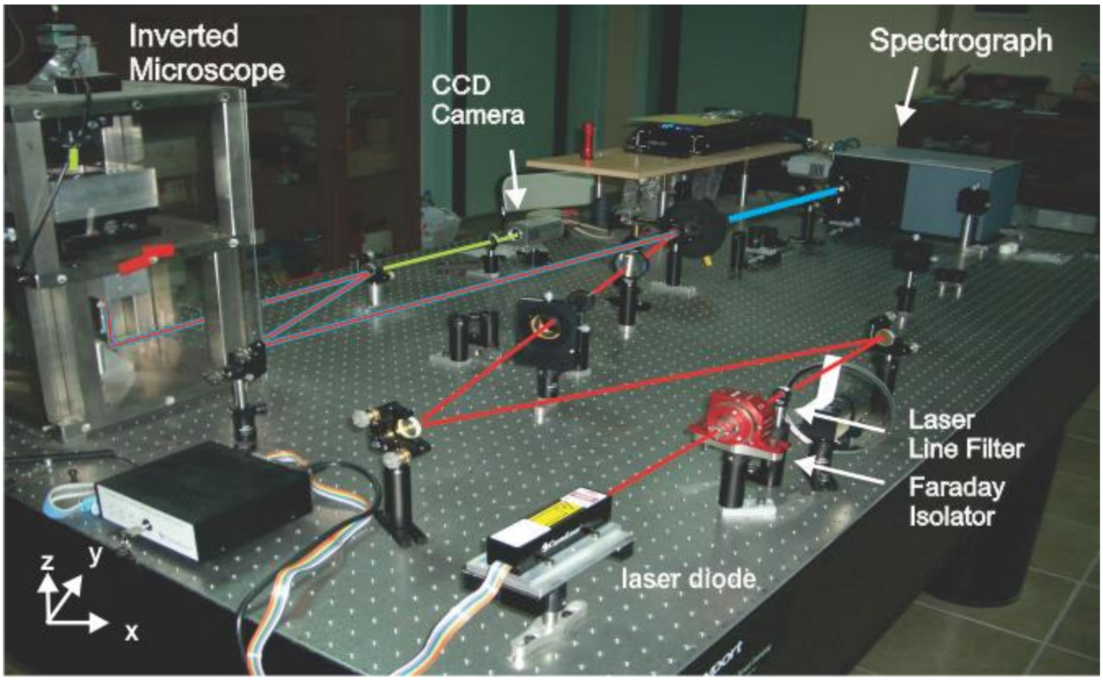


Figure 2.3 : Single particle trapping set-up of ITU.

In the central part, microscope objective (MO) is attached to a linear stage, which can move it along Z-axis as accurate as one micrometer. MO is water immersion and has 1.2 numerical aperture and magnification of 60x. Samples can be attached to another linear stage to move them in XY plane. On top of the microscope, an LED is used for illuminating the sample. In addition, for adjusting the image plane of LED, a lens, a diffuser and a 10x MO is used. After illuminating the samples, videos and images can be captured by camera placed in front of the inverted microscope.

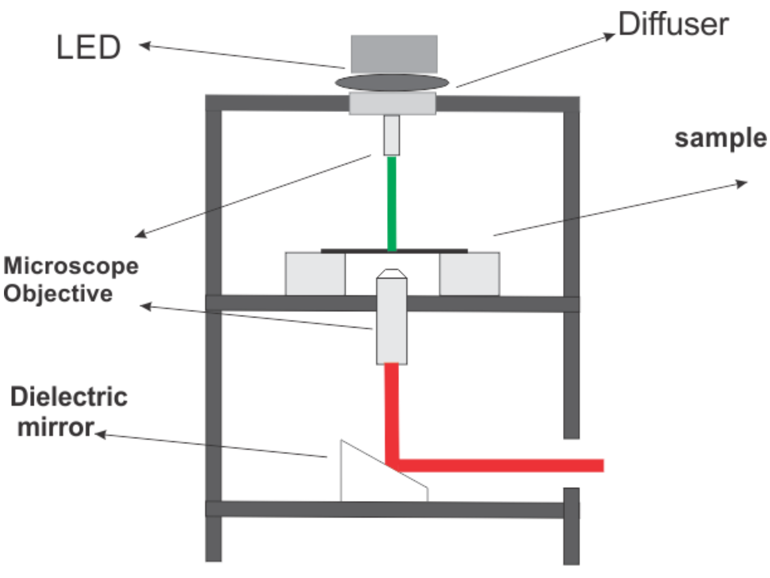


Figure 2.4 : Schematic view of inverted microscope.

As shown in figure 2.3, for preventing unwanted feedback into laser, a Faraday isolator (FI) is used. FI, which is also known as optical isolator, is one directional optical valve that prevents reflections from other optical devices, which cause instability in laser. After FI, light beam also passes through a laser line filter (LLF) in order to obtain clear emission. LLF provides high spectral resolution and blockage of emissions up to 10^{-6}. Transmission – wavelength graph of LLF for 0° incidence angle is shown in figure 2.5. Full width at half maximum (FWHM) of this LLF is around 10nm.

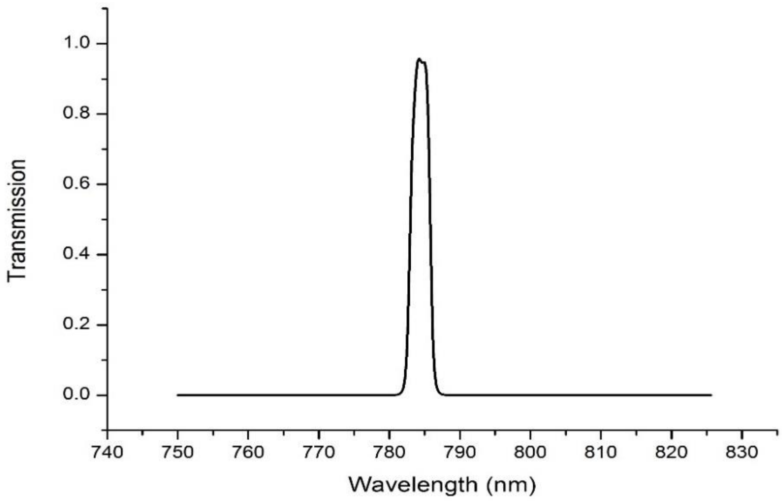


Figure 2.5 : Transmission-wavelength graph of LLF for 0°.

In order to ensure that the light beam is parallel with optical table and for adjusting beam height with spectrograph entrance, two dielectric coated mirrors are used. The reflectivity of these mirrors is around 99%. Reflectance-wavelength graph of these mirrors is shown in figure 2.6.

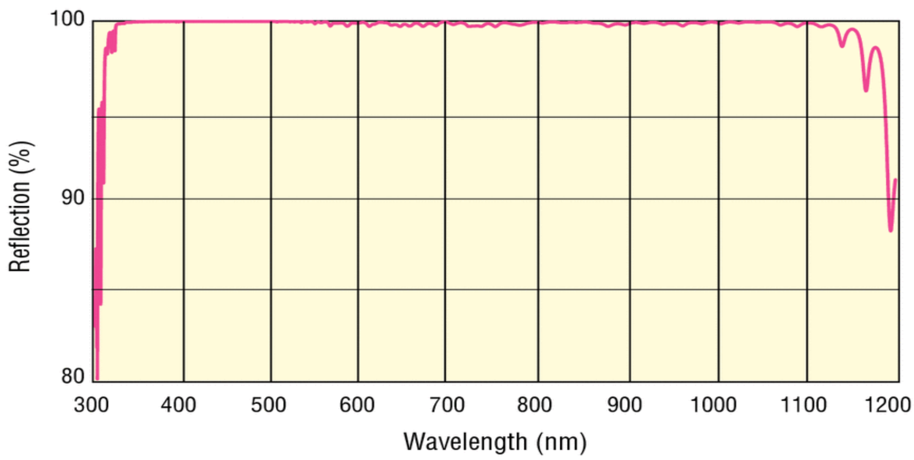


Figure 2.6 : Reflectance-wavelength graph of dichroic mirrors used for beam steering.

After mirrors, the light beam also passes through two lenses in order to adjust its diameter up to 9 mm, which fulfills MO entrance pupil. These lenses are plano-convex and have 100 mm and 300 mm focal length, respectively.

A Raman edge filter (REF) is just mounted after lenses. REF reflectivity for rays with wavelength less than 785 nm is approximately 99%. However, its transmittance for rays with wavelength more than 790 nm is around 98%. To put it another way, REF reflects most of the light in wavelenths less than 785 nm. Transmittance-wavelength graph is shown in figure 2.7 for 0° angle. It is necessary to mention that, as it is clear from the figure 2.3 the incident angle in REF is not 0°. In this case, transmittance region will shift toward smaller wavelengths around a few nanometers .

In the next step, beam height will be readjusted using a dielectric and dichroic mirror with regard to entrance pupil of inverted microscope. Dichroic mirror, reflects 98% of rays with wavelength more than 600 nm. In contrasts, wavelength less than that are mostly transmitted.

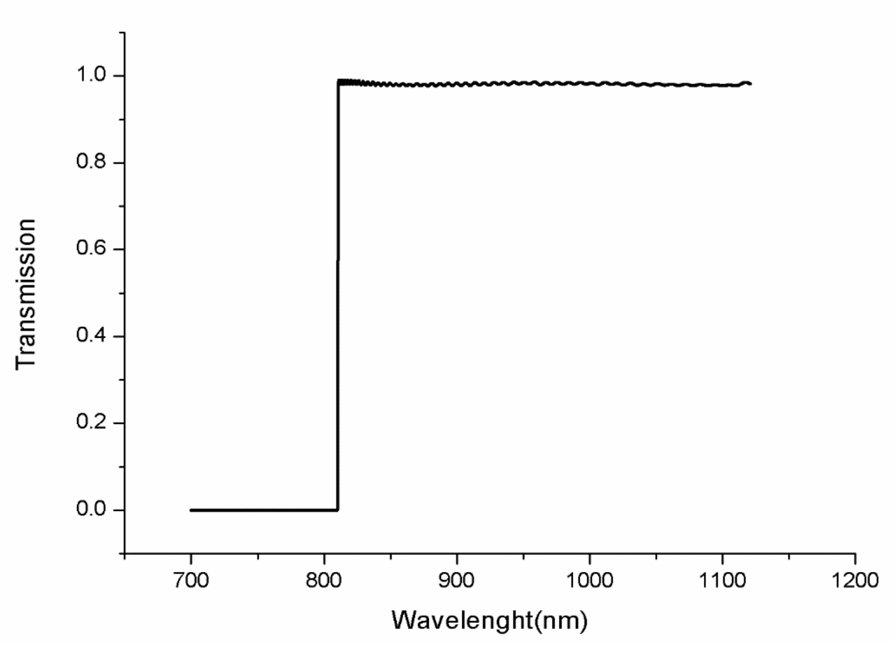


Figure 2.7 : Transmission-wavelength graph of REF.

As it was mentioned previously, after placement of samples on XY linear stage, the distance between MO focal point and samples will be adjusted by Z stage of MO. In most of the experiments for single particle trapping, samples are inside a hole of microscope slide. A hole was made in microscope slide and was covered by cover slip after putting the sample liquid in it. Illuminating the sample by green LED, makes it visible in camera, as dichroic mirrors transmit light with wavelength less than 600nm. After adjusting laser focus, particles in liquid can be trapped in the beam waist of the laser. Particle location can be manipulated in 3D, by previously mentioned linear stages.

As the trapped particle is illuminated by the laser beam, elastic (Rayleigh) scattering and inelastic (Raman) scattering occurs. As Rayleigh rays have the same wavelength with laser light they can not pass through REF; one in 10^6 of photons may pass REF. However, Raman scattering can be trasnmitted from the REF since wavelength of Raman beam is greater than threshold wavelength of the REF. To ensure maximum filtering of Rayleigh scattering, second REF is also used. At the end, Raman scattering photons are focused to the entrance pupil of the spectrograph [16]. Detailed schematic view of above-mentioned single particle trapping set-up is shown in figure 2.8.

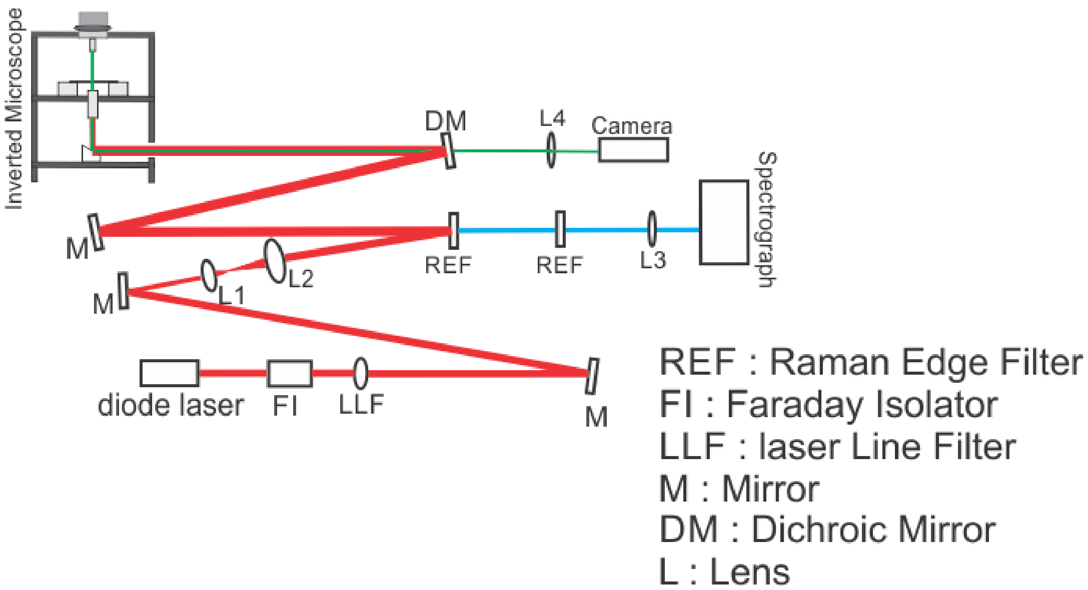


Figure 2.8 : Schematic view of single particle trapping set-up for Raman spectroscopy.

3. SPATIAL LIGHT MODULATORS BASED ON LIQUID CRYSTALS (LC-SLM)

As it was mentioned previously, many devices can be categorized as SLM. However, the most common devices in holographic optical tweezers are LC-SLM, either optically addressed or electrically addressed, which will be discussed later. Before explaining the structure of SLM, it is necesssary to take a look at liquid crystals at first.

