Holographically generated light potentials for cold-atom experiments

Paul Schroff

PhD Thesis

Experimental Quantum Optics and Photonics Group
Department of Physics
University of Strathclyde, Glasgow

Primary supervisor: Prof. Stefan Kuhr
Secondary supervisors: Prof. Jonathan Pritchard,
Dr Arthur La Rooij

Glasgow, 24th February 2025

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Signed:

Date: 24th February 2025

This thesis contains material previously published by the author during the course of their PhD. The author's contributions to each publication are stated below.

Journal article: Schroff, P., Haller, E., Kuhr, S. & Rooij, A. L. Rapid stochastic spatial light modulator calibration and pixel crosstalk optimization. *Opt. Express* **32,** 48957–48971. doi:`10.1364/OE.539548` (2024).

I implemented the simulations, built the experimental setup and performed the measurements. A. La Rooij and S. Kuhr supervised the project and helped analyse the results. All authors contributed to the preparation of the manuscript.

Journal article: Schroff, P., La Rooij, A., Haller, E. & Kuhr, S. Accurate holographic light potentials using pixel crosstalk modelling. *Sci. Rep.* **13,** 3252. doi:`10.1038/s41598-023-30296-6` (2023).

I conducted the simulations, built the experimental setup and performed the measurements. A. La Rooij, E. Haller and S. Kuhr supervised the project and helped analyse the results. All authors contributed to the manuscript.

Conference article: Schroff, P., La Rooij, A., Haller, E. & Kuhr, S. *Accurate Holographic Light Potentials For Cold-Atom Experiments* in *Optica Imaging Congress (3D, COSI, DH, FLatOptics, IS, pcAOP)* (Optica Publishing Group, 2023), HM1D.5. doi:`10.1364/DH.2023.HM1D.5`.

I carried out the simulations, constructed the experimental setup, and conducted the measurements. A. La Rooij, E. Haller and S. Kuhr supervised the project. A. La Rooij and S. Kuhr assisted with the data analysis and participated in the manuscript preparation.

Parts of the code used in this thesis have been published on GitHub, available at `https://github.com/paul-schroff/holograpdy`. The corresponding documentation can be found in Appendix B and is also available online at `https://holograpdy.readthedocs.io`.

Signed:

Date: 24th February 2025

Abstract

This thesis presents computational and experimental techniques to generate light potentials of arbitrary shapes holographically using a phase-modulating liquid-crystal spatial light modulator (SLM) for experiments with ultracold atoms. Many quantum-simulation and quantum-computing experiments using ultracold atoms have benefited from programmable local control on a microscopic scale. Inhomogeneities in the light potentials used in these experiments must be reduced to mitigate dephasing effects or heating of the atoms. Further, in applications where laser power is limited, a high efficiency is desirable. Here, I demonstrate the generation of holographic light potentials with a root-mean-squared (RMS) error below 1% and a measured efficiency of up to $\sim 40\%$. I show that in a Fourier imaging setup, for light potentials which occupy a significant fraction of the addressable area in the image plane, a parasitic effect on the SLM known as pixel crosstalk or fringing field effect limits the accuracy of the light potential. By modelling this pixel crosstalk and by compensating for its effects, the error in the light potential is reduced by a factor of ~ 5. A gradient-based optimisation algorithm is employed to calculate the SLM phase pattern for the desired light potential. To reduce experimental errors, we measure the wavefront of the incident laser beam to within $\lambda/120$ and employ an iterative camera feedback algorithm. To downscale the light potentials to a microscopic scale, a high-NA microscope objective is used. Finally, a fast method to calibrate the experimental setup is demonstrated, reducing the runtime from ~ 3 hours to ~ 5 minutes, maintaining an RMS error of below 1% in the resulting light potentials.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to begin by thanking my supervisor, Stefan Kuhr, for spreading his contagious enthusiasm for cold-atom physics and quantum optics among students, myself included. You have been an outstanding mentor during my time at Strathclyde, and I am grateful for the opportunity to have worked in your lab since my BSc degree. Your positive and supportive attitude kept me motivated throughout the last five years and encouraged me to explore new ideas. Thank you also to my postdoc, Arthur La Rooij, for his continuous support in the lab and the occasional cocktail party or pint in the pub. Your passion for physics made our journal clubs very entertaining and brought excitement to the sometimes tedious lab work. Special thanks to my good friend and colleague, Christopher Parsonage, for his irreplaceable company during our countless lunches, where we spent a significant fraction of our PhD stipend on specials at Paesanos. I am also grateful to Elmar Haller and Andrea Di Carli for their friendly help with all physics-related questions and to my secondary supervisor, Jonathan Pritchard, for proofreading this thesis. A big thank you to my fellow PhD students for their camaraderie. I would also like to thank everyone else at EQOP for making me feel so welcome in the group. I am looking forward to working with you all for another two years!

I also want to thank my parents and my sister for their unconditional support and guidance throughout my life, especially during the more challenging times of my PhD. Without you, writing this thesis would have been immeasurably more difficult, and I cannot express my gratitude enough. Finally, thank you to my friends from home and to my extended family for visiting me in Scotland every once in a while and for their presence in spirit while far away from home.

Dedicated to my parents for
their loving support.

Chapter 1

Introduction

This thesis is focused on optimising holographically generated light potentials specifically for applications in quantum-simulation and quantum-computing experiments using ultracold atoms. However, there are many other fields which might benefit from the optimisation and calibration techniques presented here. Outside the scope of cold-atom experiments, tailored light potentials are used in biomedical applications such as optogenetic stimulation [1], non-invasive imaging through tissue [2], and high-resolution 3D imaging, microscopy and tomography. In the field of forensics, holographic techniques were developed to record fingerprints [3]. Holographic beam shaping is used in manufacturing applications including metal processing or lithography [4]. In the automotive and aerospace industries, holographic head-up displays emerged as an application [5] whereas holographic near-eye displays are an active area of research for virtual and augmented reality applications [6–9]. As holographically generated light potentials are very sensitive to aberrations in the optical system used to generate them, producing light potentials with a low error is challenging. Despite the complexity associated with a holographic imaging setup, the prospect of achieving higher efficiencies and lower error has driven the development of sophisticated hologram calculation techniques such as camera-in-the-loop calibration [6]. In this work, we optimise static potentials that are generated by a liquid-crystal spatial light modulator which is imaged in the focal plane of a Fourier lens. This is a common experimental setting used in, for example, top-hat addressing beams and tweezer arrays for Rydberg atom array experiments [10] where the efficiency and the uniformity of the generated potentials is important (see

Section 1.1.3).

1.1 Programmable local control in cold-atom experiments: State-of-the-art

The capability to sculpt light into potentials of arbitrary shape has created many new opportunities in cold-atom experiments which will be discussed in this section. Applications include atomtronics [11, 12], tailored potentials for experiments with optical lattices [13–16] and quantum information platforms using Rydberg arrays [10, 17, 18]. These applications require smooth light potentials that minimise inhomogeneities and the resulting dephasing effects. Additionally, a high efficiency is desirable for experiments involving larger atom numbers or where laser power is limited. The hardware used to tailor these potentials includes liquid-crystal spatial light modulators (SLMs), binary amplitude-modulating digital micromirror devices (DMDs) and rapidly scanning acousto-optic deflectors (AODs). Each of these devices has its own benefits and drawbacks, making them suitable for different applications in cold atom experiments as outlined in this section.

1.1.1 Ultracold quantum gasses

The ability to generate arbitrary light potentials has enabled the realisation of a range of experiments that investigate out-of-equilibrium dynamics in Bose-Einstein condensates (BECs) [16, 25–29]. In these experiments, either phase- or amplitude-modulating SLMs were used to trap or anti-trap atoms, confining fermionic or bosonic atomic clouds to a tailored region of space. Fundamental quantum effects such as superfluidity, quantum turbulence, and the Kibble-Zurek mechanism which describes the formation of topological defects during phase transitions, can be investigated in these experiments. Digital micromirror devices (DMDs), a type of SLM that modulates the light's amplitude in a binary fashion, are commonplace in modern cold-atom experiments since they can be imaged directly onto the plane of the atoms and have a higher switching speed compared to liquid-crystal SLMs (see Section 1.3). Almost arbitrary control over the shape and density of Bose-Einstein condensates (BECs) was demonstrated using repulsive optical

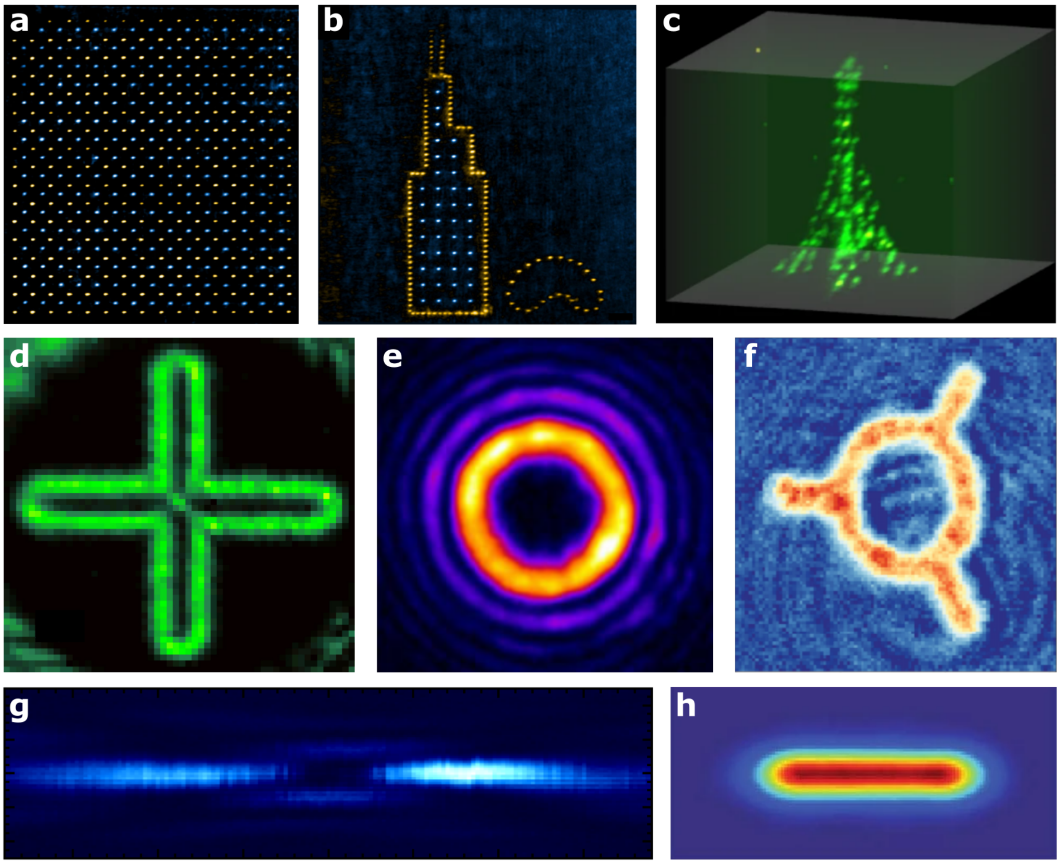


Figure 1.1: Various light potentials for cold-atom experiments. The figures are adapted from previous work. (a-b) Holographically generated 2D tweezer arrays trapping Caesium (yellow) and Rubidium (blue) atoms [19]. (c) 3D tweezers trapping atoms in the shape of the Eiffel Tower [20]. (d) Holographic potential for an atomtronic Y-junction [21]. (e) Ring trap generated with an AOD [22]. (f) DMD-generated potential for an atomtronic Aharonov-Bohm interferometer [23]. (g) Blue-detuned bottle beam trap [24]. (h) Holographically shaped top-hat potential for Rydberg addressing [10].

potentials generated by a DMD in a direct imaging setup through a high-NA microscope objective [25]. Using a DMD provides rapid and programmable reconfigurability of the atomic density without the need for modifying the optical setup. Eliminating the need for expensive custom optical elements is another benefit of this approach [25]. Imaging the DMD directly reduces the computational cost compared to Fourier imaging, however, aberrations in the optical system cannot be compensated this way (see Section 2.1). In spite of the DMD's binary nature (see Section 1.3.1), greyscales can be achieved in a direct imaging setup using binary error diffusion or time-averaging via pulse-width modulation [25]. Controlling the shape of quantum gasses will enable atomtronics applications and the investigation of superfluid dynamics [25].

Using a phase-modulating SLM, a single Gaussian beam was split into three shaped beams – one tube beam and two sheet beams [30]. The holograms for each individual shaped beam can be superimposed on the SLM with different phase gradients to spatially separate the beams from each other. This cannot be achieved using a DMD without a severe loss in efficiency. Even when disregarding the poor efficiency, a DMD would reach its limit in terms of power-handling capabilities (see Section 1.3.1) for this specific application with a combined laser power of 700 mW in the laser beams [30]. The shaped laser beams are recombined onto an atomic cloud where each of them acts as a repulsive barrier forming a box potential which confines a Bose gas in three dimensions [30]. The tube-shaped beam was created by displaying a 24π phase winding on the SLM [30], generating a hollow vortex beam with desirable self-focusing and self-healing properties [31]. By generating three shaped laser beams using a single device, this example showcases the experimental versatility and reduction in complexity that can be achieved by using phase-modulating SLMs in cold-atom experiments. This technique to generate a 3D box potential paved the way for a number of experiments that investigated out-of-equilibrium dynamics in a BEC [16, 26–29].

Ultracold Fermi gases were stirred using light potentials generated with a DMD in direct imaging to observe vortex dynamics [32–34]. The chopstick method [35] was employed using a DMD to create multiple vortex pairs, realising a 'programmable vortex collider' [32]. The chopstick method refers to an experimental technique where the atomic gas is first pierced with a repulsive, tightly-focused laser beam which is then spilt up into two beams, moving away from each other at a certain angle, creating a vortex-antivortex pair [35]. Initially, this technique was realised experimentally by steering two individual laser beams using piezo-actuated mirrors which limits the experiment to one vortex-antivortex pair. Using a DMD enabled this experiment to be scaled to multiple vortex pairs. By exploiting the high switching speed and resolution of the DMD, the chopstick method was applied simultaneously at multiple locations in the atomic gas and in different directions, enabling a vortex collider to be realised [32]. The programmable vortex collider is significant as it allows the controlled study of vortex dynamics, their decay, and their interactions, providing valuable insights into quantum turbulence and the behaviour of superfluids [32]. Understanding these dynamics is essential for fundamental research in condensed-matter physics. The researches claim that their work might pave the way to high-performance superconductors when combined with optical disorder patterns [32]. In a different experiment, a DMD was used to create a ring-shaped pattern with an azimuthal intensity gradient, imprinting a varying phase onto a fermionic atomic cloud that introduces a persistent current in the superfluid [33]. Persistent currents are important as they are crucial to understanding the quantum phase coherence of mesoscopic electronic systems and superconductors [33]. Further, persistent currents in superfluids demonstrate the ability to create and control long-lived quantum states which are crucial for coherent atomic sensing devices and for fundamental research in superfluid dynamics [33]. The DMD-generated potential in this experiment was optimised using a camera feedback algorithm producing a smooth intensity gradient that prevents perturbations in the current [33]. Optimising the uniformity of the potential is crucial for this application as any inhomogeneity in the intensity gradient would have an adverse effect to preparing the persistent current in the superfluid [33]. After stirring the atomic gas, the DMD was used to deliberately introduce local, point-like perturbations in the rotating superfluid to observe the nucleation of vortices due to defects [33]. This experiment is another example where an SLM simplified the experimental setup. Here, the DMD performs multiple tasks (stirring and local perturbations) that would otherwise require numerous laser beams and more complicated optics. Similarly, a BEC was trapped in a ring-shaped repulsive potential generated using a DMD [34]. Instead of using the chopstick technique, the DMD was used to generate a number of differently shaped stirring potentials [34] with almost arbitrary control over the shape and the dynamics of the trapping potential. This reduction in experimental complexity enabled the investigation of various different stirring techniques. This study is relevant for understanding two-dimensional turbulence and the formation of vortices in a superfluid [34].

Similar to the bosonic experiments, optimising the uniformity of a Fermi gas was demonstrated by shaping a repulsive optical potential using a liquid-crystal SLM [36]. To produce a uniform, top-hat-shaped atomic cloud, the in situ atomic density was optimised using absorption images [36] which allowed the Kibble-Zurek mechanism in a Fermi superfluid to be investigated [37]. The Kibble-Zurek mechanism describes the formation of topological defects during a phase transition which is relevant for understanding superfluidity in fermionic gasses. Arbitrary spatial control of the repulsive potential is important in this application to smooth out experimental imperfections in the trapping potential. In this experiment, the liquid-crystal SLM was used in a direct imaging setup. By rotating the polarisation of the incident light 45° with respect to the fast axis of the SLM (see Section 2.2), amplitude modulation is achieved. A benefit of this configuration compared to a DMD in direct imaging is the ability to display greyscales on each SLM pixel without the need for binary error diffusion or pulse-width modulation which can cause heating of the atoms.

In a different Fermi gas experiment, a tightly focused laser beam was generated using a DMD imaged in the Fourier plane with a high-NA microscope objective to locally probe atomic flow [38]. Further refinement might enable these probes to be used as building blocks for more intricate atomtronic circuits [38]. The small, sub-micrometre waist of the probe beam is critical to the spatial resolution of this experiment. To ensure that the waist of the tightly-focused beam is diffraction limited, aberrations in the optical system were precisely measured and corrected to within $\sim \lambda/10$ [38]. The aberrations were measured by displaying a series of patterns on the DMD, effectively using the DMD as a Shack-Hartmann wavefront sensor. The measured aberrations were then compensated using the DMD, and the position of the focussed beam was varied dynamically on a nanometre scale to locally measure the conductance of the Fermi gas [38]. The smallest displacement of the probe beam is limited to 93 nm due to the discrete nature of the binary DMD. A phase-modulating SLM can overcome this limitation which might be beneficial for this experiment since the fast switching speed of the DMD is not a critical factor here. To reiterate, the DMD is used to perform multiple tasks (beam steering, measuring, and correcting aberrations) that would otherwise require additional hardware and complicate the experimental setup.

A number of cold-atom experiments were realised using acousto-optic deflectors which, in contrast to liquid-crystal SLMs and DMDs, create time-averaged optical potentials by rapidly scanning a tightly-focussed laser beam in two dimensions. These devices were used to trap BECs [39] and to dynamically stir BECs in toroidal trapping potentials [40] (Fig. 1.1e). Using a phase-modulating SLM in combination with an AOD, an array of ring traps was generated which can be controlled dynamically [22]. Using this technique, multiple atomic clouds were trapped and moved individually along multiple dimensions. In atomtronics, a field which deals with circuits made of ultracold atoms, the realisation of a range of atomtronic circuits was enabled by structured light generated using AODs, DMDs and liquid-crystal SLMs (Fig. 1.1d and Fig. 1.1f) [18]. By designing two reservoirs connected by a channel using a DMD trapping a BEC, an acoustic Helmholtz oscillator was realised [41]. A Josephson junction was generated by trapping a BEC in a box potential with multiple repulsive barriers generated using a DMD [11]. Josephson junctions are significant in quantum computing and sensing, enabling qubits, superconducting quantum interference devices (SQUIDs), and precision metrology by exploiting quantum interference and sensitivity to external fields. Here, the DMD is imaged directly onto the atomic cloud using a high-NA microscope. Using the DMD, the width and the position of the barrier can be controlled dynamically. Further, the uniformity of the barrier was optimised using camera feedback [11]. Similarly, a Josephson junction was realised by confining a Fermi gas in a potential separated by a barrier generated using two DMDs in direct imaging [12]. By projecting an engineered potential onto a BEC by directly imaging a DMD, an Aharonov-Bohm circuit was realised, enabling high-precision rotation sensing (Fig. 1.1f) [23].

1.1.2 Optical lattice experiments with ultracold atoms

The ability to shape light on a microscopic scale has enabled new possibilities in optical lattice experiments with ultracold atoms. In these experiments, atoms are trapped in the periodic potential of an optical lattice which is generated by counter-propagating laser beams. They enable studying quantum many-body systems in condensed matter physics, quantum phase transitions, and quantum magnetism. More specifically, exotic quantum states and effects such as high-temperature superconductivity [42], and frustrated magnetism were investigated using ultracold atoms in optical lattices [43]. Optical potentials are either used in addition to the optical lattice or to form the optical lattice itself with arbitrary geometries. A specific application is, for example, projecting an anti-harmonic potential onto the optical lattice, removing the harmonic confinement caused by the Gaussian intensity profile of the lattice beams [15, 44, 45]. These light potentials in lattice experiments are often projected through a high-NA microscope objective to demagnify them to a micrometre scale. Generating micrometre-scale potentials using holographic imaging is challenging since aberrations caused by the high-NA objective must be measured in-situ to compensate for dephasing effects [46]. Light potentials generated using an SLM in a direct imaging setup are more common in optical lattice experiments since they are less sensitive to aberrations compared to holographically generated potentials. In a recent publication, an optical lattice was split into several subsystems to study Fermi-Hubbard ladders by engineering optical barriers using a DMD in a direct imaging setup with light from a superluminescent diode [14]. Additionally, the flatness of the ladder systems was optimised by feeding back on the atomic density using the DMD. Here, the incoherent light generated from the superluminescent diode reduces the speckle noise in the optical potentials that is associated with coherent laser light. In these ladder systems, locally varying energy offsets were added using a DMD-generated light potential to study magnetically mediated hole pairing [47], an underlying mechanism to unconventional superconductivity [48]. Over the past 40 years, the lack of a clear understanding of this mechanism in real materials has been driving both experimental and theoretical research [47]. In another experiment, onsite-energy and tunnelling terms in an optical lattice were programmed by projecting a DMD-generated optical potential on top of the lattice potential to simulate the tight-binding model which is essential in the study of condensed matter systems [49]. Other applications of DMD-generated potentials include making specific lattice sites inaccessible to the atoms by ‘plugging’ them with a repulsive potential [45, 50]. Further, DMDs were used to address atoms in specific lattice sites as demonstrated by a recent publication [44]. The high switching speed of DMDs enables generating dynamic light potentials on atomic timescales. In a recent publication, spot arrays were projected onto an optical lattice using a DMD to dynamically rearrange atoms to specific lattice sites [51]. Moving optical barriers were engineered using a DMD to dynamically compress multiple 1D systems of cold atoms in an optical lattice [52].

Despite the challenges associated with Fourier imaging, there are optical lattice experiments that image DMDs in the Fourier plane. In a recent publication [13], phonon modes were realised in an optical lattice by holographically projecting structured light into an optical cavity using a DMD. This specific experiment required both the phase and the amplitude of the light potentials to be controlled which is not possible when imaging the DMD directly. Using DMD-generated potentials in addition to the lattice potential requires relatively little optical power. However, to replace the optical lattice itself with a DMD-generated potential, higher intensities and optical powers are required, increasing with the number of lattice sites in the system. Compared to liquid-crystal SLMs, DMDs have a lower efficiency and power-damage threshold which poses a limitation in this specific scenario. Here, phase-modulating liquid-crystal SLMs benefit from their comparatively high efficiency and power-handling capability. In contrast to optical lattices generated using counter-propagating beams, arbitrary lattice geometries can be realised using SLM generated optical potentials. The generation of ring-shaped optical lattices was demonstrated by imaging a phase-modulating SLM in the Fourier plane to trap neutral atoms [53]. Further, it was shown that phase-modulating SLMs can improve the imaging of single atoms [54]. By imprinting a helical phase onto fluorescence light from cold atoms in an optical lattice, 3D imaging of single atoms was achieved using a phase-modulating SLM [54]. This is an important step towards realising optical lattice experiments in three dimensions since current systems are typically restricted to two dimensions.

1.1.3 Atom array experiments

In atom array experiments, optical tweezer arrays are generated holographically using LCOS SLMs to trap hundreds to thousands of cold atoms (Fig. 1.1a - c) [10, 17, 19, 20, 24, 55–58]. The tweezer array is typically relayed onto the atomic cloud using a high-NA lens or a microscope objective to achieve the desired demagnification. In these experiments, the light usage efficiency is especially important since the available laser power limits the number of tweezers that can be generated to trap atoms. To maximise the number of traps for a fixed optical power and to achieve uniform loading of the atoms into the tweezers, a high uniformity of these traps is beneficial. In quantum information platforms, increasing the number of tweezers and with it, the number of available physical qubits is the subject of current research which progresses rapidly [10, 55]. Recently, error-corrected logical qubits were realised in an atom-array experiment using 280 physical qubits [59]. The authors claim that their approach can be scaled to over 10000 physical qubits by increasing the laser power used to generate the tweezer array [59]. Error correction and increasing the number of qubits are crucial steps towards building a fault-tolerant quantum computer and to reaching practical quantum advantage – the point at which a quantum computer can solve a real-world problem faster or more accurately than a classical computer. In a recent publication [55], a large tweezer array with 12000 sites was generated, trapping 6100 caesium atoms in it. Other methods that do not require an SLM but instead use optical lattices (>1000 atoms) [60], microlens arrays (~ 1200 atoms) [61], or nanophotonic chips (64 atoms) [62] can generate tweezer arrays trapping a large number of atoms. However, in contrast to programmable SLM generated tweezer arrays, the geometry of these arrays cannot be changed without altering the experimental setup. In a recent publication [58], it was demonstrated that various quantum circuits can be realised by arranging dual-species Rydberg atoms in a specific spatial configuration of optical tweezers generated using a phase-modulating SLM. Changing the phase pattern displayed by the SLM enables the realisation of different circuit designs without requiring modifications to the experimental setup. Red-detuned tweezer arrays can trap atoms in their ground state, however, the tweezers act as a repulsive potential for atoms in the Rydberg state which can lead to atom loss. Rydberg atoms in their excited state were trapped in a repulsive, blue-detuned bottle beam trap generated using an SLM [24] (Fig. 1.1g). More recently, arrays of such holographically generated blue-detuned bottle beam traps were generated, trapping 18 atoms in a highly excited Rydberg state which possess longer lifetimes compared to atoms in the ground state [63]. Here, optimising the trapping potential is crucial to prevent atom loss and to achieve a high atom number in the Rydberg state. In a different Rydberg atom array experiment, weighted-graph optimisation was demonstrated by engineering local light shifts [64]. Local control over the light shifts was achieved using a second, phase-modulating SLM in addition to the SLM used to generate the trapping tweezers [64]. This holographically-generated potential was first optimised using camera feedback before the vacuum chamber and then further refined using single-atom spectroscopy. Using this method, a low RMS error of 2% in the relative weights was achieved [64] which is a critical factor when mapping the desired optimisation problem onto the neutral atom quantum computer. Here, lower errors in the light shifts result in a higher fidelity of the quantum algorithm.

Another application of phase-modulating SLMs in Rydberg atom array experiments is top-hat beam shaping of the addressing beam used to excite atoms to the Rydberg state. These top-hat beams are used to globally address the entire atomic array, propagating in the direction parallel to the plane of the array. As opposed to tweezer arrays which require a small diffraction-limited spot size in the order of one micrometre, these potentials are generated without any high-NA optics as they only require a spot size of several micrometres (see Section 3.8, Equation 3.3). For this application, the uniformity of the top-hat is critical since any spatial variation in the intensity will cause varying Rabi-frequencies across the atom array, resulting in dephasing effects between atoms [65]. A high intensity of the top-hat beam is beneficial in this application which necessitates a high efficiency, making this a challenging optimisation task when the available laser power is limited. In a recent experiment [10], laser light is shaped into a top-hat beam with an RMS error of 0.7% and an efficiency of 38% using an SLM in a Fourier imaging setup (Fig. 1.1h). Top-hat beam shaping is an important technique, enabling quantum information platforms to be scaled to larger atom numbers while maintaining uniform Rabi frequencies. The techniques for generating holographic light potentials presented in this thesis are directly applicable to top-hat beam shaping in Rydberg atom array experiments.

1.2 Atom-light interactions

Since we generate tailored holographic light potentials specifically for cold-atom experiments, we introduce interactions between cold atoms and an optical potential to explain the basic principles of atom trapping using laser light. Microscopic particles, including neutral atoms, can be trapped using optical forces, arising from two mechanisms: the optical dipole force and radiation pressure [66, 67]. As early as 1619, when Kepler described the sun's rays affecting comet tails, the concept of light exerting forces on particles was recognised [68, 69]. Maxwell demonstrated that the momentum flux carried by a beam of light correlates directly with its intensity, exerting a force that propels illuminated objects in the direction of the light's propagation, known as radiation pressure [67]. Using the dipole force to tap neutral atoms was first proposed by Akar'yan in 1962 [66] and later demonstrated by Chu et al. [70].

