GENERATION OF OPTICAL COMPOSITE VORTEX BEAMS USING LOGICAL OPERATIONS ON BINARY FORK GRATINGS

A Project Report

submitted by

YADUNANDA B N

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2022

THESIS CERTIFICATE

This is to certify that the thesis titled **GENERATION OF OPTICAL COMPOSITE VORTEX BEAMS USING LOGICAL OPERATIONS ON BINARY FORK GRATINGS**, submitted by **Yadunanda B N**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bonafide record of the research and project work done by him under the supervision of **Dr. Ananth Krishnan**. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Ananth Krishnan
Research Guide
Assosiate Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 27th May 2022

ACKNOWLEDGEMENTS

This project was supported by IIT Madras. The author thanks his project guide Prof. Ananth Krishnan for his immense support during the course of project and doctoral candidates Nirjhar and Anil for fruitful discussions and Subrata and Pratyush for assisting in the experimental setup.

ABSTRACT

KEYWORDS: **; Composite beams; OAM; Vortex beams; Logic operations.**

We design a simple method for the generation of arbitrary composite vortex (CV) beams using two or more binary fork gratings (BFG). These gratings were computationally generated by performing logical operations on two or more fork-gratings. The geometrical parameters of BFGs were optimized for the efficient generation of CV beams. The method was further extended to the generation of CV beams by varying the duty cycle of the binary fork gratings (BFG). This simple generation method may be useful to generate complex beam shapes with engineered phase fronts without complicated interferometry based techniques.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IITM	Indian Institute of Technology, Madras
OAM	Orbital Angular Momentum
OV	Optical Vortex
VB	Vortex Beam
CVB	Composite vortex Beam
BFG	Binary Fork Grating
LG	Laguerre Gaussian
SLM	VSpatial Light modulator
CCD	Charged coupled device
TC	Topological Charge
GPU	Graphical processing unit

NOTATION

R	Distance from grating to screen, m
ϕ	Phase in radians
l	Topological charge
p	Radial charge
(ρ, ψ, z)	cylindrical coordinate coefficients
w	Beam waist

CHAPTER 1

INTRODUCTION

As long ago as the 1600s, Kepler reasoned that light must carry a linear momentum, his logic being that the tails of comets always pointed away from the sun. Although all of light's momentum and energy properties are encapsulated in Maxwell's equations, it wasn't until 1909 that Poynting used a mechanical analogy to articulate that a circularly polarized light beam contained an angular momentum that we would now attribute to the $\hbar$ spin of individual photons as shown in Fig. 1.1. Moving beyond this spin angular momentum Darwin (grandson of the famous naturalist) pointed out that the conservation of angular momentum during higher order atomic/molecular transitions required an optical angular momentum of multiple units of $\hbar$ per photon.

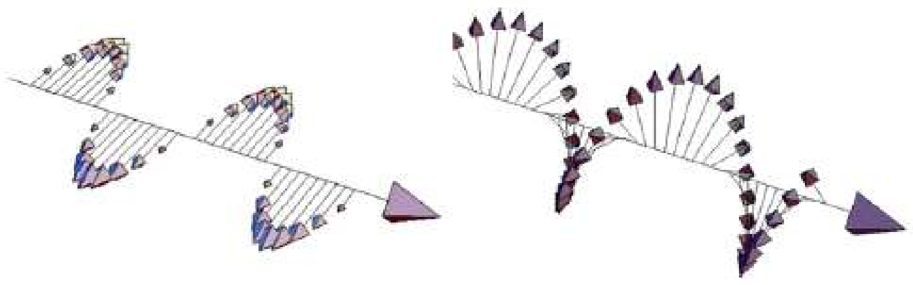

Figure 1.1: Light with linear polarization (left) carries no SAM, whereas right or left circularly polarized light (right) carries a SAM of $\pm\hbar$ per photon.

This additional angular momentum is called "orbital angular momentum" (OAM) and can be thought to arise simply from the effect of light's linear momentum acting off-axis with respect to the center of the optical beam or center of mass of the interacting object. For many decades it was implicitly assumed that this orbital angular momentum was a rare event, just as high-order transitions are themselves rare (they have low absorption cross sections).

In 1992 this assumption was overturned when Allen, Woerdman and associates established that light beams with helical phase-fronts, described by a transverse phase structure of $exp(il\phi)$, carry an orbital angular momentum equivalent to $l\hbar$ per photon, i.e. potentially an angular momentum many times greater than the spin of the photon. An important feature of all beams with helical phase-fronts is that the beam axis marks a singularity in the optical phase, akin to the time at the north pole! This phase singularity is manifested as a perfect zero in the optical intensity, meaning that OAM-carrying beams typically have annular intensity cross-sections.

In the 25 years since the '92 paper, orbital angular momentum has established itself as one of the most interesting of optical modes, with relevance to optical manipulation, imaging, quantum optics, optical communications and elsewhere. More broadly, OAM has given rise to studies of phase-structured light beams with unique that properties arise from their phase structure, not their intensity.

1.1 Helical beams: History

Central to the Allen et al. paper is the link it establishes between beams with helical phase-fronts and orbital angular momentum (OAM). However, these helically-phased beams had themselves been generated and studied earlier, not least as the examples of the transverse modes produced from suitably configured laser cavities, or as resulting from optical vortices. The study of optical vortices (or their acoustic counterpart) with phase singularities at their center had been also been extensive from the 1970s onwards. However, in none of these earlier works had any link been made between these features and the possibility of angular momentum in the beam.

The early work on OAM itself proposed and then implemented the use of cylindrical lenses to transform the high-order Hermite-Gaussian modes emitted by a conventional laser into helically-phased Laugerre-Gaussian modes. In addition, the same research group demonstrated perhaps the most obvious production method for OAM, namely the insertion into a normal laser beam of a phase plate with a thickness that increased with azimuthal angle, such that the transmitted beam acquired a transverse phase cross section of $exp(il\phi)$.

Perhaps the most significant work immediately prior to '92 relevant to future experiments involving OAM was the generation of helically-phased beams using a diffraction grating containing an l -pronged fork dislocation in the ruled lines. As shown by Soskin and associates, an incident plane-waved beam aligned co-axially with this dislocation results in a first-order diffracted beam with helical phase-fronts described again by $exp(il\phi)$. It is this use of diffractive optical elements that has been central to many subsequent studies of phase-structured beam formation and is now common to the vast majority of modern experiments using OAM. This diffractive-optic approach is made all the more applicable by the commercial availability of computer addressable, pixelated, spatial light modulators that can be controlled to act as reconfigurable diffractive optical elements. Furthermore various algorithms exist for implementing aberration correction of the SLM devices such that the beams produced are of high optical fidelity.

The majority of SLMs used for studies of OAM are based on the thin films of liquid crystal whose refractive index can be locally switched by applying an electric field and hence, if the films are laid over a pixelated electronic array, can be controlled to give a spatially dependent phase variation to the reflected light. A number of commercial devices are available capable of diffracting well in excess of 50% of the incident energy into a desired beam type. It is also possible to use intensity modulators based on digital micro-mirrors to create diffractive optical components. Although the diffraction efficiency of these digital micro-mirror devices (DMD) is much lower than an SLM, their low-cost and much higherspeed performance offer capabilities that the liquid crystal devices cannot match.

Strictly speaking the design of a diffractive optical element is only correct for one operating wavelength, so if a broad-band or a white-light beam is desired, a different technique is required. However, diffractive elements remain a possibility providing that an additional element is incorporated to compensate for their angular dispersion, such elements can be a compensation prism or an additional grating. Alternatively bespoke optical elements can use Fresnel reflections, or similar, to introduce a spatially dependent phase shift, allowing the generation of white-light vortex beams.

Beyond phase structuring alone, it is possible to use combinations of SLMs and waveplates to overlap two orthogonally polarized beams to create generalized vector vortex beams. This was first achieved with liquid crystal SLMs and more recently with DMDs.

Various other methods also exist for generating vortex beams using structured materials. Some of these methods rely upon a spatial dependent geometrical phase delay that is created using liquid crystal films, carefully oriented using structured surfaces (so called "q-plates") or rely on the structured surface itself. Finally the desire to use OAM on-chip has led to the development of chip-scale sources relying on the vertical emission from ring waveguides with small slots introduced to act as scattering centers with a defined and controllable phase relationship.

1.2 Diffractive Optical Elements for Generating Orbital Angular Momentum

In the 25 years since the recognition of OAM as an optical degree of freedom, OAM has enabled insights into various wave phenomenon. Although OAM was originally espoused in terms of optical fields, the related field of phase singularities within wavefields has a prehistory which include considerations as to the singularities that might arise in electron wave functions and, as discussed above, to studies of singularities that arise in acoustic fields. More generally phase singularities occur whenever three, or more, plane-waves interfere, an extreme example of which is optical speckle. In optical speckle each black speck is indeed a phase singularity around which the phase advances (clockwise or anticlockwise) by 2π. In 3-dimensions, these phase singularities trace out lines of perfect darkness, fractal in nature, that percolate all of space, creating topological features comprising loops and even (rarely) knots. However, this intricate 3D structure should not be directly linked to OAM since the fields in the vicinity of these singularities are super oscillatory, and therefore the associated energy lies in the space between the singularities and the total angular momentum of a random speckle pattern averages to zero as the lateral expanse of the speckle increases.

Optical beams with a helical phase front and an intensity null due to the on-axis phase singularity are commonly known as vortex beams. The transverse scalar electric field profile of these beams can be mathematically represented as a product of Gaussian and Laguerre polynomials and consequently such beams are also known as Laguerre-Gaussian (LG) beams. The normalised LG functions represent different modes of the propagating LG beams and form a complete orthonormal basis in the cylindrical co-ordinate system (r, ψ, z). The radial index p of the beam determines the number of intensity nulls in the radial direction and the azimuthal index l determines the change of azimuthal phase $\psi = 2\pi \times l$ around the intensity null. Thus the intensity null is referred to as optical vortex(OV) and l as its topological charge (TC). Due to their helical phase fronts these beams possess quantized orbital angular momentum (OAM) with a magnitude of $\hbar l$, and are also known as OAM beams. OAM renders these beams with interesting optical and opto-mechanical properties. Such beams have been explored over the last few decades for applications such as optical manipulation, communication, imaging, sensing and laser structuring of surfaces.

As an alternative to making complex refractive optics, diffractive optical elements are readily designed to mimic any refractive element of choice, albeit only at a single wavelength. A helical phase profile $exp(il\phi)$ converts a Gaussian laser beam into a helical mode whose wave fronts resemble an l-fold corkscrew, as shown in Fig. 1.2. In practice, the phase distribution of the desired optical component is typically added to a linear phase ramp and the sum expressed as modulo 2π. The result is a diffraction grating that produces the desired beam in the first diffraction order. The components are effectively holograms of the desired optical element and are thus often referred to as "computer generated holograms." To produce helical beams these holograms can be either the "forked diffraction gratings ", or spiral Fresnel lenses. The technique can be easily extended to cover both the l and the p of the generated beams. What makes the holographic approach particularly appealing is the commercial availability of spatial light modulators (SLMs). These are pixellated liquid crystal devices that can be programmed through the video interface of a computer to act as holograms. Changing their design is as simple as changing the image displayed by the computer interfacing the device.

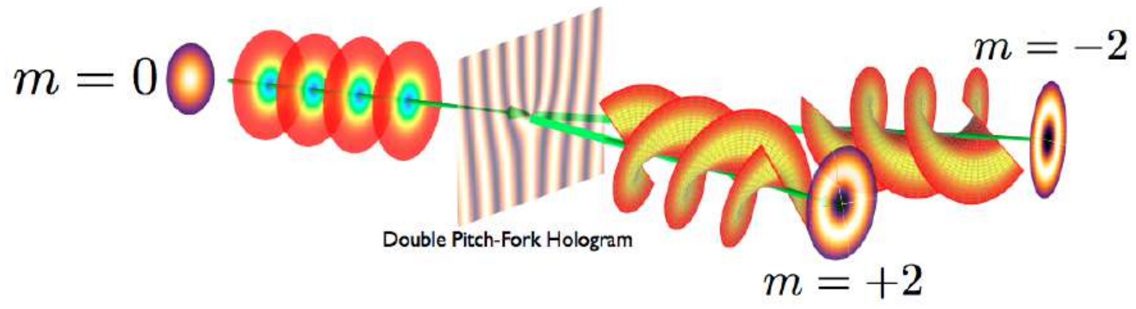

Figure 1.2: A helical phase profile $exp(il\phi)$ converts a Gaussian laser beam into a helical mode whose wave fronts resemble an l-fold corkscrew. In this case l=2.