3.1 Liquid Crystals

Discovery of LC's dates back to 1888, when Austrian botanist, F. Reinitzer realized that some materials have two melting points and tried to explain some specific characteristics of these exotic materials, such as ability to change the polarization of incident light beam. Nowadays, LCs play a great role in our life; the most prominent example is LC TVs and computer screens which use optical features of these materials. Although the name liquid crystal may sound unreasonable at first, it mentions great characteristic of this material. LC is a state which holds both properties of liquids, like fluidity and some of crystals, such as anistropy in optical features and specific ordering of molecules. To put it another way, LC is an intermediate state between liquid and crystal which is solid; that is why it is also known as meso-phases (between phases). LCs can be divided into two main groups: Thermotropic and Lyotropic. The exhibition of LC phase in both of these groups depends on the temperature and pressure, however Lyotropic LC's in addition need a solvent. Another classification is based on ordering of molecules. In nematic LCs, molecules are aligned in a specific direction which is called director. Smectic LCs are similar to nematic LCs, but with a difference that molecules are layered. Cholestric LCs exhibit chirality and have helical structure. They are also known as chiral nematic liquid crystals (CNLC) [17-19]. Figure 3.1 shows schematic view of these three kinds of LC's.

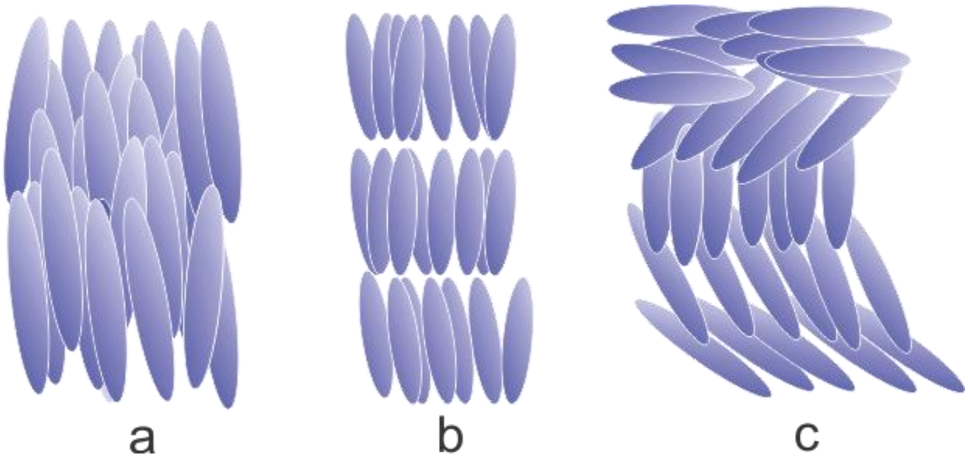


Figure 3.1 : Schematic view of a) Nematic b) Smectic c) Cholestric liquid crystals

Different kinds of LCs are used in optical devices depending on their features. The most important optical characteristic of LCs, double refractive index, is used in SLM to manipulate the intensity and phase of light. In LC-SLM, LC molecules are sandwiched between two plates. The structure of SLM cell will be discussed later. The most popular LCs exploited in SLMs are ferroelectric and nematic. In an external electric field, in a ferroelectric device (figure 3.2), molecules are perpendicular to the optical axis (figure 3.2a). However, by reversing the external electric field, molecules tend to rotate along the optical axis (figure 3.2b). As a result, optical characteristic of incident beam propagating through the LC layer can be changed after reflection from SLM. Generally, ferroelectric devices work in two different stable states; they respond fast (response time <1 ms). In contrast, nematic LCs (NLC), either parallel aligned or twisted are too slower in their response comparing with ferroelectric devices (approximately 20 ms) [20].

Figure 3.2 : Ferroelectric LC in an external electric field of a) $\vec{E}$ b) -$\vec{E}$

In nematic LCs molecules tend to align parallel to the molecule beside. Consider a case that the surface of the two plates which confine the liquid crystal are treated in such a way that they induce preferred orientation of LC molecules in their immediate vicinity using rubbing polymide layers (figure 3.3). In this figure, molecules have helical structure in whole of the cell. By applying an external field, helical structure will break; the optical characteristic of the cell will change due to this structural change. This is mostly used for intensity modulations by exploiting polarizer and analyzer before and after the LC cell [18, 21].

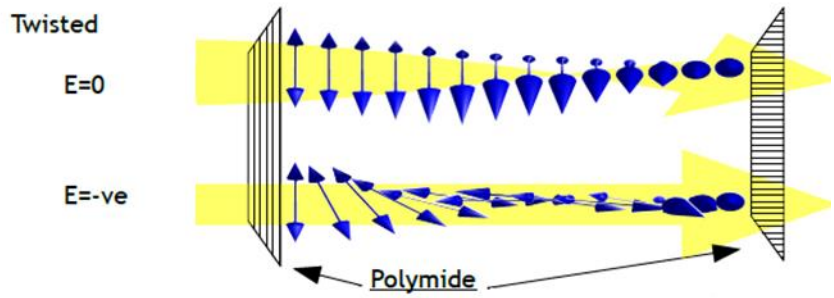


Figure 3.3 : Twisted nematic LC behaviour in an external electric field [18].

3.2 Structure of SLM Cell

LC-SLM can be either categorized as transmissive or reflective. In transmissive LC-SLMs, phase, intensity or both can be modulated after passing the light through LC layer. On the other hand, In reflective devices, the above-mentioned characteristics can be modulated while reflecting the SLM. Generally, SLM used in HOTs are phase modulating and reflective. We can also divide SLMs according to their addressing scheme. It can be either electrically addressed (EASLM) or optically addressed (OASLM).

3.2.1 Electrically addressed SLM

In both EASLM and OASLM, thick layer of LC (widely NLC) is sandwiched between two substrates. In EASLM, the front substrate is a transparent electrode; the substrate in backside is generally pixelated electrodes on silicon wafer such as a CMOS chip. All pixels can be controlled independently and simultaneously. In figure 3.4 schematic view of reflective EASLM is showed. By applying the voltage individually to each of pixels, LC molecules undergo a rotation that adjusts their optical behavior [18].

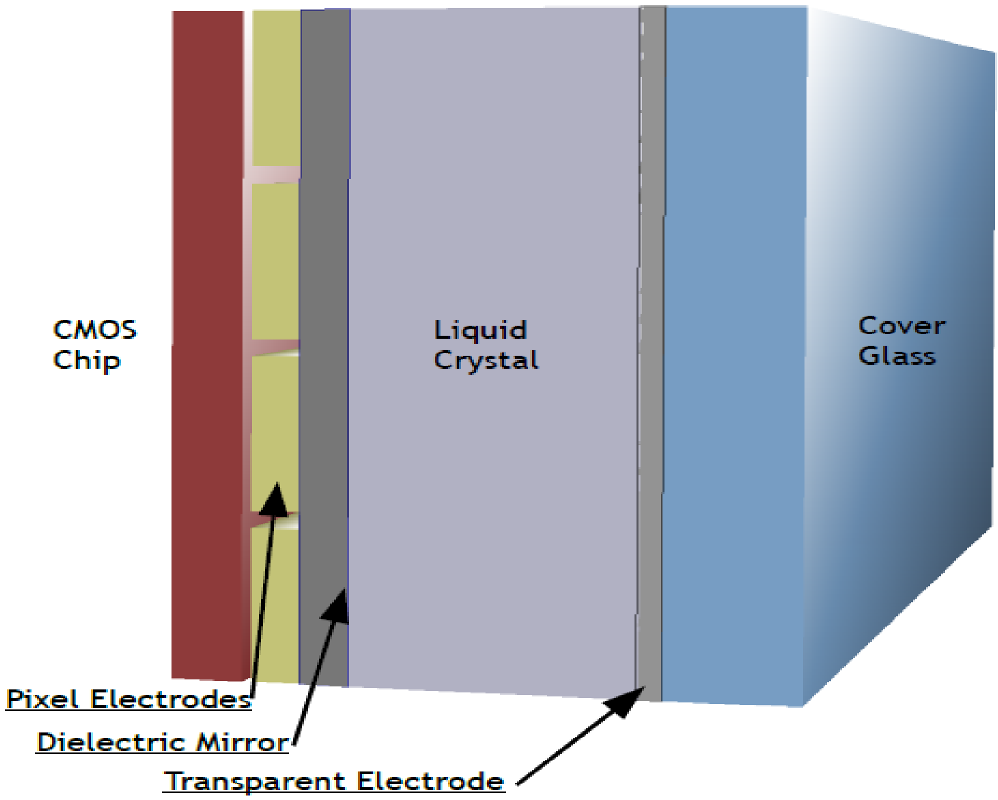


Figure 3.4 : Schematic view of electrically addressed SLM [18].

3.2.2 Optically addressed SLM

In contrast to EASLM, backplane in OASLM devices is not pixelated and a single monolith electrode is used instead. However, a thin layer of a dielectric material, generally photoconductive hydrogenated amorphous silicon (a-Si:H), is located between the electrodes (see figure 3.5). A homogeneous electric field is then applied to the electrodes. By illuminating, the resistance of dielectric material will change with regard to the intensity of the incident beam. This also changes the electric field inside the LC and, thus, the orientation of LC molecules. Major advantage of OASLM devices is that they have high fill factor (FF). FF is defined as portion of the surface which contributes to light modulation. FF in OASLM can even reach 100%. As it was mentioned previously, EASLM devices include pixelated surfaces. These pixels are potential causes of unwanted aberration in these devices; thanks to single monolith backplane, OASLM are superior from this point [18].

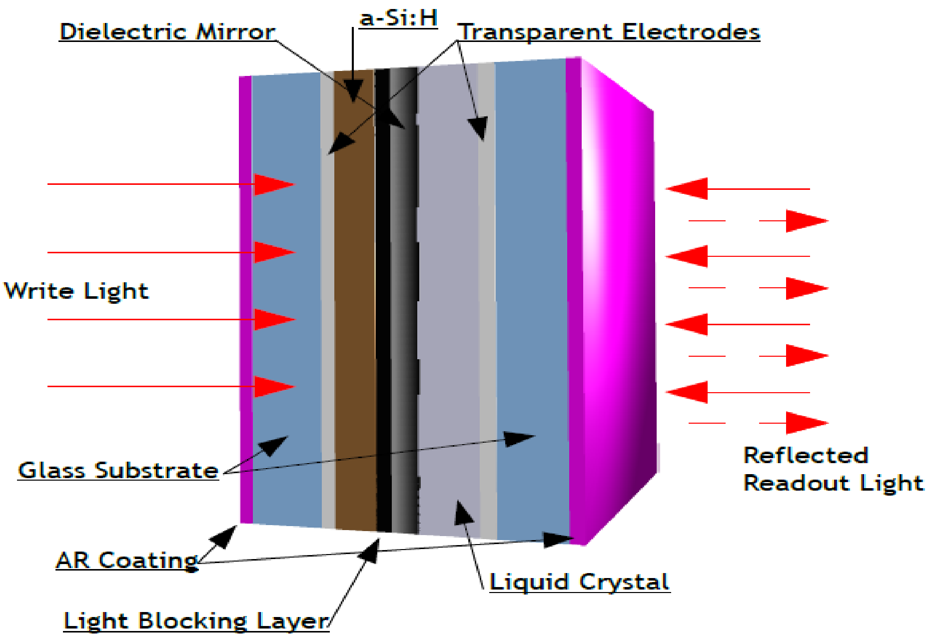


Figure 3.5 : Schematic view of optically addressed SLM [18].

3.3 Phase Modulation

Generally, Nematic LCs are used for phase modulating SLMs. Without external electric field, NLC tend to align all together in the same direction denoted by $\hat{\boldsymbol{n}}$ called director since dipole moments are induced by neighbouring molecules. In addition to that, NLCs in the absence of external electric field are uniaxially birefringent; n_e and n_o are refractive indices for light with polarization parallel and perpendicular to $\hat{\boldsymbol{n}}$.

Consider a light beam that propagates with angle θ with respect to $\hat{\boldsymbol{n}}$. In this case refractive index for polarization component in the plane of $\hat{\boldsymbol{n}}$ and propagation direction will be equal to:

$$n_{eff} = \sqrt{\frac{n_e^2 n_o^2}{n_e^2 cos^2\theta + n_o^2 sin^2\theta}} \quad \textbf{(3.1)}$$

In the presence of external electric field molecules will align in a way that elastic forces and torque could be balanced. Rotating the molecules unitedly causes the rotation of the director, too. Besides, refractive index will change because of this angle change. In SLM we can apply voltage to each of the cells so we can change their optical characteristics independently. This is the basis of phase modulation. Figure 3.6 shows the cross section of the SLM cell for phase modulation.

In this figure, (a) is in the absence of electric field. (b) and (c) are in the presence of external field (voltage is applied to substrates) however $V_2 > V_1$ [17, 18, 22].

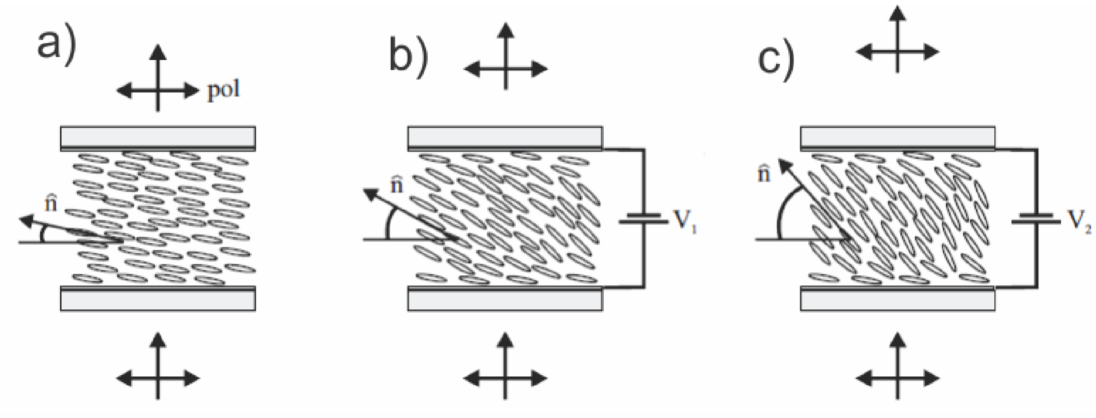


Figure 3.6 : Cross section of the SLM cell [17].

3.4 Specification of SLM Used in This Work

Liquid crystal based SLMs are manufactured by only a few companies. Major companies are Holoeye, Hamamatsu, and Boulder Nonlinear Systems. Each produce different kinds for different applications. We took several aspects into consideration for choosing an approppriate SLM. Some important charachteristics of the SLM used in this work will be discussed here. SLM that we used in this work is PLUTO-BB electrically addressed reflective phase only modulator with 1920 x 1080 resolution, manufactured by Holoeye company.

Figure 3.7 : PLUTO-BB SLM manufactured by HOLOEYE.

PLUTO version has several types, modified for different wavelengths. 2π Phase shift could be reached with PLUTO-BB in the wavelength range between 700 nm and 1000 nm. A driver is included for controling the matrix of cells. The signal is addressed via a standard DVI (Digital Visual Interface) port by a PC's graphics card and SLM works as a second monitor. It has a frame rate up to 60 Hz. Frame rate is crucial for generating holograms faster. Phase modulation for different wavelengths in PLUTO-BB type is plotted in figure 3.8 for default company configuration.