1.2.1 Lorentz oscillator model

A two-level atom in an optical field made from laser light can be described as a classical oscillator according to Lorentz's model [66]. This model assumes that the electron is elastically bound to the nucleus of the atom via a hypothetical spring. The oscillator with natural frequency, ω_0, is driven by an electric field oscillating with frequency, ω, that interacts with the electron. Due to the accelerated charge, the oscillation is damped as described by Larmor's formula. The oscillating electric field of the laser light can be expressed as [66]

$$\mathbf{E}(\mathbf{r},\, t) = \hat{\mathbf{e}}\, \tilde{E}(\mathbf{r})\ e^{-i\omega t} + c.c. \tag{1.1}$$

Here, $\hat{\mathbf{e}}$ is the unit polarisation vector, and $\tilde{E}(\mathbf{r})$ the spatially varying amplitude of the electric field. To describe the motion of the electron as a driven, damped oscillator, the equation of motion is given by Newton's second law,

$$F_{\text{net}} = m_e a = m_e \frac{d^2\mathbf{r}}{dt^2}, \tag{1.2}$$

where a is the acceleration and m_e is the mass of the electron. The net force consists of the driving force, F_{driving}, the binding force between the electron and the nucleus, F_{spring}, and the damping force, F_{damping}, The driving force, F_{driving}, is given by the Lorentz force, $F_{\text{driving}} = -e\mathbf{E}$, where e is the charge of the electron. The binding force between the electron and the nucleus, F_{spring} can be described by Hooke's law, $F_{\text{spring}} = -k\mathbf{r}$, where $\mathbf{r}$ is the displacement of the electron from its equilibrium position and k is the spring constant. Finally, the damping force is described by Larmor's formula, $F_{\text{damping}} = -m_e\Gamma_\omega \frac{d\mathbf{r}}{dt}$, where $\Gamma_\omega = \frac{e^2\omega^2}{6\pi\epsilon_0 m_e c^3}$ is the classical damping rate [66], so we can write

$$F_{\text{net}} = m_e \frac{d^2\mathbf{r}}{dt^2} = F_{\text{driving}} + F_{\text{spring}} + F_{\text{damping}}, \tag{1.3}$$

$$= -e\tilde{E}(\mathbf{r})\; e^{-i\omega t} - k\mathbf{r} - m_e \Gamma_\omega \frac{d\mathbf{r}}{dt}. \tag{1.4}$$

By introducing the resonance frequency of the oscillator, $\omega_0^2 = k/m_e$, and assuming the solution $\mathbf{r} = \tilde{\mathbf{r}}\, e^{-i\omega t}$, we can solve the equation of motion resulting in

$$\mathbf{r}(\omega) = \frac{-e}{m_e} \frac{1}{\omega_0^2 - \omega^2 - i\omega\Gamma_\omega} \mathbf{E}(\mathbf{r}, \mathbf{t})\,. \tag{1.5}$$

Due to the displacement of the electron, a dipole moment, $\mathbf{p}$, is induced by the electric field [66],

$$\mathbf{p}(\omega) = -e\mathbf{r}(\omega) = \alpha(\omega)\, \mathbf{E}(\mathbf{r}, t)\,. \tag{1.6}$$

It follows that the polarisability of the atom, α, is given by [66]

$$\alpha(\omega) = \frac{e^2}{m_e} \frac{1}{\omega_0^2 - \omega^2 - i\omega\Gamma_\omega}. \tag{1.7}$$

By introducing the on-resonance damping rate $\Gamma \equiv \Gamma_{\omega_0} = (\omega_0/\omega)^2\, \Gamma_\omega$, the polarisability can be expressed as [66]

$$\alpha(\omega) = 6\pi\epsilon_0 c^3 \frac{\Gamma/{\omega_0}^2}{{\omega_0}^2 - \omega^2 - i\left(\omega^3/{\omega_0}^2\right)\Gamma}. \tag{1.8}$$

1.2.2 Dipole force

The interaction potential, U_{dip}, between the induced dipole moment, $\mathbf{p}$, and the electric field, $\mathbf{E}$, is given by [71]

$$U_{\text{dip}} = -\frac{1}{2}\langle \mathbf{p}\mathbf{E}\rangle = -\frac{1}{2}\alpha\langle \mathbf{E}^2\rangle. \tag{1.9}$$

The factor $1/2$ in accounts for the induced nature of the dipole moment as opposed to a permanent one. Introducing the intensity of the light, $I(\mathbf{r}) = \epsilon_0 c\langle \mathbf{E}(\mathbf{r})^2\rangle$ where ϵ_0 is the permittivity of free space, and c the speed of light results in [66]

$$U_{\text{dip}} = -\frac{1}{2\epsilon_0 c} \text{Re}(\alpha)\, I(\mathbf{r})\,. \tag{1.10}$$

The real part of the polarisability, $\mathrm{Re}(\alpha)$, describes the in-phase response of the atom to the electric field, which gives rise to the dispersive nature of the interaction [71]. In the regime of low saturation and near resonance, where the detuning, $\Delta = \omega - \omega_0$, is much smaller compared to the resonance frequency, $|\Delta| \ll \omega_0$, and much larger than the on-resonance damping rate $|\Delta| \gg \Gamma$, it can be shown the real part of the polarisability can be expressed as

$$\mathrm{Re}(\alpha) = -\frac{3\pi\epsilon_0 c^3}{\omega_0^3}\frac{\Gamma}{\Delta}, \tag{1.11}$$

by using the relation $\omega_0 + \omega \approx 2\omega_0$ and by applying the rotating-wave approximation. Substituting Eq. 1.11 into Eq. 1.10 results in the dipole potential [71]

$$U_{\mathrm{dip}}(\mathbf{r}) = \frac{3\pi c^2}{2\omega_0^3}\frac{\Gamma}{\Delta} I(\mathbf{r})\,. \tag{1.12}$$

The spatially varying intensity, $I(\mathbf{r})$, results in a dipole force, $\mathbf{F}_{\mathrm{dip}}$, acting on the atom [71]

$$\mathbf{F}_{\mathrm{dip}} = -\boldsymbol{\nabla} U_{\mathrm{dip}}(\mathbf{r}) = \frac{1}{2\epsilon_0 c}\mathrm{Re}(\alpha)\,\boldsymbol{\nabla} I(\mathbf{r}) = -\frac{3\pi c^2}{2\omega_0^3}\frac{\Gamma}{\Delta}\boldsymbol{\nabla} I(\mathbf{r})\,. \tag{1.13}$$

An important result is that the dipole force is proportional to the gradient of the intensity, $\boldsymbol{\nabla} I$, and inversely proportional to the detuning, Δ. Further, the sign of the dipole force depends on the sign of the detuning, Δ. Consequently, for red-detuned light, the atom is attracted to the intensity maxima and for blue-detuned light, the atom is repelled from high-intensity regions. This classical description provides qualitative insight into the optical forces acting on a single atom, however, a semi-classical description is required to provide accurate quantitative results.

Light shift for a two-level alkali atom

Cold-atom experiments for quantum simulation and computation typically use alkali atoms. The dipole potential of a two-level alkali atom, U_{dip}, in an alternating electric field can be calculated by [71]

$$U_{\mathrm{dip}}(\mathbf{r}) = \frac{3\pi c^2}{2}\left(\frac{1}{3}\frac{\Gamma_{D1}}{\omega_{D1}^3\Delta_{D1}} + \frac{2}{3}\frac{\Gamma_{D2}}{\omega_{D2}^3\Delta_{D2}}\right) I(\mathbf{r})\,, \tag{1.14}$$

where Γ_{D1}, Γ_{D2} are the natural line widths of the D1 and D2 line which have frequencies ω_{D1}, ω_{D2} and detunings $\Delta_{D1} = (\omega^2 - \omega_{D1}^2)/(2\omega)$ and $\Delta_{D2} = (\omega^2 - \omega_{D2}^2)/(2\omega)$.

1.2.3 Scattering force

The power absorbed by the atom from the electric field is given by [66]

$$P_{\text{abs}} = \langle \dot{\mathbf{p}} \mathbf{E} \rangle = -i\omega \langle \mathbf{E}^2 \rangle = \frac{\omega}{\epsilon_0 c} \text{Im}(\alpha)\ I(\mathbf{r})\,. \tag{1.15}$$

Here, the imaginary part of the polarisability, $\text{Im}(\alpha)$, is responsible for the out-of-phase response of the dipole interaction and is given by

$$\text{Im}(\alpha) = \frac{3\pi\epsilon_0 c^3}{2\omega_0^3} \left(\frac{\Gamma}{\Delta}\right)^2 . \tag{1.16}$$

The scattering rate, Γ_{sc}, of the atom follows as [66]

$$\Gamma_{\text{sc}} = \frac{P_{\text{abs}}}{\hbar\omega} = \frac{1}{\hbar\epsilon_0 c} \text{Im}(\alpha)\ I(\mathbf{r}) = \frac{3\pi c^2}{2\hbar\omega_0^3} \left(\frac{\Gamma}{\Delta}\right)^2 I(\mathbf{r})\,. \tag{1.17}$$

Dipole traps are often treated as conservative potentials, however, this is only true if the phase across the optical potential is uniform [67, 72]. Gradients in the phase of the potential cause a force acting on the atom by redirecting the radiation pressure away from the optical axis, resulting in a transverse force component [67]. The spatially-dependent phase of the light field, $\Phi(\mathbf{r})$, imprints a transverse phase pattern, $\varphi(\mathbf{r})$, onto the wavefront of a plane wave travelling in the $\hat{\mathbf{z}}$ direction resulting in [67]

$$\Phi(\mathbf{r}) = k_z(\mathbf{r})\, z + \varphi(\mathbf{r})\,, \tag{1.18}$$

with $\hat{\mathbf{z}} \cdot \boldsymbol{\nabla}\varphi = 0$ and the wave vector, $\mathbf{k}(\mathbf{r}) = k_z(\mathbf{r})\,\hat{\mathbf{z}} + \boldsymbol{\nabla}\varphi$, provided that $k^2 = |\mathbf{k}|^2 = k_z^2 = |\boldsymbol{\nabla}\varphi|^2$ which holds for small angles, $k \gg |\boldsymbol{\nabla}\varphi|$ [67]. This modifies our driving field (Eq. 1.1) as follows

$$\mathbf{E}(\mathbf{r},\, t) = \hat{\mathbf{e}}\, \tilde{E}(\mathbf{r})\ e^{-i\omega t} e^{-i\varphi(\mathbf{r})} + c.c., \tag{1.19}$$

resulting in the scattering force [66, 67, 71]

$$\mathbf{F}_{\mathrm{sc}} = \hbar \left(k_z \hat{\mathbf{z}} + \boldsymbol{\nabla}\varphi\right) \Gamma_{\mathrm{sc}}(\mathbf{r}) = \frac{3\pi c^2}{2\omega_0^3} \left(\frac{\Gamma}{\Delta}\right)^2 \left(k_z \hat{\mathbf{z}} + \boldsymbol{\nabla}\varphi\right) I(\mathbf{r}) \tag{1.20}$$

Since the optical forces due to phase gradients are typically much weaker than the optical dipole force [67], we do not consider them in this thesis. However, there some are experiments with cold atoms utilising forces arising from phase gradients. Phase gradients were used to transfer orbital angular momentum from an optical potential to an atomic cloud in a four-wave mixing process [73]. In a different experiment, Bose-Einstein condensates were stirred using an optical potential with phase gradients to investigate vortex dynamics [74].

1.3 Spatial light modulators

Spatial light modulators (SLMs) enable local programmable control of a laser beam, generating light potentials of arbitrary shape. Two types of SLMs are commonly used in cold-atom experiments – amplitude-modulating digital micromirror devices (DMDs)

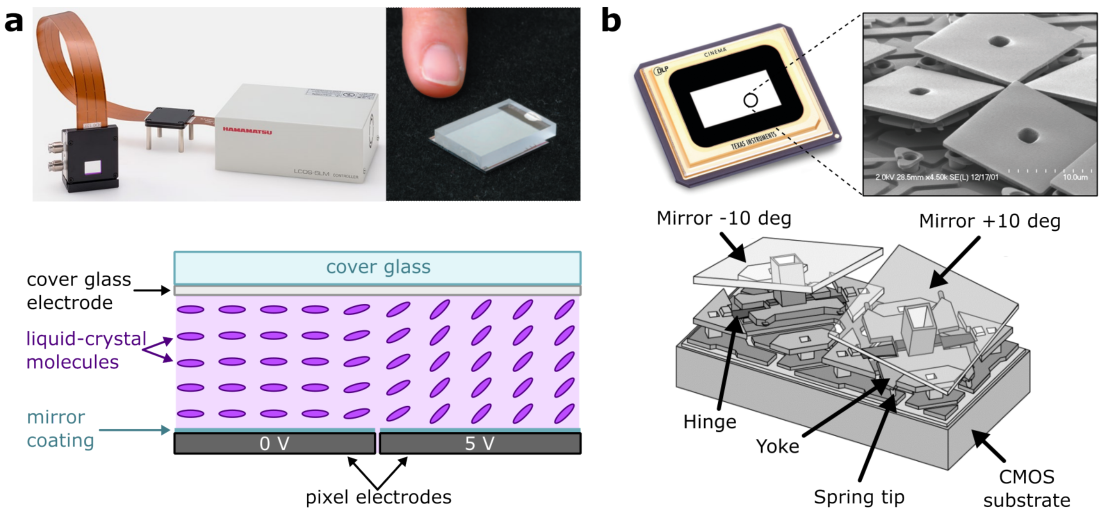


Figure 1.2: Comparison of phase-modulating liquid-crystal spatial light modulators and digital micromirror devices. (a) A liquid crystal on silicon (LCOS) SLM (top left) [75], the LCOS chip (top right) [76] and a diagram showing the cross-section of the LCOS chip (bottom). (b) Chip of a digital micromirror device (top) with a microscope image (inset) revealing the structure of the individual mirrors and a technical drawing of the device (bottom) [77–79].

and phase-modulating liquid-crystal SLMs (Fig. 1.2). Other devices used to shape light include acousto-optic deflectors and micro-electromechanical systems (MEMS) scanning mirrors [80], both of which can create time-averaged intensity patterns by rapidly scanning a tightly focussed laser beam. A phase-modulating SLM based on a programmable photonic crystal cavity array was developed, achieving high refresh rates and high spatial resolution [81]. A bandwidth of 1.5 GHz was achieved by developing a lithium niobate on silicon spatial light modulator which is scalable to millions of pixels [82]. This thesis focuses on a liquid-crystal SLM. However, to better understand the benefits and drawbacks of using a liquid-crystal SLM, we will first discuss its closest competitor in cold-atom physics – the DMD.

1.3.1 Digital mirror devices

A digital micromirror device consists of an array of microscopic mirrors which can be controlled individually to tilt to the 'on' or 'off' position (corresponding to a tilt angle of $+10°$ and $-10°$ for the specific device shown in Fig. 1.2b). This allows binary modulation of the amplitude of the incident light. Laser light incident onto mirrors in the 'on' position is directed into the desired diffraction order while any light incident onto mirrors in the 'off' position is steered away from the optical axis which makes this method of amplitude modulation inherently inefficient. The switching speed of the micromirrors is rapid, allowing for refresh rates of $\sim$ 10 kHz [83] which enables dynamically changing potentials on atomic timescales [52]. DMDs with resolutions of up to 2716×1600 pixels are available [83] with the pixel pitch ranging from $5.4\,\mu$m – $13.6\,\mu$m [83, 84]. Tailored light potentials for cold-atom experiments were realised using a DMD in direct imaging [25, 41, 52], where the efficiency of the light potential is directly proportional to the number of mirrors in the 'on' position and is limited by the diffraction efficiency of the device (typically 30% – 88%) [25, 38, 85]. Alternatively, the DMD can be imaged holographically, in the focal plane of a Fourier lens, which reduces the efficiency to 1% – 2% [46]. As opposed to direct imaging, holographic imaging allows correcting for any aberrations in the optical system in situ, enabling diffraction-limited light potentials [46]. Recently, phase-modulating DMDs were developed which, instead of tilting each mirror, control the displacement of the individual mirrors on a nanometre scale to modulate the phase of the light. These devices combine the fast switching speed of MEMS devices and the high efficiency associated with phase-only light modulation. However, at the time of writing this thesis, this technology is still in development and not readily available to purchase on the market. Using a phase-modulating DMD (Texas Instruments DLP6750Q1EVM), more efficient light potentials were generated holographically compared to amplitude-modulating DMDs [86].

1.3.2 Liquid-crystal SLMs

Liquid crystal on silicon (LCOS) SLMs consist of a liquid crystal material layer on mirror-coated pixel electrodes which are controlled individually by a complementary metal–oxide–semiconductor (CMOS) silicon backplate [87] (Fig. 1.2a). Depending on the configuration of the device, LCOS SLMs can change the polarisation or the phase of the incident light. Due to the birefringent properties of the liquid crystal material, changing the orientation of the liquid crystal molecules by applying an electric field changes the effective refractive index of the material which retards the phase of the incident light. Phase-modulating LCOS SLMs, as used in this thesis, benefit from the electrically controlled birefringence effect. This effect improves the phase modulation depth, however, it also results in a long response time of the liquid crystal molecule, limiting the refresh rate of the device especially for thick liquid crystal layers [87] to $\sim 200\,\mathrm{ms}$ for the specific device we are using (Hamamatsu X13138-07). LCOS SLMs with the twisted nematic configuration and the vertically aligned nematic configuration rely on changing the polarisation state of the light. By using a polarisation analyser after the liquid crystal layer, the amplitude of the light can be controlled. The twisted nematic and vertically aligned nematic configurations are commonly used in liquid crystal displays, however, they are unsuitable for phase modulation [87]. Phase-modulating LCOS SLMs are sensitive to the polarisation of the incident light. Depending on the polarisation, the SLM either modulates only the phase, only the amplitude, or both the phase and amplitude of the incident light (Section 2.2). Liquid crystal on silicon SLMs have an optical power damage threshold of up to $700\,\mathrm{W}$ depending on the device [88] and the wavelength of the incident light, making it suitable for high-power applications. Currently, devices with resolutions of 4160×2464 pixels [89] are available with a pixel pitch between $3.75\,\mu\text{m} - 17\,\mu\text{m}$ [89, 90]. Recently, an LCOS SLM utilising Fabry-Pérot nanocavities was developed, reducing the pixel size to $\sim 1\,\mu\text{m}$ [91]. For cold-atom experiments, a small pixel size is not always desirable since this typically increases crosstalk between neighbouring SLM pixels. Further, smaller SLM pixels reduce the spatial extent of the SLM which increases the achievable diffraction-limited spot size in the Fourier plane. The refresh rate of phase-modulating LCOS SLMs is typically 60 Hz, however, faster models with a refresh rate of $\sim 1.5\,\text{kHz}$ are available [90] enabling dynamic experiments with cold atoms [92]. Ferroelectric LCOS SLMs can modulate the phase or amplitude of the light with switching speeds of $40\,\mu\text{s}$, achieving refresh rates of $\sim 6\,\text{kHz}$ [93]. However, ferroelectric LCOS SLMs modulate light in a binary fashion, limiting their efficiency. Using a phase-modulating LCOS SLM in a holographic setup, calculated efficiencies of $18\% - 64\%$ were achieved [94–97], largely independent of the size of the light potential. After multiplying these calculated efficiencies by the diffraction efficiency of the LCOS SLM ($20\% - 90\%$, depending on the diffraction angle [98]) they are still an order of magnitude higher compared to the DMD efficiencies. The LCOS SLM can be used in a different configuration to modulate both amplitude and phase by using $45°$ polarised light. However, this reduces the efficiency of the device which makes it less interesting for our use case. Due to the high efficiency, we use the LCOS SLM in a phase-only configuration with laser light polarised parallel to the alignment direction of the liquid crystal material.

1.4 Scope of this work

In this thesis, I will discuss computational and experimental methods to generate tailored light potentials holographically using an LCOS SLM in a Fourier imaging setup. The thesis is structured as follows:

Chapter 2 The experimental Fourier imaging setup is discussed as well as the characterisation of simulated and experimentally measured light potentials. Then, computational methods to calculate the SLM phase pattern for a desired light potential are presented which produce simulated light potentials of $< 1\%$ RMS error. Finally, techniques to calibrate the SLM and reduce the error in the experimentally measured light potentials are discussed.

Chapter 3 The optimisation of experimental light potentials using a camera feedback algorithm in a Fourier imaging setup is discussed. Pixel crosstalk, a parasitic effect on the SLM, is identified as a limiting factor for the accuracy of the experimental light potentials. To improve the fidelity of the light potentials, pixel crosstalk is modelled on a sub-pixel scale. At the end of the chapter, downscaling the light potentials to a microscopic scale using a high-NA microscope objective is demonstrated.

Chapter 4 A method to calibrate the SLM via gradient-based optimisation with a greatly reduced runtime compared to the method in Chapter 2 is presented. More intricate pixel crosstalk models which further accelerate the convergence of the camera feedback algorithm and lower the error in the light potentials are discussed.

Chapter 5 The findings of this thesis are summarised, and future work is discussed along with the impact of this work on a broad range of applications.

Chapter 2

Phase-only Fourier holography using an LCOS SLM

Depending on the application, there are various experimental configurations to generate light potentials holographically using a liquid crystal on silicon (LCOS) SLM. Two common configurations to image the SLM are Fourier imaging and lensless imaging. Another experimental consideration is the polarisation state of the incident laser light which determines if the SLM modifies only the phase of the light, the amplitude of the light, or both the phase and the amplitude. Since the SLM is not imaged directly, the complex field amplitude at the SLM which generates the desired light potential in the image plane is unknown. Different algorithms which calculate the complex field amplitude displayed by the SLM to generate the desired light potential in the image plane were developed (Section 2.5). These algorithms typically minimise the error of a simulated light potential and will only generate accurate experimental results if this simulation agrees well with the experiment. To ensure that the simulation matches the experiment, the SLM must be calibrated by carefully characterising the wavefront and intensity profile of the incident light.

In this chapter, we discuss the advantages of imaging the SLM in the Fourier plane (Section 2.1) and why using the SLM in a phase-only configuration is beneficial for cold-atom experiments (Section 2.2). We then discuss our experimental Fourier imaging setup (Section 2.3) and describe how we characterise simulated and experimental holographic potentials (Section 2.4). Finally, we present how we calculate the SLM phase pattern to generate the desired light potential (Section 2.5) and discuss methods to calibrate the SLM and characterise the incident laser light (Section 2.6).

2.1 Fourier imaging and lensless imaging

In Fourier imaging, the image plane is the focal plane of a lens which is placed one focal length away from the SLM (Fig. 2.1a and Fig. 2.2), whereas in lensless imaging, the image plane is located at a certain distance away from the SLM and does not require any focusing optics (Fig. 2.1b). An advantage of Fourier imaging is that, due to the Fourier-transforming properties of a lens, the propagation of light from the SLM plane to the image plane can be modelled using a fast Fourier transform (FFT) which is computationally efficient, especially on a graphics processing unit (GPU).

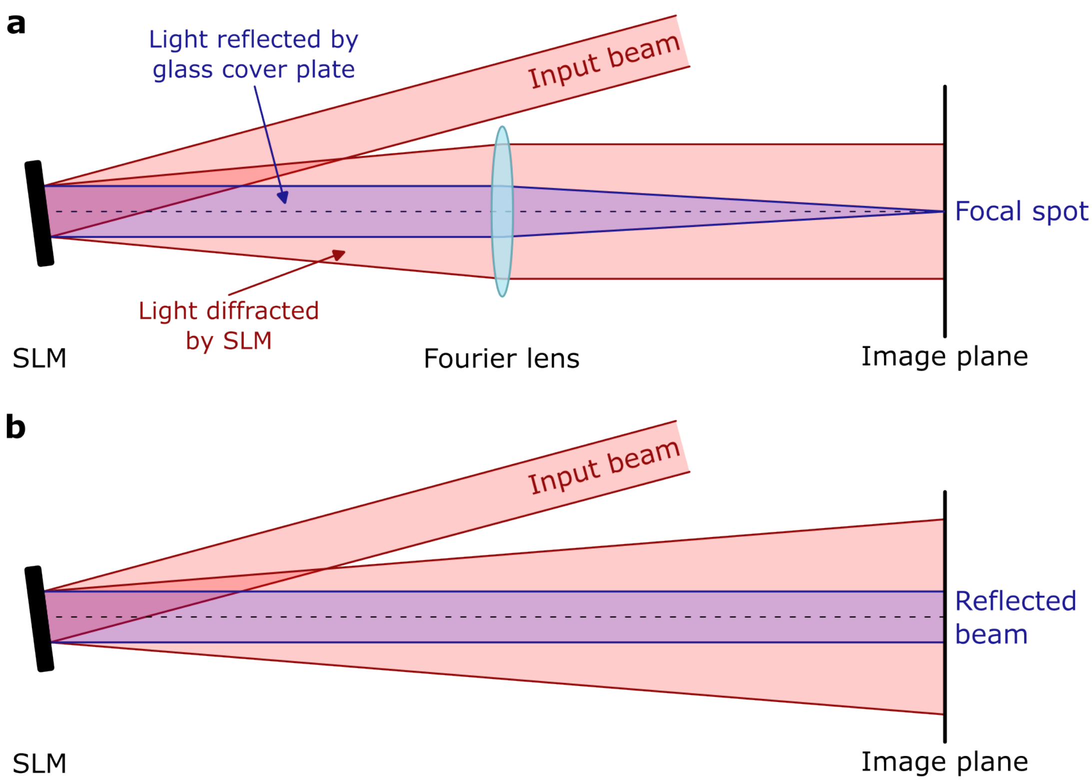


Figure 2.1: Schematic of different holographic imaging setups considering the reflection of the SLM glass cover plate. (a) In the Fourier imaging setup, the input beam is reflected by the glass cover plate and focussed onto a single spot in the image plane by the Fourier lens. (b) In lensless imaging, the reflected beam is not focussed onto a spot and interferes with the light diffracted by the SLM in a much larger area compared to (a).

In contrast, lensless imaging requires the propagation of light in free space from the SLM to the image plane to be simulated using, for example, the angular spectrum method (Appendix A). The angular spectrum method is, despite recent progress [99, 100], significantly more time-consuming to compute than the FFT. Further, a Fourier imaging setup typically produces smaller light potentials of greater spatial resolution compared to lensless imaging due to the larger diffraction angles generated by the Fourier lens. Another experimental consideration is the bright spot forming on the optical axis in the image plane when using Fourier imaging. A fraction of the light in this spot originates from light reflected by the glass cover plate on the SLM ($\sim$ 4% of the incident light) which is focused onto the optical axis in the image plane (Fig. 2.1a). The remaining light collecting on the optical axis is the zeroth diffraction order which forms due to the pixelated nature of the SLM. The light in the zeroth-order spot can be reduced by modelling structure of the SLM pixels [98, 101, 102] and by simulating the effect of multiple internal reflections caused by the glass cover plate [103]. However, since removing the zeroth order spot entirely is difficult, most light potentials are displayed off-axis. Specific patterns such as spot arrays which include the zeroth-order spot as part of the desired optical potential can be displayed on axis. Shifting the light potential away from the optical axis reduces their efficiency, however, this removes any interference caused by the light reflected from the cover glass. This spatial separation of the reflected light and the diffracted light cannot be achieved using lensless imaging, where the zeroth-order light and light reflected by the cover glass are not focused onto a spot but take up a large region in the image plane, interfering with the light potential (Fig. 2.1b). This can lead to unwanted interference fringes if the reflected light and the zeroth-order light are not taken into account during the phase retrieval process [104, 105]. An advantage is that it is not necessary to shift the light potential away from the optical axis as there is no intense spot on the optical axis. This makes lensless imaging more attractive for applications in consumer electronics involving virtual or augmented reality displays and head-up displays for the automotive and aerospace industries. However, for the purpose of cold-atom experiments, Fourier imaging is more commonly used due to the lower computational requirements, the lack of artefacts caused by light reflected by the glass cover plate, and a higher spatial resolution.

2.2 Polarisation of the incident light

By choosing the angle of polarisation of the incident laser beam with respect to the alignment layer of the liquid crystal molecules, the SLM either modulates the phase of the light, the amplitude of the light or both the phase and the amplitude. This happens due to the birefringent properties of the liquid crystal material which possesses an ordinary and extraordinary refractive index. If the incident light is polarised parallel to the alignment direction of the liquid crystal molecules, the SLM only changes the phase of the light. As only little light is absorbed by the SLM in this configuration, the device can handle particularly high optical powers (up to 700W depending on the specific SLM model [88], its cooling system and the wavelength). The phase-only mode of operation is commonly used in applications which require intense light and high efficiency such as laser sintering for metal 3D printing and cold-atom experiments – the subject of this thesis. By polarising the incident light 45° to the liquid crystal alignment direction, the SLM modulates both the amplitude and phase of the light. This configuration is often used to generate vector beams that enable spatial control over the polarisation state in the light potential [106–108]. However, using 45° polarised light is less efficient as the SLM also modulates the amplitude of the light. This decreases the power damage threshold as more light is absorbed by the SLM, causing the device to heat up at a faster rate. Light which is polarised orthogonal to the liquid crystal alignment direction is used less commonly in holographic setups as it is the least light-efficient configuration. Since the amplitude of the light can be modulated directly, using 45° polarised light or orthogonally polarised light allows using the SLM in a direct-imaging setup [36]. In this thesis, we use the SLM in a phase-only configuration with parallel polarised light due to its high efficiency.

2.3 Experimental Fourier imaging setup

For the reasons discussed in the previous two sections, we use a Fourier-imaging setup and laser light polarised parallel to the liquid crystal alignment direction for phase-only modulation (Fig. 2.2). Laser light at wavelength $\lambda = 852\,\mathrm{nm}$ from a single-mode fibre is collimated by a triplet lens (Melles Griot 06 GLC 001) with a specified

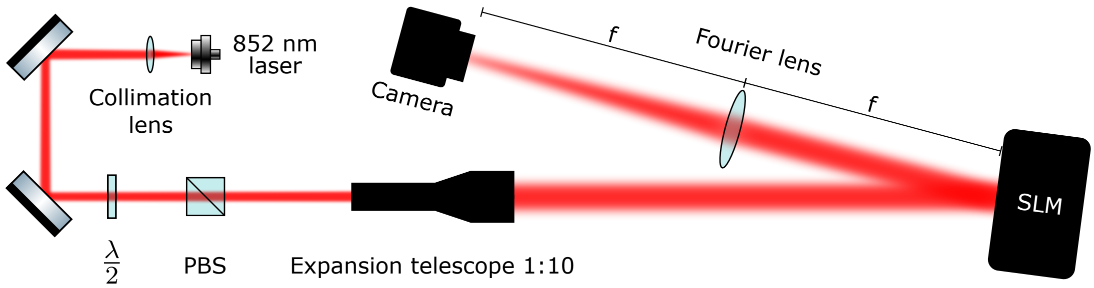


Figure 2.2: Schematic of the experimental setup.

wavefront distortion of $< \frac{\lambda}{4}$ and is polarised along the horizontal plane by a polarising beam splitter (Fig. 2.2). The beam is expanded by a telescope (Thorlabs GBE10-B) to a diameter of 9.4 mm at the SLM (Hamamatsu X13138-07, pixel pitch 12.5 μm, 1272×1024 pixels) which reflects the beam at an angle of $\sim 10°$. An achromatic doublet lens (Thorlabs ACT508-250-B) focuses the light onto the CMOS camera (Matrix Vision mvBlueFOX3-1012dG, 3.75 μm pixel pitch, 1280×960 pixels). The diameter (2") and focal length (250 mm) of the Fourier lens were chosen to avoid clipping the incident laser beam while, at the same time, capturing as much of the light diffracted by the SLM as possible.