Computer generation of holograms and their implementation for the generation of exotic beams is obviously not restricted to pure helical modes; it is a general technique that can be applied to all complex beam types or their superpositions. However, in general, the hologram design is more complicated than simply that of a phase mask alone. Accurate holograms are the complex far-field diffraction patterns of the desired objects or beams and as such are defined in terms of both their phase and intensity. For many simple beams it is sufficient to define only the phase, assuming a uniform illumination, and in such cases the far field (Fourier transform) of the hologram is a close approximation to the target beam. For example, when a forked diffraction grating is illuminated with a fundamental Gaussian beam from a conventional laser of radius w_0 the transmitted beam is a superposition of LG modes all with the same l index. These predominantly have $p = 0$, but there is also a small contribution of LG modes with higher p.

However, if precise control over the p of the resulting mode is needed, then it is necessary for the hologram to define both the phase and the intensity of the diffracted light. Unfortunately, SLMs are designed to modify only one or the other. Various approaches are possible, including the use of two SLMs in series: the first uses an algorithm such as Gerchberg–Saxton to create the desired intensity distribution, and the second then adjusts the phase to that of the target beam. This approach is particularly suited to cases where the intensity distribution is extremely localized in the hologram plane. For cases when the situation is less localized, typical of the superpositions between simple modes, a single SLM technique is possible. For diffracting holograms the efficiency with which light is diffracted to the first order depends on the depth of the blazing function, which for maximum efficiency is 2π. Varying this depth over the cross section of the hologram allows the intensity at various positions to be reduced, with any unwanted light being directed into the zero order. This gives the precise control over both phase and intensity that is required, albeit at a reduction in the overall efficiency. This technique was used prior to any interest in OAM but has since been used effectively to create many precise superpositions, including those associated with the generation of vortex lines, which are themselves linked and knotted, or precise modal measurements in quantum entanglement.

CHAPTER 2

OAM GENERATION USING AMPLITUDE DIFFRACTION GRATING

2.1 Characteristics of Orbital Angular Momentum Modes

Optical beams carrying OAM typically exhibit helical wavefronts. The pitch and handedness of the helix determine the topological charge and type (positive/negative) of the OAM beam. Various solutions of the Helmholtz wave equation can result in different kinds of beams that carry OAM. The differences arise based on the geometry or conditions that are used to solve the wave equation. Examples of some of the solutions with an OAM component are Laguerre– Gaussian (LG) modes, Bessel modes, Mathieu modes, Ince–Gaussian modes, and hypergeometric- Gaussian modes. The solutions of the paraxial wave equation in cylindrical coordinates (ρ, ψ, z) have a transverse scalar electric field given as

$$LG_{pl} = C_{lp}^{LG} \left(\frac{\rho\sqrt{2}}{w(z)}\right)^{|l|} exp\left(\frac{-\rho^2}{w^2(z)}\right) LG_p^{|l|}\left(\frac{2\rho^2}{w^2(z)}\right) exp\left(il\phi + \frac{ik\rho^2}{2R} - i\psi_G\right), \tag{2.1}$$

with

$$R = \frac{(z^2 + z_R^2)}{z}, \tag{2.2}$$

$$kw^2 = \frac{2(z^2 + z_R^2)}{z_R}, \tag{2.3}$$

$$\psi_G = (2p + |l| + 1)tan^{-1}(\frac{z}{z_R}) \tag{2.4}$$

where C_{lp}^{LG} is the normalization constant, z_R is the Rayleigh range, ψ_G is the Gouy phase, w is the beam radius, R is the radius of the spherical wavefront, $k = 2\pi/\lambda$ is the wave number, LG is the Laguerre polynomial, and l and p are the azimuthal and radial modal numbers. These modes are also called LG modes. The $exp(il\phi)$ term denotes the azimuthal phase variation, because of which, the beam exhibits OAM of $l\hbar$ per photon. The term $exp(ik\rho^2/2R)$ represents the spherical wavefront structure of the beam. LG polynomials with variables l and p are orthogonal to each other. Therefore, LG modes with beam waist w and l and p modal numbers form a complete orthogonal basis set.

An LG beam when superposed with a plane wave of the same wavelength results in an interference pattern with a fork shaped dislocation. The period of this interference pattern is determined by the wavelength and the angle of incidence of beams on the screen/camera. The charge of the fork dislocation in the interference pattern is determined by the azimuthal index l of the LG beam. The inverse of this, that is, when a plane wave is incident on a periodic grating with a charge l, the fork dislocation results in an LG beam of an azimuthal index l. This is one of the most popular methods of LG beam generation from a plane wave, where a computer generated hologram in the shape of the interference pattern with a fork dislocation known as fork grating, is used to obtain an LG beam of index l in the first diffraction order, along with the higher index beams in the higher diffraction orders. In this work, all the fork gratings are computationally generated using this interference method.

2.2 Huygen's Principle

In order to simulate the diffraction orders of a fork grating, Huygens principle was used, where each pixel (x, y) in the grating was treated as a source of secondary spherical wave. In this method, the resultant amplitude of the spherical wave from any pixel (x, y) of an amplitude grating is given as the product of the grating's pixel gray value (0 to 1) and amplitude of the incident beam, while the phase remains the same as the incident beam. Similarly, for a phase grating, the phase of the spherical wave at any pixel is given as the sum of the pixel gray value (0 to 2π) and incident beam phase, while the amplitude remains the same as that of the incident beam. The electric field at any pixel (x, y) on the screen at a distance R from the grating of size $N_x \times N_y$ pixels, was calculated using the superposition of secondary spherical waves from all the pixels of the grating, as given in Eq. 2.5 post Fresnel correction: multiplication by $(-i/\lambda)$ and inclination factor (R/r).

$$E(x', y') = \frac{R}{i\lambda} \sum_{x=0}^{N_x} \sum_{y=0}^{N_y} \frac{E(x, y)exp(ikr)}{r^2} \tag{2.5}$$

where, $r = \sqrt{(x - x')^2 + (y - y')^2 + R^2}$ and $k = \frac{2\pi}{\lambda}$

The electric field E(x, y) for every pixel on the screen was calculated using a Google Colab Python notebook executed on an online GPU server. The schematic of the simulation setup is shown in Fig. 2.1. The Grating was illuminated with a plane wave at an incident angle adjusted to obtain the 1st diffraction order incident normally at the screen center. This was done to prevent any phase distortion due to the optical path length difference across the captured cross-section of the diffraction order. The size of the screen was adjusted to be sufficient to simulate intensity and phase profile of only the 1st diffraction order. For all the simulations presented here, only amplitude gratings were used.

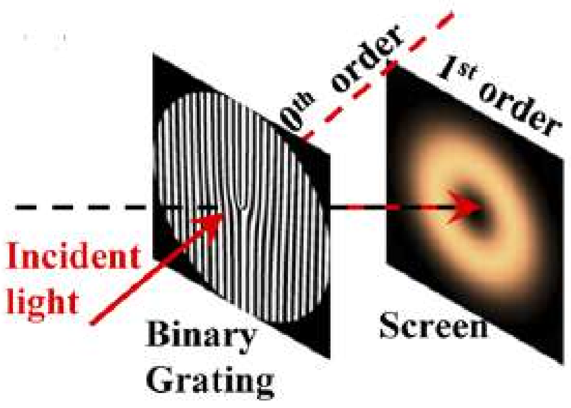

Figure 2.1: The simulated optical setup consisting of a BFG in the xy plane centered at x = 0, y = 0, z = 0, illuminated at an angle such that the 1st diffraction is incident normal to the screen in the xy plane centered at x = 0, y = 0, z = R.

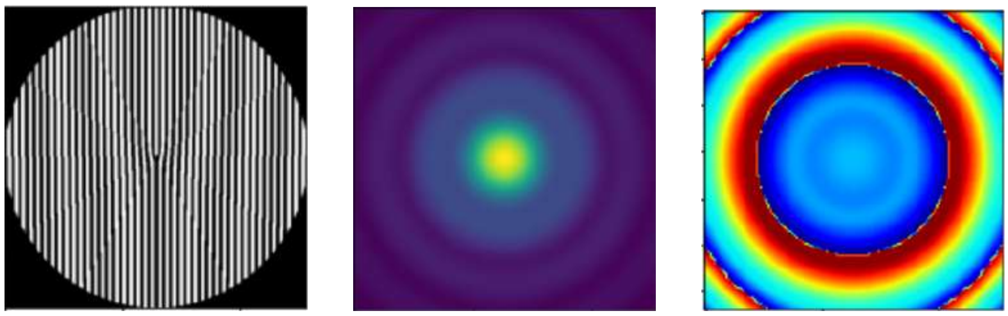

Figure 2.2: l=1 Binary grating. Intensity and phase structures of LG modes with azimuthal number $l = 0$.

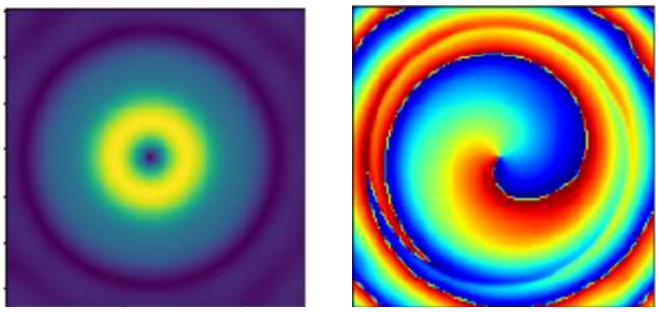

Figure 2.3: Intensity and phase structures of LG modes with azimuthal number $l = 1$.

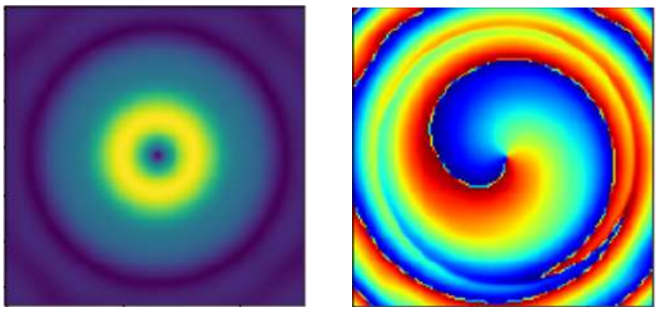

Figure 2.4: Intensity and phase structures of LG modes with azimuthal number $l = -1$.

Fig. 2.2 shows an amplitude BFG of charge l = 1 and period 0.95 m, in a circular disc shaped aperture of radius 20 m, along with the corresponding simulated intensity and the phase profile for an incident wavelength of λ = 633 nm. The obtained intensity profile was in the shape of a doughnut with an intensity null at the center, which is a well reported vortex beam intensity profile. The phase profile showed a variation of π to +π i.e. a total of +2π azimuthal phase variation around the center, thereby confirming the TC to be +1, however in a spiral shape. This was due to the fact that in the simulation each grating pixel was assumed to be a point source of a spherical wave, and thus, even though the grating itself was illuminated by a plane wave, the diffracted beam had an additional spherical phase. The effect of this spherical phase increases with the increase in R and dominates over the phase response of the fork grating. Therefore, in order to capture the phase profile of the BFG, shorter R and hence a smaller grating period was used to demonstrate the design methodology (scaled to 1/100th of the dimensions used in the experimental section).

2.3 Experimental Setup

The experiment setup diagram adopted is shown in Fig. 2.5, in which we have used a He–Ne laser(a wavelength of 633 nm) as the light source. The light beam from the source laser is first attenuated appropriately and then expanded. The expanded beam(approximately uniform plane beam at the center) is divided by a beam splitter into two parts:the reflected part and the transmitted part. The transmitted part travels towards screen or an electronic addressing reflection-type SLM controlled by a computer where the hologram are stored.The light diffracted from the screen or an SLM contains the optical vortex beams and their conjugate counterparts.The generated vortex beams are recorded on a screen or using a computer-controlled CCD camera,which is placed at a certain distance from the SLM. And the orbital angular momentum beams are observed on the screen as shown in Fig. 2.6

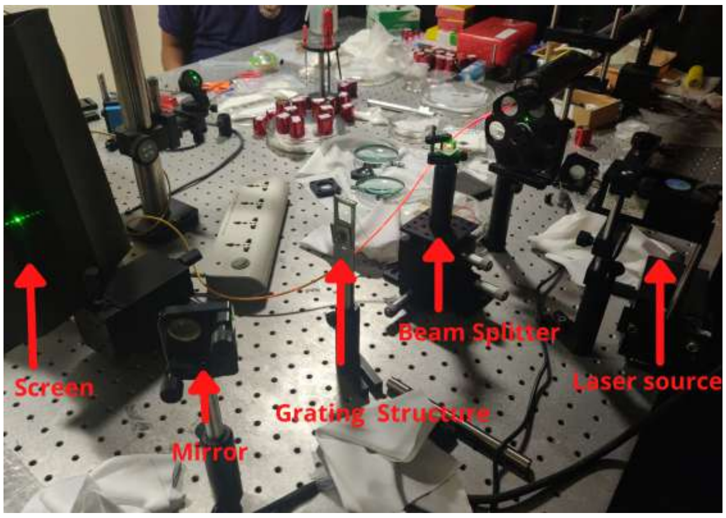

Figure 2.5: Schematic of experiment setup. Source; screen; computer; Beam Splitter; Grating structure; Screen/CCD,charge coupled device.