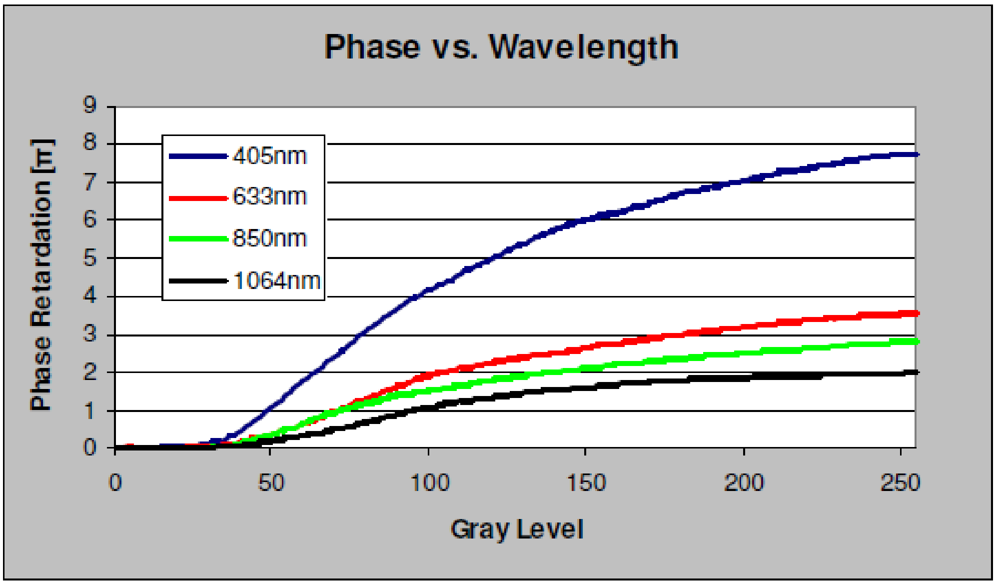


Figure 3.8 : Phase shift-gray level graph of PLUTO-BB.

Fill factor as it was defined previously is the percentage of the panel contributed for phase modulation. It has nearly 87% FF. Although EASLM devices have shorter respond time, their fill factor is lower.

Diffraction Efficiency (DE) of PLUTO-BB for 0^{th} order (non-addressed display) is around 60%. DE is defined as ratio of the power of the diffracted beam to the incident power.

The other important factor about PLUTO type is its highly programmable feature. An RS-232 cable and a interface programme enables the user to change and modify several display adjustments such as loading new gamma curve for gamma correction and loading various sequences. It is also possible to adjust dark and bright voltages using this interface which will be discussed thoroughly in further sections.

As it was indicated before, different versions of PLUTO are designed for different wavelengths. PLUTO-BB type enables 2π phase shift from 400 nm up to 1064 nm. It is also equipped with anti-reflective (AR) coating for low front-reflection. AR coating specification is shown in figure 3.9.

It is clear that in default configuration, different wavelengths will undergo different phase shifts for the same gray scale level applied to the SLM pixels. As we use 785nm and we need linear 2π phase shift, SLM must be modulated with regard to these parameters [23].

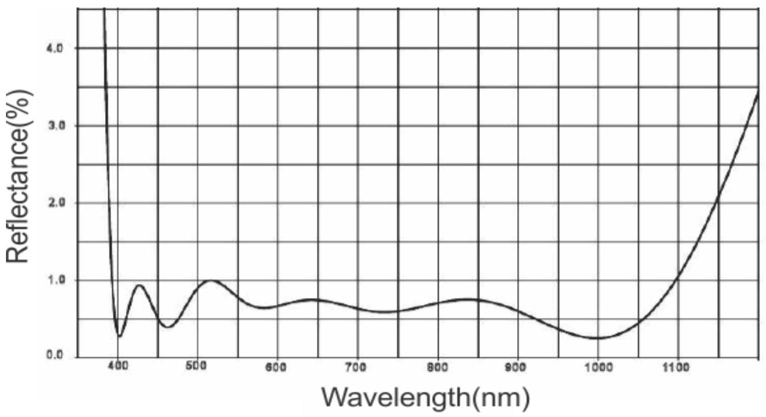


Figure 3.9 : Reflectance-Wavelength graph of SLM anti-reflective coating.

3.5 Calibration of SLM

As it is clear from figure 3.8, PLUTO SLM response for different wave lengths is distinctive. For instance, wavelegths around 1050 nm undergo 2π maximal phase shift, however beams with 400 nm wavelength can even undergo up to 8π phase shift in default factory configuration. In addition, for phase modulation we may need linear phase response. In order to obtain desired phase change with regard to input voltage we have to calibrate the device. To put it another way, calibration is performed to determine the relationship between gray level and phase shift to adjust it to the desired values as well as making the phase shift linear [24-28]. SLM has 256 phase levels (8-bit) addressing which means that the region between black and white is divided into 256 parts (gray levels). Gray levels are scaled from zero to 255.

PLUTO-BB type is connected to the computer using a DVI cable and functions as a second monitor. By extending the desktop screen, SLM can display any hologram or image on desktop.

In the proceeding sections, firstly, different methods for phase calibration will be provided. Then our phase calibration strategy and results will be discussed.

3.5.1 Methods for phase calibration

a) Michelson interferometer: By using a beam splitter we can divide the laser beam into two parts (figure 3.10). One is reflected from SLM surface (measurement beam) and the other one is reflected from a mirror (reference beam). These two beams interfere on the chip of a CCD camera. By scanning the gray level (phase level) on SLM and analyzing the interference pattern on camera one can determine phase shift with regard to gray levels [29].

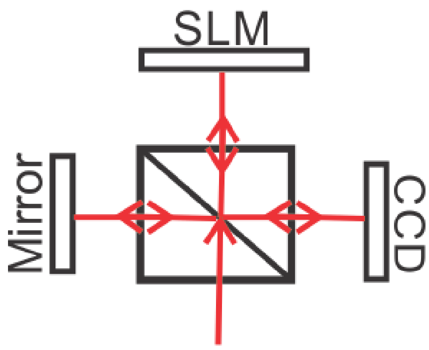


Figure 3.10 : Michelson interferometer method for SLM calibration.

b) Another method is dividing the light beam into two diffraction orders by using a suitable transmissive diffraction grating [30]. Each part hits different half of SLM. One half of SLM scans gray levels, however the other half is fixed in a stationary gray level. As described in the previous method we can determine phase shift by analyzing interference pattern in CCD camera (figure 3.11). The main drawback of these two methods is that some unwanted reflections and interference from beam splitter may make the phase measurements entangled.

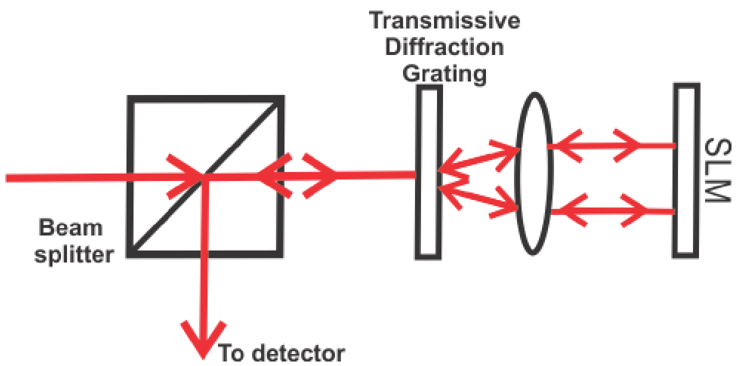


Figure 3.11 : Schematic view of SLM calibration method using grating.

c) Similar method is proposed by Holoeye manual. However, a double slit is used instead for splitting the light beam. Similar to previous methods, SLM chip is divided into two halves. One half is fixed in a stationary gray level, but the other half scans through gray levels. Similar to previous methods, analyzing the interference pattern provides phase shift with regard to gray level change. As we use this method for PLUTO-BB calibration we will discuss it further in the next sections.

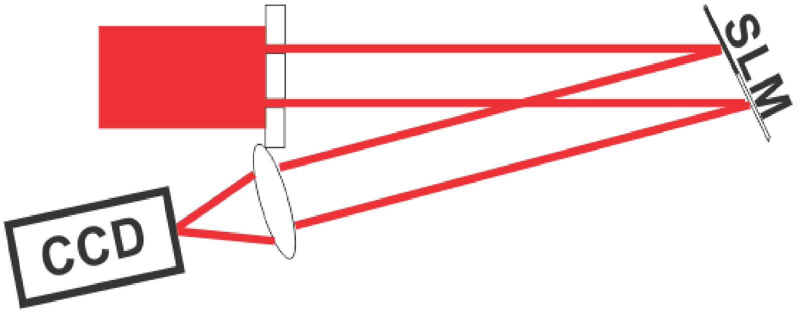


Figure 3.12 : Schematic view of SLM calibration method using a double slit.

3.5.2 Amplitude modulation

Amplitude modulation is based on polarisation change. Generally a pure polarized light undergoes a polarization change after reflecting from SLM. In order to analyze the polarization state of the reflected light, a polarizer is mounted just after the SLM to filter the light with changed polarization (figure 3.13) when the SLM scans the gray levels. This process results in amplitude modulation; amplitude change can be observed using a CCD camera or power meter. PLUTO-BB type is phase mostly device so it is mainly designed for phase changes rather than amplitude, however a similar experiment was done to ensure this statement, which will be provided in the next sections [30, 31].

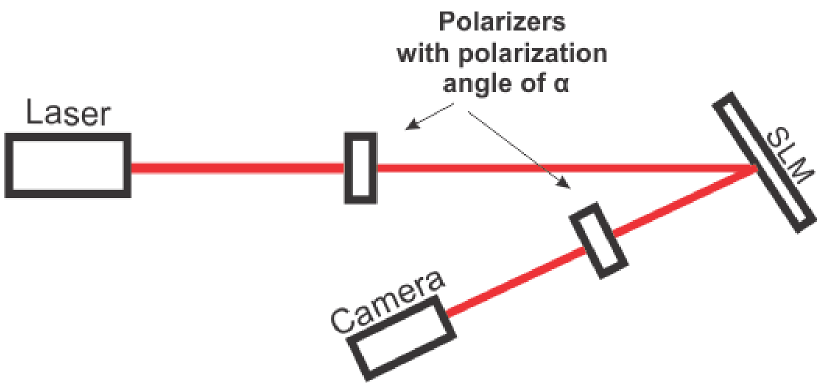


Figure 3.13 : Schematic view of a set-up for amplitude modulation.

3.5.3 Performed calibration method, results and discussion

From several above mentioned methods, double-slit experiment was chosen for SLM calibration. A 785 nm laser with output power of ~90 mW is used for illumination. Diameter of laser beam is 1.2 mm in front of the laser. A neutral density filter is used to decrease the intensity down to severel milliwats for preventing any damage to CCD camera. 10x MO as well as a lens with 75mm focal length is used for expanding the laser beam up to 9 mm which fills SLM active area. A pinhole is also used between MO and lens for spatial filtering. As the output beam from the laser has horizontal polarization and the SLM functions effectively for vertical polarization, a half wave plate is used for changing the polarization of the incident light beam.

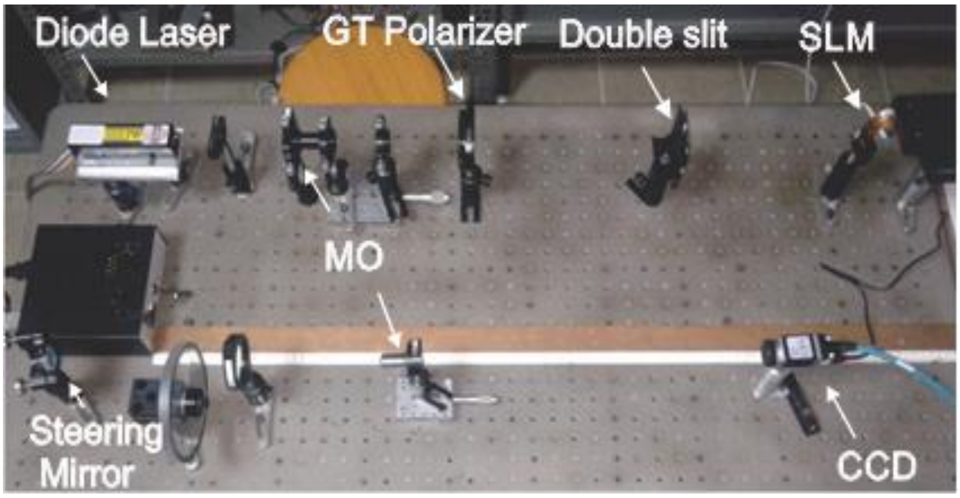


Figure 3.14 : Image of calibration set-up.

A double slit is designed in order to split the beam into two parts. Each slit is 1 mm in width and 4 mm in length. Each of the splitted beams illuminates different half of the SLM. The two beams are reflected off the SLM and are guided into 20x MO through a mirror. By using another MO, we image the interference pattern in CCD camera which occurs in focal point of lens. During the calibration procedure, one side of the pixelated SLM is stationary, however the other side scans through gray levels (phase levels). Changing the phase level of one side will cause a shift in the interference pattern. SLM is connected to the computer using a DVI cable and functions as a second monitor. By extending the desktop screen, SLM can display any hologram or image on desktop. More detailed schematic view of calibration set-up is given in figure 3.15.

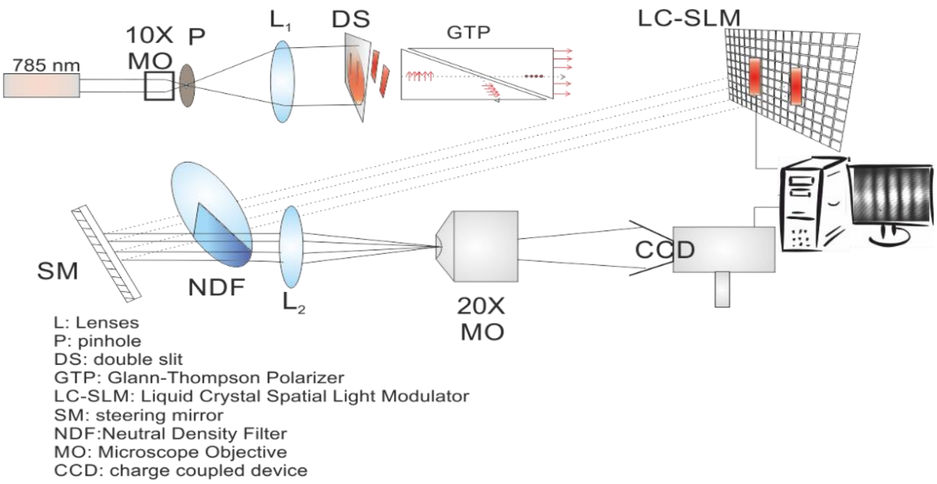


Figure 3.15 : Schematic view of calibration set-up.

MATLAB GUI is designed to perform calibration measurements (See AppendixA). At first step a color map is defined in order to divide the region between black and white into 256 gray levels. In the next step, a 1080x1920 matrix called Z matrix is created and adjusted on the screen related to the SLM. As it was indicated previously, Z matrix is also divided into two Z1 and Z2 matrix equal in size. Z2 matrix is fixed in stationary state (black) or 1st gray level during the measurement.

Since the region between black and white colors is divided into 256 (8-bit addressing), we may also have 256 steps.