Alignment

Since the holographically generated light potentials are sensitive to aberrations in the wavefront of the incident laser beam, the beam was carefully aligned and collimated as follows. After the collimation lens, a Shack-Hartman wavefront sensor (Thorlabs WFS150-5C) was used to minimise the curvature in the wavefront of the beam. As the diameter of the expanded beam is too large to fit on the Shack-Hartmann wavefront sensor, a shear-plate collimation tester (Melles Griot 09 SPM 003) was used to adjust the collimation of the beam after the expansion telescope. The manufacturer of the SLM provided a phase map to compensate for the curvature of the SLM's surface. During alignment, this phase map was displayed on the SLM to ensure that the focal spot is located in the Fourier plane of the lens, provided that the incident beam is perfectly collimated. To align the Fourier lens, the phase of a lens with focal length $2f$, in addition to the corrective phase map, was displayed on the SLM without the physical Fourier lens in the setup. The position of the resulting focal spot on the camera was measured which is placed $2f$ away from the SLM. Then, only the corrective phase map was displayed on the SLM and the Fourier lens was placed back in the setup. The transverse position of the lens was adjusted until the position of the focal spot on the camera coincided with the position of the previously measured spot without the lens in the setup. The position of the camera along the optical axis was adjusted using a micrometre translation stage to minimise the size of the focal spot on the camera. It is important to adjust the distance between the lens and the camera to one focal length within a fraction of the depth of focus since only then the propagation of light from the SLM to the camera can be described accurately by a Fourier transform. The depth of focus, DOF, is given by [109]

$$\mathrm{DOF} = \frac{8\lambda f^2}{\pi D^2} = \frac{8 \cdot 852\,\mathrm{nm} \cdot (250\,\mathrm{mm})^2}{\pi\,(9.4\,\mathrm{mm})^2} \approx 1.53\,\mathrm{mm}, \tag{2.1}$$

where λ is the wavelength and D is the beam diameter of the collimated beam at the Fourier lens with focal length, f. The distance between the SLM and the Fourier lens is less critical as any error can easily be compensated for by multiplying the electric field in the SLM plane by a quadratic phase term [110]. To ensure that the camera sensor is parallel to the image plane, it was carefully aligned as follows. The microlens array on the CMOS camera sensor acts like a diffraction grating which can be used to minimise the tilt of the camera with respect to the optical axis. The orientation of the camera was adjusted so that the zeroth-order beam diffracted by the camera sensor was aligned with the incident laser beam.

2.4 Characterisation of light potentials

To generate a light potential holographically, the phase pattern displayed by the SLM is typically calculated by an iterative algorithm which relies on optimising a simulated light potential. The simulated light potential in the computational image plane will likely deviate from the experimental potential captured by the camera due to a discrepancy between the simulation and the experiment. To characterise both the light potentials predicted by the simulation and the experimental light potentials captured by the camera, we work with two different coordinate systems in the image plane

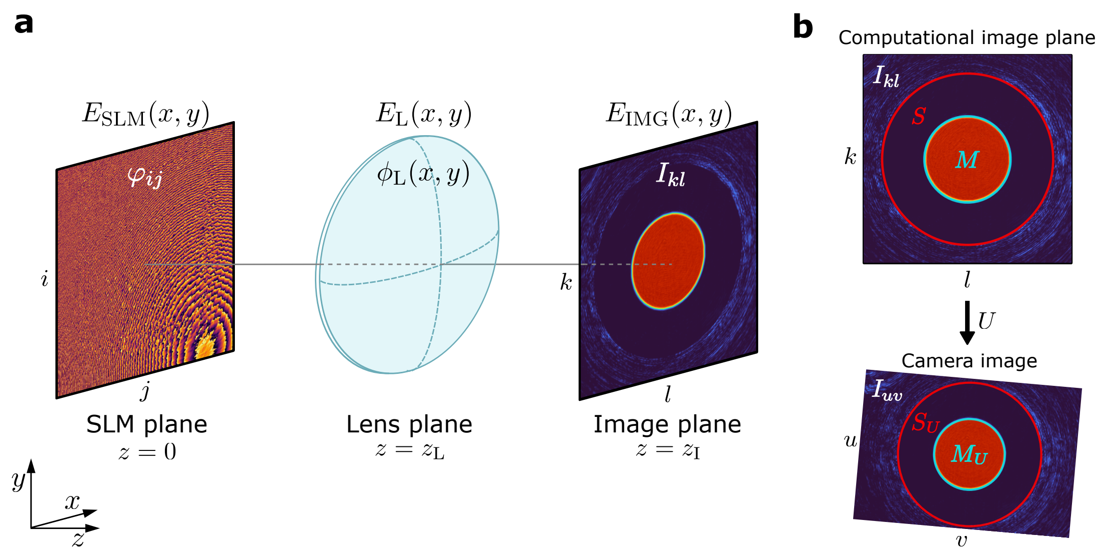


Figure 2.3: (a) Fourier imaging setup with the displayed SLM phase, φ_{ij}, in the SLM plane, shaping a light potential, I_{kl}, in the image plane which is the Fourier plane of a lens [111]. (b) The light potential in the computational image plane, I_{kl}, is mapped to the camera image, I_{uv}, via the affine transformation, U.

(Fig. 2.3b). This is necessary as the camera image and the simulation have different pixel sizes and might be translated and rotated with respect to each other due to experimental uncertainties. The coordinate system of the simulation in the computational image plane is described by row and column indices k and l, whereas the camera coordinates are described by indices u and v. An affine transformation, U, maps the camera coordinates to the coordinates of the computational image plane (Section 3.1).

To characterise the quality of our light potentials, we use the normalised root-mean-squared (RMS) error as this error metric is commonly used in the literature. We define the predicted and measured RMS error, ε_{P} and ε_{M}, respectively,

$$\varepsilon_{\mathrm{P}} = \sqrt{\frac{1}{N_M}\sum_{k,l\in M}\frac{\left(\hat{T}_{kl}-\hat{I}_{kl}\right)^2}{\hat{T}_{kl}^2}} \quad \text{and} \quad \varepsilon_{\mathrm{M}} = \sqrt{\frac{1}{N_U}\sum_{u,v\in M_U}\frac{\left(\hat{T}_{uv}-\hat{I}_{uv}\right)^2}{\hat{T}_{uv}^2}}. \tag{2.2}$$

The predicted RMS error, ε_{P}, measures the difference between the target potential, $\hat{T}_{kl}$, and the simulated light potential, $\hat{I}_{kl}$. The error is calculated in a measure region, M, which we define as the region in the computational image plane where the target intensity is larger than 50% of the maximum target intensity [112], containing N_M number of pixels (Fig. 2.3b). This is necessary since the value of ε can become very large in dark regions with small values of $\hat{T}$. We normalise both the target light potential $\hat{T}_{kl} = T_{kl} / \sum_{k,l \in M} T_{kl}$ and the simulated light potential $\hat{I}_{kl} = I_{kl} / \sum_{k,l \in M} I_{kl}$. The light potential captured by the camera, I_{uv}, is characterised by the measured RMS error, ε_{M}. To calculate ε_{M}, we first have to transform the target light potential and the measure region to camera coordinates (Fig. 2.3b), resulting in $\hat{T}_{uv}$, the normalised, transformed target light potential, and M_U, the transformed measure region containing N_U number of pixels. Other commonly used metrics are the peak signal-to-noise ratio (PSNR) and the mean structural similarity index measure (MSSIM) [113]. Both metrics are suitable to characterise dark regions in the light potentials which is why they are measured in the entire signal region, S, in the computational image plane and the transformed signal region, S_U, in the camera image, containing N_S and N_{S_U} number of pixels, respectively (Fig. 2.3b and Section 2.5). The PSNR of the camera image is defined as

$$\mathrm{PSNR}_{\mathrm{M}} = 20 \log_{10} \left(\frac{\hat{T}_{uv}^{\mathrm{MAX}}}{\sqrt{\mathrm{MSE}}} \right) \quad \text{with} \quad \mathrm{MSE}_{\mathrm{M}} = \frac{1}{N_{S_U}} \sum_{u,v \in S_U} \left(\hat{T}_{uv} - \hat{I}_{uv} \right)^2, \tag{2.3}$$

where $\hat{T}_{uv}^{\mathrm{MAX}}$ is the maximum value in $\hat{T}_{uv} \in S_U$ [114] and MSE is the mean-squared error in S_U. Similarly, for the PSNR of the simulated light potential, $\mathrm{PSNR}_{\mathrm{P}}$. The structural similarity index measure (SSIM) of the camera image is defined as

$$\mathrm{SSIM}_{\mathrm{M}}(a,b) = \frac{(2\mu_a\mu_b + c_1)\,(2\sigma_{ab} + c_2)}{\left(\mu_a^2 + \mu_b^2 + c_1\right)\left(\sigma_a^2 + \sigma_b^2 + c_2\right)}, \tag{2.4}$$

with $c_1 = \left(k_1 \hat{T}_{uv}^{\mathrm{MAX}}\right)^2$ and $c_2 = \left(k_2 \hat{T}_{uv}^{\mathrm{MAX}}\right)^2$, where $k_1 = 0.01$, $k_2 = 0.03$ are default values. The SSIM is evaluated in sliding windows, a and b, of the two images $\hat{T}_{uv}$ and $\hat{I}_{uv}$ at the same location. We use a sliding window of 7×7 pixels size as in previous studies [6]. The average intensities in each window are μ_a and μ_b, and the standard deviations are σ_a, σ_b and σ_{ab}. The MSSIM of the camera image is the SSIM averaged over all sliding windows given by

$$\mathrm{MSSIM}_{\mathrm{M}} = \frac{1}{N_{ab}} \sum_{a,b \in S_U} \mathrm{SSIM}_{\mathrm{M}}(a,b)\,, \tag{2.5}$$

with the number of sliding windows in S_U, N_{ab}. The MSSIM of the simulated light potential, $\mathrm{MSSIM_P}$, is calculated similarly. Even though RMS error and PSNR are common metrics to measure image quality, they do not represent the perceived image quality well. The MSSIM better represents the perceived image quality of the light potential on a scale from 0 to 1 [115] and is used more often in work related to holographic displays, where the perception of the human eye is important. Although less relevant for cold-atom experiments, the MSSIM is introduced here to compare our results with state-of-the-art work in the field of holographic displays as the RMS error is typically not provided. The predicted efficiency, η_P, of the light potential is given by the ratio of the power in the signal region, S, (indicated by the red rectangle in Fig. 2.3a), to the total power in the image plane [96]. We define the experimental efficiency of the light potential, η_M, as the ratio of optical power, P_S, in the transformed signal region, S_U, to the measured power of the beam before the expansion telescope, P_in, (Section 3.6)

$$\eta_\mathrm{P} = \frac{\sum_{k,l\in S} I_{kl}}{\sum_{k,l} I_{kl}} \qquad \text{and} \qquad \eta_\mathrm{M} = \frac{P_S}{P_\mathrm{in}}. \tag{2.6}$$

2.5 Phase retrieval

To generate the desired light potential in the image plane with an intensity pattern $I_\mathrm{IMG}(x,y) = |E_\mathrm{IMG}(x,y)|^2$, the phase pattern displayed by the SLM, $\varphi(x,y)$, must be found, given the constant field at the SLM, $A_\mathrm{SLM}(x,y)\exp\left[i\varphi_\mathrm{C}(x,y)\right]$. This task is commonly known as the phase retrieval problem. Various algorithms such as the mixed-region amplitude-freedom (MRAF) algorithm [96], the offset-MRAF algorithm [94], a conjugate gradient (CG) approach [95], and neural networks such as HoloNet [6] were developed to solve this purely computational problem. They produce simulated light potentials of $< 1\%$ RMS error, however, creating light potentials experimentally with this degree of accuracy is difficult due to a mismatch between the simulated and the measured light potentials caused by uncertainties in the optical setup. These effects include a distorted wavefront at the SLM, the curved surface of the SLM itself, crosstalk between neighbouring SLM pixels, aberrations caused by the Fourier lens and other alignment imperfections. Camera feedback algorithms were developed which reduce experimental imperfections to generate more accurate light potentials [10, 92, 96, 112, 116] (Chapter 3). In a recent study, experimentally accurate light potentials were generated by directly optimising the camera image using stochastic gradient descent (SGD) [6]. It was shown that the Fourier transform used to propagate the light field from the SLM to the camera can be replaced by a more sophisticated method to simulate the propagation of light which can result in more accurate experimental light potentials [94]. In our Fourier imaging setup, the electric field in the SLM plane, $E_{\mathrm{SLM}}(x, y)$, is related to the electric field in the image plane, $E_{\mathrm{IMG}}(x, y)$, via the Fourier transform (details Fig. 2.3a). The electric field in the SLM plane at $z = 0$, $E(x, y, 0) \equiv E_{\mathrm{SLM}}(x, y)$, is calculated using the amplitude of the incident laser beam, $A_{\mathrm{SLM}}(x, y)$, and the phase at the SLM (Fig. 2.3)

$$E_{\mathrm{SLM}}(x, y) = A_{\mathrm{SLM}}(x, y) \exp\left\{i\left[\varphi_{\mathrm{C}}(x, y) + \varphi(x, y)\right]\right\}. \tag{2.7}$$

The phase at the SLM is the sum of the pattern displayed by the SLM, $\varphi(x, y)$, and a constant phase, $\varphi_{\mathrm{C}}(x, y)$, which varies spatially across the SLM but does not change with the displayed phase pattern. This constant phase is caused by the curvature of the SLM's surface and distortions in the wavefront of the incident laser beam. In the image plane at $z = 2f$, the electric field, $E(x, y, 2f) \equiv E_{\mathrm{IMG}}(x, y)$, consists of the amplitude, $A_{\mathrm{IMG}}(x, y)$, and the phase, $\phi(x, y)$, of the light potential

$$E_{\mathrm{IMG}}(x, y) = A_{\mathrm{IMG}}(x, y) \exp\left[i\phi(x, y)\right]. \tag{2.8}$$

Under the paraxial approximation and the far-field approximation, the electric field in the image plane is related to the electric field in the SLM plane via the Fourier transform [110], $\mathcal{F}$,

$$\begin{aligned} E_{\mathrm{IMG}}(\kappa_x, \kappa_y) &= \frac{1}{i\lambda f} \iint_{-\infty}^{\infty} E_{\mathrm{SLM}}\left(x', y'\right) \exp\left[-2\pi i\left(\kappa_x x' + \kappa_y y'\right)\right] dx'\, dy' \\ &\equiv \mathcal{F}\left\{E_{\mathrm{SLM}}\left(x, y\right)\right\}, \end{aligned} \tag{2.9}$$

with spatial frequencies in the image plane, $\kappa_x = x/\lambda f$ and $\kappa_y = y/\lambda f$.

We implement the Fourier transform numerically by calculating its discrete version using the fast Fourier transform (FFT). There are more sophisticated methods to simulate the propagation of light in a Fourier-imaging setup. The angular spectrum method, for example, simulates the propagation of light in free space between two parallel planes without assuming a far-field or small diffraction angles. However, this method is about one order of magnitude slower compared to the FFT. Fast phase-retrieval algorithms are desirable since camera feedback algorithms typically run a phase retrieval algorithm several times to optimise a measured potential, further increasing the computation time. Additionally, slow thermal drifts can cause the light potential to move on the camera on a micrometre scale between feedback iterations. Without accounting for this positional drift, the feedback algorithm can stagnate early, resulting in an inhomogeneous potential. Since the angular spectrum method can simulate the propagation of light from the SLM plane to the Fourier lens and from the Fourier lens to the image plane, aberrations introduced by the Fourier lens and by the SLM can be modelled separately (Appendix A).

2.5.1 Iterative Fourier transform algorithms

Iterative Fourier transform algorithms (IFTAs) are a set of algorithms which solve the phase retrieval problem and rely on computationally propagating the electric field back and forth between the SLM plane and the image plane using the FFT and the inverse FFT. Even though the error of the light potential in the image plane is not minimised directly, applying amplitude or phase constraints to the electric field in each plane causes the algorithm to converge to the desired solution. The original and most simple version of an IFTA, the Gerchberg-Saxton (GS) algorithm [117], can find $\varphi(x, y)$ to produce spot patterns of 98% uniformity [118]. For spot arrays used to trap cold atoms, a modified version of the GS algorithm is still state-of-the-art due to the fast convergence and the ability to achieve high uniformity with high efficiency. However, for arbitrary and smooth light potentials, required e.g., for quantum simulation experiments with ultracold atoms in an optical lattice, the GS algorithm does not converge well. Modified versions of the original GS algorithm such as the mixed-region amplitude-freedom (MRAF) algorithm [96] and the offset-MRAF (OMRAF) algorithm [94], have produced smooth simulated light potentials approaching $\varepsilon_{\mathrm{P}} = 1\%$ root-mean-square (RMS) error, and predicted efficiencies around $\eta_{\mathrm{P}} = 24\%$ [94], depending on the target pattern (equation 2.2 and 2.6). The MRAF and OMRAF algorithms' convergence is highly dependent on the initial SLM phase guess (Section 2.5.3). In order to achieve low RMS errors, the parameters of the phase guess must be tuned which requires running the algorithm several times.

2.5.2 Gradient-based optimisation

More recently, gradient-based optimisation algorithms such as conjugate gradient (CG) minimisation were used to generate simulated light potentials with $\varepsilon_\mathrm{P} < 0.1\%$ RMS error and efficiencies of $\eta_\mathrm{P} > 60\%$ [95], outperforming the above-mentioned IFTAs [94, 96]. Note that these are RMS errors and efficiencies of simulated light potentials which differ from the experimentally obtained values (Chapter 3). Optimisation by adaptive moment estimation was used to solve the phase retrieval problem based directly on the camera image [6], known as camera-in-the-loop (CITL) calibration. Following this study, intensity control was achieved in multiple planes by regularising the phase in each plane [7], avoiding the formation of optical vortices.

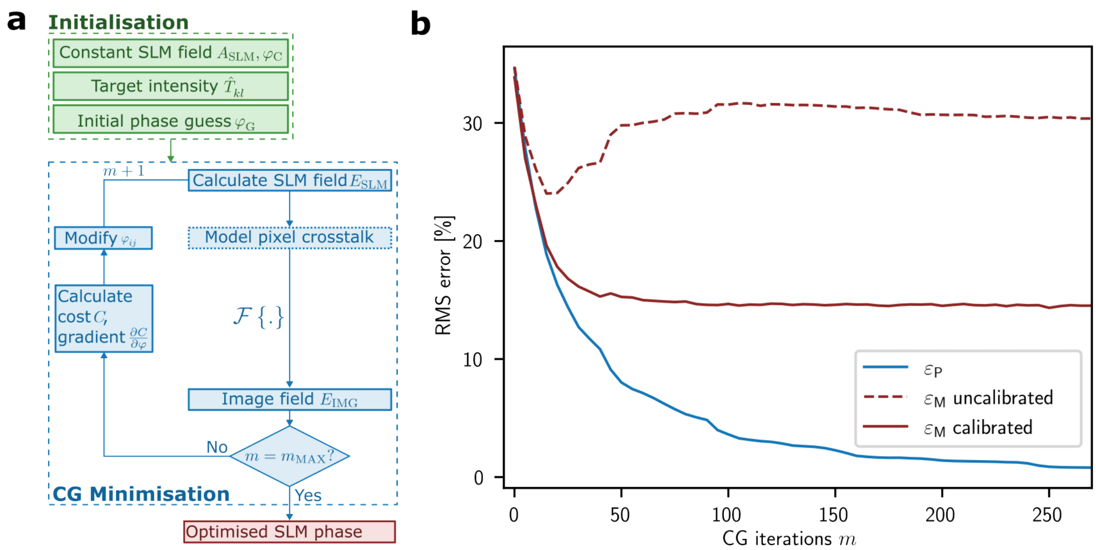


Figure 2.4: CG phase retrieval algorithm. (a) Flow diagram of the CG phase retrieval algorithm [111]. (b) Typical convergence of the CG algorithm for a disc-shaped top-hat potential. The RMS error of the simulated light potential, ε_P, (blue solid line) drops below 1% after $m = 250$. Without calibrating the constant field at the SLM, the error of the camera image, ε_M, (red dashed line) decreases initially, however, after $m = 25$, the error increases to $> 30\%$. After calibrating A_SLM and φ_C, the error of the camera image (red solid line) decreases to $\varepsilon_\mathrm{M} \approx 15\%$.

We use CG minimisation [95] due to its rapid convergence and its flexibility to define a cost function which can be chosen to meet the requirements for a specific application, e.g., the optimisation of intensity, phase, and efficiency in a specific region of interest. The minimisation improves the simulated light potential, I_{kl}, iteratively by modifying the SLM phase, φ_{ij}, based on a cost function C and its gradient $\partial C/\partial\varphi_{ij}$ (blue loop in Fig. 2.4a). We use the mean-squared error between the normalised simulated intensity pattern in the image plane, $\tilde{I}_{kl} = I_{kl}/\sum_{k,l\in S} I_{kl}$, and the normalised target intensity pattern, $\tilde{T}_{kl} = T_{kl}/\sum_{k,l\in S} T_{kl}$, in the signal region, S, as cost function for the optimisation [95],

$$C(\varphi) = s \sum_{k,l\in S} \left(\tilde{T}_{kl} - \tilde{I}_{kl}\right)^2 . \tag{2.10}$$

The sum is evaluated over k and l in the signal region, S, where s is the steepness of the cost function to aid convergence.

We use a nonlinear CG solver [119, 120], which has been implemented on a GPU using PyTorch [121]. PyTorch has automatic differentiation capabilities, a technique commonly used in machine learning which allows us to compute the gradient of the cost function, $\partial C/\partial\varphi$, efficiently without the need to derive an analytic expression. Using $s = 10^{12}$ (equation 4.3), the minimisation typically reaches $\varepsilon_{\mathrm{P}} = 1\%$ within 250 iterations (blue line in Fig. 2.4b), depending on the shape of the desired potential and provided that an initial phase guess which does not lead to optical vortices was used. As the SLM phase pattern is optimised by simulating the diffraction of light, the target intensity pattern, $\tilde{T}_{kl}$, is convolved with a Gaussian with 2 camera pixels width to remove sub-diffraction limited features which hinder convergence.

If desired, an efficiency term could be added to the cost function to optimise for a higher optical power inside the signal region [10]. Currently, we do not require control over the phase, ϕ_{kl}, in the image plane, however, it is possible to simultaneously control the intensity and the phase in the image plane at the expense of efficiency [7, 10, 116]. By saving intermediate SLM phase patterns after every fifth CG iteration and displaying them on the SLM, we investigated the experimental convergence of the error in the camera image, ϵ_{M} (red lines in Fig. 2.4b). The experimental error stagnates much sooner than the error predicted by the simulation, indicating a mismatch between the experiment and the simulation. Measuring the constant amplitude, A_{SLM}, and phase, φ_C, at the SLM significantly reduces the error in the camera image (Fig. 2.4b). Calibrating the constant field at the SLM will be discussed in Section 2.6.

2.5.3 Initial phase guess

Finding a suitable initial SLM phase guess is essential for the convergence of the CG minimisation. We choose an initial phase guess for a given light potential and remove optical vortices from a light potential if necessary (Section 2.5.4). We use a combination of a linear phase and a quadratic phase as an initial phase guess, φ_G, which is common practice in IFTAs and gradient-based phase retrieval algorithms [95, 96],

$$\varphi_G(x, y) = m_x x + m_y y + 4R\left[\gamma x^2 + (1-\gamma) y^2\right], \tag{2.11}$$

The linear terms $m_x x$ and $m_y y$ diffract the light away from the optical axis (Fig. 2.5d) and are typically determined by the shape of the target light potential, T_{kl}. The quadratic term with curvature, R, and aspect ratio, γ, creates an out-of-focus spot in

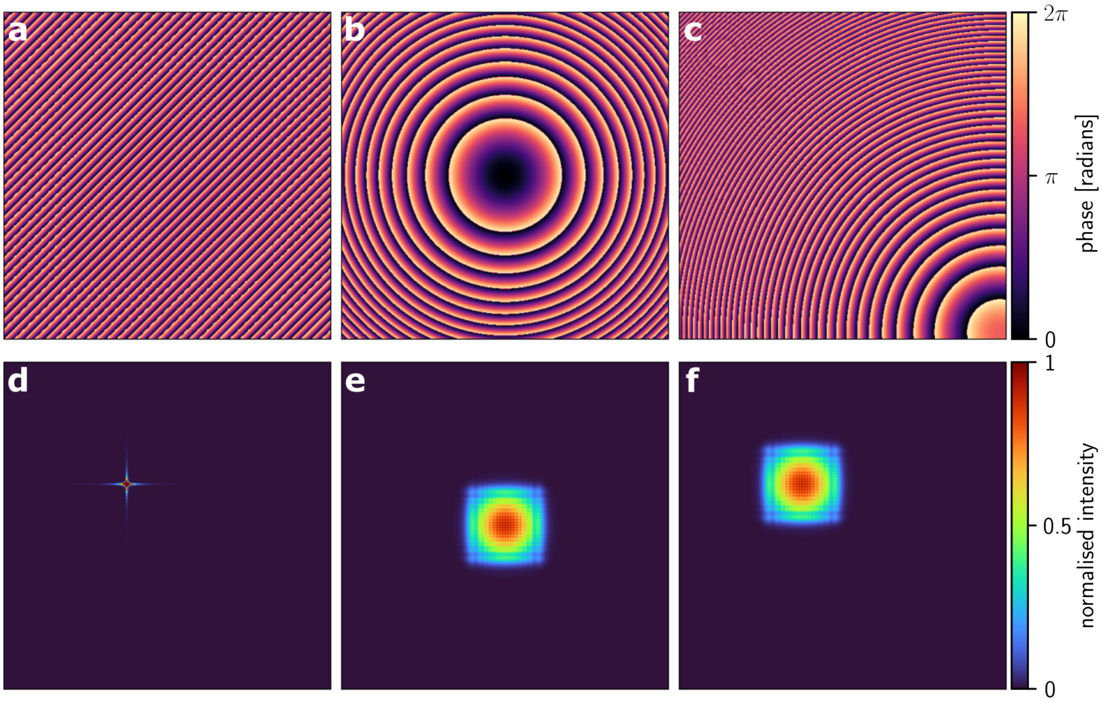


Figure 2.5: Example SLM phase patterns used as an initial guess and the corresponding far field intensity distributions. (a) Linear phase with $m_x = m_y$, (b) quadratic phase and (c) sum of linear and quadratic phase. (d - f) Far field intensity distributions corresponding to (a - c).

the Fourier plane used to control the size of the illuminated area (Fig. 2.5e). Smaller values of R produce more efficient light potentials as more light is focused into the signal region S. By combining the linear and quadratic phase terms, this defocused spot can be shifted away from the optical axis (Fig. 2.5f). Depending on the desired light potential, a different initial phase guess might be more suitable, for example, a conical phase term for ring-shaped light potentials [96]. The initial phase guess must be chosen such that optical vortices cannot form in the signal region S of the image plane [95, 96]. An optical vortex is a phase winding around a singularity at which the phase is not defined [122]. The field amplitude at this point is zero, causing 'holes' in the light potential (Fig. 2.6a). The CG minimisation cannot remove these vortices because a global phase shift is required to annihilate them [123]. By varying R, an initial guess that prevents the formation of optical vortices can be found for 'simple' target potentials. We choose a uniform disc on a dark background as a target potential and detect the number of vortices in the resulting light potential for each value of R (Fig. 2.6e). The vortices in the light potential cause a higher predicted RMS error, ε_{P} (black circles in Fig. 2.6e and blue circles in Fig. 2.6f). Certain values of R do not result in optical vortices, and the lowest RMS error was found for $R = 3.6\,\mathrm{mrad/px}^2$.