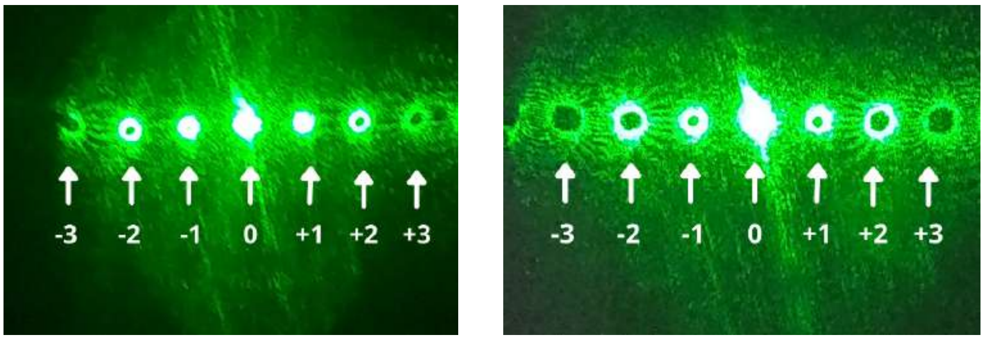

Figure 2.6: Experimental result of vortex beams observed on the Screen/CCD for l=1,2.

2.4 Translation of fork dislocation

The centre fork dislocation is moved from left side towards centre as shown in Fig. 2.7 and their respective intensity profiles are observed. Similarly the centre fork dislocation is moved from centre towards right as shown in Fig. 2.8 and their intensity profiles are observed. From the intensity plots we can observe that as the fork dislocation is moved further away from the centre, the magnitude of the intensity of the respective vortex beam is reducing. And the spot of reduction of intensity depends on the direction of translation.

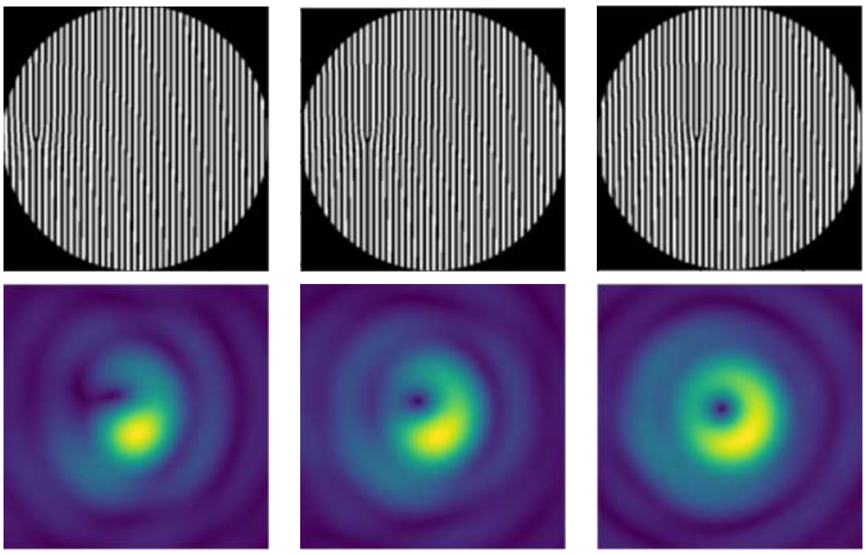
Figure 2.7: l=1 Binary grating and Intensity structures of LG modes with the translation of fork dislocation.

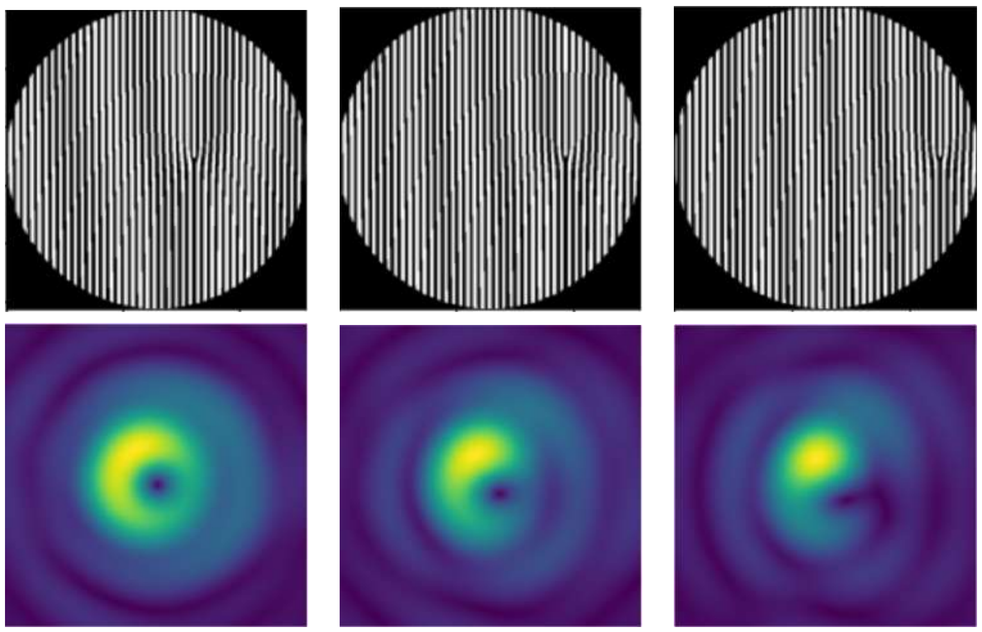

Figure 2.8: l=1 Binary grating and Intensity structures of LG modes with the translation of fork dislocation.

The location of the singularity can be controlled by moving the fork dislocation around in the complex plane. By shifting the beam slightly off the center of the singularity in the fork hologram, the singularity in the resulting intensity profile moves off-axis Fig. 2.7 and 2.8. By controlling the horizontal (x_0) and vertical (y_0) displacement of the singularity in the hologram, the singularity in the intensity profile can be positioned at any point within the complex plane.

2.5 Variation of Duty cycle in the grating

Now lets vary the duty cycle of the white(1) pixels with respect to the black(0) pixels in the binary fork grating(BFG).I have varied the duty cycle from 0.1 to 0.9 in steps of 0.1 and observed the intensity plots as shown in Fig. 2.9.

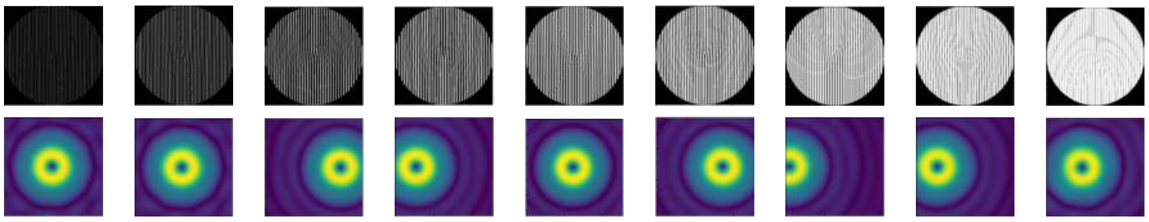

Figure 2.9: Intensity structures of LG modes with the variation of duty cycle from $10\% - 90\%$ in a l=1 Binary grating.

The Intensity of the OAM beams move along the horizontal axis as the duty cycle is varied as shown in Fig. 2.9. And hence the power at the first order on the screen also is varying as we change the duty cycle as observed in Fig. 2.10. Henceforth, a Binary fork grating(BFG) with 50% duty cycle is used because highest peak efficiency is achieved at this duty cycle.

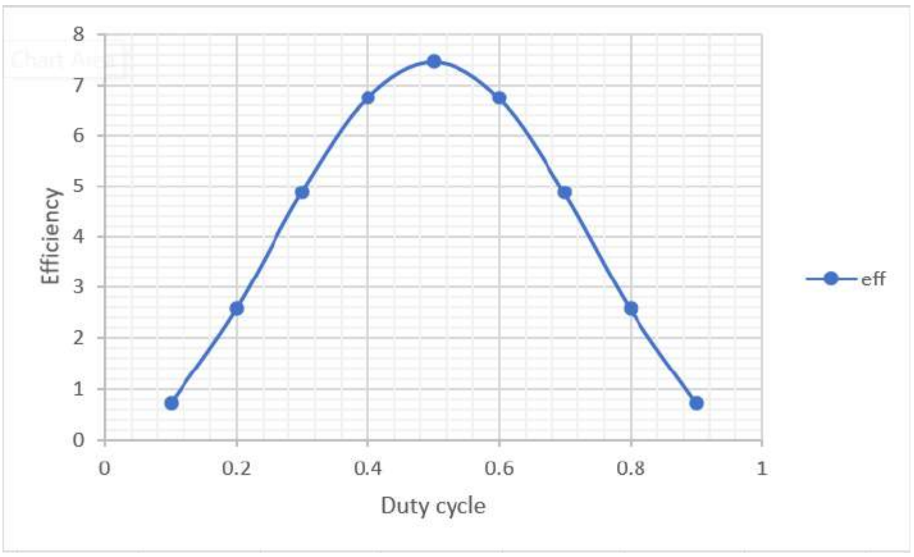

Figure 2.10: Intensity structures of LG modes with the variation of duty cycle from $10\% - 90\%$ in a l=1 Binary grating.

CHAPTER 3

COMPOSITE BEAMS

The nontrivial behaviors of phase give rise to diverse intriguing phenomena, opening up the area of singular optics. Optical vortices (known as singular light beams) have been used in many fields ranging from particle manipulation to quantum information due to their unique optical properties. The singularity structure of a wavefront plays an important role in the physical properties of vortex beams.

Unlike a vortex beam with a single singularity, composite vortices with more elaborate singularity distributions are endowed with enhanced capacity and more flexibility, which have promising applications in multiple optical traps, electromagnetically induced transparency, and quantum computation gate. Currently, such composite vortex beams (CVBs) are mainly generated based on spatial light modulators (SLMs), which suffer from large volume, high cost, and low resolution. To keep the pace of device miniaturization and system integration, tackling these challenges typically associated with CVBs generation is urgently needed.

Recently, in order to extend their functionalities, vortex beams with structured intensity profiles and multiple intensity nulls have received increased interest. Historically, the first ever observations of phase vortices were as multiple intensity nulls in the interference pattern of ultrasound and later, in optics as the interference of multiple Gaussian beams and speckle patterns generated by lasers. However, these methods cannot generate composite vortex (CV) beams - structured beams with multiple OVs of different TCs, which retain their intensity profile during propagation.

An intuitive method for the generation of CV beams is the linear superposition of two or more LG modes of different indices using interferometric techniques such as the Mach-Zehnder interferometer. However, for the efficient generation of CV beams, there are two essential criteria: (i) the constituent beams should be coaxial and (ii) there should be spatial overlap of their intensity profiles. Meeting these criteria is not straightforward using interferometric techniques, as the vortex beams of different charges have different radii and intensity profiles. Therefore, a method to generate a ‘perfect’ optical vortex was proposed, where beam radius is independent of the azimuthal charge. Even so, these methods would require stringent optical alignment for an efficient interference between the constituent beams.

In order to mitigate the problem of multiple optical path alignment, several methods involving a single optical element for the coaxial generation of all the constituent modes were proposed, such as using diffractive optical elements, plasmonic metasurfaces and holograms on spatial light modulators (SLM) for their applications in high speed communications, advanced optical manipulators and rotational sensors. These methods involve complex design processes and algorithms along with the difficulty in controlling properties of specific constituent modes. Further, since most of these generation methods are based on the phase profile holograms, they heavily rely on the properties and limitations of SLM.

Emerging optical metasurfaces have provided unprecedented capabilities to locally manipulate light’s phase, amplitude, and polarization at subwavelength scale, leading to the development of novel metasurface devices, including metalenses, multifunctional devices, polarization-sensitive holograms, vortex beam generators, and polarization structure generation. Optical metasurfaces have been used to generate vortex beams and realize the superposition of these beams. These demonstrations prove that metasurfaces can be used to create complicated singularity structures based on the superposition of two vortex beams. However, the superposition of multiple vortex beams (three or more) is needed to engineer the singularity structure of CVBs, which has not been demonstrated with optical metasurfaces. In addition, unlike SLMs, geometric optical metasurfaces are sensitive to polarization states of the light, providing a new degree of freedom to engineer CVBs.