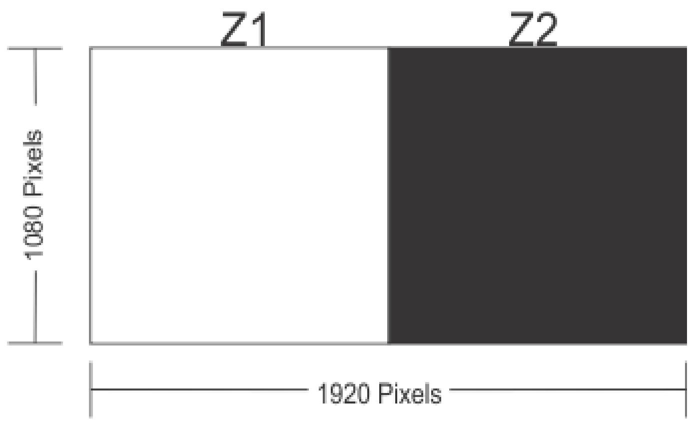


Figure 3.16 : Imaged view of Z matrix.

Z1 matrix scans through gray levels. In each step the program captures and averages three pictures from interference pattern and saves it to a three dimentional matrix called matrix of images by averaging these three images. At 256th step, we have 256 matrices of size 640 x 480, Which are camera picture dimensions.

```
 % A colormap matrix shoud be defined for 256 gray levels.
for i = 1 to 256 steps
 colormap = Gray level
end
for i = 1 to 256
 % Z1 and Z2 are matrices with dimension of 1080 x 1920 (SLM pixel dimensions)
 Z1 = i;
 Z2 = constant;
 Z = [Z1,Z2]; % matrix that will be imaged.
 image (Z);
 trigger camera
 % camera triggers 3 times.
 % Average of images are saved to "matrix of images".
end
```

In the next step program requests user to select a horizontal slice to continue (figure 3.17). After selecting a suitable slice of a sample picture, program calculates the average of each column in the slice. A slice is considered as suitable when interference pattern is clear enough for evaluation. Program uses the same slice for applying the averaging process for each of 256 steps. We would obtain a row which represents each picture and its peaks and troughs. In the next step program creates a single matrix called transfer matrix whose rows represent average line intensity profiles of individual images from the recorded stack..

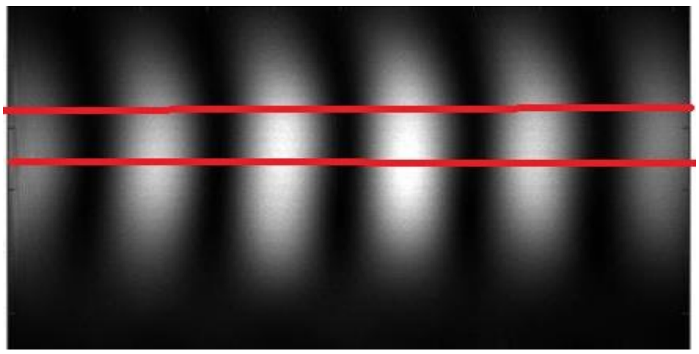


Figure 3.17 : Slice selection process from interference sample image.

```
   % vertical Slice = defined by user from sample image;
for i = 1 to 256
   % each row in transfer matrix is equal to a slice.
```

i^{th} row in transfer matrix = selected slice in i^{th}
"matrix of image"
end
for 1 to number of rows in transfer matrix
fit value = fitted "transfer matrix"
% each row in transfer matrix is fitted to a polynomial
end

Analyzing transfer matrix plays a major role in determining the phase shift caused by gray level changes. In each row, the distance (pixel numbers) between two successive peaks (or troughs) is equal to 2π.

In the next pace, user must choose brightnesses (intensity maximum) in transfer matrix to calculate the phase shift by considering the pixel difference between two successive peaks as 2π. Brightness selection is done by tracking the brightness by straight lines. For decreasing unwanted fluctuations in camera pixels caused by laser or external effects, polynomial fitting applied for each row of transfer matrix before selecting the brightness by user.

% right and left borders of a brightness defined by user.
Maximum of selected brightness = measured.
2π phase shift = pixel difference between two successive brightnesses.
Phase shift in each brightness = (pixel change in brightness * 2π)/pixel difference of 2π

Figure 3.18 shows a tranfer matrix image and a selected brightness on it before calibration. Program calculates the peak points after selecting the brightness. These peak points are then analyzed to calculate the phase shift for each brightness.

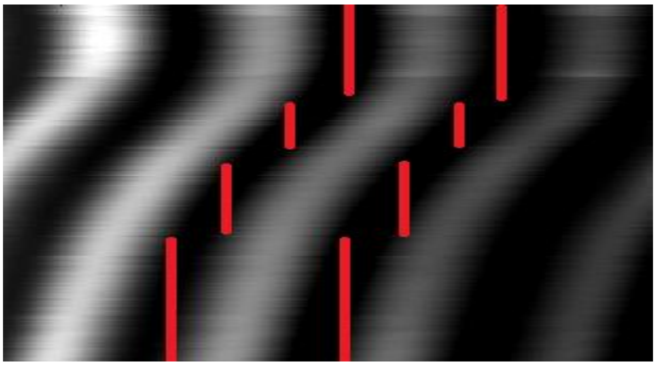


Figure 3.18 : Interference pattern shift in default SLM configuration and brightness selection process.

Phase shift-Gray level graph of this transfer matrix is given in figure 3.19. A non-linear phase shift with a maximum of 2.76 π was measured for the above mentioned transfer matrix.

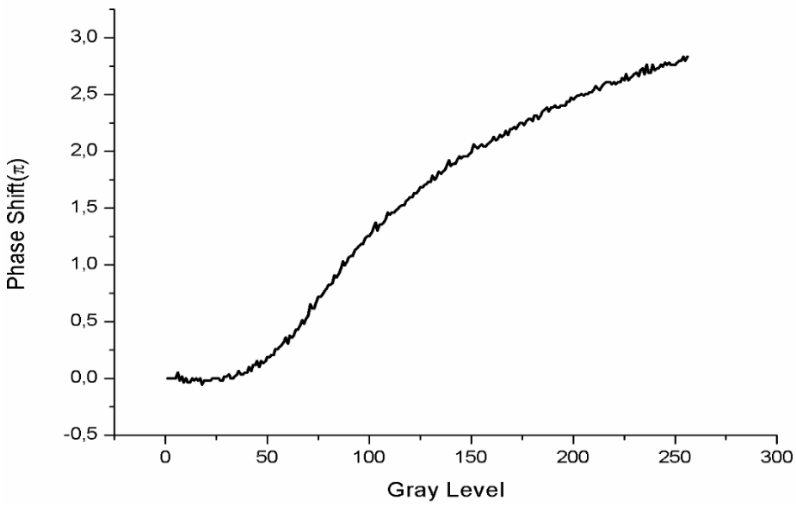


Figure 3.19 : Phase shift-Gray level graph in SLM default configuration.

Several voltages were tried for darkest and brightest gray levels to reach 2π phase shift. Some of them are given in the table below. V_{dark} and V_{bright} can be adjusted by Holoeye interface programme. V_{dark} = 0.03V and V_{bright} = 2.87V were found to be the best voltage configuration in order to reach 2π phase shift.

Table 3.1 : Measured phase shift for some dark and bright voltages.

V_{dark} (V)	V_{bright} (V)	Phase shift (π)
0,03	3,79	2,76
0,03	3,26	2,49
0,56	2,47	1,65
0,56	2,74	1,81
0,03	2,87	2,03

Transfer matrix image for V_{dark} = 0.03V and V_{bright} = 2.87V is shown in figure 3.20 after gamma correction. Gamma curve is an internal look-up table that transforms the input voltage to a suitable voltage for LC in order to obtain linear phase shift with linearly increasing gray levels. Gamma correction is done by a default excel document designed by the manufacturer. In 5:5 bitplane configuration, 192 discrete voltage levels are available. Thus, we have to map 256 Gray Levels to 192 values in order to get as linear phase shift as possible.

Figure 3.20 : Interference pattern shift after gamma correction.

Phase shift-Gray Level graph of transfer matrix above is shown in figure 3.21.

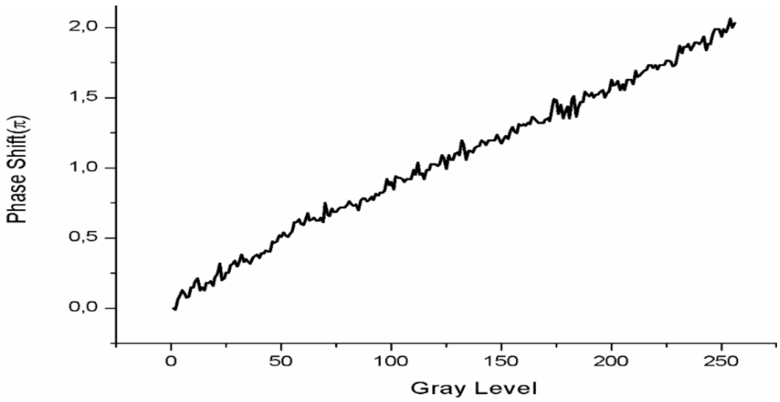


Figure 3.21 : Phase shift-gray level graph after gamma correction.

After calibration of the device and adjusting suitable voltages for brightest and darkest gray levels and also loading the corrected gamma curve, we have generated a hologram to obtain first-order Laguerre-Gaussian beam [32]. In this beam profile wave fronts form a helix. Helical structure of wave fronts for different modes of Laguerre-Gaussian beams are shown in figure 3.22.

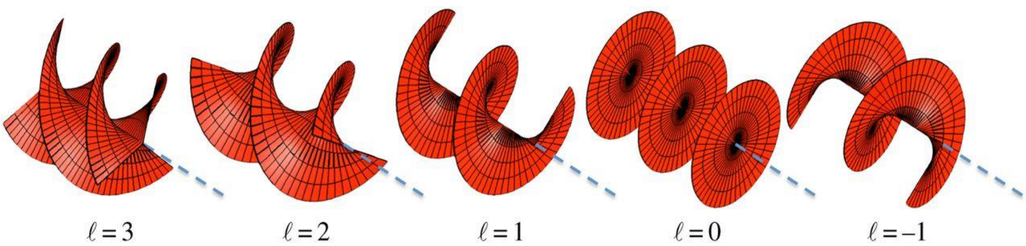


Figure 3.22 : Wave fronts for modes of Laguerre-Gaussian beam profile [33].

The hologram for obtaining first mode of Laguerre-Gaussian beam was generated in home-made program. Figure 3.23 and 3.24 show the hologram and intensity profile that we have achieved.

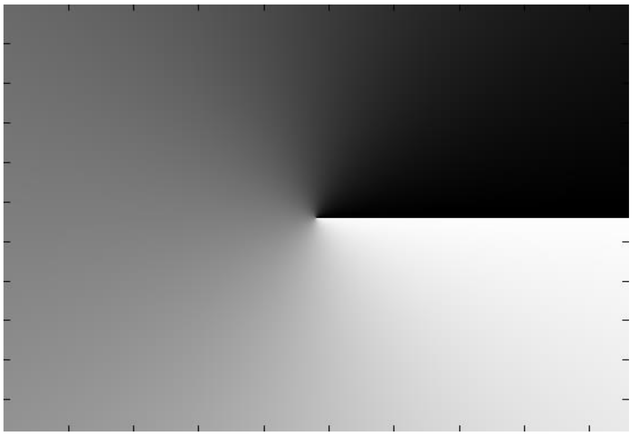


Figure 3.23 : Generated hologram for first mode of Laguerre-Gaussian profile.

Donut-shaped beam profile was obtained for above-shown hologram.

Figure 3.24 : First mode of Laguerre-Gaussian beam profile image on a screen.

Another experiment was also done to measure the changes in the intensity due to gray level change. In this experiment, identically to the previous measurements, a double slit was used to divide the light beam into two beams. Each beam strikes different half of SLM. One half is fixed in an stationary gray level, however other one scans through all 256 levels. The intensity in front of the SLM was measured by a power meter after reflection. Intensity-gray level graph is given in figure 3.25. As it is seen in this figure both beams do not experience any extreme intensity change due to gray-level scanning. Another similar experiment was done to measure the polarization change in the beam reflected from SLM plane. Similar to previous experiment, intensity was measured but after placing a polarizer. Intensity-gray level graph of this experiment is shown in figure 3.26.

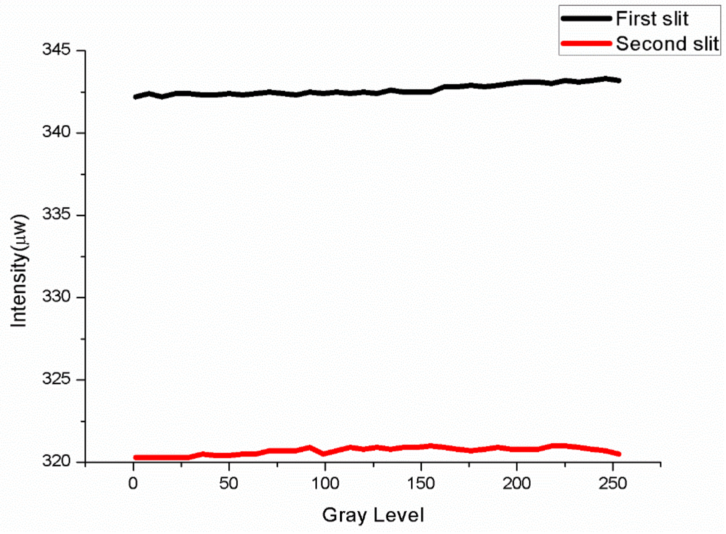


Figure 3.25 : Intensity-gray level graph measured after reflection off the SLM.

As it is clear from this figure, one of the beams undergoes a slight polarization change (first slit).

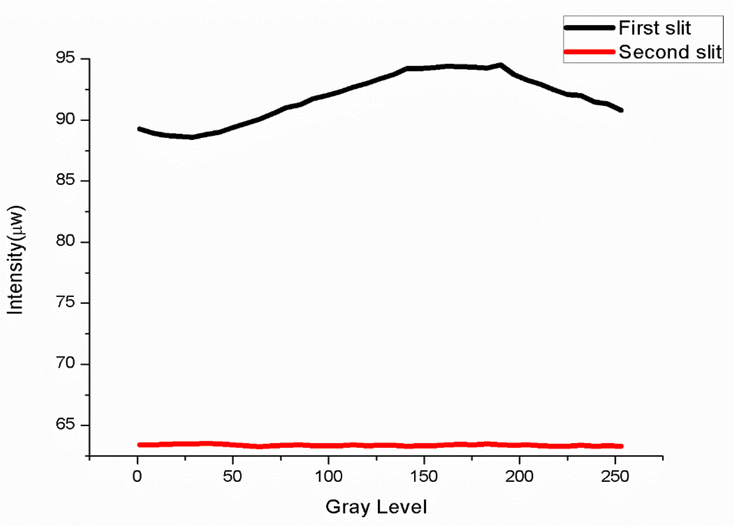


Figure 3.26 : Intensity-gray level graph measured after placing a polarizer.