2.5.4 Removing optical vortices

This procedure works well for simple patterns such as a disc-shaped flattop, however, for more intricate light potentials, it becomes difficult to find a suitable initial guess by scanning the value of R. Further, we found that using the measured intensity profile of the incident laser beam, $|A_{\mathrm{SLM}}(x,y)|^2$, instead of a perfect Gaussian can introduce vortices even for simple patterns. In our scheme, we detect optical vortices in the light potential and remove them [122, 123]. Initially, the usual CG minimisation is performed until stagnation is reached. We then detect the position of the vortices by identifying the zero crossings of the real and imaginary part of the electric field in the image plane, $E_{\mathrm{IMG}}(x,y)$. To find the charge of the vortices, a line integral around the 3×3 neighbours of these points is evaluated. The sign of the line integral indicates if the vortex is of positive or negative charge. The phase around these vortices, $\phi_{\mathrm{V}}(x,y)$,

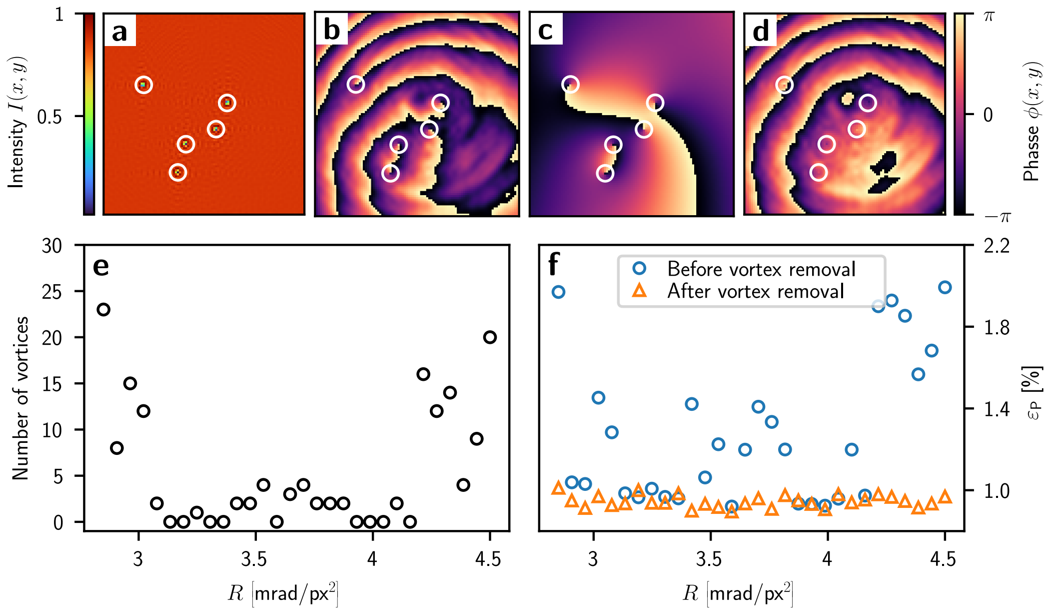


Figure 2.6: Detection and removal of optical vortices in the disc-shaped light potential [111]. (a) Intensity of the light potential, showing the central 100 × 100 pixels; (b) phase, ϕ, of the same potential, (c) phase, ϕ_v of the vortices only, (d) phase, $\phi - \phi_\mathrm{v}$, of the corrected field with vortices removed. (e) Number of vortices detected in the light potential after 100 CG iterations and 10 feedback iterations, using different values for the quadratic curvature, R, in the initial phase guess. (f) Predicted RMS error, ε_P, before vortex removal (blue circles) and after (orange triangles).

is calculated using the relation [123]

$$\phi_\mathrm{V}\left(x, y\right) = \sum_{n=1}^{N} q_n \,\mathrm{Arg}\left[\left(x - x_n\right) + i\left(y - y_n\right)\right], \tag{2.12}$$

where N is the total number of vortices, q_n the charge of the vortex and x_n and y_n its position. The phase, $\phi_\mathrm{V}(x, y)$, (Fig. 2.6c) is then subtracted from the phase of the light potential, $\phi(x, y)$, (Fig. 2.6b) which annihilates the vortices (Fig. 2.6d). The electric field consisting of the corrected phase, $\phi(x, y) - \phi_\mathrm{V}(x, y)$, and the amplitude of the light potential, $A_\mathrm{IMG}(x, y)$, is propagated back to the SLM plane using the inverse Fourier transform. The phase of the resulting electric field is used as a new initial phase guess, $\varphi_\mathrm{G}(x, y)$,

$$\varphi_\mathrm{G}(x, y) = \mathrm{Arg}\left[\mathcal{F}^{-1}\left\{A_\mathrm{IMG}(x, y)\exp\left[i\left(\phi(x, y) - \phi_\mathrm{V}(x, y)\right)\right]\right\}\right]. \tag{2.13}$$

By re-running the CG minimisation using $\varphi_{\mathrm{G}}(x, y)$, a vortex-free light potential can be produced, provided that all vortices in the light potential were detected. In case there are remaining vortices in the light potential, this process can be repeated until all vortices are detected and annihilated. Using our technique, vortex-free simulated light potentials can be generated even from an entirely random initial guess. However, starting with such a random initial phase guess results in less accurate experimental light potentials. Instead of removing optical vortices after running the CG minimisation, including a term in the cost function which constrains the phase can prevent the formation of optical vortices in the light potential [7, 97]. This was done by optimising for a target phase pattern [97] and by minimising the phase gradient in the signal region [7]. Further, optimising for the phase in the light potential makes the convergence of the CG minimisation less sensitive to the initial phase guess parameters [97].

2.6 Calibrating the SLM

Since the phase retrieval algorithms presented in the previous section only optimise a simulated light potential, it is important to reduce any discrepancies between the simulation and the experiment in order to achieve accurate experimental results. We achieve this by calibrating the SLM and by accurately characterising wavefront and the intensity profile of the incident laser beam.

2.6.1 SLM phase response curve

LCOS SLMs are usually controlled by uploading a greyscale image to the device. The greyscale of each pixel corresponds to a certain phase. In our case, the SLM comes pre-calibrated from the factory with a linear greyscale-to-phase response with predetermined slopes depending on the wavelength. This greyscale-to-phase calibration can vary locally on the SLM [6, 124] and even depends on the angle of the diffracted light [124]. We calibrated our device by removing the Fourier lens from our setup and placing a camera in the far field of the SLM (Fig. 2.1) [125]. On the left half of the SLM, we displayed a flat phase and on the right half a linear phase gradient so that the light from each half interferes with each other at the camera, forming a sine-shaped interference pattern. By varying the greyscale of the left half of the SLM

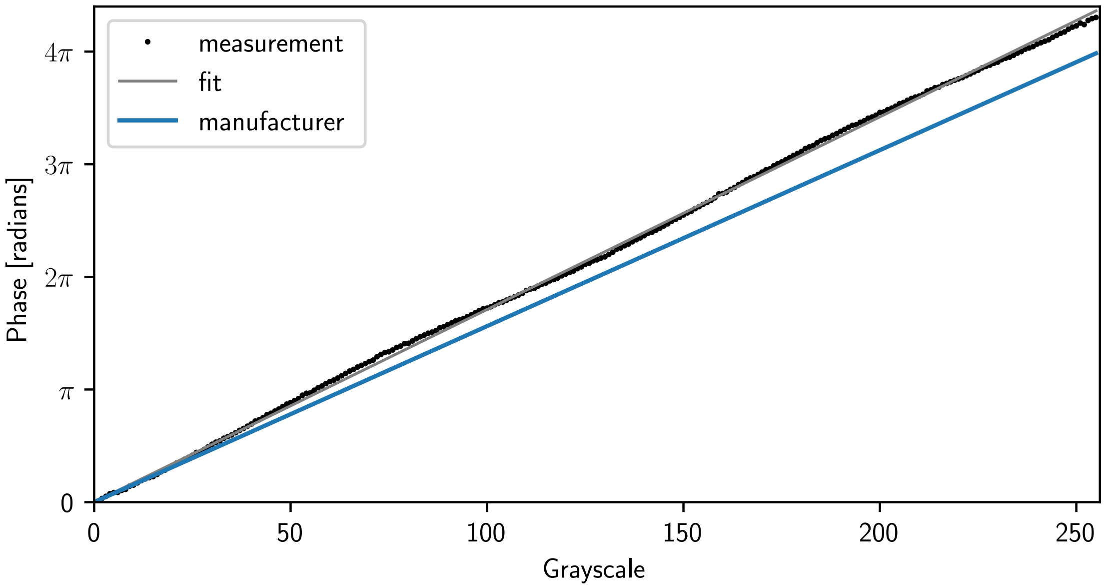


Figure 2.7: Measured phase as a function of the greyscale displayed on the SLM. The measured phase (black dots) differs significantly from the phase provided by the manufacturer (blue line). The residual of the fitted line (grey line) and the measured phase is periodic.

and measuring the spatial phase of the interference pattern on the camera, we obtain a greyscale-to-phase lookup table (Fig. 2.7). The slope of the measured phase response is $\sim 9\%$ larger than the slope provided by the manufacturer. Further, the residual of the measurement and the linear fit appears to be sine-shaped with period 2π. This periodic residual might stem from light reflected by the glass cover plate of the SLM. It was shown that, between the aluminium mirror of the SLM and the glass cover plate, a Fabry-Pérot effect can occur, causing interference in the far field which can change the observed amplitude and phase [103]. The non-linear behaviour in the phase modulation is similar to what we observe in our measurement. We conclude that measuring the phase using the described method is not suitable for our application. Even though our phase measurement in the far field deviates from the factory-calibrated values (blue line in Fig. 2.7), the phase modulation at the SLM might match the calibration. When using the camera feedback algorithm (Chapter 3), we did not see any change in the error of the light potentials when using our calibrated lookup table (black dots in Fig. 2.7). For this reason, we keep using the lookup table provided by the manufacturer instead of our measurement.

Discrete phase levels

The greyscales in the image sent to the SLM are usually discretised into 8 to 10-bit levels. The phase discretisation directly affects the diffraction efficiency of the SLM – the more phase levels are available, the higher the diffraction efficiency will be. The discretisation also impacts the quality of the light potentials. The MRAF algorithm, for example, takes care of this by discretising the SLM phase before propagating it to the image plane. This is more difficult to achieve in gradient-based phase retrieval algorithms. Here, rounding the continuous phase to discrete values creates discontinuities in the cost function, causing the gradient to become zero when the phase change of a pixel is within one phase bin which hinders the convergence of the algorithm. When the phase of an SLM pixel 'jumps' from one phase level to another, the gradient becomes very large. To resolve these issues, the cost was combined linearly with the discretised cost [126]

$$C_{\mathrm{mixed}}\left(\varphi\right) = \alpha C\left(\varphi_{\mathrm{discrete}}\right) + \left(1-\alpha\right) C\left(\varphi\right), \tag{2.14}$$

with the mixing parameter α between 0 and 1. The continuous SLM phase pattern φ is binned to the number of available phase levels on the SLM according to the gray-value-to-phase lookup table, resulting in the discretised phase $\varphi_{\mathrm{discrete}}$. Choosing a large value for $\alpha = 0.9$ is sufficient to aid the convergence significantly.

2.6.2 Laser intensity profile

To generate experimental light potentials that match the simulated ones, it is essential to measure the beam profile, $A_{\mathrm{SLM}}(x, y)$, and constant phase, $\varphi_{\mathrm{C}}(x, y)$, at the SLM plane. We use an interferometric method [46] which displays a sequence of patterns on subsections on the SLM. We measure the constant phase, φ_{C}, across the SLM using a scheme introduced in a previous study [46]. To measure the intensity profile across the SLM, we sample the local intensity by displaying a square pattern on an area of 32×32 pixels containing a linear phase gradient (Fig. 2.8a), while on the remaining area of the SLM, a flat phase is displayed. This phase gradient generates a diffraction spot away from the optical axis, and the light incident onto the remaining area of the SLM collects on the optical axis. We vary the position of the square pattern, d_x and

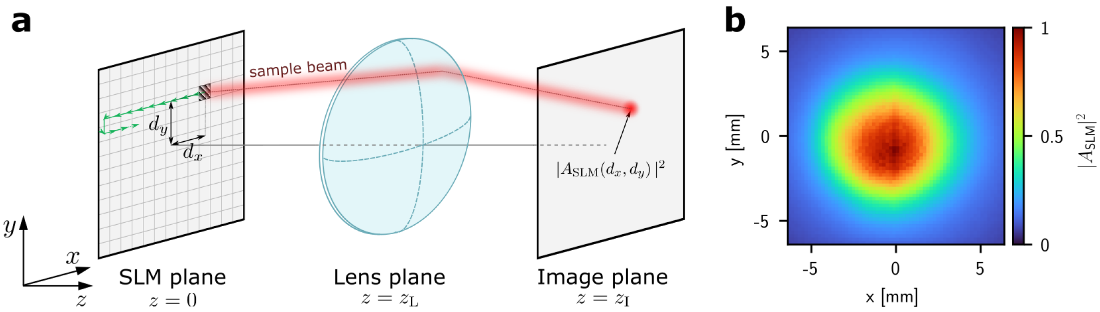


Figure 2.8: (a) Scheme illustrating the measurement of the laser intensity profile by displaying a series of apertures containing a linear gradient on the SLM [114]. (b) Resulting laser intensity profile [111].

d_y, across the entire area of the SLM and measure the intensity of each diffraction spot, $|A_{\mathrm{SLM}}(d_x, d_y)|^2$, on the camera, and as a result, the intensity profile of the laser beam across the SLM is reconstructed (Fig. 2.8b) [114]. The position of the square is moved on an equally spaced grid using 64×64 measurements. The diffraction angle of the linear phase gradient is $\alpha_x = \alpha_y = 0.5°$ both in x- and y-direction. Initially, the square is displayed at the centre of the SLM and a Gaussian is fitted to the resulting diffraction spot on the camera, in a square region of interest of 300 camera pixels. The intensity of each spot is calculated as the sum of all pixel values in the region of interest. In the resulting laser intensity profile (Fig. 2.8b), a faint vertical stripe is visible in the centre of the SLM. We suspect this line appears since the pixels in each half of the SLM are driven separately, which might cause the left half of the SLM to be slightly out of phase with the right half of the SLM (private communication with Hamamatsu [127]).

2.6.3 Laser wavefront

To measure the constant phase, the position of a square sample phase pattern is varied across the entire area of the SLM, similar to our scheme used to measure the intensity. In addition, a reference square pattern is displayed at the centre of the SLM (Fig. 2.9a). The beams originating from the two phase patterns interfere at the camera, causing sine-shaped fringes. The spatial phase, ϕ_{M}, of this interference pattern is detected by fitting a 2D sine pattern to the camera image [46]

$$I_{\mathrm{IMG}}(x, y) = A^2 + B^2 + 2AB\cos\left[k\left(x\sin\gamma_x + y\sin\gamma_y\right) + \phi_{\mathrm{M}}\right], \qquad (2.15)$$

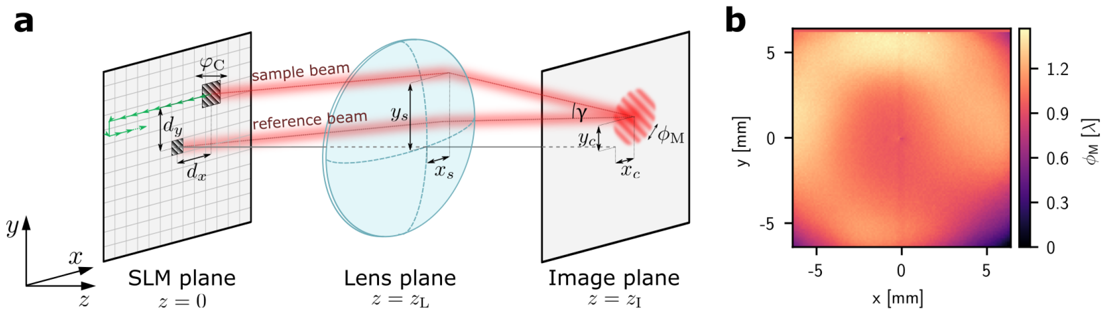


Figure 2.9: (a) Scheme illustrating the measurement of the constant phase at the SLM using an interferometric approach by displaying a sequence of patterns on sub-regions of the SLM (adapted from Zupancic et al. [46]). (b) Resulting measured phase, φ_M, expressed in units of λ.

where $\gamma_x = \arctan(d_x/f)$ and $\gamma_y = \arctan(d_y/f)$. Here, d_x and d_y are the position of the sample pattern with respect to the reference pattern and f is the focal length of the Fourier lens. A and B are the amplitudes of the diffracted beams caused by the reference and the sample square pattern, respectively. Assuming perfect positioning of the lens at $z = f$ and the camera at $z = 2f$ and assuming a thin and parabolic lens, the measured phase, ϕ_M, corresponds to the phase difference between the reference aperture and the sampling aperture $\varphi_C = \phi_M$. The parameters A, B and ϕ_M are fitted while γ_x and γ_y are calculated. Due to the Gaussian shape of the beam incident onto the SLM, the intensity of the light at the SLM drops off towards the edges. This causes the intensity of the sampling beam B to become very small compared to A as the sampling aperture moves away from the centre of the SLM, resulting in a low interference contrast, $2AB$, and a poor fit. To counteract this, the size of the sampling patch is increased as it moves away from the centre of the SLM to keep the power contained in the sampling aperture equal to the power contained in the reference aperture. This increases the contrast of the interference pattern on the camera and improves the uncertainty of the phase measurement at darker regions of the SLM. We use 124×124 different positions of the phase pattern, equally spaced across the SLM with a reference phase pattern of 16×16 SLM pixels, resulting in the measured constant phase φ_C in Fig. 2.9b. Similar to the measured laser intensity profile, a vertical stripe is visible in the centre of the SLM caused by timing specifications of the SLM related to updating the phase pattern [127]. Displaying $-\varphi_C$ on the SLM and re-running the measurement results in a flat phase of $\sim \lambda/40$ RMS error. The wavefront reconstruction will not give accurate results with a severely distorted wavefront which deflects the sampling beam and spatially separates it from the reference beam on the camera, causing a lower contrast of the interference pattern. As the measured intensity and phase have 32×32 and 124×124 data points, they are up-scaled to the native resolution of the SLM (central 1024×1024 pixels) using fourth-order Lanczos interpolation [128]. Before up-scaling, the phase is unwrapped and both measurements are smoothed using a 3×3 uniform filter. It takes approximately 30 minutes to calibrate the intensity pattern and 2 hours to obtain the phase calibration.

Compensating for beam pointing variations

We observed that the interference patterns on our camera move on a micrometre scale, caused by moving air from the laminar flow air-conditioning above our optics table. Further, there is a long-term thermal drift which moves the interference pattern several micrometres throughout the 30 min air-conditioning cycle. The airflow and thermal drift cause pointing variations of the laser beam which manifest in position variations of the interference pattern on the camera. Any movement of the interference pattern will lead to an error in φ_{M}, degrading the accuracy of our measurement. To remove any phase error caused by the movement of the laser beam on the camera, we create a 2D optical lattice on the camera, spatially separated from the sine-shaped interference pattern (Fig. 2.10b). This is done by displaying a square phase pattern containing a linear phase gradient in each corner of the SLM (Fig. 2.10a). The phase gradient of those patterns is larger than the reference and sample pattern to achieve the spatial separation on the camera. We then measure the position of the 2D optical lattice by detecting the phase in x- and y-direction to remove any movement of the laser beam from the measurement of φ_{M}. This decreases the RMS error in the phase measurement from $\sim \lambda/40$ to $\sim \lambda/120$. Using a reference pattern like the optical lattice works well in this calibration, however, this is more difficult to implement during the camera feedback process to generate arbitrary potentials as the light potentials might interfere with the reference pattern (Chapter 3). For this reason, we enclose the entire optical path to shield it from moving air. To check if the enclosure reduced the movement

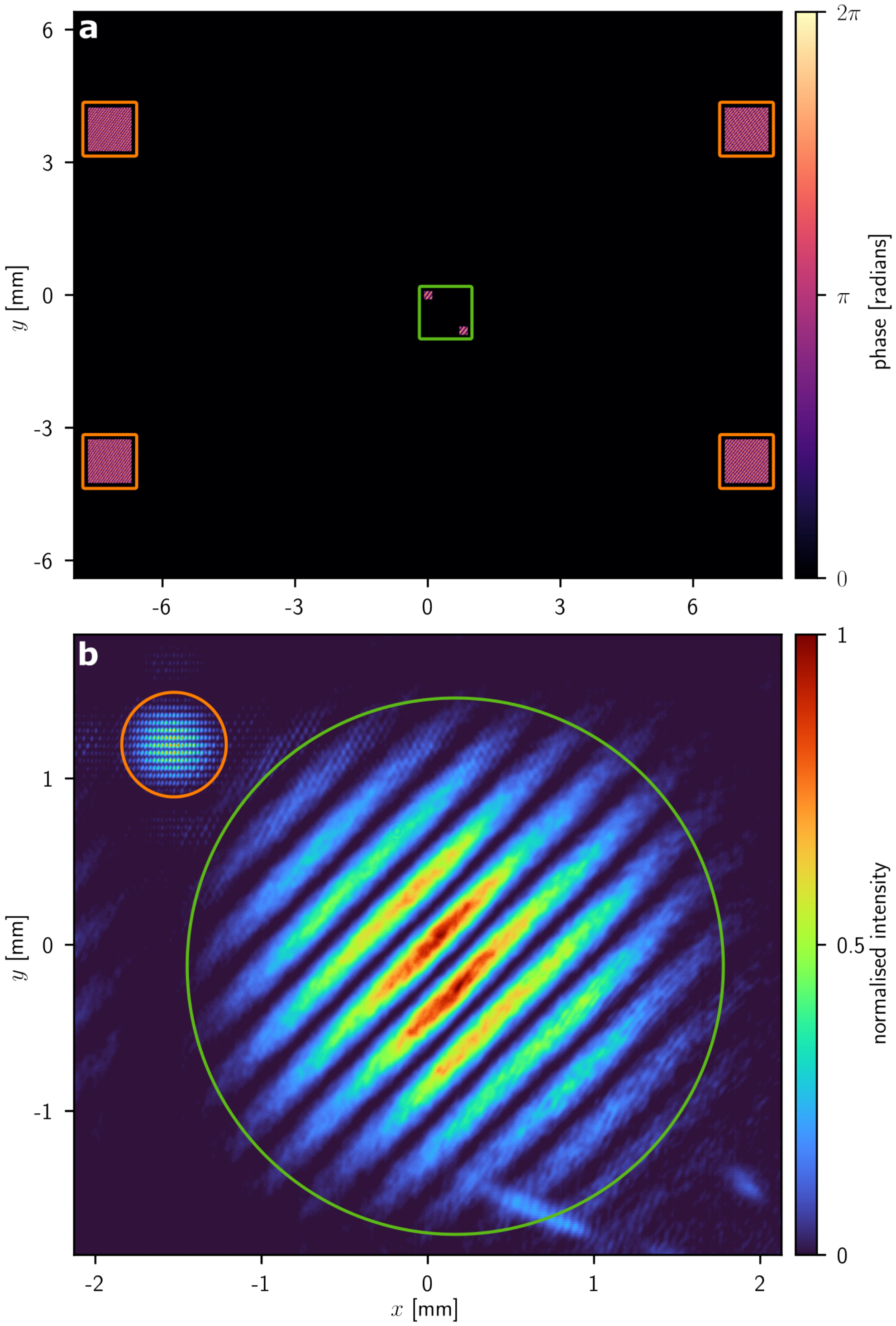


Figure 2.10: Scheme to compensate for the pointing instability of the laser beam during the phase measurement. (a) SLM phase pattern with sample and reference pattern (green square) and the lattice patterns (orange squares). (b) Resulting intensity pattern on the camera with the sine-shaped pattern (green circle) and the 2D optical lattice (orange circle).

of the light potentials on the camera, we run the stabilised wavefront calibration with and without the enclosure and track the position of the optical lattice on the camera throughout the measurement (Fig. 2.11). A periodic drift of the lattice position on the camera with a period of several minutes can be observed in both measurements which might be caused by thermal contraction and expansion of the optics table throughout the air-conditioning cycle. This results in a $\pm 2\,\mu$m drift of the potential on the camera throughout the measurement. Without the enclosure, moving air causes a high-frequency displacement of the potential on the camera (Fig. 2.11a). Shielding the optical path from moving air using the enclosure removes the high-frequency movement of the potential on the camera, however, the thermal drift remains (Fig. 2.11b). We identified the movement of the light potential on the camera as a limiting factor for the accuracy of the light potential after we took the data presented in Chapter 3. The results in Chapter 4 were taken with the enclosure in place.

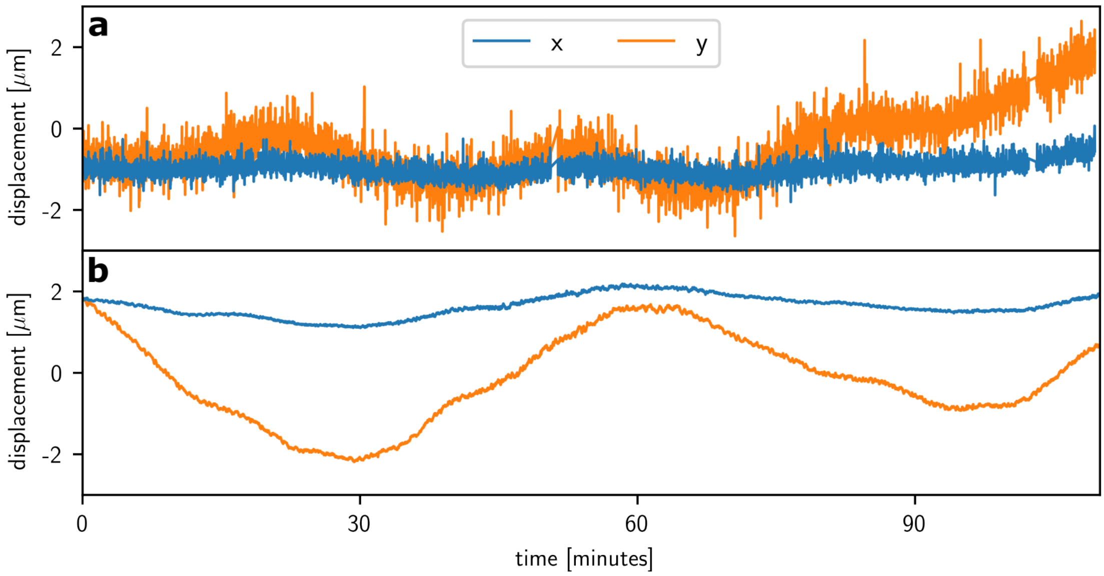


Figure 2.11: Measuring the displacement of the optical lattice on the camera (a) without enclosure and (b) with enclosure.

2.7 Outlook

The phase retrieval algorithms presented in Section 2.5 are all iterative and converge in the order of a few seconds on our GPU. This is around three orders of magnitudes slower than the refresh rate of LCOS SLMs which lies in the range between 60 Hz − 1 kHz, depending on the device. Consequently, to display phase patterns sequentially at the refresh rate of the SLM, it is necessary to pre-calculate them using iterative phase retrieval methods. To accelerate the calculations, machine-learning approaches were developed to solve the phase retrieval problem by training a neural network with pairs of phase patterns and simulated images [6, 7, 56]. Training these networks is time-consuming (multiple hours on a GPU), however, once they are trained, they can then generate an SLM phase pattern corresponding to a target potential in a single step with accuracies approaching the iterative algorithms. This enables real-time holography without the need to pre-calculate phase patterns which is especially useful for applications where the sequence of light potentials is not predetermined. In a recent study [56], a neural network based on the U-net architecture [129] was trained specifically to generate spot arrays. The trained network was able to generate phase patterns in only 160 ms which produced optical tweezers trapping ultracold strontium atoms. Another neural network, HoloNet [6], can generate SLM phase patterns to create arbitrary light potentials at a rate of 40 frames per second. However, the quality of the generated potentials is lower compared to the potentials generated by gradient-based optimisation.

To calculate the initial phase guess, we simply used a combination of a linear and a quadratic phase term, however, more sophisticated methods to calculate the initial phase guess have been developed [130, 131]. These techniques will further accelerate the convergence of the CG minimisation, preventing the formation of optical vortices even for intricate target light potentials. The method we use to measure the intensity profile and the wavefront of the laser beam incident onto the SLM (Section 2.6) takes around 3 hours to complete. This is mostly due to the large number of phase patterns displayed on the SLM. In total, we display $\sim$ 17000 phase patterns to calibrate the laser beam intensity and phase, which requires 200 ms for each pattern. The remaining time is taken up by image acquisition, fitting a 2D sine to each camera image (phase calibration only) and saving the data. Any change in the optical setup caused by, for example, re-aligning the laser beam on the SLM or adjusting its collimation after a thermal drift requires re-running the time-consuming calibration procedure. In Chapter 4, we introduce a faster calibration method based on optimisation by gradient-descent, greatly reducing the runtime from 3 hours to 2 minutes.

Chapter 3

Optimising experimental light potentials using camera feedback

In this chapter, we discuss reducing the error in experimental light potentials using a camera feedback algorithm. To compare simulated light potentials discussed in the previous chapter to camera images of experimental potentials, we discuss determining the coordinate transform between the camera image and the computational image plane in Section 3.1. In Section 3.2, we present a simple camera feedback algorithm which reduces the error in the experimental light potential. We find that pixel crosstalk, a parasitic effect occurring between neighbouring pixels on the SLM, causes artefacts in the light potentials and hinders the convergence of our feedback algorithm. In Section 3.3, we discuss modelling pixel crosstalk and compensating for its effects, reducing the error in the experimentally measured light potentials. Using the above-mentioned techniques, we then generate various light potentials relevant for cold-atom experiments in Section 3.4. Since cold-atom experiments require local control on a microscopic scale, we downscale the light potentials in the Fourier plane using a high-NA microscope objective as discussed in Section 3.8 and compare the microscopic potentials with our previous results.

The phase patterns calculated by the phase retrieval algorithms presented in the previous chapter only generate accurate simulated light potentials. Even after carefully measuring the intensity profile and the wavefront of the incident laser beam, the experimentally measured light potentials can vary significantly from the simulated ones due to experimental effects which are not taken into account in the simulation. To reduce errors in the light potential caused by this mismatch between the simulation and the experiment, various camera feedback algorithms were developed [6, 112, 131]. A simple and effective approach, commonly used in cold-atom experiments, is to modify the target light potential used in the phase retrieval algorithm (Fig. 3.3 and Section 2.5). This empirical approach has produced uniform spot arrays (1.4% standard deviation of the trap depths [57]) and accurate top-hat potentials (0.7% RMS error [10]). Another approach is to reduce the discrepancy between the experiment and the simulation by deploying a more sophisticated model of the experimental setup [6, 7]. This model includes experimental effects that were previously not accounted for by "learning", for example, optical aberrations in the experimental setup from a set of camera images using machine learning techniques [6, 7] (Chapter 4).