3.1 Composite Beam Generation using superposition of two or more beams

The general superposition of multiple LG modes with specific polarization states can be expressed as

$$|LG|_{sup} = \sum_{pl,\sigma}^{N} C_{pl} e^{i\delta_{pl}} |LG_{pl,\sigma}| \tag{3.1}$$

where N is the total number of modes for superposition, C_{pl} and δ_{pl} are the amplitude coefficient and the initial phase, respectively, and σ is the circular polarization state. The composite vortex beams are generated for different topological charges as shown in Fig. 3.1, 3.2 and 3.3.Consider the superposition of two collinear LG beams of TC $l1$ and $l2$, giving rise to the formation of complex amplitude written as(with $p = 0$):

$LG_{pl} = LG_{0l_1} + LG_{0l_2}$

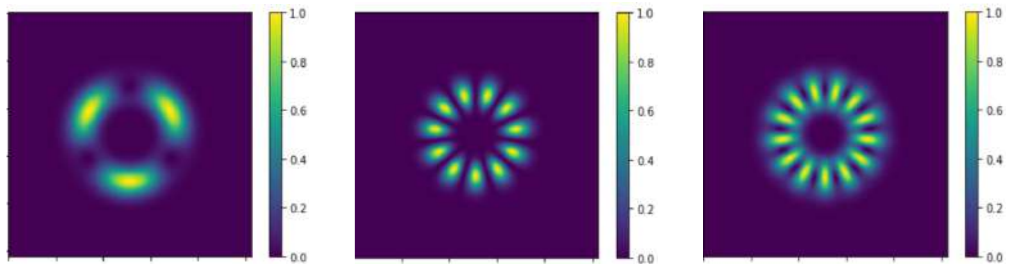

Figure 3.1: Intensity structures of composite vortex beams generated by combining two beams (a)l_1=-9,l_2=-6; (b)l_1=-6,l_2=5; (c)l_1=-9,l_2=4.

Similarly superpositioning three and four LG beams, resulting in complex amplitude written as(with $p = 0$) and shown in Fig. 3.2and 3.3:

For three beams, $LG_{pl} = LG_{0l_1} + LG_{0l_2} + LG_{0l_3}$

For four beams, $LG_{pl} = LG_{0l_1} + LG_{0l_2} + LG_{0l_3} + LG_{0l_4}$

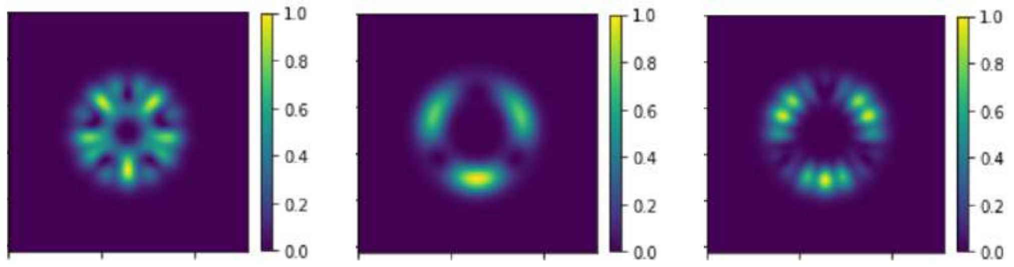

Figure 3.2: Intensity structures of composite vortex beams generated by combining two beams (a)l_1=-7,l_2=-2,l_3=6; (b)l_1=-9,l_2=-6,l_3=-5; (c)l_1=-9,l_2=-6,l_3=7.

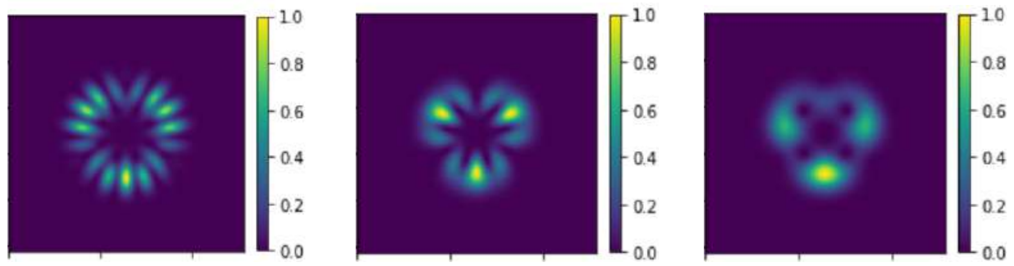

Figure 3.3: Intensity structures of composite vortex beams generated by combining two beams (a)l_1=-9,l_2=-6,l_3=2,l_4=9; (b)l_1=-9,l_2=-6,l_3=-3,l_4=3; (c)l_1=-9,l_2=-6,l_3=-5,l_4=-2.

3.2 Composite Beam Generation by performing logical operations on two binary fork gratings

In this section we will be generating composite vortex beams by superposition of two binary fork gratings of different topological charges. We are doing superposition by performing logical operations such as AND, OR, NAND and NOR on these gratings. The binary fork grating is constructed based on white(1) and black(0) pixels which are binary (0,1), therefore the resultant grating after the logical operations are performed would also be binary based from Table 3.1.

Table 3.1: Truth table of AND, OR, NOR and NAND logical operations.

A	B	AND
0	0	0
0	1	0
1	0	0
1	1	1

A	B	OR
0	0	0
0	1	1
1	0	1
1	1	1

A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

The composite vortex beams thus generated by performing logical operations on two gratings of different topological charges is shown in Fig. 3.4, 3.5, 3.6 and 3.7.

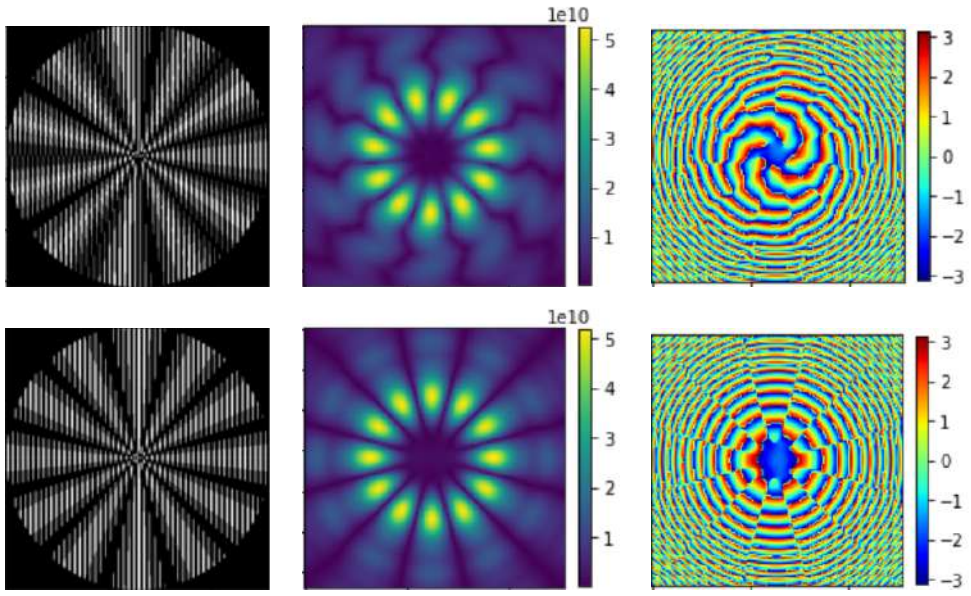

Figure 3.4: Grating, Intensity and phase structures of composite vortex beams generated by ANDing two gratings of charge (a) l_1=-6 and l_2=5; (b)l_1=-6 and l_2=6.

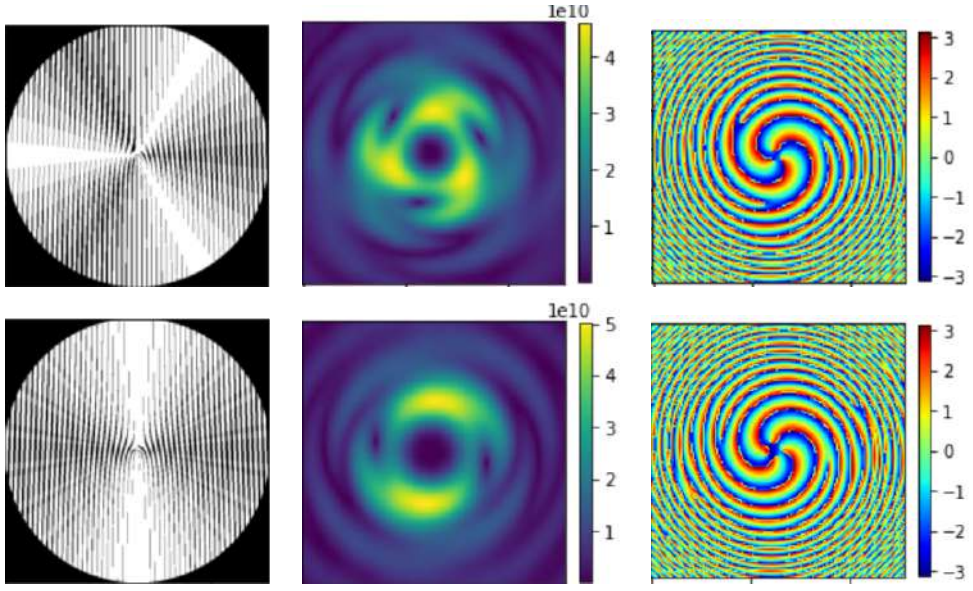

Figure 3.5: Grating, Intensity and phase structures of composite beams generated by ORing two gratings of charge (a) l_1=-6 and l_2=-3; (b)l_1=-6 and l_2=-4.

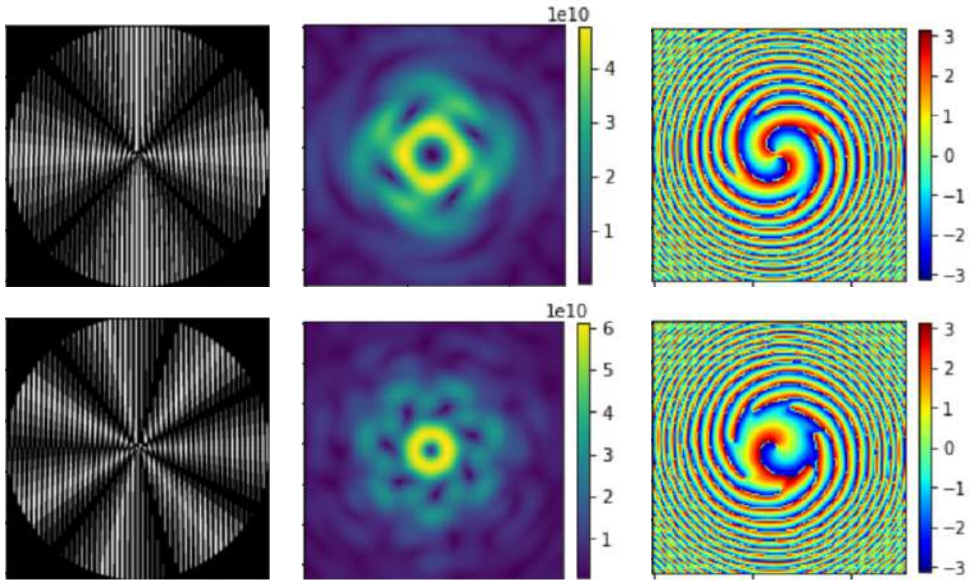

Figure 3.6: Grating, Intensity and phase structures of composite vortex beams generated by NORing two gratings of charge l_1=-6 and l_2=-2; (b)l_1=-6 and l_2=1.