4. HOLOGRAPHIC BEAM SHAPING FOR OPTICAL TWEEZERS

To create and manipulate desired trap positions and intensities we have to find suitable optical fields on SLM's screen. The first major problem is to calculate the desired optical traps positions and relate them to the hologram shown on SLM plane. The field will produce the traps after propagating in optical system. In fact generating the calculated hologram on SLM is far from reality because of pixelated characteristic of the chip and limited gray level, however we will try to get as close as ideal hologram obtained from mathematical calculation with different algorithms that will be discussed in next sections. Since phase mostly SLMs are used for holographic beam shaping, the main idea in calculating the hologram is that each pixel can change the phase from zero to 2π. Scalar diffraction theory plays a major role in calculating the propagation so we will discuss it in the next section.

4.1 Scalar Diffraction

In this section we will discuss scalar diffraction theory that deals with wave propagation and is used for calculating the hologram [17, 34].

In 1678 Christiaan Huygens, attempted to describe the diffraction phenomenon. According to Huygens, Any point on wavefront can be imagined as a new source for secondary wavefronts. In fact, the wavefront in other instances could be found by constructing wavelets. Another prominent work was done by Thomas Young in 1804. He brought in the concept of interference to the wave theory of light. In the 19th century, Augustine Jean Fresnel, introduced the Huygens-Fresnel principle. Fresnel obtained a mathematical description for diffraction of light when it passes through an aperture. Consider a case with a point source with vibration frequency of f. Complex amplitude of U_0 describes the disturbance of this point source. Complex amplitude at a distance r_0 from the point source is described by:

$$U(r_0) = \frac{U_0 e^{ikr_0}}{r_0} \tag{4.1}$$

In which, λ is wavelength and k is wave number $\frac{2\pi}{\lambda}$. Complex amplitude at further point P can be found by summing all contributions of source points lying on circle (sphere in 3D) with radius r_0 using the principle of super-position. In order to obtain agreement with experimental results, Fresnel found that two additional multiplication factors must be included in the above mentioned formula:

$$U(P) = \frac{-i}{\lambda} U(r_0) \int_S \frac{e^{iks}}{s} \mathrm{K}(\chi) dS \tag{4.2}$$

In which U(P) is the complex amplitude at further point P and $\mathbf{K}(\chi)$is the inclination factor. S denotes integration over the surface area and s is the distance between source point and P. In 1882 Gustav Kirchhof combined Fresnel and Maxwell ideas and showed that by assuming two boundary conditions of the light incident on the surface, similar derivation could be reached by applying Green's theorem to Helmholtz equation in the case of monochromatic wave. According to Kirchhoff's boundary conditions, Both field and its derivative in the plane outside the aperture are assumed to be zero. Although these boundary conditions were not absolutely true, Kirchhoff's treatment showed excellent agreement with experimental results. In 1896 Arnold Sommerfeld modified the theory presented by kirchhoff. Using Greens theorem and implementing other boundary conditions, Sommerfeld obtained an expression which is known as Rayleigh-Sommerfeld (RS) diffraction integral. Optical field in 3D at point (u,v,z) for z > 0 is given by:

$$U(u, v, z) = \frac{1}{i\lambda} \iint_{\Sigma} U(x, y, 0)\, e^{ikr} \frac{\cos\theta}{r} dxdy \tag{4.3}$$

Σ is the surface of aperture in z = 0, $U(x, y, 0)$ is the field in aperture and $\vec{r}$ is the vector between the point in the aperture and desired point. In this case $|\, r|$ will be equal to $\sqrt{z^2 + (x-u)^2 + (y-v)^2}$. The angle between aperture surface normal and this vector is equal to θ. Analytical solution for this integral is impossible, however some geometrical simplification could be applied. The most popular simplification is paraxial approximation which is valid for small values of angle θ. Using Taylor series expansion below:

$$\sqrt{1+u} = 1 + \frac{u}{2} - \frac{u^2}{8} + \cdots \qquad \textbf{(4.4)}$$

We can rewrite r as:

$$r \approx z + \frac{(x-u)^2 + (y-v)^2}{2z} \qquad \textbf{(4.5)}$$

By replacing this expression in the main formula we will obtain Fresnel diffraction integral:

$$U(u,v,z) = \frac{e^{ikz}}{i\lambda z}\int_{-\infty}^{+\infty} U(x,y,0)\, e^{i\frac{k}{2z}[(x-u)^2+(y-v)^2]} dxdy \qquad \textbf{(4.6)}$$

Now let us consider a simple case in which a lens with focal length of f is mounted in front of the SLM as in figure below:

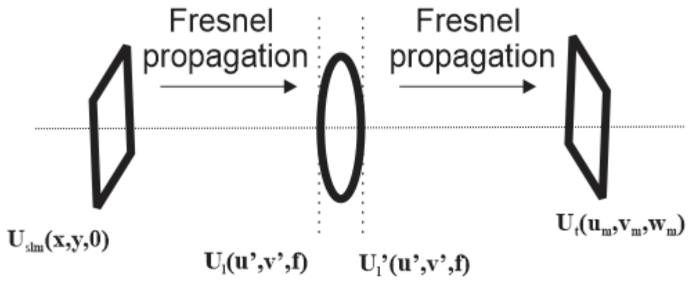


Figure 4.1 : Schematic view of Fresnel propagation from SLM plane to image plane.

For obtaining desired trap positions in back focal plane we have to calculate SLM field propagation up to this plane called as target plane through lens and free space. Thus we have to apply Fresnel diffraction integral to the SLM field U_{slm}= U(x,y,0):

$$U_l(u',v',f) = \frac{e^{ikf}}{i\lambda f}\iint_{-\infty}^{+\infty} U_{slm}(x,y) e^{\frac{ik}{2f}\left[(x-u')^2+(y-v')^2\right]} dxdy \qquad \textbf{(4.7)}$$

In which u' and v' are coordinates on front plane of the lens. By multiplying $e^{\frac{-ik}{2f}(u'^2+v'^2)}$ we can find the field on the other plane of the lens (thin lens approximation). For field propagation from lens to target plane we have to apply diffraction integral one time more, which yields:

$$U_t(u,v,w) = \frac{1}{i\lambda f} e^{i\frac{2\pi(2f+w)}{\lambda}} \iint_{-\infty}^{+\infty} U_{slm}(x,y) e^{-i\frac{\pi w}{\lambda f^2}(x^2+y^2)} e^{-i\frac{2\pi}{\lambda f}(xu+yv)} dxdy \qquad \textbf{(4.8)}$$

In this expression, w is spot's axial deviation from target plane. Here we have considered that the lens aperture is big enough to transfer all the light diffracted from SLM. However, in real cases we have to apply another term for lens aperture as well as lens function. Since the SLM screen is pixelated we have to change the integral to summation. By omitting normalization factor we have:

$$U_t(u_m, v_m, w_m) = e^{i\frac{2\pi w_m}{\lambda}} \sum_{x,y} U_{slm}(x,y) e^{-i\left[\frac{\pi w_m}{\lambda f^2}(x^2+y^2)+\frac{2\pi}{\lambda f}(xu_m+yv_m)\right]} \qquad \textbf{(4.9)}$$

Here m is assigned to trap numbers. For finding the U_{slm} we can consider field propagation from target plane to SLM plane, or in other words the inverse of the previously mentioned process. Thus:

$$U_{slm}(x,y) = \sum_m U_t(u_m, v_m, w_m)\, e^{i(-\frac{2\pi w_m}{\lambda}+\frac{\pi w_m}{\lambda f^2}(x^2+y^2)+\frac{2\pi}{\lambda f}(xu_m+yv_m))} \qquad \textbf{(4.10)}$$

It is clear that by applying this equation to desired values of U_t we can obtain field of hologram on SLM. As $\frac{\pi w_m}{\lambda f^2}(x^2+y^2)$ and $\frac{2\pi}{\lambda f}(xu_m+yv_m)$ are phase changes due to lens and prism respectively, this method is known as Prisms and Lenses algorithm. We will discuss it further in section 4.2.1.

The general form of an optical beam can be shown by a complex field of $U = aexp(i\phi)$, in which a is the amplitude and ϕ is the phase. Since we use phase modulation for optical trapping, we may consider that the beam after striking the SLM screen will undergo a phase change. Let us consider ϕ_h the phase change due to SLM. Hence, the general form of the beam after hitting the screen of the SLM will be $U_h = a\exp[i(\phi+\phi_h)]$. Algorithms that we will discuss serve to calculate and implement this phase shift ϕ_h. Thanks to Fourier transformation we are able to calculate and implement the hologram for creating desired spots in lens focal plane. In other words Fourier transformation of hologram on SLM screen will give desired spots in Fourier plane (focal plane of lens).

$$U_i(x_i, y_i) = \mathcal{F}\{U_h(x_h, y_h)\} \qquad \textbf{(4.11)}$$

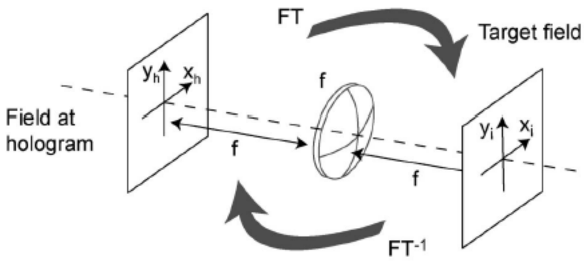


Figure 4.2 : Schematic view of transformation of SLM and Fourier plane [13].

If one of $U_i(x_i, y_i)$ or $U_h(x_h, y_h)$ is given we can find other one by applying inverse or forward Fourier transformation. It is also clear that target intensity is equal to $|U_t(x_i\,, y_i)|^2$ [17, 34].

4.2 Algorithms for Generating Holograms

4.2.1 Grating and lenses

Grating and Lenses algorithm is the fastest and the most straightforward algorithm for generating holograms. In this method phase shift introduced by basic optical devices are combined to obtain desired spots. Before discussing this method let us consider how some simple optical devices such as prism and lens alter phase. Figure 4.3 shows a plane wave incident on one surface of prism. The refractive index is equal in all part of the prism. However, optical depth varies because of changing thickness of the prism. Hence the beam undergoes a different phase changes in different parts of the prism with different thicknesses. More phase altering happens in the thickest part of the prism. Due to this phase change and refraction on prism surface, propagation direction of the plane wave changes.

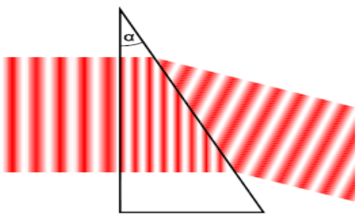


Figure 4.3 : Schematic view of plane wave hitting a prism.

In the lenses either concave or convex the same process happens. Again consider a case that plane wave hits the surface of a lens. In larger thickness of the lens, phase shift will be greater, so by changing the propagation direction the beam will be convergent or divergent.

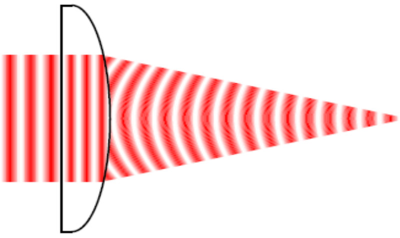


Figure 4.4 : Schematic view of plane wave hitting a lens.

As we have discussed previously, an optical beam is represented by $a \exp i(\phi + \phi_h)$ after striking the SLM. If ϕ_h is zero (non-addressed SLM), single spot will be created in the Fourier plane. It is easy to change the position of this spot by showing lens and prism holograms on SLM. For instance we can alter its position in XY plane and in Z direction by generating prism and lens holograms respectively. Phase change that a lens applies on a beam is equal to:

$$\phi_{lens}\ (x, y) = \frac{\pi w_m (x^2 + y^2)}{\lambda f^2} \qquad \textbf{(4.12)}$$

In which, w_m is spot deviation from Fourier plane in Z direction. x and y are coordinates on the lens (Lens Hologram). λ is wave length and f is the focal length of additional Fourier lens.

Phase change that a prism applies on incident beam is:

$$\phi_{prism}\ (x, y) = \frac{2\pi(ux + vy)}{\lambda f} \qquad \textbf{(4.13)}$$

In this formula u and v are coordinates of the spot in the Fourier plane. If we want to have both lateral and axial change in the spot position we have to combine these two phase changes. Since the SLM can introduce maximal phase shift of 2pi, this is done by modulo 2π:

$$\phi_h = (\phi_{lens} + \phi_{prism})\ mod\ 2\pi \qquad \textbf{(4.14)}$$

For creating the spot in u,v and w coordinates in Fourier plane we have to calculate this hologram and show it on SLM. For creating several traps we have to calculate necessary phase shift for each one independently and add all of them. In this case:

$$\phi_h = arg\left(\sum_{n=1}^{N} U_{h,n}\right) \tag{4.15}$$

In this formula, N is the total number of spots[13, 22, 35].

4.2.2 Two-dimensional Gerchberg–Saxton algorithm

This algorithm is also based on Fourier transformation. Multiple traps in Fourier plane can be created by Gerchberg–Saxton algorithm. Iteratively optimizing the intensities in Fourier plane by changing the ϕ_h is the basis of this algorithm. Consider that I_t is desired trap intensity in Fourier plane, our purpose is to find ϕ_h in a way that $\mathcal{F}\{\exp(i\phi_h)\} = I_t$. For accomplishing this we start with a random ϕ denoted by ϕ_r.

For first iteration we have:

$$u_{h,1} = \exp(i\phi_r) \tag{4.16}$$

In each step we propagate the field of hologram to Fourier plane by taking its Fourier transformation. Thus for l^{th} iteration we have:

$$u_{i,l} = \mathcal{F}\{u_{h,l}\} \tag{4.17}$$

Phase in image space will be:

$$\phi_{i,l} = \arg(u_{i,l}) \tag{4.18}$$

We replace the new field:

$$u_{i,l'} = \sqrt{I_t}\exp(i\phi_{i,l}) \tag{4.19}$$

By taking its inverse Fourier transformation, we propagate back to SLM plane:

$$u_{h,l'} = \mathcal{F}^{-1}\{u_{i,l'}\} \tag{4.20}$$

By applying the same process:

$$\phi_{h,l+1} = \arg(u_{h,l'}) \qquad \textbf{(4.21)}$$

$$u_{h,l+1} = \exp(i\phi_{h,l+1}) \qquad \textbf{(4.22)}$$

After several iterations, hologram will be convergent in a way that argument of the field on SLM is desired hologram for obtaining spots [13].

4.2.3 Random mask encoding technique

In this method, SLM pixels are divided into k subdomains, in which k is the number of desired traps. Again consider the general form of the field after reflection from SLM plane, $U_h = a \exp i(\phi + \phi_h)$, ϕ and ϕ_h are phase of the field before striking the SLM, and phase change due to hologram shown on SLM, respectively. Similar to prisms and lenses algorithm we exploit prism phase change formula on SLM:

$$u_h = \sum_{k=1}^{N} e^{i\frac{2\pi}{\lambda f}(xu_k + yv_k)} \qquad \textbf{(4.23)}$$

Where xy and uv are coordinates on SLM plane and image plane respectively. Thus in Fourier plane we have:

$$u_i = \sum_{k=1}^{N} \iint e^{-i\frac{2\pi}{\lambda f}(x(u-u_k)+y(v-v_k))} dxdy = \sum_{k=1}^{N} \delta(u - u_k, v - v_k) \qquad \textbf{(4.24)}$$

Random mask encoding technique is a non-iterative algorithm that exploits a linear multiplication:

$$u_h = \sum_{k=1}^{N} h_k(x,y) e^{i\frac{2\pi}{\lambda f}(xu_k + yv_k)} \qquad \textbf{(4.25)}$$

In which:

$$h_k(x,y) = \begin{cases} 1 & if\ x, y \in I_k \\ 0 & other\ wise \end{cases} \qquad \textbf{(4.26)}$$

Simply, in this method whole modulator area (pixalated area) is divided into subdomains in a way that these domains do not overlap. The major advantage of random mask technique is that each domain is assigned to a single trap and can be updated independently so it is fast relative to other methods. In contrast, changing spot's location in prisms and lenses algorithm and Gerchberg–Saxton Algorithm, needs whole SLM chip calculation and update. The other important factor that is reported for this method is the absence of ghost traps .