3.1 Camera calibration

As already mentioned in Section 2.4, to compare the camera image of the light potential to the simulated potential, we have to find a coordinate transform which maps the camera image to the computational image plane and vice versa. The coordinate transform must account for the different pixel sizes in the camera image and the computational image plane and any translation and or rotation which might occur between the two planes. We use an affine transformation to model the coordinate transform between the two planes which accounts for scaling, translation, rotation in the $xy-$plane, and shear between the two images. A rotation of the camera image around the $x-$ and $y-$axis could be modelled by using a perspective transformation instead of an affine transformation, however, this is not necessary since we adjusted the rotation of the camera to be perpendicular with respect to the optical axis (Section 2.3). The coordinate transform between the computational image plane with indices k and l and the camera image with indices u and v is given by

$$\begin{bmatrix} u \\ v \end{bmatrix} = U \begin{bmatrix} k \\ l \end{bmatrix}, \tag{3.1}$$

where U is the affine transformation matrix (Fig. 2.3).

To calculate the affine transformation matrix, we generate a chequerboard-shaped light potential on the camera and detect the corner points of the chequerboard in the camera image [132] (Fig. 3.1a). We first measure the constant field at the SLM (Section 2.6) and use this measurement to calculate the phase pattern for a chequerboard-shaped light potential using the conjugate gradient algorithm. When displaying the resulting phase pattern on the SLM, the chequerboard on the camera features fringes caused by pixel crosstalk. Specifically, these fringes are caused by 0 to 2π phase jumps in the phase pattern on the SLM, where pixel crosstalk is most noticeable. The fringes degrade the chequerboard pattern and cause the corner detection to yield unreliable results or to fail entirely. To resolve this issue, we add a global phase of 0.1 radians to the SLM phase pattern and re-wrap it to 2π. This slightly shifts the position of the 0 to 2π phase jumps on the SLM and with it, the corresponding fringes on the camera. By repeating this process 10 times and averaging the corresponding camera images, the fringes caused by pixel crosstalk average out and the resulting chequerboard is detected reliably. Using a corner detection algorithm [132], the chequerboard corner points are detected in the averaged camera image. These points are then fitted to the corresponding corner points in the computational image plane (Fig. 3.1b) by optimising the parameters in the affine transformation matrix, U. In previous studies, a spot array [118] and an array of discs [6] were generated on the camera to obtain the transformation matrix. We use a chequerboard pattern since fast and robust detection algorithms already exist [132] and due to their ability to find corner points with sub-pixel accuracy. Errors in the translation between the camera image and the computational image plane (Fig. 3.2), even on a sub-pixel scale, can cause the camera feedback algorithm to stagnate early as it relies on comparing the camera image to the desired target light potential. To show the effect of an error in translation, we define a square light potential of 100 camera pixels width with a blurred edge (Fig. 3.2a). Using an affine transformation, we shift the light potential along the y-axis from 0 to 0.5 camera pixels and calculate the RMS error between the shifted potential and the original potential (Fig. 3.2c). Even a small translation of the potential, below 0.1 camera pixels, causes a significant increase in RMS error. A translation of the light potential on the

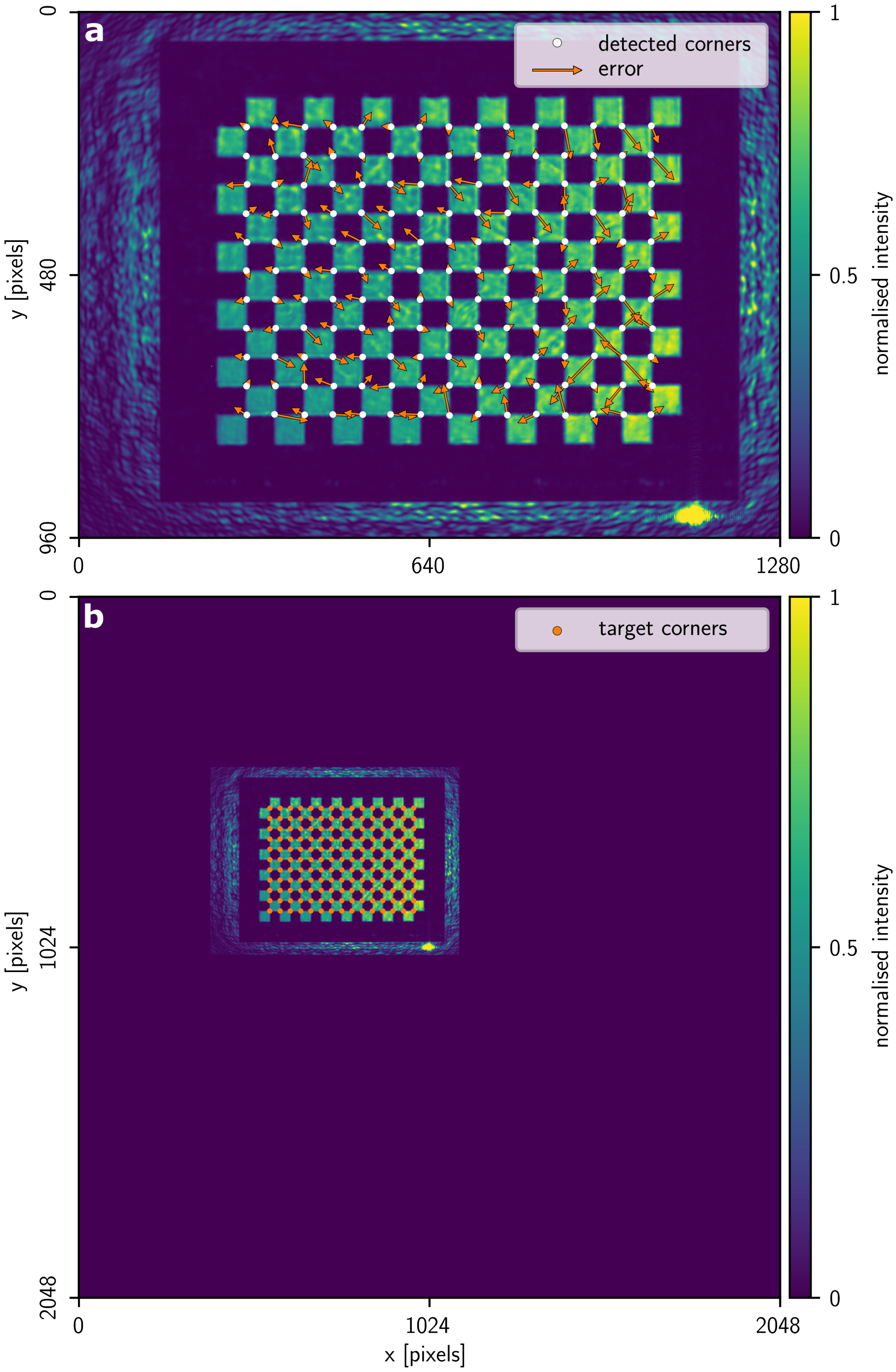


Figure 3.1: Camera calibration to find the affine coordinate transformation matrix. (a) Camera image with the detected corner points (white circles). The orange arrows indicate the error between the transformed corners of the target chequerboard and the detected corners in the camera image. The length of the arrow is 30 times longer than the displacement between the points. (b) Transformed camera image in the computational image plane containing the target corner points (orange circles).

camera can occur in the experiment caused by, for example, vibrations from moving air or thermal drift (Section 2.6.3). Currently, we determine the affine transformation matrix at the start of the camera feedback process, however, the light potential might move on the camera while running the feedback process. To improve the robustness of the feedback algorithm against any displacement of the light potential on the camera, it would be beneficial to determine the affine transformation matrix after taking each camera image. This might be done by generating a reference pattern on the camera which moves with the light potential similar to the approach in Section 2.6.3. Alternatively, after capturing each camera image, the parameters of the affine transformation could be re-optimised to minimise the difference between the target light potential and the measured light potential. However, this might slow down the feedback algorithm.

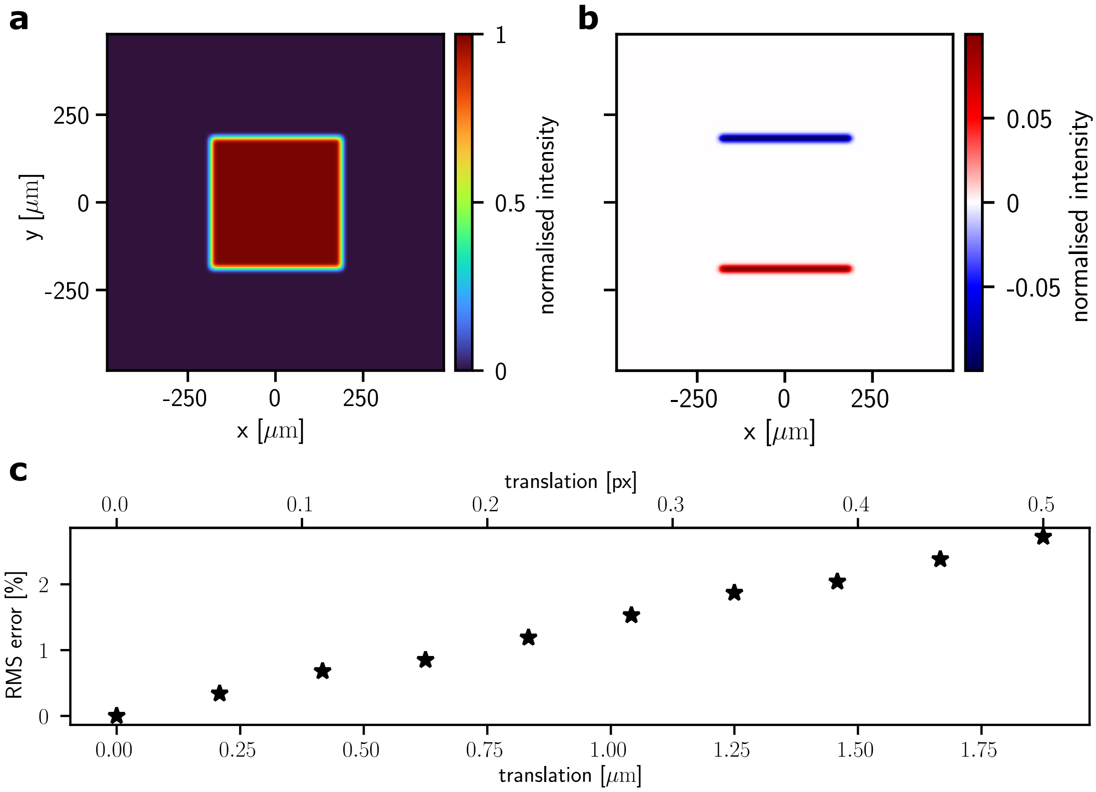


Figure 3.2: Effect of shifting the light potential on the RMS error. (a) The normalised target light potential. (b) The difference between the target light potential and the potential shifted to the bottom by 0.5 camera pixels. (c) RMS error between the shifted light potential and the target light potential as a function of the vertical translation.

3.2 Camera feedback algorithm

Here, we employ an iterative camera feedback algorithm from Bruce et al. [112] (red loop in Fig. 3.3). First, a phase pattern calculated by a phase retrieval algorithm (blue loop in Fig. 3.3) is displayed on the SLM and an image of the experimental light potential is captured by the CMOS camera. In areas of the camera image with too little light, the intensity in the target potential is raised and vice versa [112]. Re-running the phase retrieval algorithm with the modified target potential will then result in a more accurate experimental potential. Initially, at feedback iteration $n = 0$, an SLM phase pattern, $\varphi_{ij}^{(0)}$, with row and column indices i and j is calculated for a given target light potential, $\hat{T}_{kl}^{(0)}$, by running the conjugate gradient minimisation for $m_{\max}$ iterations (blue loop in Fig. 3.3). We then display this pattern on the SLM and take a camera image, I_{uv}, of the light potential. We map the initial target light potential from the coordinate system of the computational image plane, $\hat{T}_{kl}^{(0)}$, to the coordinate system of the camera image, $\hat{T}_{uv}^{(0)}$, using the affine transformation matrix, U (Section 3.1). Then, the camera image, I_{uv}, and the transformed initial target light potential, $T_{uv}^{(0)}$, are normalised [112] and subtracted from each other. This difference $D_{uv} = \hat{T}_{uv}^{(0)} - \hat{I}_{uv}$ is then transformed back to the coordinates of the computational image plane and added to the previous target light potential $\hat{T}_{kl}^{(n-1)}$, resulting in a new target light potential $\hat{T}_{kl}^{(n)} = \hat{T}_{kl}^{(n-1)} + D_{kl}$ for the next feedback iteration, n. We then re-run the conjugate gradient minimisation using the updated target light potential and the previous optimised phase pattern, $\varphi_{ij}^{(n-1)}$, as an initial guess. Before the new target potential is calculated, the difference D_{uv} is blurred with a Gaussian kernel to ensure that there are no features in the new target that are smaller than the diffraction limit (such as camera noise) as the conjugate gradient minimisation cannot produce light potentials containing sub-diffraction-limited features. The feedback algorithm typically converges within $n = 15$ iterations. Initially, the predicted error of the light potential converges well to $\varepsilon_{\mathrm{P}} \sim 1\%$ (dashed line in Fig. 3.4d), however, the experimental error stagnates at $\varepsilon_{\mathrm{M}} \sim 18\%$ (solid line in Fig. 3.4d). After updating the target light potential the first time, ε_{M} converges at a faster rate before stagnating again. The slight increase in ε_{M} towards the end of each feedback iteration might be caused by positional drift of the light potential on the camera on a micrometre scale since we did not enclose our opti-

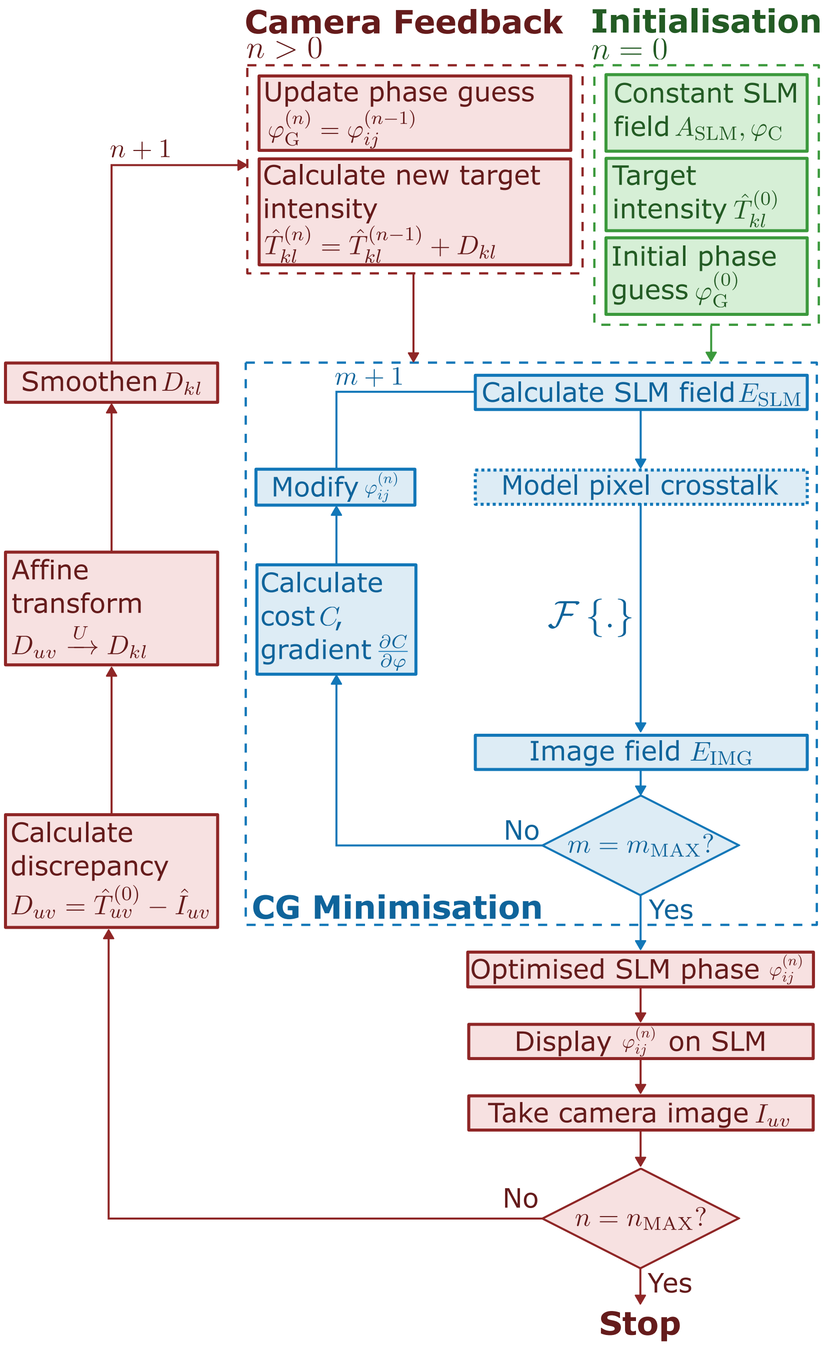


Figure 3.3: Flow diagram visualising the process of generating a light potential [111] which includes the conjugate gradient minimisation [95] (inner, blue loop) nested inside of the camera feedback algorithm [112] (outer, red loop).

cal setup to shield it from airflow (Section 2.6.3). Interestingly, stopping the conjugate gradient algorithm when the smallest value of ε_{M} is reached causes the algorithm to stagnate sooner in the next feedback iteration. Even though the ε_{M} slightly increases towards the end of the conjugate gradient optimisation, running the conjugate gradient algorithm longer aids global convergence. Defining an optimal number of conjugate gradient iterations per feedback iteration is difficult since this number changes as the feedback algorithm progresses and further depends on the target light potential and the initial phase guess. However, we found that more conjugate gradient iterations are required during the first feedback iterations than in later feedback iterations.

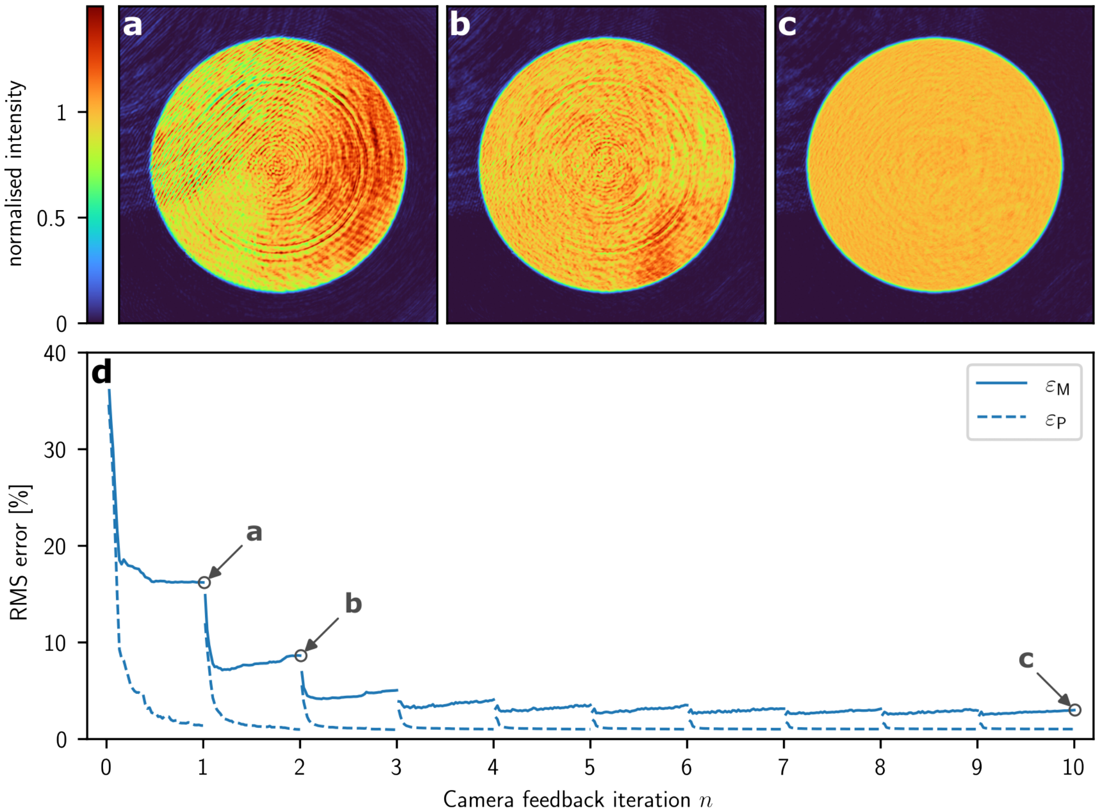


Figure 3.4: Convergence of the camera feedback algorithm. (a-c) Camera images, I_{uv}, of disc-shaped light potentials after feedback iterations $n = 0$, $n = 1$, and $n = 10$. (d) Updating the target light potential after each feedback iteration causes the experimental error, ε_{M}, to converge. At the start of each feedback iteration, the error of the simulated light potential, ε_{P}, increases as a result of changing the target light potential.

3.3 Pixel crosstalk on the SLM

By modelling a single SLM pixel with a single computational pixel, we assume that the phase across the SLM pixel is uniform. However, due to the nature of the liquid crystal material inside the SLM, neighbouring pixels affect each other at their boundary region. This effect is known as pixel crosstalk or fringing field effect [102, 104, 133–136]. Artefacts in holographic light potentials caused by pixel crosstalk were reduced by modelling pixel crosstalk to generate spot arrays [98, 134, 137] and smooth, arbitrary potentials [9, 105, 111].

3.3.1 Modelling pixel crosstalk

We model the effect of pixel crosstalk on our light potentials by up-scaling the SLM phase such that one SLM pixel is represented by 3×3 computational pixels and con-

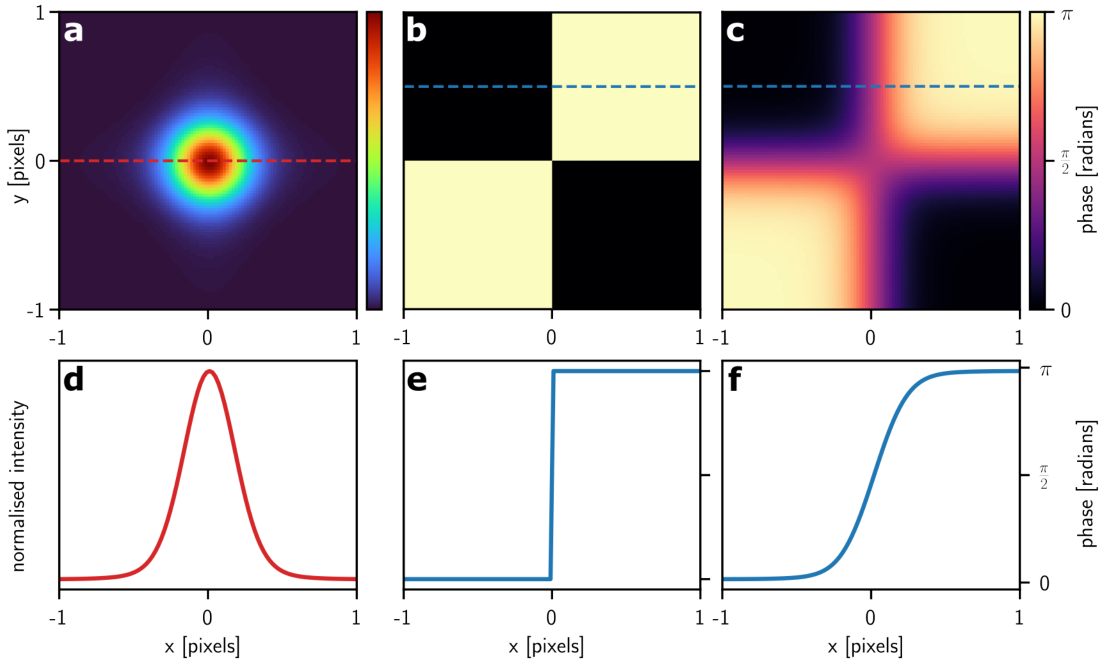


Figure 3.5: Blurring effect of the pixel crosstalk on the SLM phase. (a) Crosstalk kernel (equation 3.2). (b) Phase of 2×2 SLM pixels before applying the crosstalk model. (c) SLM phase after convolving the phase in (b) with the kernel in (a). (d-f) Profiles of the kernel and the phase patterns along the dashed lines indicated in (a-c).

volving it with a kernel, K, [136]

$$K(x,y) = \mathcal{F}^{-1}\left\{\exp\left[-\left(\frac{|\kappa_x|^q + |\kappa_y|^q}{\sigma^q}\right)\right]\right\}, \tag{3.2}$$

of order, q, and width, σ, with the spatial frequencies, κ_x and κ_y. To demonstrate the blurring effect of the pixel crosstalk, we convolved a chequerboard phase pattern of 2×2 SLM pixels with the crosstalk kernel, up-scaling one SLM pixel to 50×50 computational pixels. Simulating the entire SLM array with this degree of up-scaling is not possible due to memory limitations (we run our computations on an Nvidia RTX A5000 graphics card with 24 GB of graphics memory). As an example, we calculated the SLM phase for a spot array target potential using the conjugate gradient minimisation and observed fringes in the camera image (Fig. 3.6b) which do not appear in the simulated light potential (Fig. 3.6a). After up-scaling and convolving the same SLM phase pattern, we propagate the field from the SLM plane to the image plane using the Fourier transform. The resulting simulated light potential (Fig. 3.6c) features fringes similar to those in the camera image, however, with reduced contrast. Since we use Fourier imaging, increasing the spatial resolution in the SLM plane increases the spatial extent of the computational image plane but has no influence on the spatial resolution of the light potential in the computational image plane.

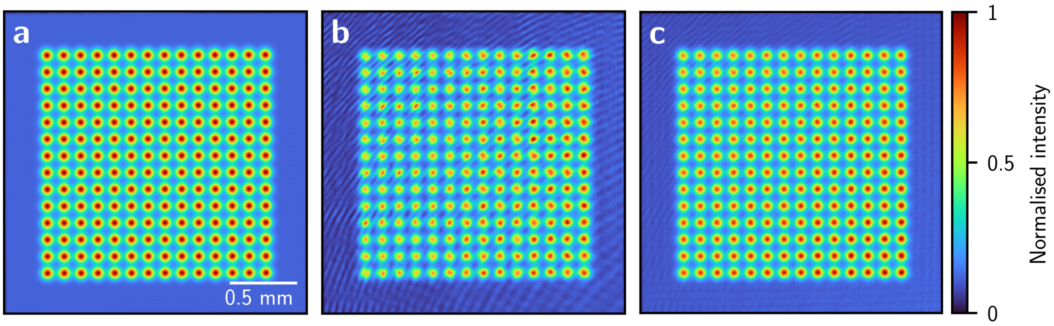


Figure 3.6: Simulated and experimental images illustrating the effect of pixel crosstalk [111]. (a) Simulated light potential for a spot array target light potential. (b) Camera image of the experimental light potential showing fringes and an intensity gradient, with less intense spots in the top left of the image. (c) Simulated light potential after up-scaling and convolving the SLM phase pattern with kernel K. The fringes and the intensity gradient seen in the camera image (b) are reproduced in the simulation, however, with reduced contrast.

3.3.2 Compensating for pixel crosstalk

In the conjugate gradient minimisation, we account for pixel crosstalk by upscaling the displayed phase, $\varphi(x, y)$, and restricting its values to a range between 0 and 2π to ensure that the cost $C(\varphi)$ remains a continuous, differentiable function. We convolve the up-scaled phase with the kernel, K, before propagating the light field to the image plane. The parameters $\sigma = 1.24\,\mathrm{px}^{-1}$ and $q = 1.80$ were found by a 2D optimisation to minimise ε_M after 150 conjugate gradient iterations without camera feedback for a disc-shaped target potential. Using the camera feedback algorithm with the pixel crosstalk model further reduces the RMS error. The final RMS error and the effect of the pixel crosstalk correction depend on the size of a specific target light potential (Fig. 3.7). Upscaling the SLM pixels by a factor of 3 is computationally expensive, however, we accelerate our calculations using a GPU which reduces the runtime of our algorithm to ~ 10 minutes (15 feedback iterations with 100 conjugate gradient iterations each).