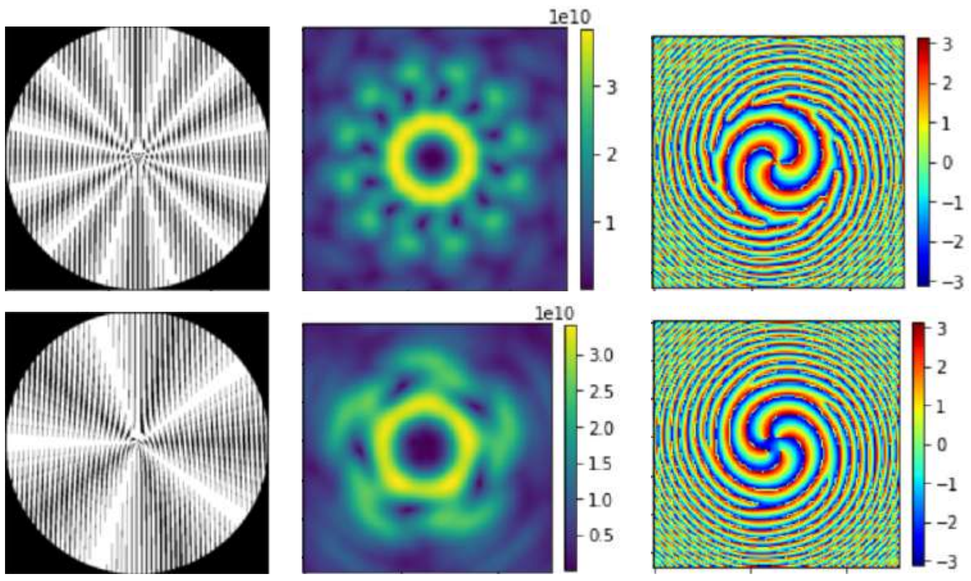

Figure 3.7: Grating, Intensity and phase structures of composite vortex beams generated by NANDing two gratings of charge (a) l_1=-9 and l_2=3; (b) l_1=-9 and l_2=-4.

This is a simple approach to generating composite vortex beams without needing any optical LG beams and without worrying about the stringent optical alignment for an efficient interference between the constituent beams. Because meeting these criteria is not straightforward using interferometric techniques, as the vortex beams of different charges have different radii and intensity profiles.

3.2.1 Translation and Rotation

Now we will perform translation (i.e. shift the fork dislocation horizontally) on one grating and performing logical operation with another grating without translation to generate composite vortex beams as shown in Fig. 3.8 and 3.9

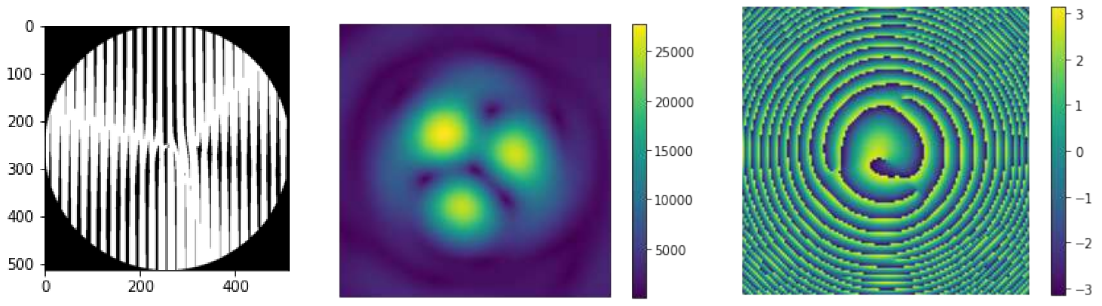

Figure 3.8: Grating, Intensity and phase structures of composite vortex beams generated by ORing translated l_2=1 with l_1=-2.

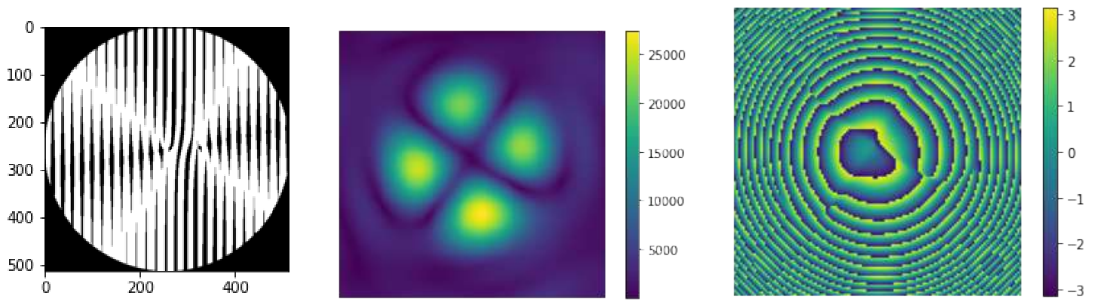

Figure 3.9: Grating, Intensity and phase structures of composite vortex beams generated by ORing translated l_2=-2 with l_1=2.

Similarly we will perform rotation (i.e. rotate the fork dislocation by $1°$) on one grating and performing logical operation with another grating without rotation to generate composite vortex beams. As shown in Fig. 3.10 second grating is rotated by $3°$ with respect to first and ORed to result composite beams. Similarly shown in Fig. 3.11 second grating is rotated by $4°$ with respect to first and ORed to result composite beams. If we keep increasing the angle of rotation their beam axis may not interfere, resulting in no output at first order.

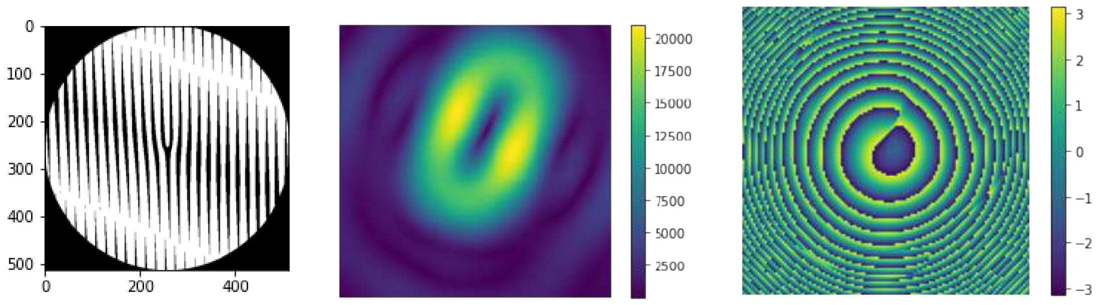
Figure 3.10: Grating, Intensity and phase structures of composite vortex beams generated by ORing rotated(by $3°$) l_2=1 with normal l_1=1.

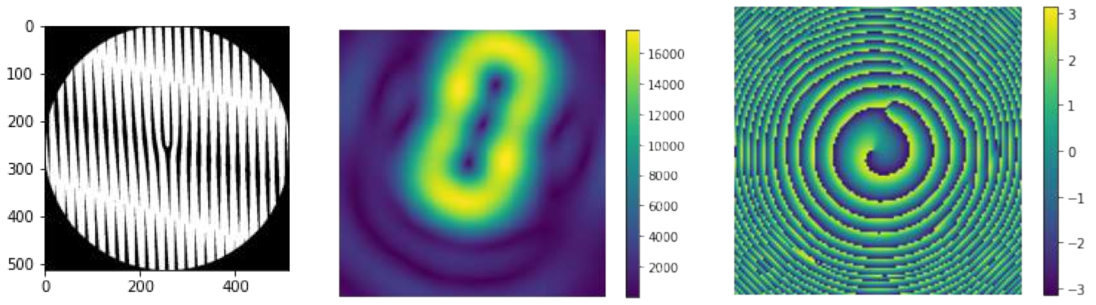
Figure 3.11: Grating, Intensity and phase structures of composite vortex beams generated by ORing rotated (by $4°$) l_2=1 with normal l_1=1.

3.3 Composite Beam Generation by performing logical operations on three binary fork gratings

Similar to previous section, we will now be performing logical operations on a set of three gratings based on 3-input logic truth tables(8 possible combinations). The complex vortex beams thus generated using three gratings of different topological charges is shown in Fig. 3.12, 3.13, 3.14 and 3.15

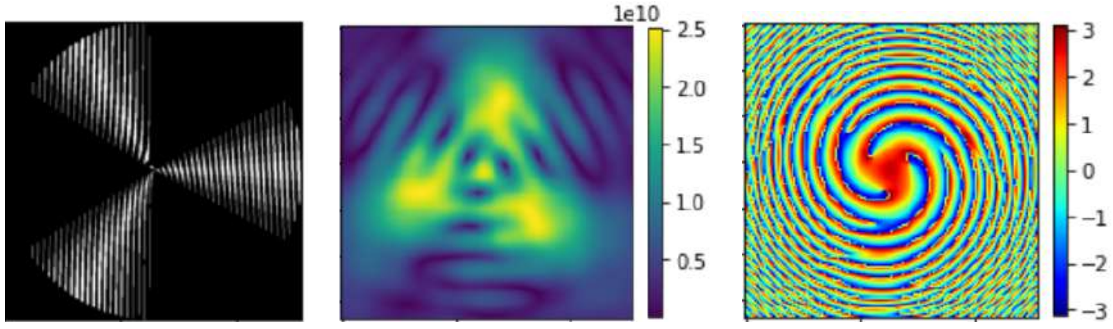

Figure 3.12: Grating, Intensity and phase structures of composite vortex beams generated by three gratings; l_1=-9 AND l_2=-6 AND l_3=-3.

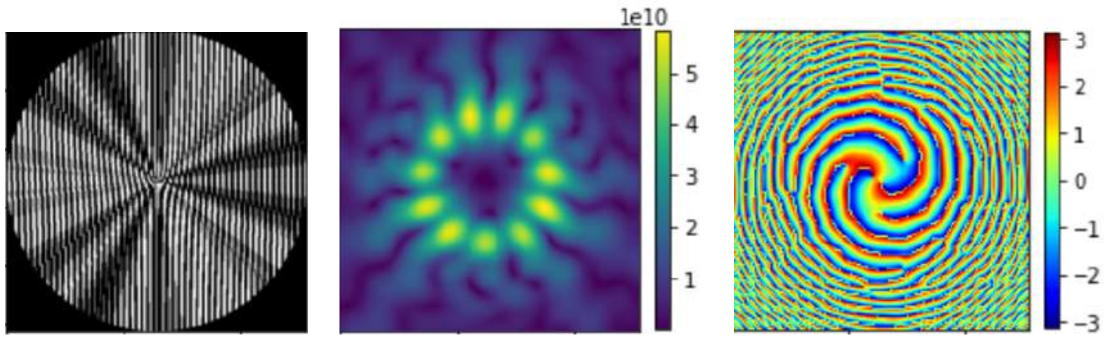

Figure 3.13: Grating, Intensity and phase structures of composite beams generated by three gratings; l_1=-9 AND l_2=-6 NOR l_3=5.

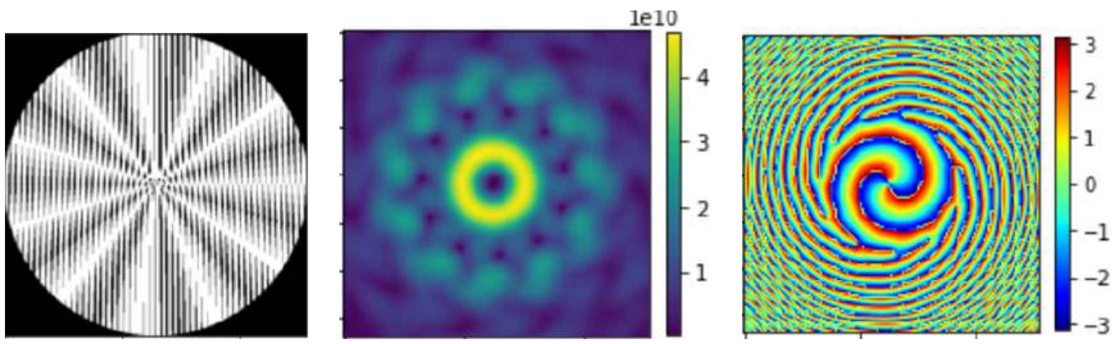

Figure 3.14: Grating, Intensity and phase structures of composite vortex beams generated by three gratings; l_1=-9 NOR l_2=-6 OR l_3=2.

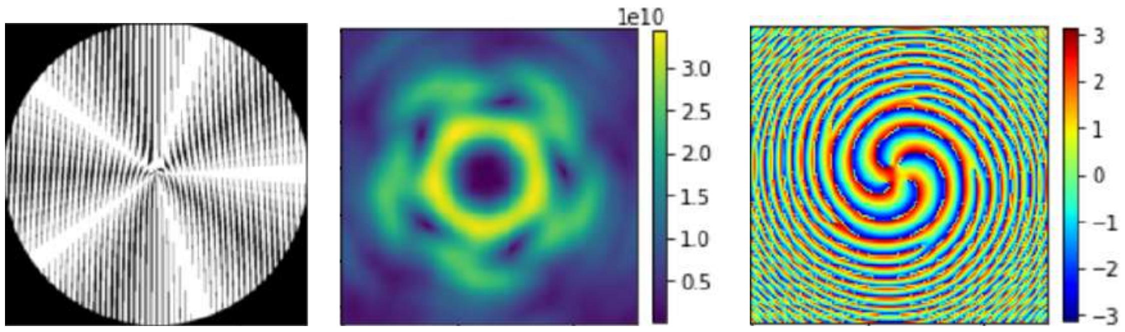

Figure 3.15: Grating, Intensity and phase structures of composite vortex beams generated by three gratings; l_1=-9 NOR l_2=-6 OR l_3=-4.