On the other hand, the major drawback of above mentioned technique is low efficiency. By adding traps the efficiency (ratio of trap energy to the whole energy), decreases gradually [36].

4.3 Holographic Optical Tweezer Set-Up And User Interface Interactive Program Development

We have constructed a Holographic Optical Tweezer set-up (HOT) and designed a graphical user interface (GUI) in order to calculate and create holograms using a spatial light modulator controlled by the prisms and lenses algorithm described in Section 4.2.1.

Our designed HOT set-up and its schematic view are shown in figure 4.5. Several changes were made in single particle trapping set-up section presented in 2.2 for HOT system construction.

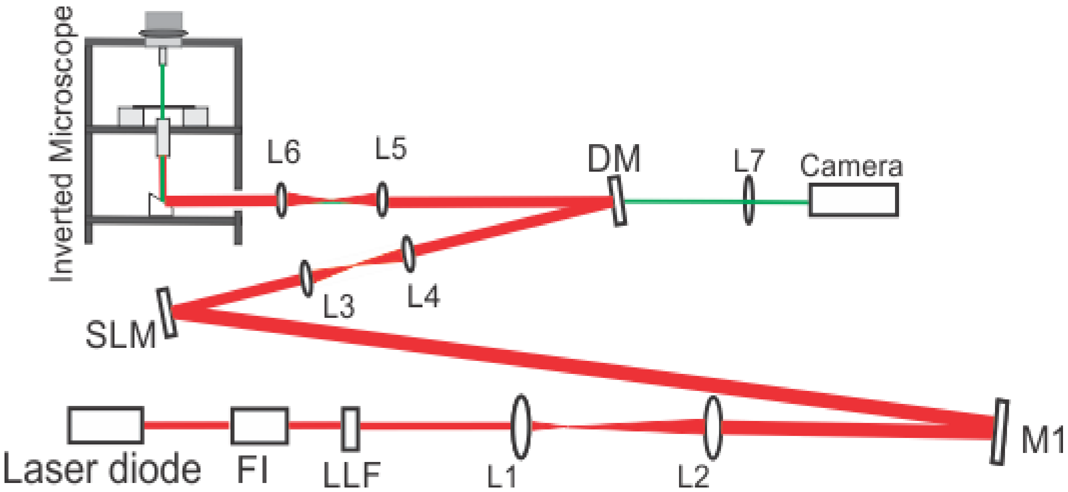


Figure 4.5 : Schematic view of designed HOT set-up.

Raman edge filter was eliminated in order to simplify the set-up. An expanding system of Keplerian Telescope including L1 and L2 Lenses as well as a pinhole for spatial filtering was used. Focal length of L1 and L2 are 100mm and 750mm respectively. Similar to single particle trapping set-up a Faraday isolator is used in order to prevent unwanted feedbacks into diode laser. Just after the FI, a LLF was used to provide high spectral resolution. Expanded laser beam is guided to SLM through a dielectric mirror. SLM's height is adjusted to 8.5 cm (Height of inverted microscope entrance). We have observed that without using appropriate telescope systems, beams from SLM overflowed the MO and even the mirror just before the inverted microscope. Thus, we have used two telescope systems, first between SLM and dielectric mirror and second between this dieletric mirror and the inverted microscope in order to prevent overflowing the beam from optical elements. A CCD camera was also placed after dielectric mirror. This mirror reflects 98% of rays with wavelength more than 600 nm. In contrasts, wavelength less than that are mostly transmitted. So we are able to see inside the sample by illuminating it with an LED.

The other important factor is that, lenses used in telescope systems, in front of the camera can cause some unwanted reflections in camera. To decrease these feedbacks into CCD camera, usage of anti-reflection-coated lenses is inevitable. Again similar to single particle-trapping set-up, one side of the hole in microscope slide is covered with a cover slip before putting the sample liquid. The other side is covered with another cover slip after injecting the sample into the hole. Generally, in our HOT experiments we used polystyrene beads with 4μm mean diameter diluted in water.

As it was mentioned previously, waves with different incident angles on SLM, experience different effective refractive indices because of anisotropy in LC molecules; in most of NLC devices, change in diffraction efficiency is lower than 15% for incident angles less than 30°; that's why we tried to decrease this angle as much as possible [18].

We have also developed a user interface interactive program in order to control and manipulate particles in 3D independently. Graphical user interface GUI of this program is shown in figure 4.6 (See Appendix B for codes).

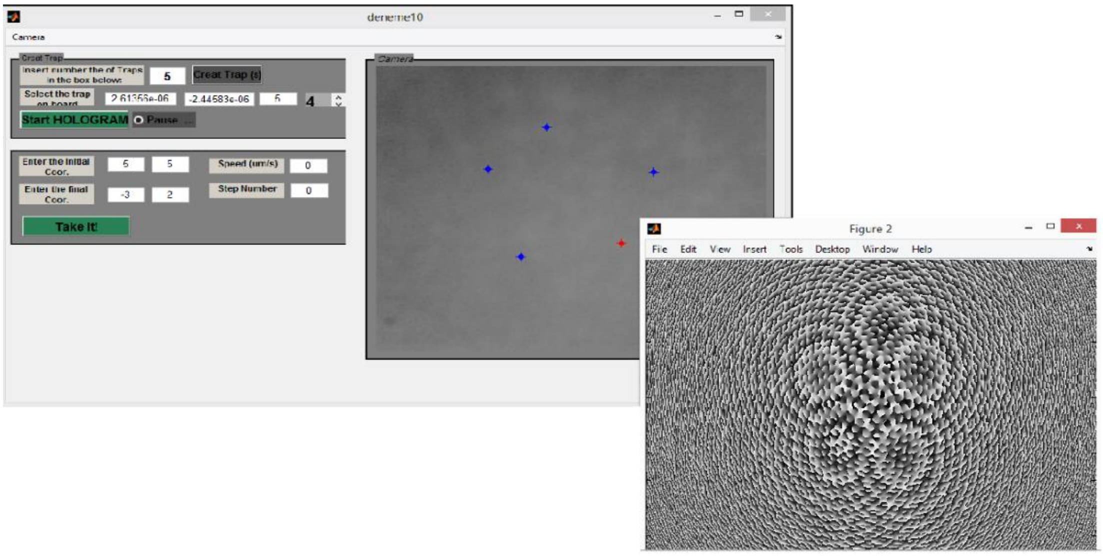
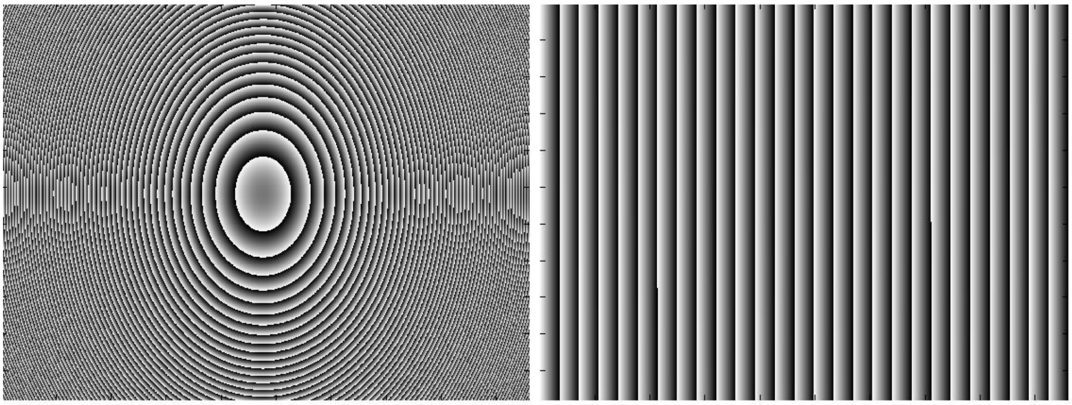


Figure 4.6 : Image of desinged GUI and a generated sample hologram.

Camera screen in GUI displays sample plane of the MO of the inverted microscope. User is able to select number of traps and adjust their positions in 3D using this camera screen. By selecting a desired trap (spot), coordinates of its location are shown in two boxes assigned for XY coordinates. On the other hand, after selecting a spot, one can change its depth or in other words deviation from Fourier plane that was discussed in section 4.2.1. User is able to obtain desired hologram after pushing the button named as “Start Hologram”. However, by changing the position coordinates in 3D of selected spot in camera screen by user, hologram updates itself automatically.

Holograms in this program are generated using grating and lenses algorithm. Spot coordinates are obtained and updated automatically from camera screen in GUI. Hologram for manipulating each spot is calculated by equation 4.14.

ϕ_{lens} and ϕ_{prism} are also calculated using equations 4.12 and 4.13 respectively. In these formulas, xy and uv are coordinates on hologram plane and image plane repectively. Deviation from Fourier plane that enables depth control of the particle is denoted by w. λ is the wavelength and f is the focal length of the lens (MO in our case). Figure 4.7 shows the generated hologram for lateral shift and axial shift respectively both for one trap:



a) b)

Figure 4.7 : a) hologram of lens for axial shift b) hologram of prism for lateral shift.

In our designed program after calculating ϕ_h for each trap independently, we can find final hologram for combination of several traps by:

$$\phi_{h,several\ traps} = arg\left(\sum_{n=1}^{N} u_{h,n}\right) \qquad \textbf{(4.27)}$$

In figure 4.8 hologram for three spots without axial shift is shown:

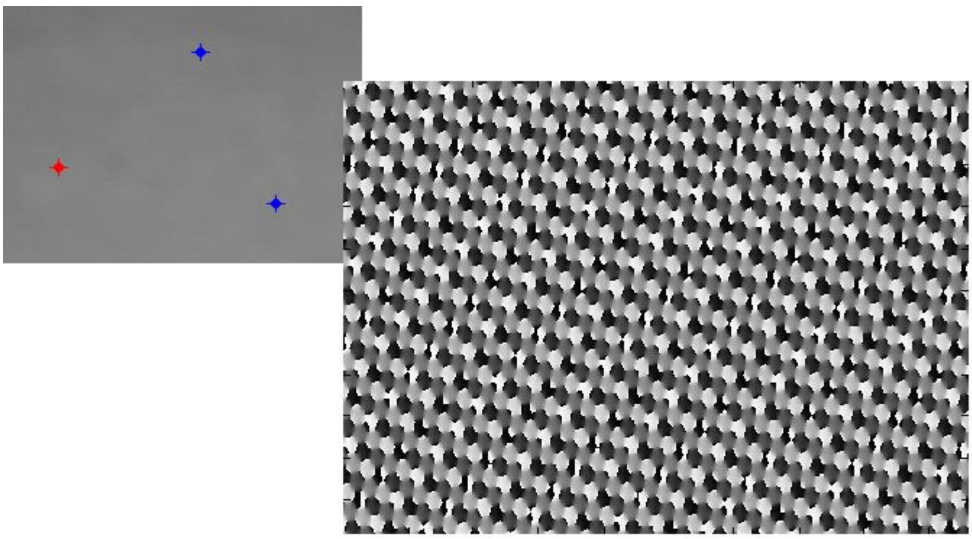


Figure 4.8 : The hologram shown in the right image serves to generate three traps shown in the left image.

Figure 4.9 shows hologram for generating several traps all with axial deviation from Fourier plane:

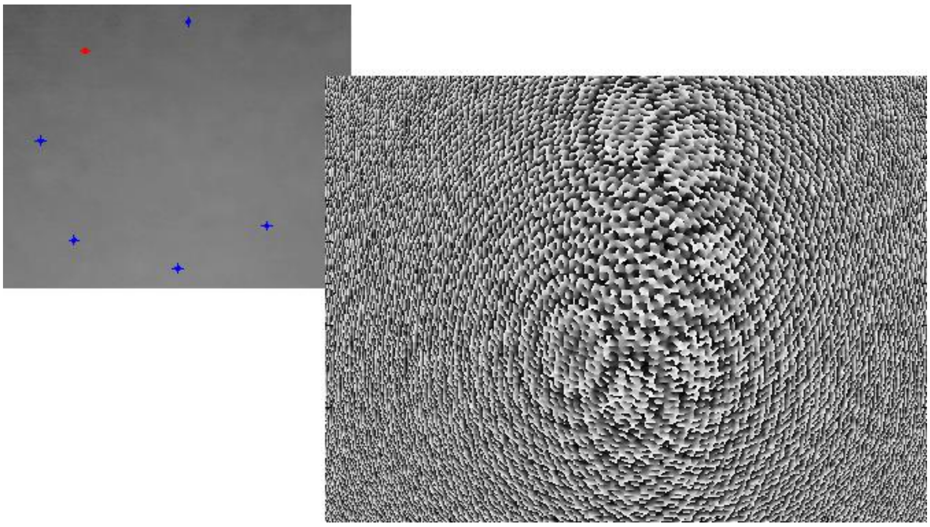


Figure 4.9 : Image of the hologram for spots with axial shift.

Figure 4.10 also shows an image of created spots in an screen in front of the SLM captured by CCD camera.

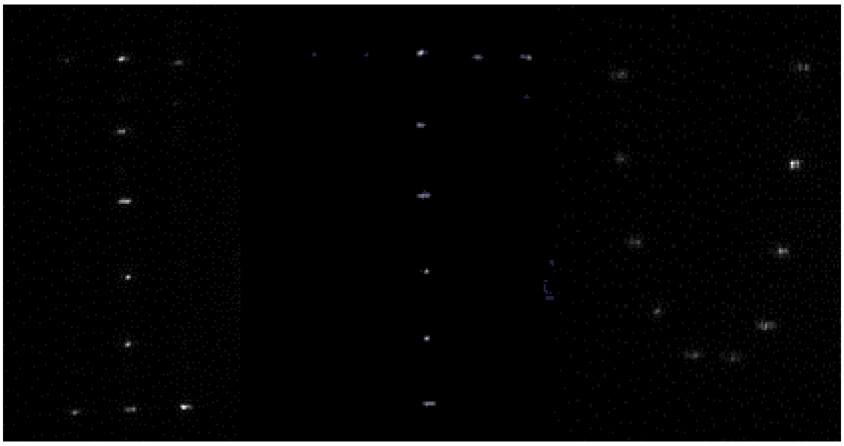


Figure 4.10 : Image of created spots in front of the SLM.

Figure 4.11, shows image of axial shift of an unknown trapped particle (levitated in water) by using designed GUI.