To study how the pixel crosstalk model affects our light potentials, we produced disc-shaped light potentials of different diameters, D, between $0.64\,\mathrm{mm}$ and $3.3\,\mathrm{mm}$, with and without accounting for pixel crosstalk (Fig. 3.7). The target light potential was convolved with a Gaussian kernel of 2 pixels width to ensure that the edge of the disc is not sharper than the diffraction limit. For the initial phase guess, the quadratic phase curvature was adjusted proportionally to the disc diameter (Section 2.5.3). This ensures that the predicted efficiency of the differently sized light potentials remains similar ($\eta_\mathrm{P} = 74\% - 87\%$). Without accounting for pixel crosstalk, we achieved the lowest error ($\varepsilon_\mathrm{M} = 1.1\%$) for small discs of $D = 0.85\,\mathrm{mm}$, and less accurate potentials ($\varepsilon_\mathrm{M} = 3.6\%$) for larger discs of $D = 3.2\,\mathrm{mm}$ with measured efficiencies $\eta_\mathrm{M} = 33\% - 40\%$. We found that ε_M is inversely proportional to the measured intensity, I', in the flat part of the disk (Fig. 3.7d). To obtain I', we measure the average intensity in the flat part of the disc using the camera (Section 3.6). Smaller discs are of higher intensity since the same amount of optical power is focused onto a smaller area. We found that the pixel crosstalk causes a ghost image [98, 131] which can interfere with the light potential and cause fringes. By accounting for pixel crosstalk in our model, any interference with the light potential caused by the ghost image is attenuated which lowered the final experimental RMS error by a factor of ~ 0.4 ($D = 2.8\,\mathrm{mm}$). We found that

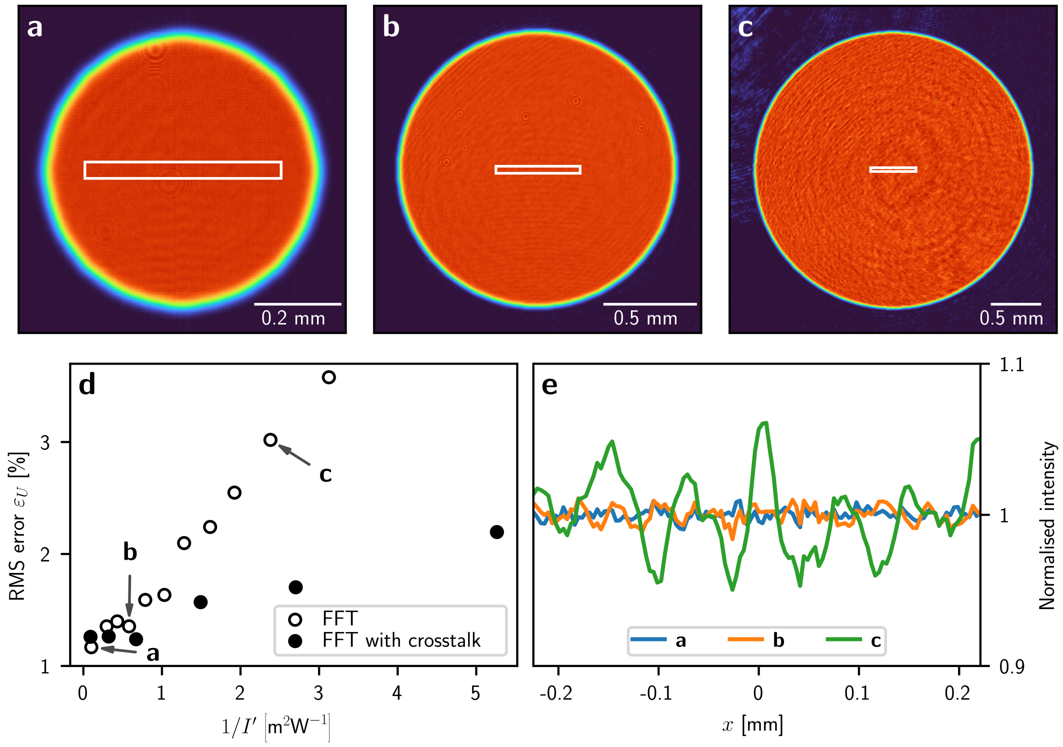


Figure 3.7: Effect of pattern size and pixel crosstalk on the RMS error [111]. (a)-(c) Disc-shaped potentials (diameters $D = 0.6\,\mathrm{mm}$, $D = 1.5\,\mathrm{mm}$ and $D = 2.8\,\mathrm{mm}$), generated using camera feedback without the pixel crosstalk model, and normalised by the average intensity in the flat part of the disc. (d) RMS error of disc-shaped light potentials of different diameters with and without the pixel crosstalk model. (e) Horizontal profiles of the light potentials averaged over 10 rows within the white rectangles in (a)-(c).

accounting for pixel crosstalk has little effect on smaller light potentials, where the overlap between the ghost image and the light potential is smaller (Fig. 3.7d). When taking the pixel crosstalk model into account, the RMS error, ε_M, remains smaller as the ghost image caused by pixel crosstalk is attenuated (Fig. 3.7d), and the measured efficiency decreases from $\eta_\mathrm{M} = 41\%$ ($D = 0.85\,\mathrm{mm}$) to $\eta_\mathrm{M} = 20\%$ ($D = 3.2\,\mathrm{mm}$). We found that the predicted efficiency, η_P, is proportional to the measured efficiency, η_M. The efficiency predicted by the pixel crosstalk model is lower and closer to the measured efficiency as multiple diffraction orders are simulated. We did not see an improvement in ε_M when increasing the resolution of an SLM pixel even further to 5×5 or 7×7 computational pixels. The simple crosstalk model (equation 3.2) only attenuates the fringes seen in the camera image, they are not removed completely. In Chapter 4, we present a more sophisticated pixel crosstalk model, significantly improving on the results presented here.

3.4 Potentials for cold-atom experiments

To characterise our method, we produced various light potentials relevant for cold-atom experiments. We created a ring with a Gaussian profile relevant for atomtronic experiments [40] (Fig. 3.8a), a Gaussian potential with an offset as used to cancel the harmonic confinement in optical lattices [15] (Fig. 3.8b) and a Gaussian spot array with a non-zero background for tweezer arrays [10] (Fig. 3.8c). We also generated a potential resembling an 'atomtronic' OR gate as used by previous studies [94, 96, 131] (Fig. 3.8d). For the Gaussian potential and the spot array, we achieved the best experimental results by using an initial phase guess according to equation 2.11 (Section 2.5.3). For the ring-shaped potential (Fig. 3.8a) and the OR gate (Fig. 3.8d), an initial phase guess resulting in vortex-free potentials could not be found in the same way. For these patterns, the remaining optical vortices were removed [122] (Section 2.5.4).

For all patterns, we used 15 feedback iterations with 100 conjugate gradient iterations each, accounting for pixel crosstalk during the optimisation. The experimental RMS error of all four patterns varies between $\varepsilon_{\mathrm{M}} = 1.4\% - 1.6\%$, with measured efficiencies between $\eta_{\mathrm{M}} = 15\% - 31\%$ (Table 3.1). The remaining imperfections are most visible in the cuts of holograms with flat regions. The peak signal-to-noise ratios (PSNR) [114] measured in the transformed signal region, S_U, of the light potentials in Fig. 3.8a-d are $45.7\,\mathrm{dB}$, $40.7\,\mathrm{dB}$, $43.8\,\mathrm{dB}$, and $39.9\,\mathrm{dB}$, respectively.

Compared to previous studies (Table 3.1), we can generate experimental light potentials of low RMS error and higher efficiencies. Using the conjugate gradient method, a small line-shaped potential ($105\,\mu$m length) of 0.7% RMS error was generated [10] by optimising the intensity and phase in the image plane as well as the efficiency ($\varepsilon_{\mathrm{P}} = 38\%$). Using such a phase constraint, it becomes increasingly difficult to generate light potentials which are accurate and efficient for larger patterns. A larger line-shaped potential ($\sim 400\,\mu$m length) was generated in a different study [97, 116] by constraining the phase, however, with a much lower efficiency of 8.3%. If the phase of the target light potential is constrained, more accurate light potentials are typically

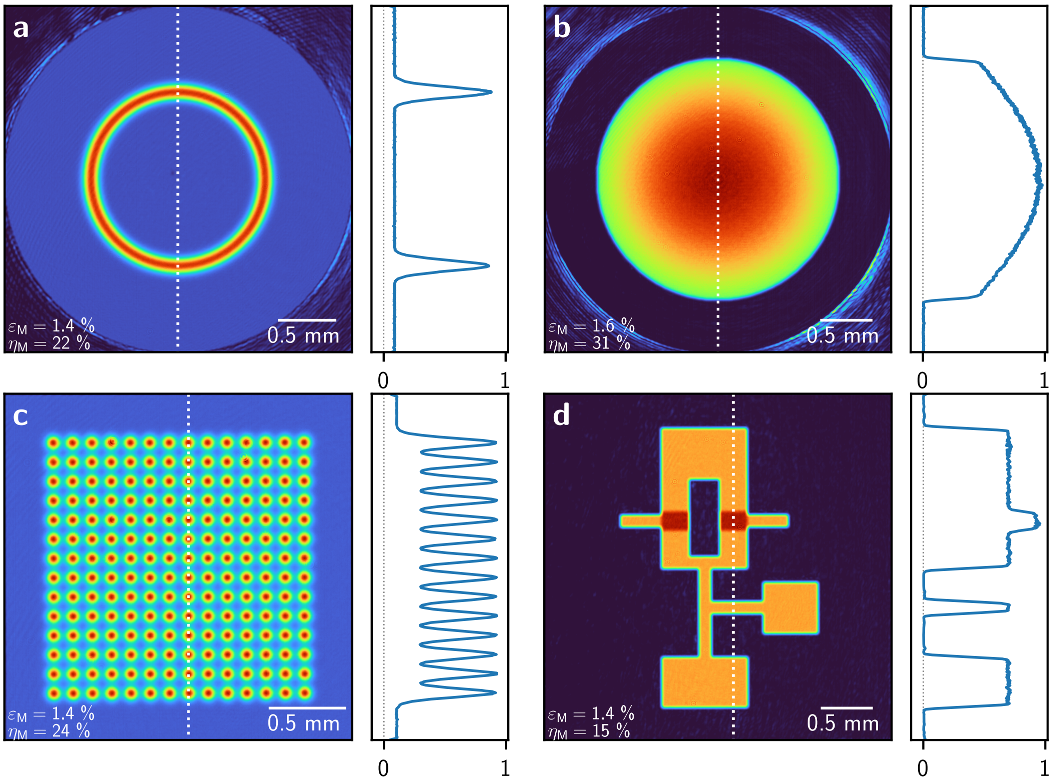


Figure 3.8: Camera images and their normalised profiles (along the white dashed lines) after 15 feedback iterations using the FFT with the crosstalk model [111]. (a) Ring with a Gaussian profile on a non-zero background. (b) Gaussian potential with offset. (c) Gaussian spot array on a non-zero background. (d) An 'atomtronic' logical OR gate [96].

less efficient and vice versa [10, 97]. By removing the phase constraint, accurate and efficient light potentials were generated computationally using the conjugate gradient method [95], however, the unrestrained phase makes it difficult to realise these experimentally [116]. Here, we minimised experimental errors by characterising our optical system and by using camera feedback. This allows us to generate accurate and efficient light potentials experimentally, without constraining the phase in the image plane. Previous studies have characterised their optical system and used camera feedback without constraining the phase [94, 112], however, using an IFTA (MRAF [96] or OMRAF [94]) instead of the conjugate gradient algorithm, resulting in less accurate and less efficient experimental light potentials than presented here. In our work, accounting for pixel crosstalk further reduced the RMS error, especially for large light potentials, while lowering the efficiency by $\sim 20\%$ (bottom of Table 3.1).

3.5 Rigorous modelling of the propagation of light

We investigated if a more accurate method to model the propagation of light, the angular spectrum method, reduces the error in our light potentials. In contrast to the Fourier transform, the angular spectrum method provides a direct solution to the Helmholtz equation [110] and does not require the paraxial approximation or the far field approximation. The increased accuracy comes at the expense of computation speed. For our optical setup, the angular spectrum method requires a larger amount of zero-padding compared to the FFT since the pixel size of the input and output planes are identical. For each free-space propagation from the SLM to the Fourier lens and from the Fourier lens to the image plane, the angular spectrum method requires one FFT and an inverse FFT, resulting in a total of four Fourier transforms to model the propagation from the SLM to the image plane. We did not see an improvement in the

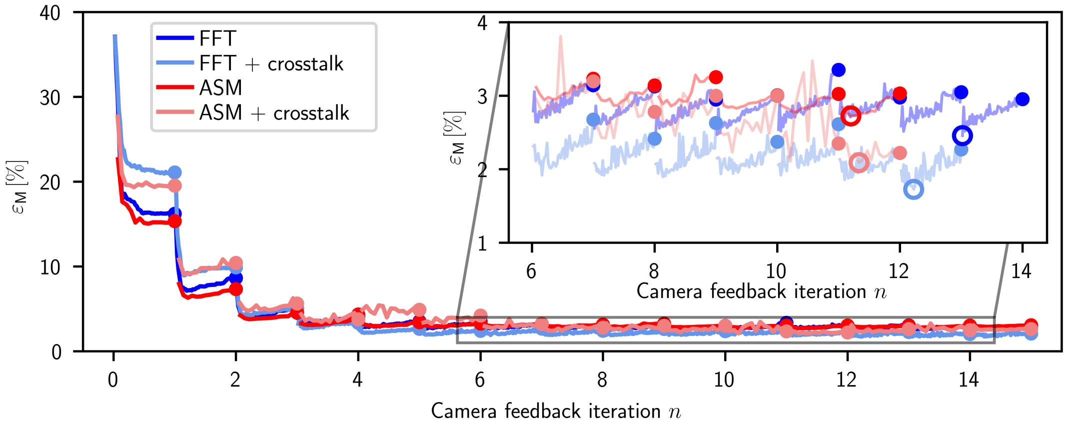


Figure 3.9: Convergence of the feedback procedure using the FFT and the angular spectrum method (ASM), with and without pixel crosstalk modelling [111]. The main figure shows ε_M as it converges for $n = 15$ camera iterations with $m = 100$ conjugate gradient iterations in between. The values ε_M used in the camera feedback process are shown as filled circles. To investigate the behaviour of ε_M during the conjugate gradient minimisation, we saved intermediate phase patterns and analysed the resulting light potentials (lines in main figure). For $n > 1$, the experimental error, ε_M, is smallest for $m < 100$. The inset shows the convergence during the final 8 camera feedback iterations. The lowest experimental error was found between $n = 11$ and $n = 14$ (hollow circles in the inset).

RMS error when using the angular spectrum method (Appendix A) instead of the FFT, however, other experimental uncertainties such as a displacement of the Fourier lens in the xy-plane or a tilt of the Fourier lens could be modelled with the angular spectrum method to improve the accuracy of the light potentials before any camera feedback. The light potential might have drifted on the camera during the longer computation time of the angular spectrum method, increasing the RMS error (Section 2.6.3 and Section 3.1). Cold-atom experiments require microscopic potentials to be projected using a high-NA objective, which will be discussed in Section 3.8. The FFT might not be sufficient to model this high-NA objective due to the large diffraction angles and the angular spectrum method could lead to more accurate potentials in this scenario, even without restricting the phase.

3.6 Efficiency measurement

To obtain the power in the signal region, P_S, we measure the optical power that corresponds to a certain pixel value and exposure time of the camera image. We display a circular mask on the SLM containing a linear phase gradient and place an iris in the image plane to block the zeroth-order light. Only the power of the first-order spot caused by the SLM phase pattern is measured using a power meter. We then take a camera image of this spot with a certain exposure time and relate the pixel sum of the camera image to the measured power. Using this calibration, the optical power, P_S, is calculated from the pixel sum of the camera image inside the transformed signal region, $\sum_{u,v\in S_U} I_{uv}$, and the exposure time. The predicted efficiency, η_{P}, is always higher than the measured efficiency, η_{M}, as it does not take the diffraction efficiency of the SLM into account. When displaying a flat phase on the SLM, the measured power of the zeroth-order spot is 69% of the incident power, P_{in}.

3.7 Camera artefacts

We observe a chequerboard-like effect on our camera (Matrix Vision BlueFox 3), similar to a Bayer pattern on a colour RGB camera. The intensity of every other camera pixel is offset by $\sim 0.5\%$, causing high-frequency noise in the measured light potential which

				Simulation				Experiment			
Publication	Pattern	Method	Propagation	ε_{P} [%]	η_{P} [%]	$\mathrm{PSNR}_{\mathrm{P}}$ [dB]	$\mathrm{MSSIM}_{\mathrm{P}}$	ε_{M} [%]	η_{M} [%]	$\mathrm{PSNR}_{\mathrm{M}}$ [dB]	$\mathrm{MSSIM}_{\mathrm{M}}$
Ebadi et al. [10]	Gaussian line	CG	FFT	-	38	-	-	0.7	-	-	-
Bowman [116]	Gaussian line	CG	FFT	0.5	8.3	-	-	-	3.5	-	-
Bruce et al. [92]	Gaussian ring	MRAF	FFT	0.6	-	-	-	3.9	-	-	-
This work	Gaussian ring (Fig. 3.8a)	CG (CT)	FFT	0.56	34	52.4	0.99	1.4	22	45.7	0.98
Gaunt et al. [94]	OR gate	OMRAF	ASM	1	24	-	-	7	-	-	-
Van Bijnen [131]	OR gate	MRAF	FFT	-	-	-	-	6	-	-	-
This work	OR gate (Fig. 3.8d)	CG (CT)	FFT	0.81	24	53.2	0.99	1.4	15	39.9	0.87
Harte et al. [95]	Power-law potential	CG	FFT	0.07	64	-	-	-	-	-	-
This work	Gaussian potential (Fig. 3.8b)	CG (CT)	FFT	0.80	55	48.4	0.94	1.6	31	40.7	0.87
Gaunt et al. [94]	Top-hat	OMRAF	ASM	-	-	-	-	6	-	-	-
Van Bijnen [131]	Top-hat	MRAF	FFT	-	-	-	-	1.7	-	-	-
Peng et al. [6]	Full colour scenes	Adam, CITL	ASM	-	-	34.3	0.96	-	-	18.5	0.66
Choi et al. [7]	Full colour scenes	ADMM, CNN	ASM	-	-	38.8	-	-	-	22.7	0.79
This work	Small disc (Fig. 3.7a)	CG	FFT	0.91	87	51.9	0.92	1.1	40	43.7	0.95
This work	Spot array (Fig. 3.8c)	CG (CT)	FFT	0.74	41	49.1	0.99	1.4	24	43.8	0.99
This work	Large disc (Fig. 3.9)	CG	FFT	1.1	78	43.4	0.90	2.7	34	32.5	0.67
This work	Large disc (Fig. 3.9)	CG	ASM	1.0	67	-	-	2.8	33	31.7	0.64
This work	Large disc (Fig. 3.9)	CG (CT)	FFT	0.92	54	48.7	0.88	1.9	28	35.3	0.73
This work	Large disc (Fig. 3.9)	CG (CT)	ASM	0.87	55	-	-	2.1	27	31.4	0.54

Table 3.1: Simulated (predicted) and experimental errors, $\varepsilon_{\mathrm{P/M}}$, efficiencies, $\eta_{\mathrm{P/M}}$, peak signal-to-noise ratio, $\mathrm{PSNR}_{\mathrm{P/M}}$, and mean structural similarity index measure, $\mathrm{MSSIM}_{\mathrm{P/M}}$, of previous studies compared to this work. In the last four rows, we compare different methods using the disc-shaped target light potential (convergence shown in Fig. 3.9).

negatively affects the convergence of the feedback algorithm, limiting the accuracy of our light potentials to around $\varepsilon_{\mathrm{M}} \sim 1.4\%$ (Section 3.4). By characterising this effect and compensating for it, we lowered the RMS error to around $\varepsilon_{\mathrm{M}} \sim 1.2\%$, however, there were other artefacts present such as vertical stripes on the sensor which were difficult to remove. We later tested a different camera (Andor Zyla 5.5) with higher sensor uniformity which resulted in significantly more accurate potentials with $\varepsilon_{\mathrm{M}} \sim 0.7\%$ (Chapter 4).

3.8 Microscopic light potentials

The light potentials presented above are sufficiently accurate for most applications in cold-atom experiments, however, these experiments require local control on a micrometre scale. In this section, we discuss generating microscopic light potential by downscaling the existing potentials using a high-NA microscope objective.

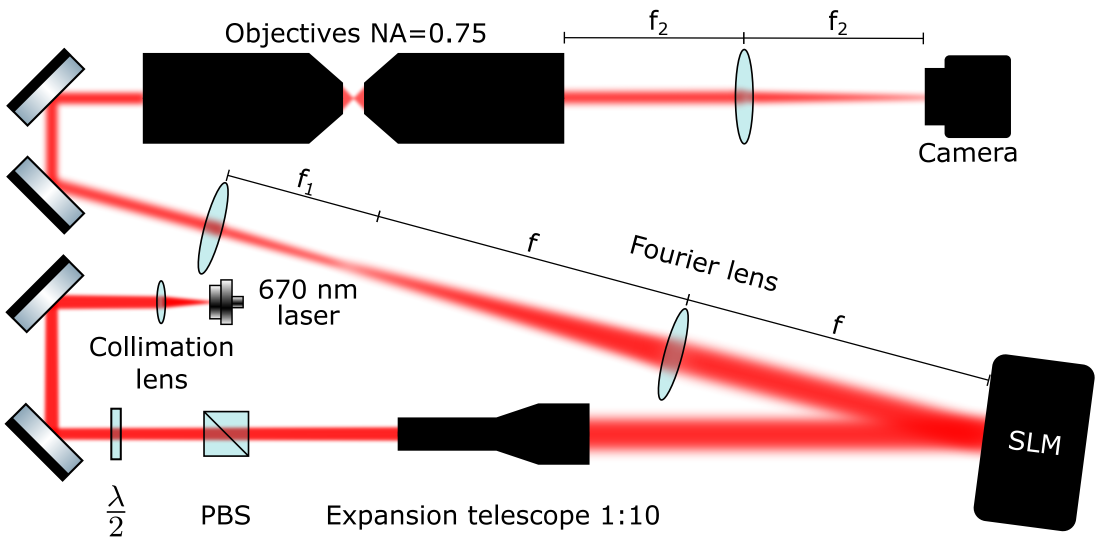


Figure 3.10: Holographic setup using two microscope objectives (Olympus MPLN50X) [138]. An image forms in the focal plane of the Fourier lens ($f = 250$mm) which is demagnified by the first tube lens ($f_1 = 150$mm) and the first microscope objective. The second microscope objective and tube lens ($f_2 = 200$mm) image the microscopic potentials onto a CCD camera.

3.8.1 Setup

To create microscopic light potentials, we demagnify the potentials in the Fourier plane by extending our existing setup (Fig. 2.2) by a tube lens ($f_1 = 150\,\mathrm{mm}$) and a high-NA microscope objective (Olympus MPLN50X, NA = 0.75, Fig. 3.12). Due to the large demagnification (a factor of 42), it is not feasible to directly place a camera in the focal plane of the microscope objective to perform camera feedback since the pixel size of the camera ($3.75\,\mu$m) is too large to resolve the potentials. Instead, to measure the microscopic potentials, we use a second, identical microscope and another tube lens ($f_2 = 200\,\mathrm{mm}$) to magnify the potentials in the focal plane of the objective onto the feedback camera. Due to the low transmission of the microscopes at 852 nm, we changed the wavelength to 670 nm.

3.8.2 Calibration

The collimation of the laser beam after the first tube lens and after the second microscope objective is adjusted using a shearing interferometer. The position of the second microscope objective along the optical axis was fine-tuned using a piezo stage to minimise the size of the focal spot on the camera. By running the intensity measurement for the incident laser beam (Section 2.6.2) using the camera after the microscopes, we image the aperture of the first microscope objective on the SLM. When running the conjugate gradient minimisation, we only use the central circular region on the SLM which is not truncated by the aperture of the objective – a flat phase is displayed on the remaining pixels. We correct for aberrations in the system by running our phase calibration (Section 2.6.3) and displaying the negative measured phase on the SLM. To verify that our imaging system is diffraction-limited, we investigate its point spread function. The diffraction-limited point spread function of our imaging system is an Airy disc of radius, r_{lim}, given by

$$r_{\mathrm{lim}} = \frac{1.22\lambda}{2\mathrm{NA}}, \tag{3.3}$$

resulting in $r_{\mathrm{lim}} = 545\,\mathrm{nm}$ using $\lambda = 670\,\mathrm{nm}$ and NA = 0.75. This corresponds to the radius of the focal spot in the focal plane of the microscope objective, provided that

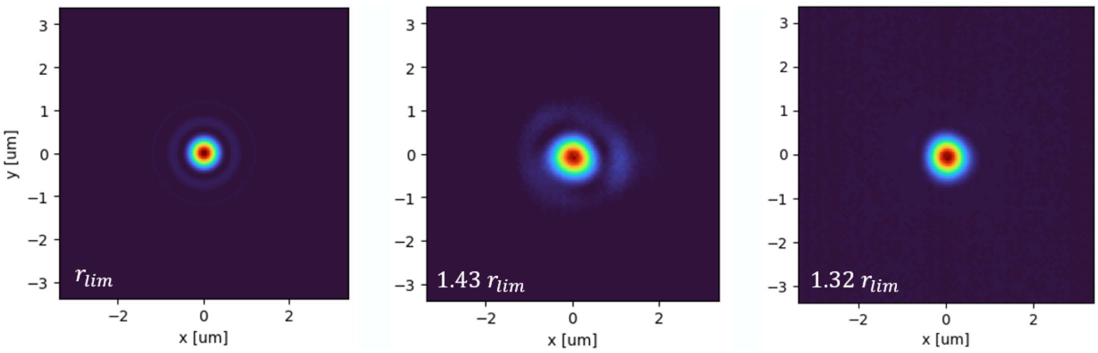


Figure 3.11: Focal spot in the focal plane of the microscope objective. (a) The calculated, diffraction-limited PSF. (b) Measured PSF before the phase calibration and (c) after the phase calibration.

the entire aperture of the microscope objective is illuminated uniformly (Fig. 3.11a). However, the Gaussian laser beam at the SLM ($1/e^2$ beam diameter $d_{\mathrm{SLM}} = 7.25\,\mathrm{mm}$) is demagnified by the Fourier lens and the first tube lens to a $1/e^2$ beam diameter at the microscope objective, $d_{\mathrm{MO}} = 4.35\,\mathrm{mm}$. Since the beam at the microscope objective is smaller than the diameter of the microscope objective aperture, $a_{\mathrm{MO}} = 6\,\mathrm{mm}$, we expect a larger focal spot with the radius, r_f,

$$r_f = \frac{a_{\mathrm{MO}}}{d_{\mathrm{MO}}} \cdot r_{lim} = 1.38\, r_{lim} \tag{3.4}$$

Before calibrating the constant phase at the SLM, the measured PSF of the focal spot is larger than the expected radius, r_f (Fig. 3.11b). However, after calibrating the phase, the width of the PSF is slightly smaller than r_f ($1.32\, r_{\mathrm{lim}}$, Fig. 3.11c) which indicates that our imaging system is diffraction-limited.

3.8.3 Results: Comparison to macroscopic potentials

We generated the same patterns used in Section 3.4, Fig. 3.8 on a microscopic scale, corresponding to a demagnification by a factor of 42. The experimental RMS error of our microscopic potentials varies between $\varepsilon_{\mathrm{M}} = 3.6 - 6.2\%$ which is ~ 3 times larger than the macroscopic potentials generated without the objectives [111]. Previously, microscopic ring traps with a root-mean-squared (RMS) error of $< 5\%$ were used in cold-atom experiments to investigate Bose-Einstein condensates [40, 92]. The increased number of optical elements in our setup might lead to larger errors due to possible mis-

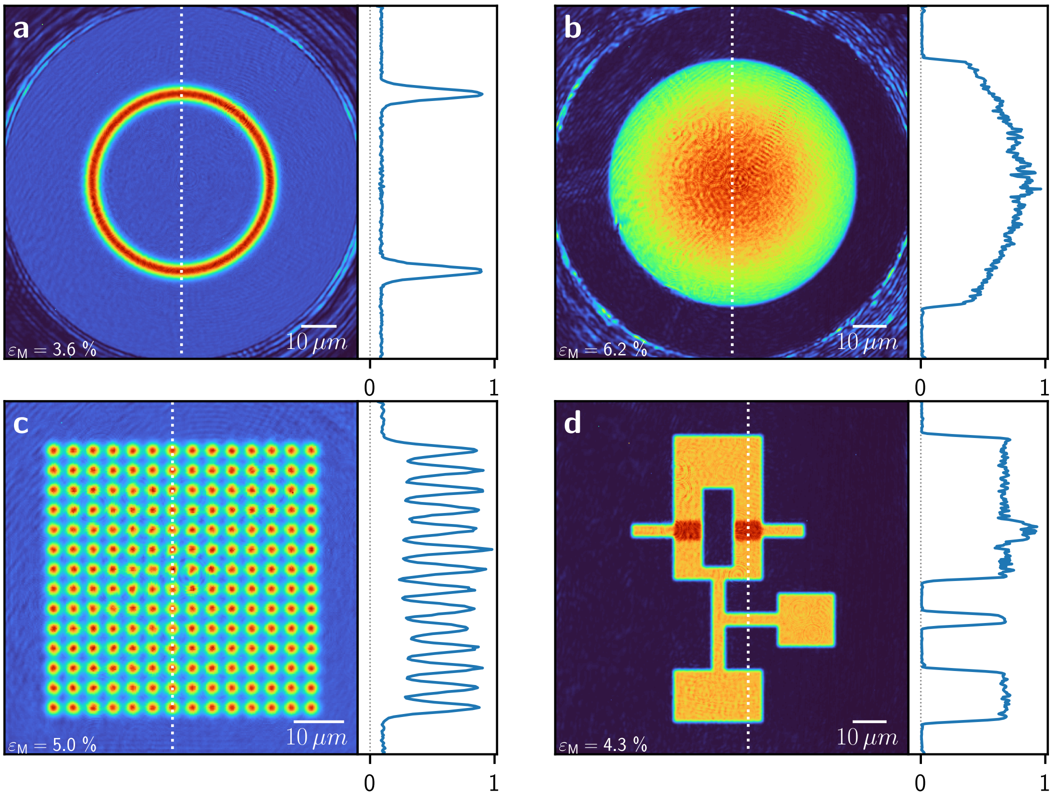


Figure 3.12: Camera images of microscopic light potentials and their normalized profiles (along the white, dashed lines) using patterns from previous work [138]. The scale bar indicates the size of the potentials in the focal plane of the microscope objective. (a-d) Ring with Gaussian profile, Gaussian profile with offset, Gaussian spot array, and an 'atomtronic' logical OR gate.

alignment. Further, the Fourier transform used to simulate the propagation of light in our phase-retrieval algorithm (Section 2.5) does not take the large diffraction angles and polarization effects caused by the high-NA microscope objective into account. These errors could be further reduced by using the angular spectrum method to model possible misalignment of optical elements and to model the large diffraction angles caused by the high-NA objectives.