3.4 Composite Beam Generation by performing logical operations on four binary fork gratings

Similar to previous section, we will now be performing logical operations on a set of four gratings based on 4-input logic truth tables(16 posiible combinations). The complex vortex beams thus generated using four gratings of different topological charges is shown in Fig. 3.16, 3.17, 3.18 and 3.19

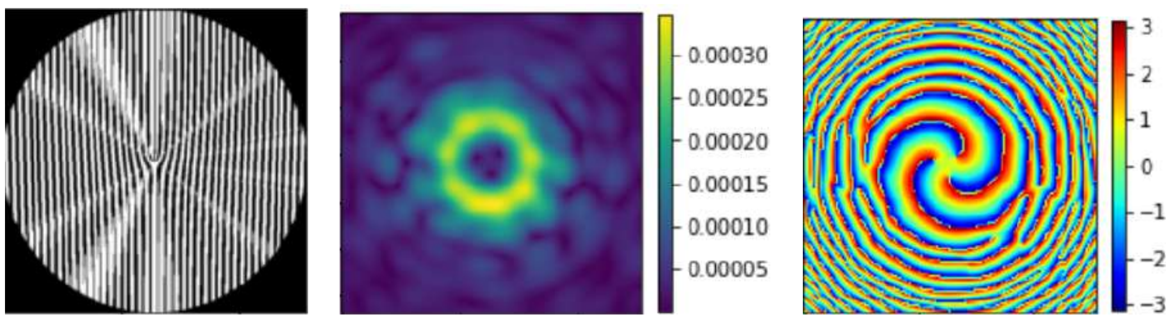

Figure 3.16: Grating, Intensity and phase structures of composite vortex beams generated by four gratings; l_1=-9 NOR l_2=-6 AND l_3=5 OR l_4=3.

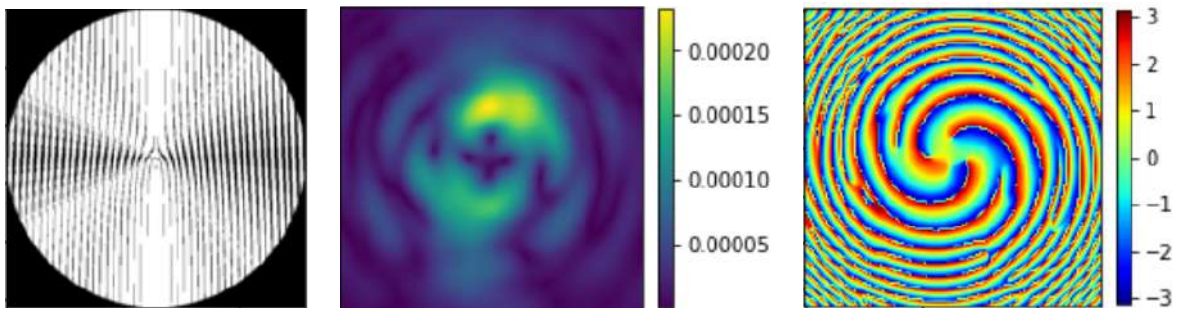

Figure 3.17: Grating, Intensity and phase structures of composite beams generated by four gratings; l_1=-9 AND l_2=-6 OR l_3=-5 OR l_4=-3.

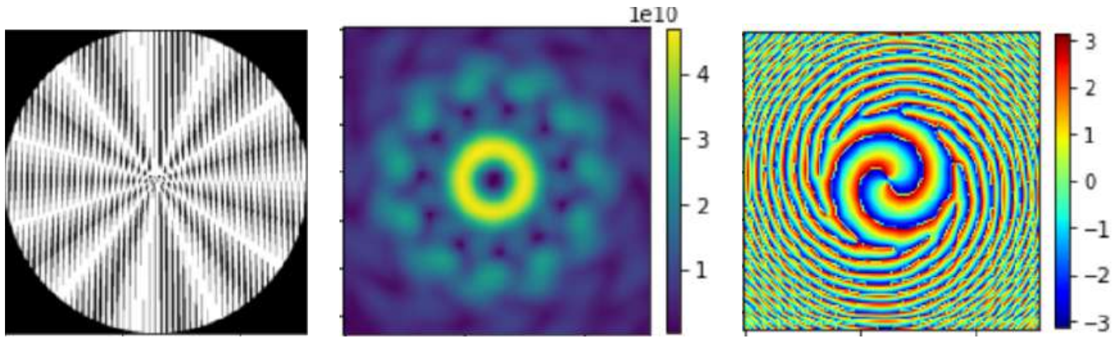

Figure 3.18: Grating, Intensity and phase structures of composite vortex beams generated by four gratings; l_1=-9 AND l_2=-6 OR l_3=-5 OR l_4=-3.

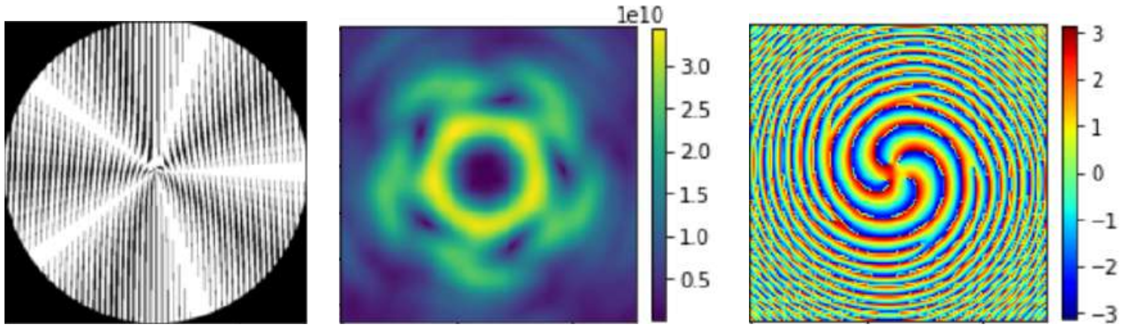

Figure 3.19: Grating, Intensity and phase structures of composite vortex beams generated by three gratings; l_1=-9 NOR l_2=-6 OR l_3=-4.

CHAPTER 4

CONCLUSION

We successfully demonstrated various Composite vortex beams in the 1st diffraction order by performing logical operations on two or more binary fork gratings through simulation. The geometrical parameters of BFGs were optimized for the efficient generation of CV beams. The method was further extended to the generation of CV beams by varying the duty cycle of the binary fork gratings (BFG). This simple generation method may be useful to generate complex beam shapes with engineered phase fronts without complicated interferometry based techniques. The proposed CV beam generation method may have potential applications in diverse fields such as optical micromanipulation and optical communication.

APPENDIX A

APPENDIX

Code Used: Reference from Nirjhar Kumar

```
//Import the required modules
import torch
import matplotlib.pyplot as plt
import matplotlib
import PIL
import numpy as np
import random
import math
import cmath
import cv2
from scipy.special import comb, factorial
from scipy.special import genlaguerre as genlag
from scipy.special import eval_genlaguerre as eval_genlag
from sklearn import preprocessing as norm
from scipy import ndimage
from torch.nn.functional import normalize
import sys
from matplotlib import image
from matplotlib import pyplot


class diffraction:
  size_x_in_m,size_y_in_m,period_x,period_y,Nx,Ny = 0,0,0,0,0,0
  screen_size_x_in_m,screen_size_y_in_m,improve_fact_x,improve_fact_y
    ,R,R2 = 0,0,0,0,0,0
  pre_improvement_screen_size_x_in_pixel,
    pre_improvement_screen_size_y_in_pixel = 128,128
  lambda_m,k = 0,0
  order_xpos_in_m,order_ypos_in_m = 0,0
  incidence_theta_x,incidence_theta_y=0,0

```

```
    def initialize_grating_parameters(self,size_x_in_m = 16.384e-6, size_y_in_m = 16.384e-6,period_x=800e-9,period_y=800e-9,Nx=512,Ny=512):
        self.size_x_in_m = size_x_in_m
        self.size_y_in_m = size_y_in_m
        self.period_x=period_x
        self.period_y=period_y
        self.Nx = Nx; # No. of pixels in x-dimension
        self.Ny = Ny;  # No. of pixels in y-dimension

    def initialize_screen_parameters(self,screen_size_x_in_m = 204.8e-6,screen_size_y_in_m = 204.8e-6,improve_fact_x=1,improve_fact_y=1,R=0.16e-2):
        self.screen_size_x_in_m = screen_size_x_in_m
        self.screen_size_y_in_m = screen_size_y_in_m

        self.improve_fact_x = improve_fact_x
        self.improve_fact_y = improve_fact_y

        self.R=R # distance of screen from gating
        self.R2=R**2

    def inicident_beam_parameters(self,lambda_m,input_intensity = 1.0, RI=1):
        self.lambda_m=lambda_m
        self.RI=RI
        self.k=-(2*np.pi)*complex(np.imag(RI),np.real(RI))/lambda_m

        E0_per_pixel= np.sqrt(input_intensity/np.double(self.Nx*self.Ny));

        return np.ones([self.Nx,self.Ny])*E0_per_pixel



    def inicident_LGbeam_intensityNphase(self,lambda_m=632.8e-9,RI=1, input_intensity = 1.0, p = 0, l = 1, w0 = None, z = 0.0 , Dcenter_x_in_m=0.0, Dcenter_y_in_m=0.0):
        self.lambda_m=lambda_m
        self.RI=RI
```

```python
    # phase constant: -ik :
    self.k=(-(2*np.pi)*complex(np.imag(RI),np.real(RI))/lambda_m) # wavelength of light in vaccuum
    if w0 is None:
      w0 = self.size_x_in_m/4;               # Beam waist

    zR = self.k*w0**2.0/2;       # Calculate the Rayleigh range

    C=np.sqrt(2*factorial(p)/(np.pi*factorial(p+l)))
    # Setup the cartesian grid for the plot at plane z
    xx, yy = np.meshgrid(np.linspace(-self.size_x_in_m/2-Dcenter_x_in_m, self.size_x_in_m/2-Dcenter_x_in_m,self.Nx), np.linspace(-self.size_y_in_m/2-Dcenter_y_in_m, self.size_y_in_m/2-Dcenter_y_in_m,self.Ny));

    # Calculate the cylindrical coordinates
    r = np.sqrt(xx**2 + yy**2);
    phi = np.arctan2(yy, xx);

    U00 = 1.0/(1 + z/zR) * np.exp(-r**2.0/w0**2/(1 + z/zR));
    #w = w0 * np.sqrt(1.0 + z**2/zR**2);
    w= w0 * np.abs(1 + z/zR)
    R = np.sqrt(2.0)*r/w;

    # Lpl from OT toolbox (Nieminen et al., 2004)
    Lpl = comb(p+l,p) * np.ones(np.shape(R));   # x = R(r, z).^2
    for m in range(1, p+1):
        Lpl = Lpl + (-1.0)**m/factorial(m) * comb(p+l,p-m) * R**(2.0*m);

    U = C*U00*R**l*Lpl*np.exp(1j*l*phi)*np.exp(-1j*(2*p + l + 1)*np.arctan(z/np.abs(zR)));

    return np.sqrt(input_intensity/np.double(self.Nx*self.Ny))*U;


  def order_at_screen_center(self,order_num_x=1,order_num_y=0):
    return np.arcsin(order_num_x*self.lambda_m/self.period_x), np.arcsin(order_num_y*self.lambda_m/self.period_y)