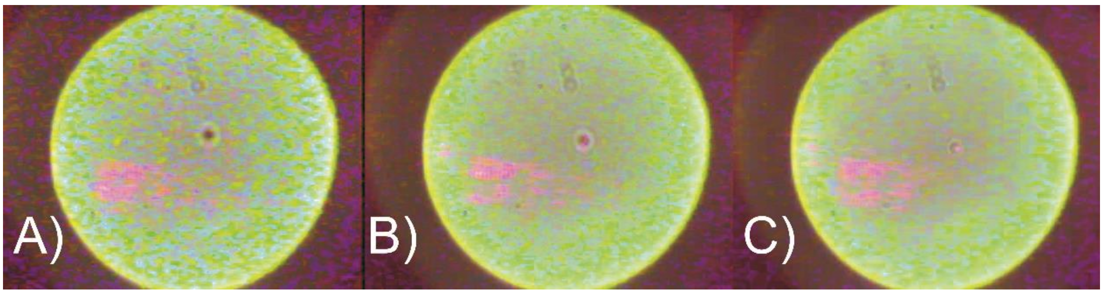


Figure 4.11 : Axial shift of a sample particle.

Figure 4.12, shows image of lateral shift of that particle.

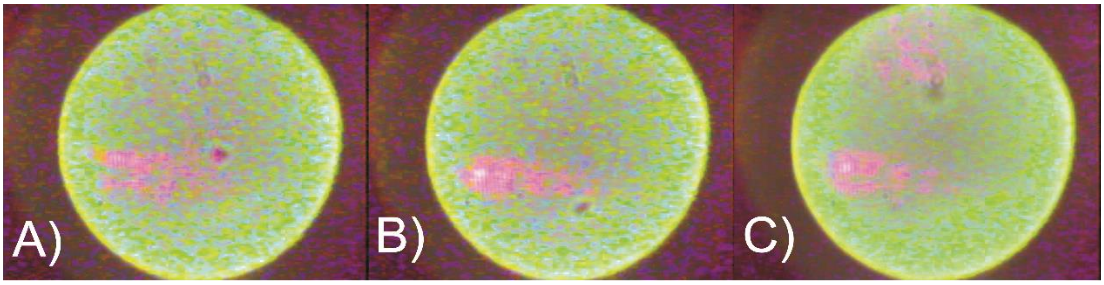


Figure 4.12 : Lateral shift of a sample particle.

Another experiment also was done in order to measure intensity distribution in spots. Table below shows measured intensity for each spot up to five spots totally.

Table 4.1 : Measured spot intensities.

Number of total spots	Intensities (mW)				
1 spot	9.5				
2 spot	4.45	4.13			
3 spot	2.22	2.15	2.35		
4 spot	1.65	1.48	1.85	1.57	
5 spot	1.34	1.65	1.2	1.3	1.15

As it is clear from table 4.1, it was observed that by adding more spots, total intensity was divided (approximately equally) between spots; For more stable trapping and more trap numbers, higher laser intensity is inevitable.

5. CONCLUSION

Without reservation, optical tweezers and especially HOT systems play major role in biological researches and several groups around world are working to improve and expand its horizons. In this thesis I tried to explain ongoing construction of Holographic optical tweezer set-up of Istanbul technical university from the first step and find solutions for the obstacles that we encountered. Many parameters from buying SLM to designing program for generating holograms were considered in order to have efficient trapping. In addition several measurements were done to understand SLM functionality.

In fact, technical problems were among major obstacles that should be removed in future works. Better Keplerian telescopes with efficient and maximum anti-reflection coating lenses, undoubtedly will increase efficiency and more precise imaging.

Other programs like Java and Labview seem to have faster performance that should be considered in future works. Besides, Although prisms and lenses is the most straightforward method, it has some drawbacks such as ghost traps that should be modified in future works.

REFERENCES

[1] **Padgett, M. J., Molloy, J., & McGloin, D.** (Eds.). (2010). *Optical Tweezers: methods and applications*. CRC Press.

[2] **Maxwell, J. C.** (1954). Electricity and magnetism.

[3] **Lebedev, P. N.** (1883). Experimental examination of light pressure. *Nuovo Cimento*, ***15***(195), 195.

[4] **Nichols, E. F., & Hull, G. F.** (1903, April). The pressure due to radiation. In *Proceedings of the American Academy of Arts and Sciences* (pp. 559-599). American Academy of Arts and Sciences.

[5] **Ashkin, A.** (1970). Acceleration and trapping of particles by radiation pressure. *Physical review letters*, ***24***(4), 156.

[6] **Ashkin, A., Dziedzic, J. M., Bjorkholm, J. E., & Chu, S.** (1986). Observation of a single-beam gradient force optical trap for dielectric particles. *Optics letters*, ***11***(5), 288-290.

[7] **Verdeny, I., Farré, A., Mas Soler, J., López Quesada, C., Martín Badosa, E., & Montes Usategui, M.** (2011). Optical trapping: a review of essential concepts. *Óptica Pura y Aplicada, 2011, vol.* ***44****, num. 3, p. 527-551*.

[8] **Soni, G. V., Hameed, F. M., Roopa, T., & Shivashankar, G. V.** (2002). Development of an optical tweezer combined with micromanipulation for DNA and protein nanobioscience. *Current Science*, ***83***, 1464-1470.

[9] **Rao, S., Raj, S., Balint, S., Fons, C. B., Campoy, S., Llagostera, M., & Petrov, D.** (2010). Single DNA molecule detection in an optical trap using surface-enhanced Raman scattering. Applied Physics Letters, ***96***(21), 213701.

[10] **Bustamante, C., Smith, S. B., Liphardt, J., & Smith, D.** (2000). Single-molecule studies of DNA mechanics. *Current opinion in structural biology*, ***10***(3), 279-285.

[11] **Wang, M. D., Yin, H., Landick, R., Gelles, J., & Block, S. M.** (1997). Stretching DNA with optical tweezers. *Biophysical journal*, ***72***(3), 1335.

[12] **Andrews, D. L.** (2011). Structured light and its applications: An introduction to phase-structured beams and nanoscale optical forces. Academic Press.

[13] **Leach, J., Wulff, K., Sinclair, G., Jordan, P., Courtial, J., Thomson, L., Gibson, G., Karunwi, K., Cooper, J., Laczik, Z. J., Padgett, M.** (2006). Interactive approach to optical tweezers control. Applied optics, ***45***(5), 897-903.

[14] **Harada, Y., & Asakura, T.** (1996). Radiation forces on a dielectric sphere in the Rayleigh scattering regime. Optics communications, ***124***(5), 529-541.

[15] **Ashkin, A.** (1992). Forces of a single-beam gradient laser trap on a dielectric sphere in the ray optics regime. *Biophysical journal*, ***61***(2), 569.

[16] **Şeninak, Ş.** (2010). "Preeclampsia hastalığının Raman Cımbızlama Yöntemiyle incelenmesi," Master, Fen Bilimleri Enstitüsü, İstanbul Üniversitesi.

[17] **Persson, M.** (2013). Advances in holographic optical trapping.

[18] **Preece, D.** (2011). *Novel uses of spatial light modulators in optical tweezers* (Doctoral dissertation, University of Glasgow).

[19] **Konforti, N., Wu, S. T., & Marom, E.** (1988). Phase-only modulation with twisted nematic liquid-crystal spatial light modulators. Optics letters, ***13***(3), 251-253.

[20] **Clark, N. A., & Lagerwall, S. T.** (1980). Submicrosecond bistable electro-optic switching in liquid crystals. Applied Physics Letters, ***36***(11), 899-901.

[21] **Mukohzaka, N., Yoshida, N., Toyoda, H., Kobayashi, Y., & Hara, T.** (1994). Diffraction efficiency analysis of a parallel-aligned nematic-liquid-crystal spatial light modulator. Applied optics, ***33***(14), 2804-2811.

[22] **Burnham, D. R.** (2009). Microscopic applications of holographic beam shaping and studies of optically trapped aerosols (Doctoral dissertation, University of St Andrews).

[23] **HOLOEYE.** (2013). PLUTO Phase Only Spatial Light Modulators Manual, vol. v***1.3.2***.

[24] **Martín-Badosa, E., Montes-Usategui, M., Carnicer, A., Andilla, J., Pleguezuelos, E., & Juvells, I.** (2007). Design strategies for optimizing holographic optical tweezers set-ups. Journal of Optics A: Pure and Applied Optics, ***9***(8), S267.

[25] **Bergeron, A., Gauvin, J., Gagnon, F., Gingras, D., Arsenault, H. H., & Doucet, M.** (1995). Phase calibration and applications of a liquid-crystal spatial light modulator. Applied optics, ***34***(23), 5133-5139.

[26] **Ferreira, F. P., & Belsley, M. S.** (2010). Direct calibration of a spatial light modulator by lateral shearing interferometry. Optics express, ***18***(8), 7899-7904.

[27] **Otón, J., Ambs, P., Millán, M. S., & Pérez-Cabré, E.** (2007). Multipoint phase calibration for improved compensation of inherent wavefront distortion in parallel aligned liquid crystal on silicon displays. Applied optics, ***46***(23), 5667-5679.

[28] **Xun, X., & Cohn, R. W.** (2004). Phase calibration of spatially nonuniform spatial light modulators. *Applied optics*, ***43***(35), 6400-6406.

[29] **Frumker, E., & Silberberg, Y.** (2007). Phase and amplitude pulse shaping with two-dimensional phase-only spatial light modulators. *JOSA B*, ***24***(12), 2940-2947.

[30] **Vaughan, J. C.** (2005). *Two-dimensional ultrafast pulse shaping and its application to coherent control and spectroscopy* (Doctoral dissertation, Massachusetts Institute of Technology).

[31] **Sprünken, D. P.** (2008). A 2D spatial light modulator for spatio-temporal shaping.

[32] **Leach, J., Dennis, M. R., Courtial, J., & Padgett, M. J.** (2004). Laser beams: knotted threads of darkness. *Nature*, ***432***(7014), 165-165.

[33] **Padgett, M.** (2014, December). Light's twist. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences (Vol. ***470***, No. 2172, p. 20140633). The Royal Society.

[34] **Goodman, J. W.** (2005). Introduction to Fourier optics. Roberts and Company Publishers.

[35] **Padgett, M., & Di Leonardo, R.** (2011). Holographic optical tweezers and their relevance to lab on chip devices. Lab on a Chip, ***11***(7), 1196-1205.

[36] **Montes-Usategui, M., Pleguezuelos, E., Andilla, J., & Martín-Badosa, E.** (2006). Fast generation of holographic optical tweezers by random mask encoding of Fourier components. Optics express, ***14***(6), 2101-2107.

APPENDICES

APPENDIX A: MATLAB codes for Calibration
APPENDIX B: MATLAB codes for HOT

APPENDIX A:

```
x_pixel = 1080;
y_pixel = 1920;
waiting_time = 3;
number_of_steps = 256;
x_of_point=1;
y_of_point=1;
ccd_row=1296;
ccd_column=966;
% z matrix is divided into 2 matrices z1 and z2.
% z2 has constant values, however z1 values change in each step.
% As I have divided black and white difference into 256, I will have 256 steps.
% in other words, B_W_division = number of steps
 % black and white has been divided in to 256. Starting from zero at first
% row and finishing at one at the 256th row.
mycolormap = zeros(number_of_steps,3);
for i= 1 : (number_of_steps)-1
    mycolormap(i+1,:) = i / ((number_of_steps)-1) ;
end
% preview of images in camera
uiwait(msgbox('Welcome to SLM Calibration Software','Welcome','modal'));
vid = videoinput('winvideo', 2, 'UYVY_1024x768');
src = getselectedsource(vid);
vid.FramesPerTrigger = 1;
vid.ReturnedColorspace = 'grayscale';
preview(vid);
vid.TriggerRepeat = Inf;
triggerconfig(vid, 'manual');
start(vid);
vid.ReturnedColorspace = 'grayscale';
```

```
pause(2);
% % %
for j=1:256
z1 = zeros (x_pixel, y_pixel/2);
z2 = zeros (x_pixel, y_pixel/2);
z1(z1==0)=j;
z=[z1,z2];
image(z);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
set(gca,'Unit','normalized','Position',[0 0 1 1]);
colormap(mycolormap);
clear z1
clear z2
clear z
pause(1.5);
trigger(vid);
matrix_of_images_1(:,:,j) = getdata(vid);
pause(0.5);
matrix_of_images(:,:,j)=(matrix_of_images_1(:,:,j)+matrix_of_images_2(:,:,j)+matri
x_of_images_3(:,:,j))/3;
pause(0.2);
end

stoppreview(vid);
stop(vid);
% The point will be selected for choosing the slice that the process
% will be done
foto=image(matrix_of_images(:,:,100));
colormap(mycolormap);
uiwait(msgbox('Please select upside','Slice selection','modal'));
h = impoint(gca,[]);
uiwait(msgbox('Please select downside','Slice selection','modal'));
g = impoint(gca,[]);
pos_h=getPosition(h);
pos_g=getPosition(g);
```

```
row_h=floor(pos_h(1,2));
row_g=floor(pos_g(1,2));

for k=1:256
   for t=1:ccd_row
     transfer_matrix(1,t,k)= mean (matrix_of_images(row_h:row_g,t,k));
   end
end

for h=1:256
   transfer_matrix_1(h,:)=transfer_matrix(1,:,h);
end


for i=1:256
   tm1(i,:)=transfer_matrix_1(i,:);
   fit(i,:)=polyfit(1:ccd_row,tm1(i,:),50);
   fit_value(i,:)=polyval(fit(i,:),1:ccd_row);
end
% Fit_value is the fit of transfer_matrix_1.
image(fit_value);
colormap(mycolormap);
% in this program any brightness can be selected from the fit_value matrix
% (which found in calibration_taslak4 previously)
image(fit_value);
colormap(mycolormap);
hold on
% %
uiwait(msgbox('please select the brightness that you want to analyse','Slice selection','modal'));
uiwait(msgbox('please select the right border','Slice selection','modal'));
hh_1 = impoint(gca,[]);
pos_hh_1 = getPosition(hh_1);
% row_hh_1 = floor(pos_hh_1(1,2));
column_hh_1 = floor(pos_hh_1(1,1));
line([column_hh_1,column_hh_1],[0,50],'LineWidth',4,'color','r');
```

```
uiwait(msgbox('please select the left border','Slice selection','modal'));
gg_1 = impoint(gca,[]);
pos_gg_1 = getPosition(gg_1);
% row_gg_1 = floor(pos_gg_1(1,2));
column_gg_1 = floor(pos_gg_1(1,1));
line([column_gg_1,column_gg_1],[0,50],'LineWidth',4,'color','r');
% Border selection will be done afew time in order to select whole the brightmess
for  i=1:50
[m_1(i),I_1(i)] = max(fit_value(i,column_gg_1:column_hh_1),[],2);
I_1(i)=I_1(i)+column_gg_1-1;
end
for  i=51:75
[m_1(i),I_1(i)] = max(fit_value(i,column_gg_2:column_hh_2),[],2);
I_1(i)=I_1(i)+column_gg_2-1;
end
for  i=76:100
[m_1(i),I_1(i)] = max(fit_value(i,column_gg_3:column_hh_3),[],2);
I_1(i)=I_1(i)+column_gg_3-1;
end
for  i=101:150
[m_1(i),I_1(i)] = max(fit_value(i,column_gg_4:column_hh_4),[],2);
I_1(i)=I_1(i)+column_gg_4-1;
end
for  i=151:200
[m_1(i),I_1(i)] = max(fit_value(i,column_gg_5:column_hh_5),[],2);
I_1(i)=I_1(i)+column_gg_5-1;
end
for  i=200:256
[m_1(i),I_1(i)] = max(fit_value(i,column_gg_6:column_hh_6),[],2);
I_1(i)=I_1(i)+column_gg_6-1;
end
for i=1:256
I_2(i,1)=I_1(1,i);
end
% I_2 is the matrix of each brightness, which can be copied to excel.
```