3.9 Outlook

The camera feedback algorithm we are using produces light potentials of low RMS error in a Fourier imaging setup, however, it has several drawbacks. Running the feedback algorithm only removes errors for a specific light potential – if a different shape is required, this new potential must be re-optimised. Another approach, which does not have this limitation, is to model a digital twin of the experimental setup which captures as many sources of error as possible [6]. Experimental imperfections in the physical setup can be mimicked by optimising the model's parameters to minimise the difference between camera images and the corresponding simulated output of the model. After the model is trained, experimentally accurate potentials can be generated without relying on iterative optimisation using the camera image [6]. If required, the resulting potential can be further optimised using an iterative method. This approach would be especially interesting for the generation of microscopic light potentials, where one could parameterise the entire optical path and learn the aberrations introduced by each optical element and diffraction effects caused by, for example, the aperture of the microscope objective. A downside of this approach is that any change in the experiment, for example, a position drift of the incident laser beam, requires the digital twin to be re-trained, which is time-consuming. Another iterative feedback method known as camera-in-the-loop optimisation [6] can directly solve the phase retrieval problem by optimising the camera image using adaptive moment estimation. Gradient-based optimization of a simulated image is possible since the phase of the SLM is connected to the simulated image via mathematical operations which allows the computation of the gradient with respect to the SLM phase using automatic differentiation in, for example, PyTorch. However, when calculating the cost function directly from the camera image, the gradient of the cost function with respect to the SLM phase is no longer accessible [6]. To circumvent this problem, camera-in-the-loop optimisation uses an approximation of the gradient

$$\frac{\partial C}{\partial \varphi} = \frac{\partial C}{\partial f} \cdot \frac{\partial f}{\partial \varphi} \approx \frac{\partial C}{\partial f} \cdot \frac{\partial \hat{f}}{\partial \varphi}, \tag{3.5}$$

where f is the experimental propagation (with inaccessible gradient) and $\hat{f}$ is the simulated propagation of which we can calculate the gradient (see our definition of the gradient in Section 2.5.2). Using this approximate gradient, the phase retrieval problem can be solved by optimising the camera image directly. Camera-in-the-loop optimisation requires fewer total iterations to converge compared to the simple feedback algorithm we are using, however, more camera images are required as each cost function evaluation requires a camera image. Camera-in-the-loop optimisation produces very accurate results and is state-of-the-art for full-colour holographic displays [6, 7]. This method produces results of $\mathrm{MSSIM_M} = 0.79$ for full-colour images by using three different wavelengths (red, green, and blue) [7]. Our method can produce light potentials with a mean structural similarity index measure of $\mathrm{MSSIM_M} = 0.73 - 0.99$ (Table 3.1), however, we only use a single wavelength and our target light potentials are less complex in comparison. It would be interesting to directly compare this technique to our implementation of camera feedback for the same patterns we used in this work.

When generating microscopic light potentials, we model the propagation of light using an FFT which is inadequate due to the large diffraction angles introduced by the high-NA microscope objective. The angular spectrum method can model large diffraction angles, however, it has a drawback which prevents us from using it in our simulations. In our initial implementation of the angular spectrum method, the physical size of a pixel in the input plane and the output plane are identical. This becomes a problem since we require starkly varying pixel sizes in the SLM plane and the focal plane of the objective. While the SLM pixel size is $12.5\,\mu$m, a resolution of $\sim 250\,$nm is desired in the focal plane of the microscope objective. A naive application of the angular spectrum method would require upscaling the SLM plane by a factor of 50, resulting in a zero-padded array of 102400×102400 pixels which is far beyond the capabilities of our desktop workstation. Modified versions of the angular spectrum method with varying pixel sizes in the input and output planes were developed to reduce the computational requirements. By treating the quadratic phase term introduced by the microscope objective analytically, a semi-analytical angular spectrum method was developed [99, 139], greatly reducing the sampling requirements. More recently, a scalable angular spectrum propagation was proposed [100] which performs a zoom-in operation using three FFTs, further lowering the sampling requirements over the semi-analytical angular spectrum method. These propagation methods would enable to model the entire optical path of the setup discussed in Section 3.8 during the phase retrieval process which will be the subject of future work. Alternatively, microscopic light potentials of high accuracy were generated by constraining the phase of the light potential to be flat [10]. While constraining the phase makes the light potentials less sensitive to experimental errors such as misalignment of optical elements, it typically leads to less efficient potentials [10, 97].

Chapter 4

Fast SLM calibration using Fourier imaging and adaptive moment estimation

In this chapter, we present a new calibration technique that is faster than previous methods while maintaining the same level of accuracy (Section 4.1). By employing stochastic optimisation and random speckle intensity patterns, we calibrate a digital twin that accurately models the experimental setup. This approach allows us to measure the wavefront at the SLM to within $\lambda/170$ in $\sim$5 minutes using only 10 SLM phase patterns, a significant speedup over state-of-the-art techniques. Additionally, our digital twin models pixel crosstalk on the liquid-crystal SLM, enabling rapid calibration of model parameters and reducing the error in light potentials by a factor of $\sim$5 without losing efficiency (Section 4.2). Our fast calibration technique will simplify the implementation of high-fidelity light potentials in, for example, quantum-gas microscopes and neutral-atom tweezer arrays where high-NA objectives and thermal lensing can deform the wavefront significantly. Applications in the field of holographic displays that require high image fidelity will benefit from the novel pixel crosstalk calibration, especially for displays with a large field of view and increased SLM diffraction angles. To compare our method to previous results, we generate light potentials of various sizes and characterise them (Section 4.3).

The previously presented methods to calibrate the intensity (Section 2.6.2) and the wavefront (Section 2.6.3) at the SLM result in accurate holographic light potentials, however, these calibration methods take several hours to run. Any change in the experimental setup, for example, any drift of the incident laser beam, requires a re-calibration. Further, to find the optimal parameters for the pixel crosstalk kernel (Section 3.3), an exhaustive 2D optimisation of both kernel parameters was necessary.

4.1 Constant field at the SLM

Methods to measure the constant phase at the SLM with high speed and accuracy using a Twyman-Green interferometer were proposed [141, 142], however, they require additional optical elements and a flat reference mirror. Self-interfering calibration schemes (Section 2.6) that do not require additional equipment are typically slow as they involve displaying a large number of phase patterns on the SLM to achieve the desired spatial resolution and are not straightforward to implement [46, 111, 124, 143–145]. With our

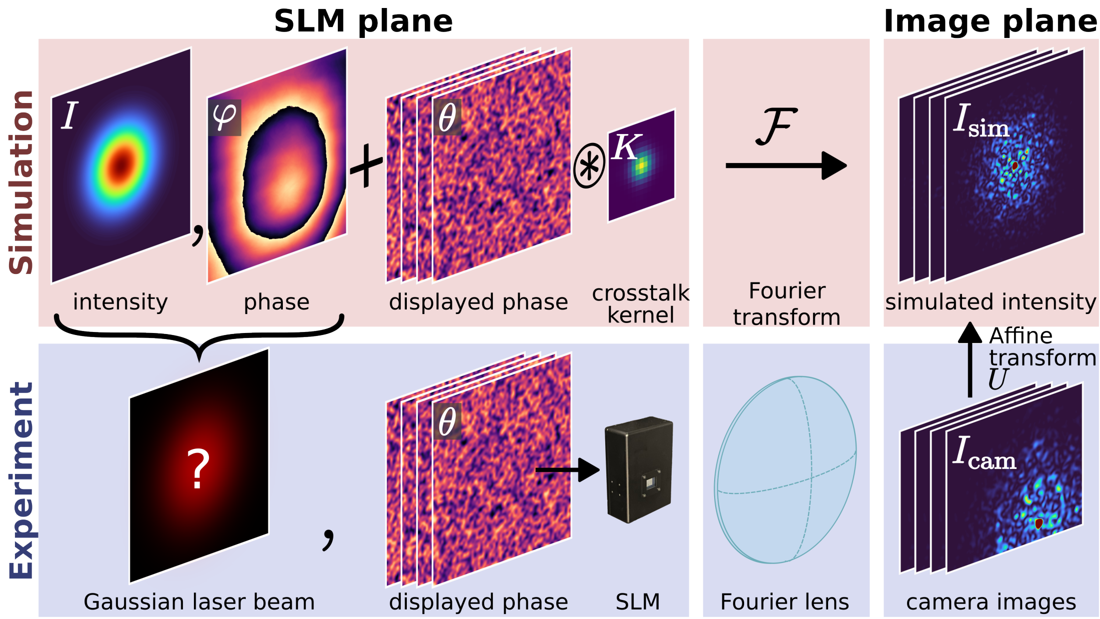


Figure 4.1: A digital twin (red box) simulates our experimental Fourier imaging setup (blue box) [140]. To train the digital twin, parameters of the simulation in the red box (the constant intensity and wavefront at the SLM, I and φ, the pixel crosstalk kernel, K, and the affine transformation matrix, U) are adjusted to minimise the difference between the simulated images, I_{kl}, and the camera images, I_{uv}, when displaying semi-random phase patterns, θ, on the SLM. The physical lens in the experimental setup is modelled by a Fourier transform.

specific hardware, measuring the wavefront at the SLM at a resolution of 128×128 data points takes over an hour, since due to the response time of the liquid crystal material, it takes $\sim 200\,\text{ms}$ for each of the phase patterns to stabilize before we can take a camera image.

As reported recently, the phase and amplitude profile of test objects were recovered by displaying only eight random phase patterns on the SLM and recording the corresponding speckle images [3]. To recover the unknown electric field, an iterative Fourier transform algorithm (IFTA) was employed to minimise the difference between the camera images and the simulated images. In a different study [6], thousands of pre-calculated phase patterns were displayed on the SLM. Their corresponding camera images were recorded and used to train a parameterised model of the experimental setup by optimisation using adaptive moment estimation. The parameterisation included modelling the amplitude and phase at the SLM using a sum of Gaussians and Zernike polynomials, respectively.

In this chapter, I present a method that allows us to precisely measure the intensity profile of the incident laser beam and its wavefront in a matter of minutes, requiring less than 10 camera images without a reference mirror or other additional hardware. A sequence of random phase patterns is displayed on the SLM to recover the constant field at the SLM using optimisation by adaptive moment estimation, a form of stochastic gradient-based optimisation. We introduce gradient terms to our cost function to optimise for the smoothness of the constant phase and amplitude as opposed to modelling the constant field using smooth, analytical functions. This removes any geometric constraints from the intensity profile and the wavefront that might have been imposed by, for example, radially symmetric Zernike polynomials used in previous work [6].

4.1.1 Fast calibration method

To measure the constant phase, φ, and intensity at the SLM, I, we display several smooth, semi-random phase patterns, θ, on the SLM (Fig. 4.1) and record the corresponding speckle patterns, I_{cam}, using the camera (Andor Zyla 5.5, $6.5\,\mu$m pixel pitch, 2560×2160 pixels). The phase patterns are generated from an array of 128×128 pixels with uniformly distributed phase values between 0 and 3π. This array is upscaled to the native resolution of the SLM by a factor of 8 using nearest-neighbour interpolation and is convolved with a Gaussian kernel with 8 SLM pixels width. The convolution produces smooth, semi-random phase patterns which suppress the effect of pixel crosstalk [3] since neighbouring pixels do not display starkly different phase values in these patterns. This is necessary as we do not account for pixel crosstalk when calibrating the constant intensity and wavefront at the SLM. The expected camera images, I_{sim}, are simulated by propagating the electric field from the SLM plane to the image plane using a Fourier transform, $\mathcal{F}$,

$$I_{\mathrm{sim}} = \left|\mathcal{F}\{E\}\right|^2 , \tag{4.1}$$

with $E_{kl} = A_{kl}\, e^{i(\varphi_{kl} + \theta_{kl})}$, where A_{kl} is the amplitude profile of the laser beam with its intensity, $I_{kl} = |A_{kl}|^2$, its spatially varying phase, φ_{kl}, and the phase displayed on the SLM, θ_{kl}, with indices, k, l. In the computational implementation, we now use a type 2 non-uniform fast Fourier transform which allows us to arbitrarily choose the pixel pitch and the region of interest in the Fourier plane. This saves memory at the cost of execution speed compared to the regular FFT, provided that the camera has a smaller field of view and fewer pixels than the computational image plane with the regular FFT. Here, we choose the pixel pitch in the Fourier plane to match the pixel pitch of our camera and only compute the area of the Fourier plane which is covered by the camera. To find the coordinates in Fourier space which the camera occupies, we detect the position of the zeroth-order diffraction spot on the camera by fitting a 2D Gaussian to it. We then calculate the desired pixel pitch in the Fourier plane, p_{NUFT} (in radians, ranging from $-\pi$ to π), given by

$$p_{\mathrm{NUFT}} = \frac{2\pi}{\lambda f}\, p_{\mathrm{SLM}}\, p_{\mathrm{CAM}}, \tag{4.2}$$

using the camera's pixel pitch, p_{CAM}, the pixel pitch of the SLM, p_{SLM}, with the wavelength, λ, and the focal length of the Fourier lens, f. In practice, this mapping of the camera coordinates to the coordinates of the Fourier plane is not accurate enough due to experimental errors such as a slightly rotated camera. To account for a slight mismatch between the camera image and the simulated image, we employ a partial affine transformation which models the rotation, translation, and scaling in the x- and

y-directions of the camera image but no shear. By adjusting the value of every pixel in the constant phase and intensity as well as tuning the parameters in the partial affine transformation matrix, we minimise a cost function using optimisation by adaptive moment estimation (Adam solver [146] using PyTorch [121]). Our cost function, C_{MSE}, computes the mean-squared error (MSE) of the difference between the transformed camera image and the simulated image,

$$C_{\mathrm{MSE}}(\varphi,\, I,\, U) = \frac{1}{N_{\mathrm{F}} N^2} \sum_{n=1}^{N_{\mathrm{F}}} \sum_{k,\, l}^{N} \left[\left(I^n_{\mathrm{sim},\, kl} - \mathcal{T}_U \{ I^n_{\mathrm{cam}} \}_{kl} \right)^2 \right], \tag{4.3}$$

where N_{F} is the number of phase patterns and camera images used in the optimisation, N the number of pixel rows and columns on the SLM ($N = 1024$ for our specific SLM) and $\mathcal{T}_U\{\cdot\}$ is the affine transformation operator with the affine transformation matrix, U. We introduce two smoothness terms, C_φ and C_A, to minimise the gradient of the phase and the amplitude, respectively, which is calculated using the forward difference given by

$$C_\varphi = \frac{1}{(N-1)^2} \sum_{i,\, j}^{N-1} \left[(\varphi_{i,\, j} - \varphi_{i+1,\, j})^2 + (\varphi_{i,\, j} - \varphi_{i,\, j+1})^2 \right] \text{ and} \tag{4.4}$$

$$C_A = \frac{1}{(N-1)^2} \sum_{i,\, j}^{N-1} \left[(A_{ij} - A_{i+1,\, j})^2 + (A_{i,\, j} - A_{i,\, j+1})^2 \right]. \tag{4.5}$$

C_φ and C_A ensure that the constant wavefront and the intensity at the SLM remain smooth as one would expect from the wavefront of a Gaussian beam and its intensity profile. We weigh our cost function terms like follows

$$C = s \left(C_{\mathrm{MSE}} + s_\varphi C_\varphi + s_A C_A \right), \tag{4.6}$$

with the overall steepness of the cost, s, and the weighing parameters, s_φ and s_A. By changing the weighing terms in the cost function, the smoothness of the amplitude and phase can be controlled individually at the cost of accuracy (higher mean-squared error). Excluding the smoothness terms from the cost function and only minimising the mean squared error causes the optimiser to introduce unwanted high-frequency artefacts in the recovered constant phase and intensity. The smoothness terms minimise variations between neighbouring pixels in the recovered wavefront and intensity which suppresses high spatial frequencies. This form of regularisation prevents overfitting, promoting global convergence by reducing the likelihood of getting stuck in a local minimum.

4.1.2 Calibration results

To run the optimisation, we use $N_\mathrm{F} = 10$ different random SLM phase patterns and the corresponding camera images. The cost function steepness, $s = 10^{14}$, and the weighing parameters, $s_\varphi = 5 \times 10^{-3}$ and $s_A = 2 \times 10^{-2}$ were chosen empirically. To find these values for s_φ and s_A, we calculated C_A from an analytical Gaussian beam profile of 7.4 mm beam diameter and C_φ from the SLM phase correction pattern provided by the manufacturer. We then adjusted s_φ and s_A to keep C_A and C_φ approximately at the previously calculated values throughout the optimisation. Without prior knowledge of the constant field at the SLM, the constant phase, φ, was initialised with an array of zeros and the constant amplitude, A, with an array of ones. After 2000 iterations (around 7 minutes on an Nvidia RTX A5000 GPU for $N_\mathrm{F} = 10$), we stop the optimisation as stagnation is reached, resulting in a smooth intensity profile (Fig. 4.2b) and wavefront (Fig. 4.3b). To validate the results from our new stochastic approach, we compare them to the results from our previous calibration method based on shifting the position of

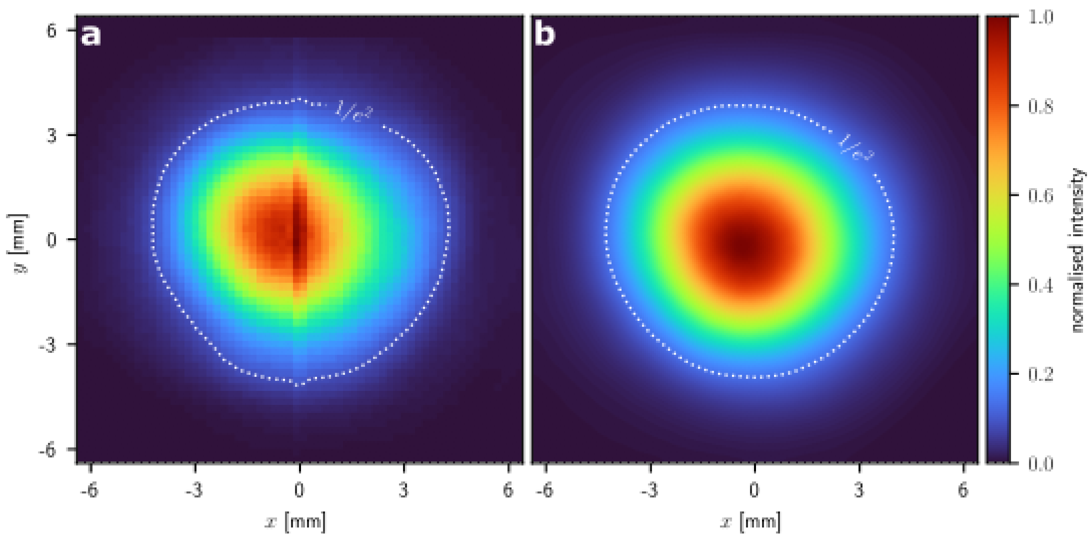


Figure 4.2: Laser intensity profiles recovered using (a) local sampling method [46] using 64×64 measurements and (b) the stochastic approach presented here [140]. The $1/e^2$ intensity threshold is indicated by the dashed white line.

local phase patterns across the SLM (we will refer to this as local sampling method [46], Section 2.6). The beam diameter calculated from the intensity measurement using the stochastic approach is larger compared to the local sampling method (8.1 mm compared to 7.4 mm). The local sampling method generated a visible vertical line in the intensity profile in the centre of the SLM, caused by how the SLM updates the phase pattern (private communication with Hamamatsu [127], Fig. 4.2a). This artefact does not appear in the measurement using the stochastic approach (Fig. 4.2b). The wavefronts agree well in the centre of the SLM (Fig. 4.3), however, the wavefront recovered using the stochastic approach is noticeably flatter in darker regions of the laser beam towards the edges of the SLM.

To characterise φ, we only consider a region, $\mathcal{B}$, on the SLM in which the intensity is larger than $1/e^2$ of the maximum intensity (region within the white, dashed line in Fig. 4.2b). To determine the recalibration error of the phase measurement using the stochastic method, we subtract the initially measured phase from the semi-random phase patterns and display the resulting phase, $\theta - \varphi$, on the SLM. When re-running the stochastic method using those wavefront-corrected phase patterns, we obtain a residual

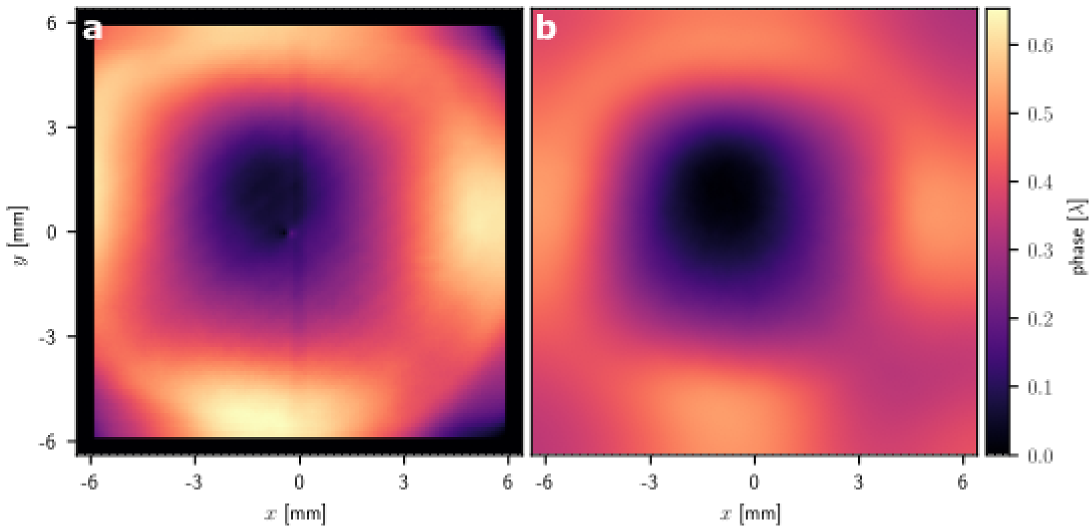


Figure 4.3: Wavefronts recovered using (a) the local sampling method [46], φ_C^{zup}, using 124×124 measurements and (b) the stochastic approach, φ_C^{ada}, using adaptive moment estimation [140]. Due to technical reasons (Section 2.6.3), the local sampling method cannot measure the wavefront at the very edge of the SLM, indicated by the black border in (a).

phase, $\delta\varphi$, after removing the tilt within $\mathcal{B}$. To quantify the recalibration error, we calculate the standard deviation of the residual phase

$$\sigma = \sqrt{\frac{1}{N_{\mathcal{B}}k^2} \sum_{i,j \in \mathcal{B}} \left(\delta\varphi_{ij} - \overline{\delta\varphi}\right)^2}, \tag{4.7}$$

where $\overline{\delta\varphi}$ is the mean value of $\delta\varphi$ in $\mathcal{B}$, containing $N_{\mathcal{B}}$ pixels with indices i, j and $k = 2\pi/\lambda$. With the stochastic approach ($N_{\mathrm{F}} = 10$), we obtain a similarly small standard deviation, $\sigma = \lambda/170$, compared to the local sampling method, $\sigma = \lambda/180$.

To investigate the accuracy of the phase measured using the stochastic approach, φ_{SGD}, we quantify its deviation from the phase measured using the local sampling method, φ_{LS}, by calculating the RMS error of their difference, $\Delta\varphi = \varphi_{\mathrm{SGD}} - \varphi_{\mathrm{LS}}$, in region $\mathcal{B}$,

$$\epsilon = \sqrt{\frac{1}{N_{\mathcal{B}}k^2} \sum_{i,j \in \mathcal{B}} \left(\Delta\varphi_{ij} - \overline{\Delta\varphi}\right)^2}, \tag{4.8}$$

where $\overline{\Delta\varphi}$ is the mean value of $\Delta\varphi$ in $\mathcal{B}$. Using our stochastic approach with only one SLM phase pattern ($N_{\mathrm{F}} = 1$) in the optimisation, we already obtain $\epsilon = \lambda/97$. To investigate if the RMS error, ϵ, decreases further when using more than one SLM

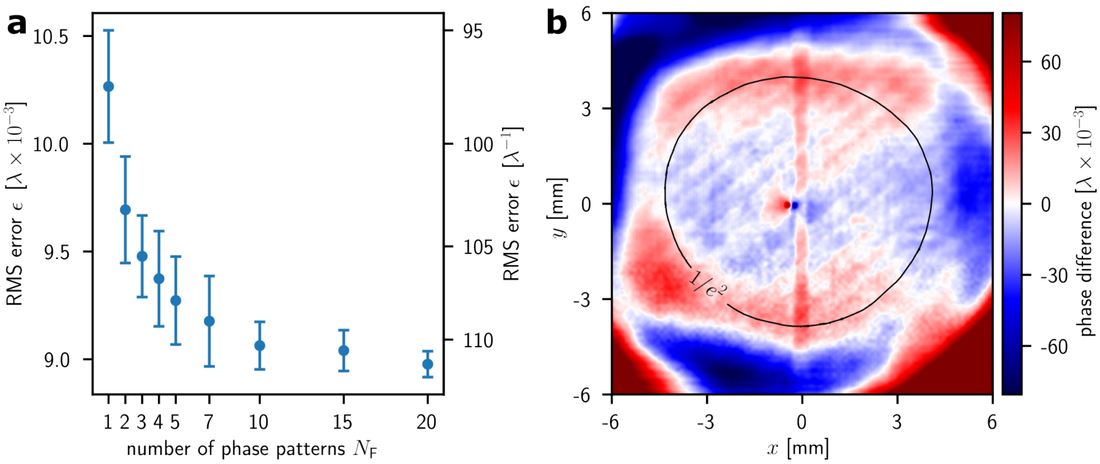


Figure 4.4: Comparing the wavefront of the stochastic approach to the local sampling method [46, 140]. (a) Residual RMS error, ϵ_{diff}, between the wavefronts obtained from each method as a function of the number of phase patterns, N_{F}, used during the stochastic approach. Only the area of the wavefront within the $\frac{1}{e^2}$ intensity threshold (dotted line in b) of the laser beam intensity is considered to calculate the RMS error. (b) Difference between the wavefronts measured using the stochastic approach ($N_{\mathrm{F}} = 10$) and the local sampling method.

phase pattern, we measure φ_{SGD} using different values of N_{F} and calculate ϵ for each measurement (Fig. 4.4a).

To investigate if the RMS error, ϵ, decreases further when using more than one SLM phase pattern, we measure φ_{SGD} using different values of N_{F} and calculate ϵ for each measurement (Fig. 4.4). When increasing N_{F}, the error decreases and reaches $\epsilon = \lambda/110$ at $N_{\mathrm{F}} = 10$. When the number of SLM phase patterns is increased further to $N_{\mathrm{F}} = 20$, the RMS error only decreases slightly. Artefacts caused by the local sampling method become evident when plotting the difference between the two wavefronts (Fig. 4.4b). Similar to the intensity measurement, a vertical stripe is visible in the centre of the SLM when using the local sampling method (Section 2.6). This line appears since the pixels on the left and right half of the SLM are driven separately (private communication with Hamamatsu [127]). Periodic diagonal stripes across the entire SLM were observed which are likely caused by the linear phase gradient used in the measurement [111] (Section 2.6.3). Combining the repeatability of both methods in quadrature yields an error of $\Delta\sigma = \sqrt{\sigma_{\mathrm{SGD}}{}^2 + \sigma_{\mathrm{LS}}{}^2} = \lambda/124$, which explains why the RMS error between the two measurements does not decrease significantly when using more than $N_{\mathrm{F}} = 10$ phase patterns (Fig. 4.4a). The runtime of the stochastic approach presented here is much shorter and requires far fewer camera images compared to the local sampling method (7 minutes runtime and 10 images compared to 2.5 h and ~ 17000 images). Another advantage of the stochastic approach is the spatial resolution of 1×1 SLM pixel, whereas the spatial resolution of the local sampling method depends on the number of images taken and limited to around 8×8 SLM pixels. For this reason, the local sampling method relies on upscaling the measured wavefront and intensity to the native resolution of the SLM using interpolation which is not required with the stochastic approach.

4.2 Pixel crosstalk

As discussed in Section 3.3, neighbouring SLM pixels which display different phase values affect each other at their bordering region which is known as pixel crosstalk or fringing field effect, leading to a non-uniform phase across a single SLM pixel. In a Fourier-imaging setup, this effect is especially noticeable for large light potentials since they require high spatial frequencies to be displayed by the SLM. Specifically, these high-frequency SLM phase patterns contain many 0 to 2π phase jumps which are impacted most by pixel crosstalk, causing artefacts in the image. The magnitude of pixel crosstalk heavily depends on the properties of the LCOS SLM such as the thickness of the liquid crystal layer and the size of the pixel electrodes - smaller pixel sizes and thicker liquid crystal layers amplify the effect. As manufacturers steadily increase the resolution of modern LCOS SLMs and consequently shrink their pixel pitch, it becomes increasingly important to model pixel crosstalk and compensate for its effects.

Here, we characterise the pixel crosstalk on the SLM by calibrating more sophisticated pixel crosstalk models compared to previous work [111] using optimisation by adaptive moment estimation, similar to the calibration of the constant field at the SLM described in the previous Section. This method accelerates the convergence of the camera feedback process significantly and further reduces the error in light potentials compared to previous work [111], especially for large potentials.