```

```python



# depreciated function. instead use 'design_1D_grating' (see just below)

def design_1D_grating(self,period_in_m=None,grating_vect_angle_in_rad=0,grating_duty_cycle=0.5,r_in_m=None,grating_phase_shift_in_rad= 0.0,fork_center_x_in_m=0,fork_center_y_in_m=0,TC=1,miscellenous_parameter=1):
    # populate function parameters with None value with values of the corresponding class variable
    is_grating_vector_odd_multiple_of_piby2 = np.mod(np.round(2*grating_vect_angle_in_rad/np.pi),2)
    if r_in_m is None:
        r_in_m=self.size_x_in_m/2
    if period_in_m is None:
        period_taken_from_class_variable = True
        if is_grating_vector_odd_multiple_of_piby2:
            period_in_m=self.period_y
        else:
            period_in_m=self.period_x
    else:
        period_taken_from_class_variable = False

    if is_grating_vector_odd_multiple_of_piby2:
        on_pixel_size_in_m=self.size_y_in_m/(grating_duty_cycle*self.Ny)
    else:
        on_pixel_size_in_m=self.size_x_in_m/(grating_duty_cycle*self.Nx)

    number_of_on_pixel=np.round(period_in_m/on_pixel_size_in_m)
    period_possible_in_m=number_of_on_pixel*on_pixel_size_in_m
    if np.abs(period_possible_in_m-period_in_m)/period_in_m > 1e-5:
        period_in_m=period_possible_in_m
        if period_taken_from_class_variable:
            if is_grating_vector_odd_multiple_of_piby2:
                self.period_y=period_in_m
    print('Warning: class period_y adjusted to: ',period_in_m,' m')
   else:
    self.period_x=period_in_m
    print('Warning: class period_x adjusted to: ',period_in_m,' m')
  else:
   print('Warning: grating period adjusted to: ',period_in_m, ' m')

 # pixel coordinates of the center of the image
 x0 =int(self.Nx*(fork_center_x_in_m/self.size_x_in_m+0.5));
 y0 = int(self.Ny*(fork_center_y_in_m/self.size_y_in_m+0.5));

 # determines grating period along a given direction 'grating_vect_angle_in_rad'
 x_fringe_density=np.cos(-grating_vect_angle_in_rad)/self.Nx*self.size_x_in_m/period_in_m
 y_fringe_density=np.sin(-grating_vect_angle_in_rad)/self.Ny*self.size_y_in_m/period_in_m
 # unwraped grating phase profile
 xr,yr = np. meshgrid (np.arange(self.Nx)-x0,np.arange(self.Ny)-y0)
 r=np.sqrt(xr**2+yr**2);
 phi=np.arctan2(yr,xr)/(2*np.pi);
 sph = -0*0.1*r*x_fringe_density
 phi1 = TC*phi + x_fringe_density*(xr)+y_fringe_density*(yr) + sph + grating_phase_shift_in_rad/(2*np.pi) +1e-10
 #phi1 = TC*phi + x_fringe_density*(xr)+y_fringe_density*(yr) + sph + np.floor(miscellenous_parameter*phi)/miscellenous_parameter*grating_phase_shift_in_rad/(2*np.pi) +1e-10
 # change the aperture shape to circular
 phi1=phi1 * np.where(r>self.Nx*r_in_m/self.size_x_in_m, 0,1)
 # wraped grating phase profile
 r1 = np.mod(phi1,1); #phase mod 2 pi in units of 2pi
 return np.where(r1>=(1-grating_duty_cycle), 1, 0)


def design_1D_grating_empty(self):
 Ny=self.Ny
```

```
  Nx=self.Nx
  C = np.ones([Ny,Nx]);
  for x in range(Nx):
    for y in range(Ny):
      x0 = Nx/2; # coordinates of the center of the image
      y0 = Ny/2;
      xr=x-x0;
      yr=y-y0;
      r=np.sqrt(xr**2+yr**2); # radial coordinate
      #if((xr<=Nx/2) or (yr<=Ny/2)):
      if r<=Nx/2:#aperture:
        C = C;
      else:
        C[y,x]=0
  return C

####angular illumination along x
def angular_illumination(self,theta):
  self.incidence_theta_x=theta[0]
  self.incidence_theta_y=theta[1]
  phasex=(2*np.pi*np.sin(self.incidence_theta_x)/self.lambda_m)*np.
 linspace(-self.size_x_in_m/2,self.size_x_in_m/2,self.Nx)
  phasey=(2*np.pi*np.sin(self.incidence_theta_y)/self.lambda_m)*np.
 linspace(-self.size_y_in_m/2,self.size_y_in_m/2,self.Ny)
  phasexx, phaseyy = np.meshgrid(phasex, phasey, sparse=True)
  C_phase= phasexx+phaseyy
  return C_phase


def incident_beam_phase_profile(self,TC_inc=0):
  Ny=self.Ny
  Nx=self.Nx
  C2 = np.zeros([Ny,Nx]);
  for x in range(Nx):
      #m=m-0.1
      for y in range(Ny):
          x0 = Nx/2; # coordinates of the center of the image
          y0 = Ny/2;
          xr=x-x0;
          yr=y-y0;
          r=np.sqrt(xr**2+yr**2); # radial coordinate
          if r<=Nx/2:
            phi=np.arctan2(yr,xr);  # angular coordinate
            C2[y,x] = TC_inc*phi
  return C2

def set_screen_position(self,order_num_x,order_num_y):
  self.order_xpos_in_m=np.tan(np.arcsin(self.lambda_m*order_num_x/
 self.period_x)-self.incidence_theta_x)*self.R
  self.order_ypos_in_m=np.tan(np.arcsin(self.lambda_m*order_num_y/
 self.period_y)-self.incidence_theta_y)*self.R
def set_screen_position2(self,order_num_x,order_num_y):
  self.order_xpos_in_m=np.tan(np.arcsin(self.lambda_m*order_num_x/
 self.period_x))*self.R
  self.order_ypos_in_m=np.tan(np.arcsin(self.lambda_m*order_num_y/
 self.period_y))*self.R
def set_screen_position3(self,order_num_x,order_num_y,n1=1,n2=1):
  self.order_xpos_in_m=np.tan(np.arcsin(self.lambda_m*order_num_x/
 self.period_x)-np.arcsin((n1/n2)*np.sin(self.incidence_theta_x)))*
 self.R
  self.order_ypos_in_m=np.tan(np.arcsin(self.lambda_m*order_num_y/
 self.period_y)-np.arcsin((n1/n2)*np.sin(self.incidence_theta_y)))*
 self.R
def set_screen_position4(self,order_num_x,order_num_y):
  self.order_xpos_in_m=np.tan(-self.incidence_theta_x)*self.R
  self.order_ypos_in_m=np.tan(-self.incidence_theta_y)*self.R
def set_screen_position5(self,t1=16e-4,t2=16e-4,incidence_theta
 =(0,0)):
  self.order_xpos_in_m=np.tan(-((t1+t2)/t2)*np.tan(incidence_theta
 [0]))*self.R
  self.order_ypos_in_m=np.tan(-((t1+t2)/t2)*np.tan(incidence_theta
 [1]))*self.R

def run_huygens_construct(self,C_amp,C_phase,C_grating,
 grating_type='amplitude grating', lambda_m=500e-9, RI=1):
  self.lambda_m = lambda_m
  self.RI = RI
  self.k=(-(2*np.pi)*complex(np.imag(RI),np.real(RI))/lambda_m)
  if (grating_type=='amplitude grating'):
```

```python
  C=C_grating*C_amp#*np.exp(self.R*np.real(self.k)) # grating and amplitude multiplied togather to create a single matrix for amplitude modulation
else:
  C=C_amp#*np.exp(self.R*np.real(self.k))
  C_phase=C_phase+C_grating*np.pi

#Grating vector mesh:
g_division_fact_x =int(C.shape[0]/128)
g_division_fact_y =int(C.shape[1]/128)

grating_split_x=np.linspace(-0.5,0.5,1+g_division_fact_x)*self.size_x_in_m
grating_split_y=np.linspace(-0.5,0.5,1+g_division_fact_y)*self.size_y_in_m

grating_p_split_x=(np.linspace(0,1,1+g_division_fact_x)*C.shape[0]).astype('int')
grating_p_split_y=(np.linspace(0,1,1+g_division_fact_y)*C.shape[1]).astype('int')

screen_size_x_in_pixel = self.pre_improvement_screen_size_x_in_pixel
screen_size_y_in_pixel = self.pre_improvement_screen_size_y_in_pixel

screen_split_x=np.linspace(-0.5,0.5,1+self.improve_fact_x)*self.screen_size_x_in_m
screen_split_y=np.linspace(-0.5,0.5,1+self.improve_fact_y)*self.screen_size_y_in_m

#Screen vector mesh:
pf= torch.zeros(screen_size_x_in_pixel*self.improve_fact_x,0,device=cuda)
for r_unit in range(self.improve_fact_x):
  S_x_vec=torch.linspace(self.order_xpos_in_m+screen_split_x[r_unit],self.order_xpos_in_m+screen_split_x[r_unit+1],screen_size_x_in_pixel,device=cuda)
  pf_y= torch.zeros(0,screen_size_y_in_pixel).to(cuda)
  for c_unit in range(self.improve_fact_y):
    S_y_vec=torch.linspace(self.order_ypos_in_m+screen_split_y[c_unit],self.order_ypos_in_m+screen_split_y[c_unit+1],screen_size_y_in_pixel,device=cuda)
    S_vec=torch.zeros((S_x_vec.size()[0],S_y_vec.size()[0])).to(cuda)

    S_vec=S_y_vec.view(S_y_vec.size()[0],1)*1j + S_vec
    S_vec=(S_x_vec + S_vec)

    fp_unit= torch.zeros(  S_x_vec.size()[0]*S_y_vec.size()[0],1,device=cuda)
    for gr_unit in range(g_division_fact_x):
      G_x_vec=torch.linspace(grating_split_x[gr_unit],grating_split_x[gr_unit+1],int(C.shape[0]/g_division_fact_x),device=cuda)
      for gc_unit in range(g_division_fact_y):
        G_y_vec=torch.linspace(grating_split_y[gc_unit],grating_split_y[gc_unit+1],int(C.shape[1]/g_division_fact_y),device=cuda)
        G_vec=torch.zeros((G_x_vec.size()[0],G_y_vec.size()[0]),device=cuda)

        G_vec=G_y_vec.view(G_y_vec.size()[0],1)*1j + G_vec
        G_vec=(G_x_vec + G_vec)

        E0=torch.polar(torch.tensor(C[grating_p_split_y[gc_unit]:grating_p_split_y[gc_unit+1],grating_p_split_x[gr_unit]:grating_p_split_x[gr_unit+1]],dtype=torch.float,device=cuda),torch.tensor(C_phase[grating_p_split_y[gc_unit]:grating_p_split_y[gc_unit+1],grating_p_split_x[gr_unit]:grating_p_split_x[gr_unit+1]],dtype=torch.float,device=cuda))

        #Calculate optical path from each point on the grating to each point on the screen:
        path_diff2 = torch.zeros((S_x_vec.size()[0]*S_y_vec.size()[0],G_x_vec.size()[0]*G_y_vec.size()[0]),device=cuda)
        path_diff2=path_diff2+G_vec.view(1,G_x_vec.size()[0]*G_y_vec.size()[0])
        path_diff2=path_diff2-S_vec.view(S_x_vec.size()[0]*S_y_vec.size()[0],1)
```

```
          path_diff2=path_diff2*path_diff2.conj()+self.R2
          #print(self.k)
          intk=torch.sqrt(path_diff2)
          #print("intk = ",intk)
          path_diff2=torch.exp(intk*self.k)/path_diff2
          #path_diff2=torch.exp(intk*np.imag(self.k)*1j)/path_diff2

          #print("intk1j = ",intk/1j)
          #print("path_diff21j =", torch.exp(intk/1j))
          #print("path_diff2 =", path_diff2)

          #Calculate the diffractio  n pattern:

          E0=(E0.reshape(G_x_vec.size()[0]*G_y_vec.size()[0],1))
          #E0=(E0.reshape(G_x_vec.size()[0]*G_y_vec.size()[0],1))*torch.exp(intk*np.real(self.k))
          #fp_unit=fp_unit-1j*self.R*torch.mm(path_diff2,E0)/self.lambda_m
          fp_unit=fp_unit-1j*self.R*torch.mm(path_diff2,E0)*self.RI/self.lambda_m

      pf_y= torch.cat((  pf_y,fp_unit.view(S_x_vec.size()[0],S_y_vec.size()[0])    ),0)
    pf= torch.cat((  pf,pf_y   ),1)
  return pf.to(cpu).numpy()






def plot_inputs(self,input_beam_amplitude,input_beam_phase,grating,img_file_name_prefix='binary'):
  print("Input: ")
  suffix='_grating'
  image_type='.jpg'
  screen_size_x_in_pixel=self.Nx
  screen_size_y_in_pixel=self.Ny
  fig = plt.figure(figsize=(10, 10)) # set the height and width in inches
```

```python

        # plot input beam Intensity profile
        plt.subplot(1,3,1)
        input_beam_amplitude[0,0]=0
        plt.imshow(input_beam_amplitude, interpolation='nearest')#,cmap='hot')
        plt.colorbar(fraction=0.045)


        # plot input beam Phase profile
        plt.subplot(1,3,2)
        plt.imshow(input_beam_phase,cmap='jet')
        plt.colorbar(fraction=0.045)

        # input grating
        plt.subplot(1,3,3)
        plt.imshow(grating,cmap='gray')
        # save as image
        matplotlib.image.imsave(img_file_name_prefix+suffix+image_type, grating, cmap='gray')

        # setting subplots' title
        ax_list = fig.axes
        ax_list[0].set_title('Incident_beam_intensity_profile')
        ax_list[2].set_title('Incident_beam_phase_profile')
        ax_list[4].set_title(img_file_name_prefix+suffix)

        plt.show()

    def plot_inputs_grat(self,grating1,grating2,gratinge, img_file_name_prefix='binary'):
        print("Input: ")
        suffix='_grating'
        image_type='.jpg'
        screen_size_x_in_pixel=self.Nx
        screen_size_y_in_pixel=self.Ny
        fig = plt.figure(figsize=(10, 10)) # set the height and width in inches

        # plot input beam Intensity profile
    plt.subplot(1,3,1)
    plt.imshow(grating1,cmap='gray')

    # save as image
    matplotlib.image.imsave(img_file_name_prefix+suffix+image_type, grating1, cmap='gray')


    # plot input beam Phase profile
    plt.subplot(1,3,2)
    plt.imshow(grating2,cmap='gray')
    # save as image
    matplotlib.image.imsave(img_file_name_prefix+suffix+image_type, grating2, cmap='gray')

    # input grating
    plt.subplot(1,3,3)
    plt.imshow(gratinge,cmap='gray')
    # save as image
    matplotlib.image.imsave(img_file_name_prefix+suffix+image_type, gratinge, cmap='gray')

    # setting subplots' title
    ax_list = fig.axes
    ax_list[0].set_title('Grating_1')
    ax_list[1].set_title('Grating_2')
    ax_list[2].set_title("Equi_Gratiing")

    plt.show()


  def plot_grating(self,grating,fig_handle= None, img_file_name_prefix='binary',image_type='.bmp',cmap='gray'):
    if fig_handle is None:
      fig_handle = plt.figure(figsize=(10, 10)) # set the height and width in inches
    print("Input: ")
    suffix='_grating'
    screen_size_x_in_pixel=self.Nx
    screen_size_y_in_pixel=self.Ny
```

```python

    plt.imshow(grating,cmap = cmap)
    # save as image
    if image_type is not '.':
      matplotlib.image.imsave(img_file_name_prefix+suffix+image_type, grating, cmap = cmap)
    if fig_handle is None:
      plt.show()



  def plot_outputs(self,diffraction_pattern,img_file_name_prefix='1stOrder',intensity_plot_clim_min=None,intensity_plot_clim_max=None):
    print("Output: ")
    suffix_ip='_intensity_profile'
    suffix_pp='_phase_profile'
    image_type='.jpg'
    fig = plt.figure(figsize=(10, 10)) # set the height and width in inches

    # plot output Intensity Profile
    plt.subplot(1,3,1)
    plt.imshow(np.abs(diffraction_pattern))##'viridis',cmap='copper'
    plt.colorbar(fraction=0.045)

    # plot output Phase Profile
    plt.subplot(1,3,3)
    plt.imshow(np.angle(diffraction_pattern),cmap='jet') # cmap= 'twilight',cmap='jet'
    plt.colorbar(fraction=0.045)


#
d1=diffraction()
Nx=512
Ny=512
wavelength=633e-9 #633e-9
period = 950e-9  #950
size_x_in_m = 40e-6
```

```
size_y_in_m = 40e-6
screen_size_x_in_m = 1.0e-4 #2.7e-4
screen_size_y_in_m = 1.0e-4

#tcs=[-7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7]
#tcs=[-16,-15,-14,-13,-12,-11,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9,10,11,1
#tcs=[-2,-1,0,1,2]
dc=0.5#dc1=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
D_order_num=1
d1.initialize_grating_parameters(size_x_in_m ,size_y_in_m ,period_x=period,period_y=period,Nx=Nx,Ny=Ny)
d1.initialize_screen_parameters(screen_size_x_in_m ,screen_size_y_in_m ,improve_fact_x=1,improve_fact_y=1,R=(1000e-6))

for thet1 in [0]:#range(0,360,5):

  for thet2 in [0]:#range(10,180,10):

    for tc1 in [1]:

      for tc1 in [1]:

#Initializing gratings, Incident beams
        grating1=d1.design_1D_grating(TC=tc1,grating_duty_cycle=dc,grating_vect_angle_in_rad=np.deg2rad(0),grating_phase_shift_in_rad=np.deg2rad(0))
        #grating2=d1.design_1D_grating(TC=tc2,grating_vect_angle_in_rad=np.deg2rad(thet1),grating_phase_shift_in_rad=np.deg2rad(thet2))

        input_beam_amplitude=d1.inicident_beam_parameters(lambda_m=wavelength)
        #input_beam_amplitude=d1.inicident_LGbeam_intensityNphase(lambda_m=wavelength)
        input_beam_phase=d1.angular_illumination(d1.order_at_screen_center(D_order_num,0))

#Grating logical operations
```

```
    gratinge=grating1
    #gratinge=np.logical_or(grating1,grating2)
    #gratinge=np.logical_not(np.logical_and(grating1,grating2))
    print("tca=",tc1,"tc1=",tc1," dc=",dc,"operation=OR-OR"," grating_vect_angle=",thet1,"grating_phase_shift=",0)
    for x in range(Nx):
      for y in range(Ny):
        x0 = Nx/2; # coordinates of the center of the image
        y0 = Ny/2;
        xr=x-x0;
        yr=y-y0;
        r=np.sqrt(xr**2+yr**2); # radial coordinate
        #if((xr<=Nx/2) or (yr<=Ny/2)):
        if r<=Nx/2:#aperture:
          gratinge = gratinge;
        else:
          gratinge[y,x]=0



    d1.plot_inputs(input_beam_amplitude,input_beam_phase,gratinge)

    if (1):
      d1.set_screen_position(order_num_x=D_order_num,order_num_y=0)
      #print(d1.set_screen_position(order_num_x=D_order_num, order_num_y=0))

      diffraction_pattern=d1.run_huygens_construct3(input_beam_amplitude,input_beam_phase,gratinge,'amplitude grating')



      d1.plot_outputs(diffraction_pattern)

      P_out=np.sum((np.abs(diffraction_pattern)**2)*((screen_size_x_in_m/128)**2))##(size_x)**2)
      print(P_out)
```

```
        #I_op=(((np.abs(diffraction_pattern)(size_x)2)2))(
    screen1_size/128)**2
        P_in=np.sum((np.abs(input_beam_amplitude)**2)*((size_x_in_m
    /512)**2))
#Efficiency

        Eff = ((P_out/P_in))*100
        print("P_out=",P_out,"P_in=",P_in,"Eff%=",Eff)
```