APPENDIX B:

```
function varargout = hologram10(varargin)
% HOLOGRAM10 MATLAB code for hologram10.fig
% HOLOGRAM10, by itself, creates a new HOLOGRAM10 or raises the existing
% singleton*.
% H = HOLOGRAM10 returns the handle to a new HOLOGRAM10 or the handle to
% the existing singleton*.
% HOLOGRAM10 ('CALLBACK',hObject,eventData,handles,...) calls the local
% function named CALLBACK in HOLOGRAM10.M with the given input
% arguments.
% HOLOGRAM10 ('Property','Value',...) creates a new HOLOGRAM10 or raises
% the existing singleton*.  Starting from the left, property value pairs are
% applied to the GUI before hologram10_OpeningFcn gets called.  An
% unrecognized property name or invalid value makes property application
% stop.  All inputs are passed to hologram10_OpeningFcn via varargin.
% Edit the above text to modify the response to help hologram
% Begin initialization code - DO NOT EDIT

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @hologram10_OpeningFcn, ...
                   'gui_OutputFcn',  @hologram10_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% Executes just before hologram10 is made visible.
function hologram10_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to hologram10 (see VARARGIN)
% choose default command line output for hologram10
handles.output = hObject;

guidata(hObject, handles);
% UIWAIT makes hologram10 wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% Outputs from this function are returned to the command line.

function varargout = hologram10_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;

function number_of_traps_Callback(hObject, eventdata, handles)
% hObject handle to number_of_traps (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of number_of_traps as text
% str2double(get(hObject,'String')) returns contents of number_of_traps as a double
% Executes during object creation, after setting all properties.

function number_of_traps_CreateFcn(hObject, eventdata, handles)
```

```
% hObject handle to number_of_traps (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in creat_traps.

function creat_traps_Callback(hObject, eventdata, handles)
% hObject handle to creat_traps (see GCBO)
% handles structure with handles and user data (see GUIDATA)
number_of_traps=str2double(get(handles.number_of_traps,'string'));
for i=1:number_of_traps
     handles.trap(i)=impoint(handles.camera_axes);
end

guidata(hObject, handles);
function z_of_trap_Callback(hObject, eventdata, handles)
% hObject handle to z_of_trap (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of z_of_trap as text
% str2double(get(hObject,'String')) returns contents of z_of_trap as a double
% Executes during object creation, after setting all properties.

function z_of_trap_CreateFcn(hObject, eventdata, handles)
% hObject handle to z_of_trap (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end

function y_of_trap_Callback(hObject, eventdata, handles)
% hObject handle to y_of_trap (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of y_of_trap as text
% str2double(get(hObject,'String')) returns contents of y_of_trap as a double
% Executes during object creation, after setting all properties.

function y_of_trap_CreateFcn(hObject, eventdata, handles)
% hObject handle to y_of_trap (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function x_of_trap_Callback(hObject, eventdata, handles)
% hObject handle to x_of_trap (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of x_of_trap as text
% str2double(get(hObject,'String')) returns contents of x_of_trap as a double
% Executes during object creation, after setting all properties.

function x_of_trap_CreateFcn(hObject, eventdata, handles)
% hObject handle to x_of_trap (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% Executes on selection change in trap_on_board.

function trap_on_board_Callback(hObject, eventdata, handles)
% hObject handle to trap_on_board (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: contents = cellstr(get(hObject,'String')) returns trap_on_board contents as
% cell array
% contents {get(hObject,'Value')} returns selected item from trap_on_board
% Executes during object creation, after setting all properties.

function trap_on_board_CreateFcn(hObject, eventdata, handles)
% hObject handle to trap_on_board (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: listbox controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% Executes on button press in start_hologram.

function start_hologram_Callback(hObject, eventdata, handles)
% hObject handle to start_hologram (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

number_of_traps = str2double(get(handles.number_of_traps,'string'));
hologram_matrix = zeros(1080,1920,number_of_traps);
```

```
hologram_matrix_total = zeros(1080,1920,1);
matrix_of_positions_1 = zeros(number_of_traps,2);

w = zeros(number_of_traps,1);

f = 0.2 * (10^-2);
lambda = 785*(10^-9);
A = pi/(lambda*(f^2));
B = (2*pi)/(lambda*f);

mycolormap = zeros(256,3);
for iii= 1 : (256)-1
    mycolormap(iii+1,:) = iii / ((256)-1) ;
end
% The position of traps will be written in the matrix below.
for i=1:number_of_traps
    matrix_of_positions_1(i,:) = getPosition(handles.trap(i));
end
% V is x, and u is the y of screen.
% u and v are positions in fourier plane.

for j=1:number_of_traps
   v(j,1)=((((matrix_of_positions_1(j,1)-320)*(10/320))))*(10^-6);
   u(j,1)=(((240-matrix_of_positions_1(j,2))*(10/240)))*(10^-6);
end

% i and j are pixel coordinates.
for l=1:number_of_traps
  for i = 1:1080
      for j = 1:1920
        first_term =((i-540)*8*10^(-6))^2;
        second_term =((j-960)*8*10^(-6))^2;
        phi_lens(i,j,l) = A * w(l,1) * (first_term + second_term);
        first_term_prim = (u(l,1))*(i-540)*(8*(10^-6));
        second_term_prim = (v(l,1))*(j-960)*(8*(10^-6));
```

```
            phi_prism(i,j,l) = B * ( first_term_prim + second_term_prim );
          end
        end
end

for l=1:number_of_traps
        hologram_matrix(:,:,l)=  mod(phi_lens(:,:,l) + phi_prism(:,:,l) , 2*pi) ;
end
dlmwrite('mods.asc',hologram_matrix_total)
hologram_matrix_total = exp(1i* hologram_matrix(:,:,1));
for k=2:number_of_traps
hologram_matrix_total = hologram_matrix_total+exp(1i*hologram_matrix(:,:,k));
end

hologram_matrix_total=angle(hologram_matrix_total);
hologram_matrix_total = hologram_matrix_total+repmat(pi,1080,1920);

max_of_columns = max(hologram_matrix_total);
max_of_matrix=max(max_of_columns);
coef=255/max_of_matrix;
hologram_matrix_total=floor(coef.*hologram_matrix_total);
figure(2);
colormap(mycolormap);
image(hologram_matrix_total);
set(gca,'Unit','normalized','Position',[0 0 1 1]);
while 2>1
        pause(0.1);
        a=get(handles.trap_on_board,'value');
        matrix_of_positions_1(a,:) = getPosition(handles.trap(a));
        v(a,1)=(((matrix_of_positions_1(a,1)-320)*(10/320)))*(10^-6);
        u(a,1)=(((240-matrix_of_positions_1(a,2))*(10/240)))*(10^-6);

    for k=1:number_of_trap
if a==k
        setColor(handles.trap(k),'r');
```

```
        set(handles.x_of_trap,'string',v(k,:));
        set(handles.y_of_trap,'string',u(k,:));
        w(k,1)= (str2double(get(handles.z_of_trap,'string')))* 10^-6;
else
        setColor(handles.trap(k),'b');
end
end
    aa=get(handles.trap_on_board,'value');

    p=get(handles.pause_radio_button,'value');
if p==1
       pause(5);
end

for i = 1:1080
   for j = 1:1920
       first_term =((i-540)*8*10^(-6))^2;
       second_term =((j-960)*8*10^(-6))^2;
       phi_lens(i,j,a) = A * w(a,1) * (first_term + second_term);
       first_term_prim = u(a,1)*(i-540)*(8*(10^-6));
       second_term_prim = v(a,1)*(j-960)*(8*(10^-6));
       phi_prism(i,j,a) = B * ( first_term_prim + second_term_prim );
   end
end

hologram_matrix(:,:,a)=  mod(phi_lens(:,:,a) + phi_prism(:,:,a) , 2*pi) ;
hologram_matrix_total = exp(1i* hologram_matrix(:,:,1));

for k=2:number_of_traps
  hologram_matrix_total =  hologram_matrix_total + exp(1i*hologram_matrix(:,:,k));
end

hologram_matrix_total=angle(hologram_matrix_total);
hologram_matrix_total = hologram_matrix_total+repmat(pi,1080,1920);
```

```
max_of_columns = max(hologram_matrix_total);
max_of_matrix=max(max_of_columns);
coef=255/max_of_matrix;
hologram_matrix_total=floor(coef.*hologram_matrix_total);

figure(2);
colormap(mycolormap);
image(hologram_matrix_total);
set(gca,'Unit','normalized','Position',[0 0 1 1]);

p=get(handles.pause_radio_button,'value');
if p==1
    pause(5);
end
end

function camera_Callback(hObject, eventdata, handles)
% hObject handle to camera (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

function open_camera_Callback(hObject, eventdata, handles)
% hObject handle to open_camera (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
vid = videoinput('winvideo', 1, 'YUY2_640x480');
src = getselectedsource(vid);
vid.FramesPerTrigger = 1;
vid.ReturnedColorspace = 'grayscale';
vidRes = vid.VideoResolution;
% nBands = vid.NumberOfBands;
hImage = image( zeros(vidRes(2), vidRes(1)) );
axes(handles.camera_axes);
% Update handles structure
guidata(hObject, handles);
```

```
preview(vid,hImage);

function close_camera_Callback(hObject, eventdata, handles)
% hObject handle to close_camera (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
cla(handles.camera_axes);
% Executes on button press in pause_radio_button.

function pause_radio_button_Callback(hObject, eventdata, handles)
% hObject handle to pause_radio_button (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of pause_radio_button

function initial_z_Callback(hObject, eventdata, handles)
% hObject handle to initial_z (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of initial_z as text
% str2double(get(hObject,'String')) returns contents of initial_z as a double
% Executes during object creation, after setting all properties.

function initial_z_CreateFcn(hObject, eventdata, handles)
% hObject handle to initial_z (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function initial_y_Callback(hObject, eventdata, handles)
```

```
% hObject handle to initial_y (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of initial_y as text
% str2double(get(hObject,'String')) returns contents of initial_y as a double
% Executes during object creation, after setting all properties.

function initial_y_CreateFcn(hObject, ~, handles)
% hObject handle to initial_y (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function initial_x_Callback(hObject, eventdata, handles)
% hObject handle to initial_x (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of initial_x as text
% str2double(get(hObject,'String')) returns contents of initial_x as a double

function initial_x_CreateFcn(hObject, eventdata, handles)
% hObject handle to initial_x (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function final_z_Callback(hObject, eventdata, handles)
% hObject handle to final_z (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of final_z as text
% str2double(get(hObject,'String')) returns contents of final_z as a double
% Executes during object creation, after setting all properties.

function final_z_CreateFcn(hObject, eventdata, handles)
% hObject handle to final_z (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function final_y_Callback(hObject, eventdata, handles)
% hObject handle to final_y (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of final_y as text
% str2double(get(hObject,'String')) returns contents of final_y as a double
% Executes during object creation, after setting all properties.

function final_y_CreateFcn(hObject, eventdata, handles)
% hObject handle to final_y (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end

function final_x_Callback(hObject, eventdata, handles)
% hObject handle to final_x (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of final_x as text
% str2double(get(hObject,'String')) returns contents of final_x as a double
% Executes during object creation, after setting all properties.

function final_x_CreateFcn(hObject, eventdata, handles)
% hObject handle to final_x (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% Executes on button press in take_trap.

function take_trap_Callback(hObject, eventdata, handles)
% hObject handle to take_trap (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

initial_x = (str2double(get(handles.initial_x,'string')));
initial_y = (str2double(get(handles.initial_y,'string')));
final_x = (str2double(get(handles.final_x,'string')));
final_y = (str2double(get(handles.final_y,'string')));
step_number = (str2double(get(handles.step_number,'string')));
speed = (str2double(get(handles.speed,'string')))
d1=sqrt((initial_x)^2+(initial_y)^2);
d2=sqrt((final_x)^2+(final_y)^2);
```

```
distance=d1+d2;
total_time = distance/speed;

step_time = total_time/step_number;
step_distance = (final_x-initial_x)/step_number;
x_steps_matrix = [initial_x:step_distance:final_x];
x_steps_matrix = (transpose(x_steps_matrix))*10^-6;
[r_x,nimp]=size(x_steps_matrix);
hologram_matrix = zeros(1080,1920,1);
f = 0.15 * (10^-2);
lambda = 785*(10^-9);
A = pi/(lambda*(f^2));
B = (2*pi)/(lambda*f);

mycolormap = zeros(256,3);
for iii= 1 : (256)-1
        mycolormap(iii+1,:) = iii / ((256)-1) ;
end

for i = 1:r_x
y_steps_matrix(i,1)=((final_y*(10^-6)-initial_y*(10^-6))/(final_x*(10^-6)-
initial_x*(10^-6)))*(x_steps_matrix(i,1)-initial_x*(10^-6))+initial_y*(10^-6);
end
      hologram_matrix = zeros(1080,1920,1);

for l=1:r_x
          pause(step_time);
for i = 1:1080
   for j = 1:1920
            first_term_prim = y_steps_matrix(l,1)*i*(8*(10^-6));
            second_term_prim = x_steps_matrix(l,1)*j*(8*(10^-6));
            phi_prism(i,j,l) = B * ( first_term_prim + second_term_prim );
    end
end
hologram_matrix(:,:,l)= mod(phi_prism(:,:,l),2*pi);
```

```
max_of_columns = max(hologram_matrix(:,:,l));
max_of_matrix=max(max_of_columns);
coef=255/max_of_matrix;
hologram_matrix(:,:,l)=floor(coef*hologram_matrix(:,:,l));
figure(2);
colormap(mycolormap);
image(hologram_matrix(:,:,l));
set(gca,'Unit','normalized','Position',[0 0 1 1]);
    end

function speed_Callback(hObject, eventdata, handles)
% hObject handle to speed (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of speed as text
% str2double(get(hObject,'String')) returns contents of speed as a double
% Executes during object creation, after setting all properties.

function speed_CreateFcn(hObject, eventdata, handles)
% hObject handle to speed (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function step_number_Callback(hObject, eventdata, handles)
% hObject handle to step_number (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of step_number as text
% str2double(get(hObject,'String')) returns contents of step_number as a double
```

```
% Executes during object creation, after setting all properties.

function step_number_CreateFcn(hObject, eventdata, handles)
% hObject handle to step_number (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

CURRICULUM VITAE

Name Surname: Nima BAVILI

Place and Date of Birth: Tabriz, 09/05/1989

E-Mail: bavili@itu.edu.tr, bavilinima@gmail.com

EDUCATION:

B.Sc.: Tabriz University