Listing A.1: Python code for CVB Generation

REFERENCES

1. Bouchal, Z. and R. Celechovsky, vortex states of light as information carriers," New Journal of Physics, Vol. 6, No. 6, 1-15, 2014.

2. Allen, L., M. W. Beijersbergen, R. J. Spreeuw, et al., "Orbital angular momentum of light and the transformation of Laguerre-Gaussian laser modes," Physical Review A, Vol. 45, No. 11, 8185-8189, 1992.

3. Paterson, C., "Atmospheric turbulence and orbital angular momentum of single photons for optical communication," Phys. Rev. Lett., Vol. 94, No. 15, 477-481, 2005.

4. Su, T., R. P. Scott, S. S. Djordjevic, et al., "Demonstration of free space coherent optical communication using integrated silicon photonic orbital angular momentum devices," Opt. Express, Vol. 20, No. 9, 9396-9402, 2012.

5. Bouchal, Z. and R. Celechovsky, "Mixed vortex states of light as information carriers," New Journal of Physics, Vol. 6, No. 6, 1-15, 2004.

6. Gelechovsky, R. and Z. Bouchal, "Generation of variable mixed vortex fields by a single static hologram," Journal of Modern Optics, Vol. 53, No. 4, 473-480, 2006.

7. Liu, Y. D. and C. Q. Gao, "Superposition and detection of two helical beams for optical orbital angular momentum communication," Opt. Commun., Vol. 281, 3636-3639, 2008.

8. Wang, J., J. Y. Yang, I. M. Fazal, et al., "Terabit free-space data transmission employing orbital angular momentum multiplexing," Nature Photonics, Vol. 6, No. 7, 488-496, 2012.

9. Huang, H., Y. Cao, G. D. Xie, et al., "Crosstalk mitigation in a free-space orbital angular mo- mentum multiplexed communication link using 4×4 MIMO equalization," Opt. Lett., Vol. 39, No. 15, 4360-4363, 2014.

10. Huang, H., G. D. Xie, Y. X. Ren, et al., "4 × 4 MIMO equalization to mitigate crosstalk degradation in a four-channel free-space orbital-angular-momentum-multiplexed system using heterodyne detection," European Conference and Exhibition on Optical Communication, 708- 710, IET, 2013.

11.Ren, Y. X., Z. Wang, G. D. Xie, et al., "Demonstration of OAM-based MIMO FSO link using spatial diversity and MIMO equalization for turbulence mitigation," Optical Fiber Communications Conference and Exhibition(OFC), 2016.

12. Sztul, H. H. and R. R. Alfano, "Double slit interference with Laguerre-Gaussian beams," Opt. Lett., Vol. 31, No. 7, 999-1001, 2006.

13. Soares, W. C., D. P. Caetano, E. J. S. Fonseca, et al., "Direct determination of light beams' topological charges using diffraction," Conference on Quantum Electronics and Laser Science, 1-2, 2008.

14. Ghai, D. P., P. Senthilkumaran, and R. S. Sirohi, "Single-slit diffraction of an optical beam with phase singularity," Optics and Lasers in Engineering, Vol. 35, No. 23, 123-126, 2009.

15. Brandao, P. A. and S. B. Cavalcanti, "Topological charge identification of partially coherent light diffracted by a triangular aperture," Phys. Lett. A, Vol. 380, No. 47, 4013-4017, 2016.

16. Chen, J., X. Z. Ke, and Y. M. Yang, "Laguerre-Gaussian beam diffraction and dispersion of the orbital angular momentum," Acta Optica Sinica, Vol. 34, No. 4, 0427001-0427001-7, 2014.

17.Ke, X. Z., J. Chen, and H. Lv, "Study of double-slit interference experiment on the orbital angular momentum of LG beam," Scientia Sinica Physica, Mechanica Astronomica, Vol. 42, No. 10, 996-1002, 2012.

18.Single-shot generation of composite optical vortex beams using hybrid binary fork

gratings Nirjhar Kumar, Ankit Arora, and Ananth Krishnan.
19. Creating Composite Vortex Beams with a Single Geometric Metasurface Yang Ming,Yuttana Intaravanne,Hammad Ahmed,Mitchell Kenney,Yan-qing Lu,Xianzhong Chen.
20. Implementing digital holograms to create and measure complex-plane optical fields. American Journal of Physics 84, 106 (2016).
21. Orbital angular momentum: origins, behavior and applications. Alison M. Yao and Miles J. Padgett. Published May 15, 2011 (Doc. ID 136333).
22. Orbital angular momentum 25 years on. MILES J. PADGETT School of Physics and Astronomy, University of Glasgow, Glasgow, Scotland, UK.
23. Srinivas Pachava, Raghu Dharmavarapu, Anand Vijayakumar, Sruthy Jayakumar, Amogh Manthalkar, Awakash Dixit, Nirmal K. Viswanathan, Balaji Srinivasan, Shanti Bhattacharya, “Generation and decomposition of scalar and vector modes carrying orbital angular momentum: a review,” Opt. Eng. 59(4), 041205 (2019).
24. Yuan, X.; Xu, Y.; Zhao, R.; Hong, X.; Lu, R.; Feng, X.; Chen, Y.; Zou, J.; Zhang, C.; Qin, Y.; et al. Dual-Output Mode Analysis of Multimode Laguerre-Gaussian Beams via Deep Learning. Optics 2021, 2, 87–95.
25. Generation of composite vortex beams by independent Spatial Light Modulator pixel addressing. Mateusz Szatkowski , Jan Masajada , Ireneusz Augustyniak , Klaudia Nowacka.
26. Composite vortex beams by coaxial superposition of Laguerre–Gaussian beams. Sujuan Huang , Zhuang Miao,Chao He,Fufei Pang,Yingchun Li,Tingyun Wang.

LIST OF PAPERS BASED ON THESIS

1. Yadunanda B N, Nirjhar Kumar, Anil. GENERATION OF OPTICAL COMPOSITE VORTEX BEAMS USING LOGICAL OPERATIONS ON BINARY FORK GRATINGS *Journal*, Submition in progress, (2022).