4.2.1 Modelling pixel crosstalk

In this section, we discuss modelling pixel crosstalk on the SLM, introduce several pixel crosstalk models used in previous work, and present new methods to model crosstalk. Previously, pixel crosstalk was modelled on a sub-pixel scale by upscaling the SLM phase pattern and convolving it with a parameterised crosstalk kernel [9, 98, 104, 105, 136],

$$\Theta(x, y) = \theta(x, y) \circledast K(x, y)\,, \tag{4.9}$$

where Θ is the phase after applying the pixel crosstalk model. In previous work, the crosstalk kernel, K, has been parameterised using a symmetric super-Gaussian given by equation 3.2 [102, 104, 111, 136], with two parameters (Section 3.3). We will refer to the convolution with the super-Gaussian kernel, K, as pixel crosstalk model 1a. To capture possible asymmetries, we introduce model 1b, which involves a convolution with a piecewise kernel, K_{pw} [102],

$$K_{\mathrm{pw}}(x,y) = \begin{cases} \mathcal{F}^{-1}\Big\{\exp\Big[-\Big(\frac{|\kappa_x|}{\sigma_{\mathrm{xn}}}\Big)^{q_{\mathrm{xn}}} - \Big(\frac{|\kappa_y|}{\sigma_{\mathrm{yn}}}\Big)^{q_{\mathrm{yn}}}\Big]\Big\} & x \le 0,\, y \le 0 \\ \mathcal{F}^{-1}\Big\{\exp\Big[-\Big(\frac{|\kappa_x|}{\sigma_{\mathrm{xn}}}\Big)^{q_{\mathrm{xn}}} - \Big(\frac{|\kappa_y|}{\sigma_{\mathrm{yp}}}\Big)^{q_{\mathrm{yp}}}\Big]\Big\} & x \le 0,\, y > 0 \\ \mathcal{F}^{-1}\Big\{\exp\Big[-\Big(\frac{|\kappa_x|}{\sigma_{\mathrm{xp}}}\Big)^{q_{\mathrm{xp}}} - \Big(\frac{|\kappa_y|}{\sigma_{\mathrm{yn}}}\Big)^{q_{\mathrm{yn}}}\Big]\Big\} & x > 0,\, y \le 0 \\ \mathcal{F}^{-1}\Big\{\exp\Big[-\Big(\frac{|\kappa_x|}{\sigma_{\mathrm{xp}}}\Big)^{q_{\mathrm{xp}}} - \Big(\frac{|\kappa_y|}{\sigma_{\mathrm{yp}}}\Big)^{q_{\mathrm{yp}}}\Big]\Big\} & x > 0,\, y > 0 \end{cases} \tag{4.10}$$

with different orders, q_{i}, and widths, σ_{i}, along the x and y directions for each quadrant of the crosstalk kernel. To remove any geometric constraints from the crosstalk kernel, we introduce model 1c which involves convolving the phase pattern, θ, with an entirely unconstrained kernel, K_{uc}, where each pixel value in the kernel is a learnable parameter. A more sophisticated model was introduced by Moser et al. [102] which accounts for nonlinearities that a convolution cannot model. Inspired by this approach, we implemented pixel crosstalk model 2 as follows (Fig. 4.5). [102],

$$\begin{aligned} \Theta_{m',n'} = \theta_{m,n} &+ T^0_{s,t}(\theta_{m-1,n-1} - \theta_{m,n}) + T^1_{s,t}(\theta_{m-1,n} - \theta_{m,n}) \\ &+ T^2_{s,t}(\theta_{m-1,n+1} - \theta_{m,n}) + T^3_{s,t}(\theta_{m,n-1} - \theta_{m,n}) \\ &+ T^4_{s,t}(\theta_{m,n+1} - \theta_{m,n}) + T^5_{s,t}(\theta_{m+1,n-1} - \theta_{m,n}) \\ &+ T^6_{s,t}(\theta_{m+1,n} - \theta_{m,n}) + T^7_{s,t}(\theta_{m+1,n+1} - \theta_{m,n}), \end{aligned} \tag{4.11}$$

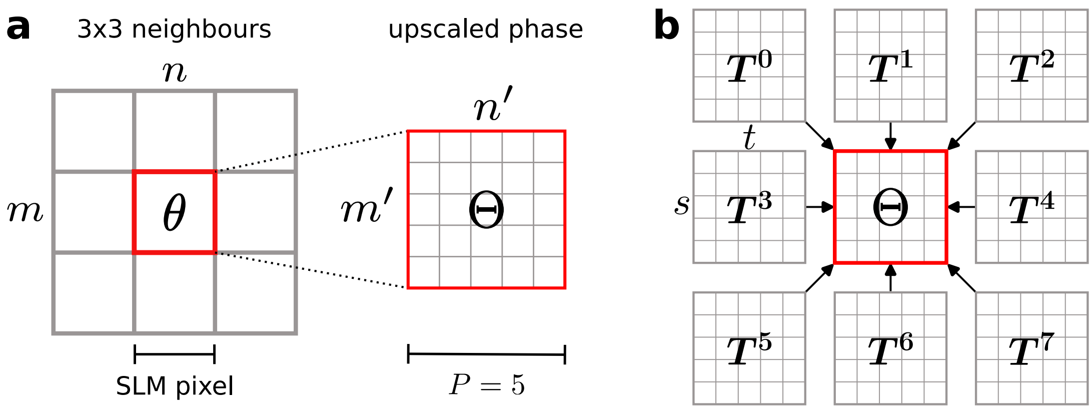


Figure 4.5: Pixel crosstalk model 2 inspired by Moser et al. [102, 140]. The displayed SLM phase θ is upscaled by a factor of P by applying the crosstalk model (equation 4.11).

with the indices of the upscaled phase m', $n' \in 0, 1, ..., PN$ and the indices of the SLM pixels $m = \lfloor m'/P \rfloor$ and $n = \lfloor n'/P \rfloor$. The matrices $T^i_{s,t}$ with indices $s = \mathrm{mod}(m', P)$ and $t = \mathrm{mod}(n', P)$, each correspond to the crosstalk caused by the eight neighbouring pixels, each containing $P \times P$ pixels, with the upscaling factor, P. Each pixel value in the arrays T^i is a learnable parameter. The difference between our model and the previous implementation [102] is that each pixel T^i is a free parameter. In the original model, the two-dimensional T^i were constructed from analytical one-dimensional transition functions with varying parameters depending on the pixel values of two neighbouring pixels. Model 2 is computationally efficient since no convolution is required and all upscaled SLM pixels can be calculated in parallel using a GPU. For both model 1c and model 2, we investigate how the upscaling factor, P, affects the performance of the model.

4.2.2 Optimised crosstalk model

Finding the correct parameters to model pixel crosstalk using model 1a has previously been achieved using an exhaustive search approach, where both parameters (width σ and order q) of the model were optimised to obtain the best agreement between a simulated image and the camera image [111]. This method is time-consuming as it requires running the phase retrieval algorithm and analysing the resulting camera image for each combination of parameters. For this reason, optimising models with a larger number of parameters (such as models 1b, 1c, and 2) is impractical.

We present a faster and more convenient method to find the optimal crosstalk parameters, similar to the approach used in Section 4.1 to calibrate the wavefront and the intensity of the laser beam. Instead of using smooth, semi-random SLM phase patterns, we generate entirely random phase patterns with uniformly distributed values between 0 and 2π for each SLM pixel. The effect of pixel crosstalk generated by the random phase patterns is especially noticeable since neighbouring pixels display starkly varying phase values. As the previously calibrated laser intensity profile and the wavefront are measured at the native resolution of the SLM, we upscale them by a factor P using Lanczos interpolation [128] to model the incident laser beam. After applying the crosstalk model to the phase displayed by the SLM, θ, the camera images are simulated by propagating the electric field from the SLM plane to the image plane via a Fourier transform (Fig. 4.1)

$$I = \left| \mathcal{F}\left\{ A e^{i(\varphi + \Theta)} \right\} \right|^2 . \tag{4.12}$$

Using optimisation by adaptive moment estimation, the parameters of the different pixel crosstalk models are found by optimising them together with the parameters of the affine transformation matrix, U, to minimise the mean-squared-error between the camera image and the simulated image,

$$C_{\mathrm{ct}} = s\, C_{\mathrm{MSE}}, \tag{4.13}$$

where C_{ct} is the cost used in the optimisation. In Section 2.6.3, we showed that the optical potential can move on the camera on a micrometre scale due to thermal drift and moving air. Generating a reference pattern on the camera (Section 2.10) might enable to adjust the affine transformation for each camera image individually instead of using a global affine transformation for all camera images, reducing translation errors of the image (Section 3.1).

We use $N_{\mathrm{F}} = 10$ phase patterns for the optimisation which converges after around 300 iterations ($\sim$ 4 minutes runtime with models 1a, 1b, and 1c and $\sim$ 2 minutes with model 2) with an upscaling factor of $P = 3$. For model 1a (equation 3.2), the

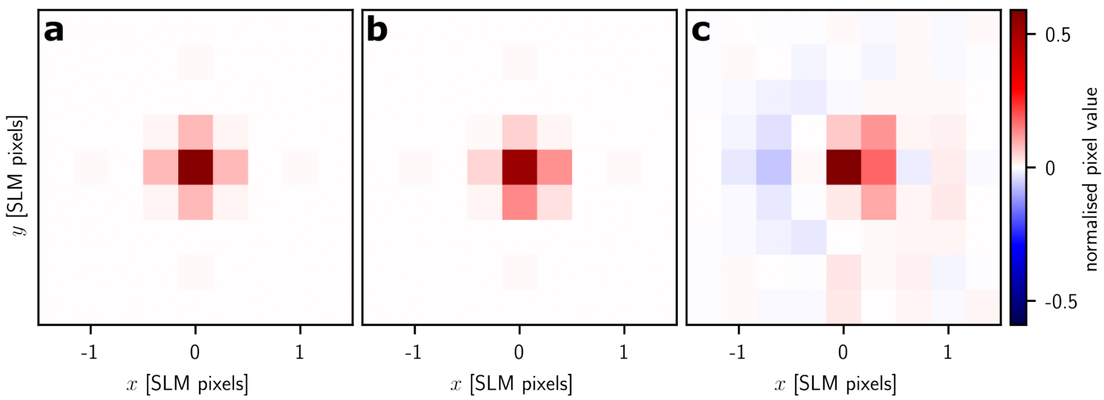


Figure 4.6: Optimised pixel crosstalk kernels using different models, normalised by their pixel sum [140]. (a) Kernel, K, in model 1a according to equation 3.2, (b) piecewise kernel, K_{pw}, in model 1b according to equation 4.10, and (c) the unconstrained kernel, K_{uc}, in model 1c, where each pixel is a learnable parameter.

parameters $q = 1.20$ and $\sigma = 2.03\,\mathrm{px}$ were found (Fig. 4.6a), where px is one SLM pixel (px = $12.5\,\mu$m). These values differ slightly from the previously found parameters (Section 3.3), however, we are using light of wavelength $\lambda = 670\,\mathrm{nm}$, whereas previously, $\lambda = 852\,\mathrm{nm}$ was used. Optimising model 1b (equation 4.10) resulted in the orders $q_{\mathrm{xp}} = 1.85$, $q_{\mathrm{xn}} = 1.15$, $q_{\mathrm{yp}} = 1.75$, $q_{\mathrm{yn}} = 1.12$ and $\sigma_{\mathrm{xp}} = 1.02p_{\mathrm{SLM}}$, $\sigma_{\mathrm{xn}} = 2.72p_{\mathrm{SLM}}$, $\sigma_{\mathrm{yp}} = 1.06p_{\mathrm{SLM}}$, $\sigma_{\mathrm{yn}} = 3.04p_{\mathrm{SLM}}$ (Fig. 4.6b). The kernels, K and K_{pw}, in models 1a and 1b were initialised with parameters $q = 2$ and $\sigma = 1\,\mathrm{px}$, based on values found in the literature for a similar LCOS SLM device [104, 111]. In model 1c, by optimising each pixel value in the kernel instead of the parameters of an analytic function, we obtain the kernel, K_{pw}, shown in figure 4.6c. The kernel in model 1c was initialised with the central pixel set to one and the remaining pixels set to zero. The spatial extent of the kernels in models 1a, 1b, and 1a was set to 3×3 SLM pixels.

For model 2, we vary the upscaling factor, P, to investigate if the error of the light potentials can be further reduced by using a larger upscaling factor (Fig. 4.7). At the

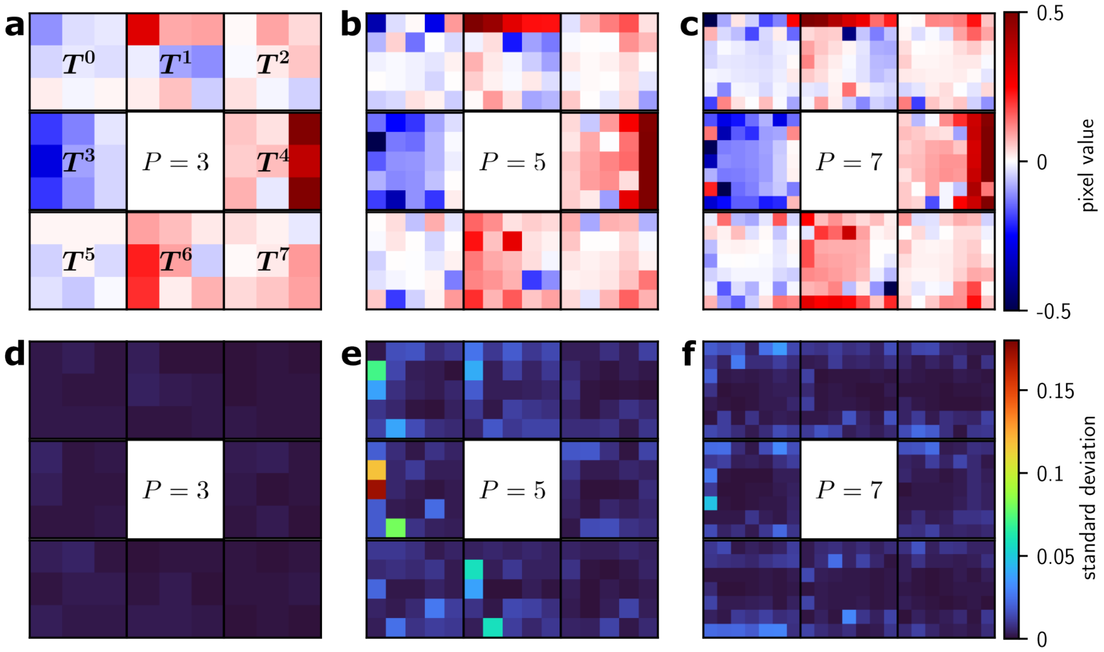


Figure 4.7: Optimised pixel crosstalk model 2 for various upscaling factors (Fig. 4.5) [140]. (a - c) Arrays T^i after running the optimisation using the upscaling factor $P = 3$, $P = 5$, and $P = 7$. (d - f) Standard deviation of arrays T^i for $P = 3$, $P = 5$, and $P = 7$ after performing the optimisation three times for each value of P using different sets of phase patterns and camera images.

start of the optimisation, each array T^i is initialised as an array of zeros. The structure of the optimised arrays T^i is similar for different values of P with a noticeable asymmetry along the x-axis (Fig. 4.7a-4.7c). These results suggest that the pixel crosstalk is affected most by the immediate left- and right-hand-side neighbours of each SLM pixel. A similar asymmetry along the x-axis can be observed in the optimised kernel of model 1c (Fig. 4.6c). The structure of each array T^i becomes more intricate when increasing the upscaling factor, P. To investigate the repeatability of these results, we performed the optimisation of model 2 three times for each of the upscaling factors, $P = 3$, $P = 5$, and $P = 7$ using different sets of random phase patterns and camera images during each run. The optimised models are very similar for $P = 3$, with the standard deviation in the arrays T^i not exceeding 1% (Fig. 4.7d). However, for larger upscaling factors, the standard deviation of individual pixels in the arrays T^i is increased due to the larger number of parameters (Fig. 4.7e and 4.7f). We conclude that the number of phase patterns used during the optimisation, $N_\mathrm{F} = 10$, is sufficient to train the crosstalk model since we obtain very similar results when using different training datasets.

4.3 Results

To benchmark the performance of the different pixel crosstalk models and to show that the stochastic approach to calibrate the SLM produces accurate light potentials, we generate a square, top-hat-shaped light potential using conjugate gradient minimisation (Section 2.5) and a camera feedback algorithm (Section 3). We define our target light potential in the image plane in the units of the Fourier pixel pitch, $p_\mathcal{F}$, in the computational image plane assuming twofold zero-padding of the electric field at the SLM, given by

$$p_\mathcal{F} = \lambda f \,/\, (N \cdot p_\mathrm{SLM}) = 670\,\mathrm{nm} \cdot 250\,\mathrm{mm} \,/\, (2048 \cdot 12.5\,\mu\mathrm{m}) \approx 6.54\,\mu\mathrm{m} \tag{4.14}$$

after performing the FFT, where the zero-padded SLM plane contains $N \times N$ pixels with the pixel pitch, p_SLM. The target light potential is a square of $600\,p_\mathcal{F}$ side length with a dark border of $100\,p_\mathcal{F}$ width, offset from the optical axis by $420\,p_\mathcal{F}$ along the x- and y-

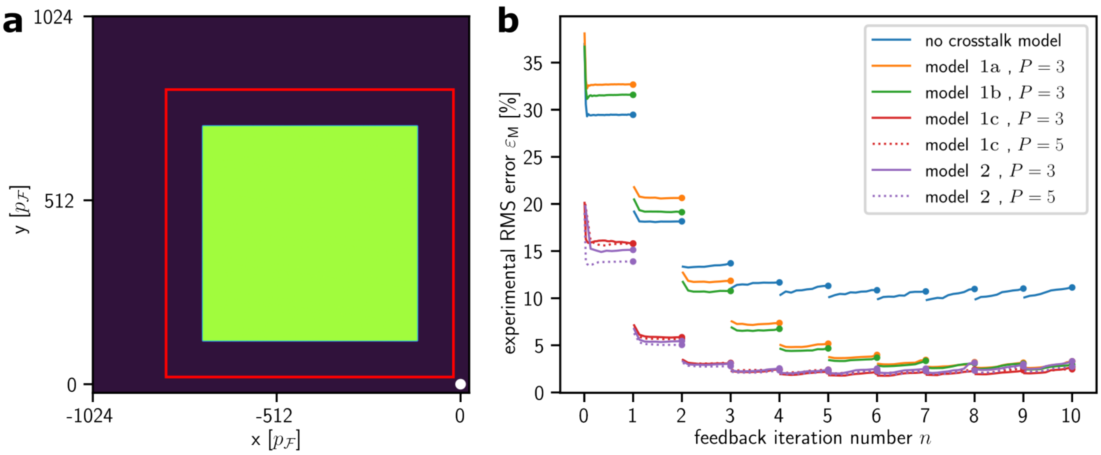


Figure 4.8: Benchmark of different crosstalk models used to generate a square target light potential (a) with the optical axis at $x = 0$, $y = 0$ (white dot) and a square signal region, S (red square) [140]. The maximum steering angle of the SLM corresponds to $\pm 1024\,\mathrm{p}_\mathcal{F}$, where $p_\mathcal{F}$ is the width of one Fourier pixel. (b) Convergence of the RMS error, ε_M, during the camera feedback process with the target potential in (a) as a function of feedback iterations, n, using various pixel crosstalk models and upscaling factors, P.

directions (Fig. 4.8a). We use a large light potential which occupies a significant fraction of the addressable area in the image plane (within the bounds of the first diffraction order). Diffraction angles approaching the maximum steering angle of the SLM, which corresponds to $\pm 1024\,p_\mathcal{F}$ in the computational image plane, must be displayed on the SLM to generate this pattern. The target pattern is convolved with a Gaussian kernel of $2\,p_\mathcal{F}$ width to avoid sub-diffraction-limited edges in the target light potential. As an initial phase guess, we use a linear phase to offset the potential from the optical axis by $480\,p_\mathcal{F}$ in the x- and y-directions and a quadratic phase term with $R = 1.6\,{}^{\mathrm{mrad}}/_{\mathrm{p_{SLM}}^2}$ (equation 2.11). Using the convergence of the camera feedback process, we evaluate the performance of the different crosstalk models (Fig. 4.8 and Fig. 4.9). Without modelling pixel crosstalk, the RMS error of the square, top-hat-shaped light potential reaches its minimum of 10.7% after 6 feedback iterations. Using model 1a, the error reduces to 3.2% after 8 feedback iterations. Model 1b performs similarly with an error of 2.9% after 9 feedback iterations. Interestingly, the error before any camera feedback ($n = 0$) using models 1a and 1b is slightly larger than without any crosstalk model (top row in Fig. 4.9 and Table 4.1). Model 1c results in significantly lower RMS error before any camera feedback (16% compared to 29%). Further, the error converges faster

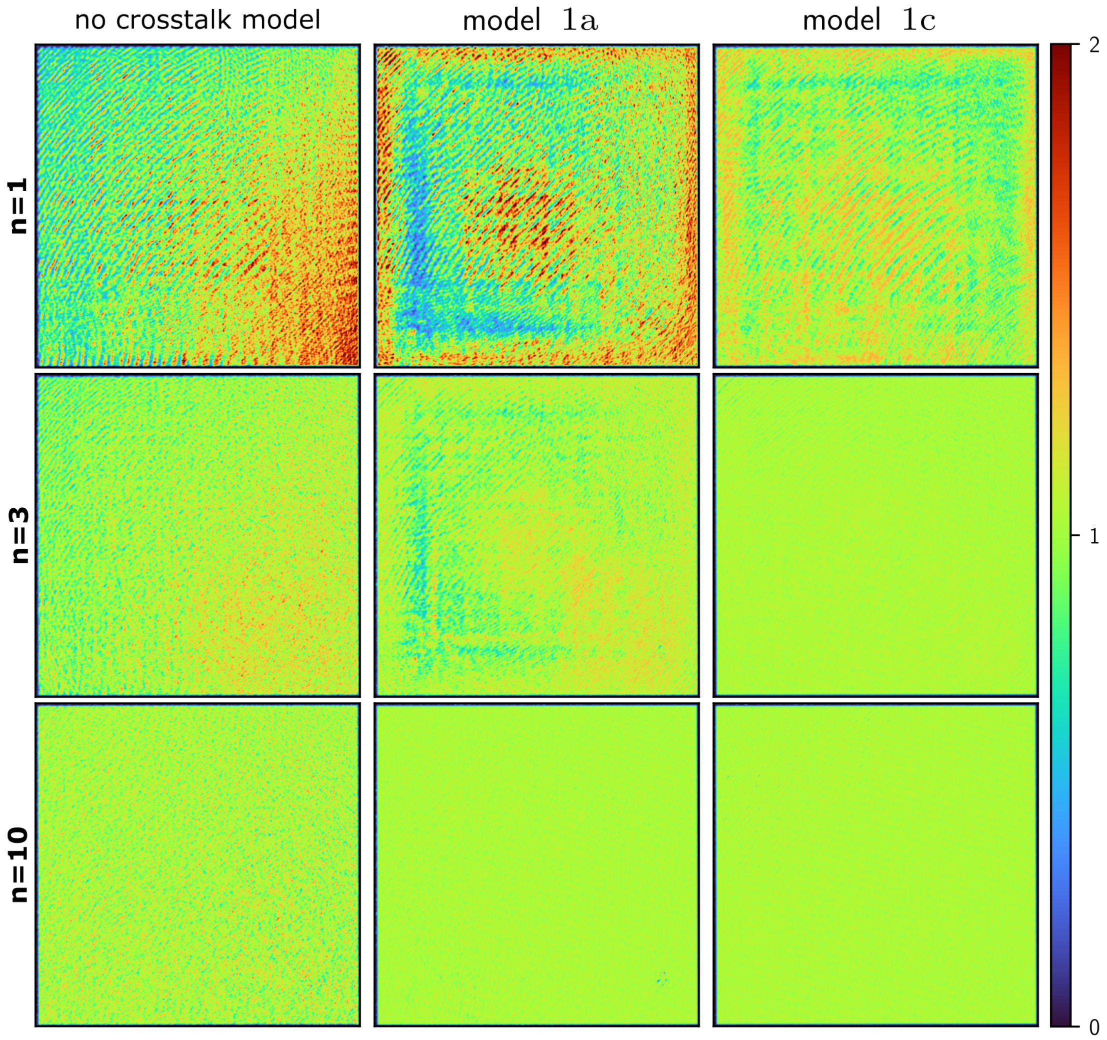


Figure 4.9: Zoom in on the central, flat region of the top-hat (Fig. 4.8a) after $n = 1$, $n = 3$ and $n = 10$ feedback iterations (left column) without pixel crosstalk model, (centre column) using model 1a and (right column) using the model 1c. The upscaling factor of the crosstalk models is $P = 3$. Using model 1c, the top-hat potential after $n = 3$ is flatter than the potential using model 1a after $n = 10$.

compared to model 1a and model 1b. Using model 1c, an error of 3.1% is reached after 3 feedback iterations which took 8 − 9 feedback iterations with model 1a and model 1b (middle row in Fig. 4.9). The lowest error reached with this kernel is 2.2% after 5 feedback iterations which is 30% lower compared to model 1a. Increasing the upscaling factor from $P = 3$ to $P = 5$ does not reduce the RMS error for this particular target potential further using model 1c. Model 2 converges to similar error levels as model 1c. Before any camera feedback, the RMS error is slightly lower (15% compared to

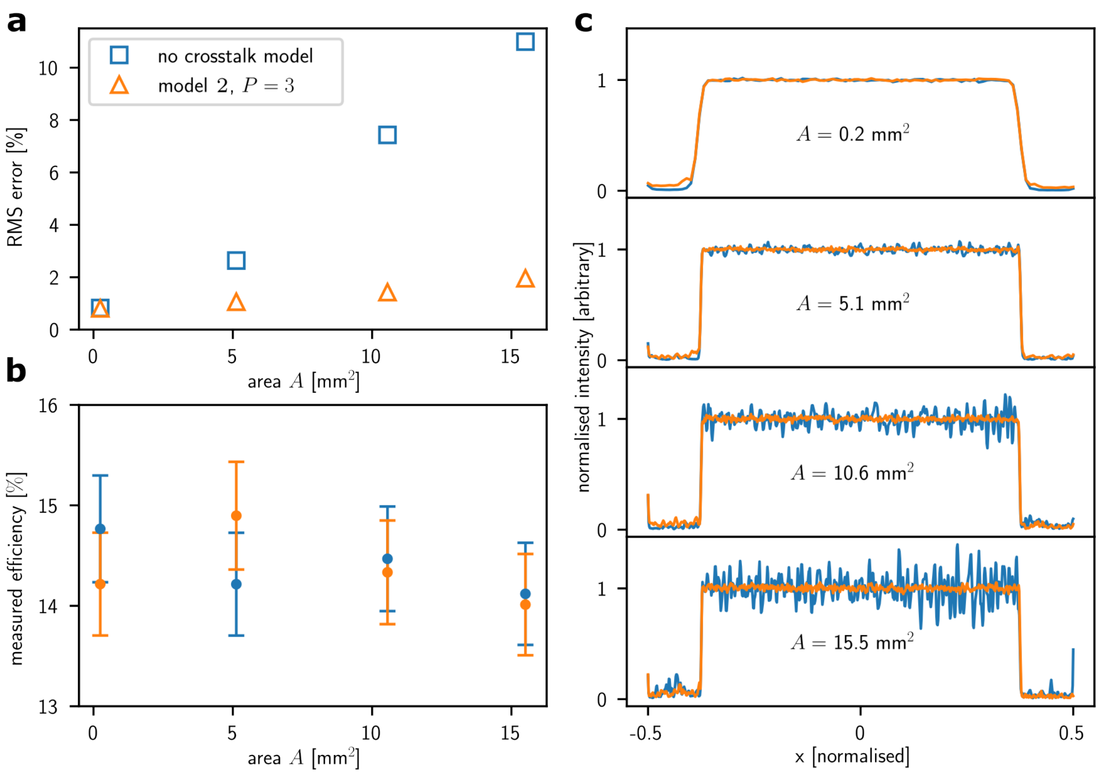


Figure 4.10: Varying the size of the top-hat-shaped light potential using 10 feedback iterations with (yellow) and without (blue) modelling pixel crosstalk. (a) The lowest experimental RMS error reached during the feedback process increases as a function of the area of the light potential, A. (b) Experimentally measured efficiency of the light potential as a function of the area of the light potential. (c) Profiles of the camera images through the centre of the square potential along the horizontal for varying sizes. The intensity is normalised by the mean value of the flat region of the top hat.

16%) and the lowest error of 2.4% is reached after 4 feedback iterations. Increasing the upscaling factor to $P = 5$ is beneficial using model 2 – the error before camera feedback is reduced to 14% and reaches 2.3% after 6 feedback iterations. However, we did not see a further reduction in RMS error when using $P = 7$.

We also investigated the experimental efficiency of the top-hat potentials generated using different crosstalk models (Table 4.1). Without modelling pixel crosstalk, the experimental efficiency of the top hat is $\sim 14\%$. When using model 1a and model 1b, the efficiency of the potential drops to $\sim 11\%$, which is consistent with our previous findings [111]. Model 1c and model 2 do not decrease the efficiency of the potentials significantly, resulting in an efficiency of $\sim 13\%$ and $\sim 14\%$, respectively. The efficiencies of the top hats are relatively low since a large value of R in the initial phase guess was necessary

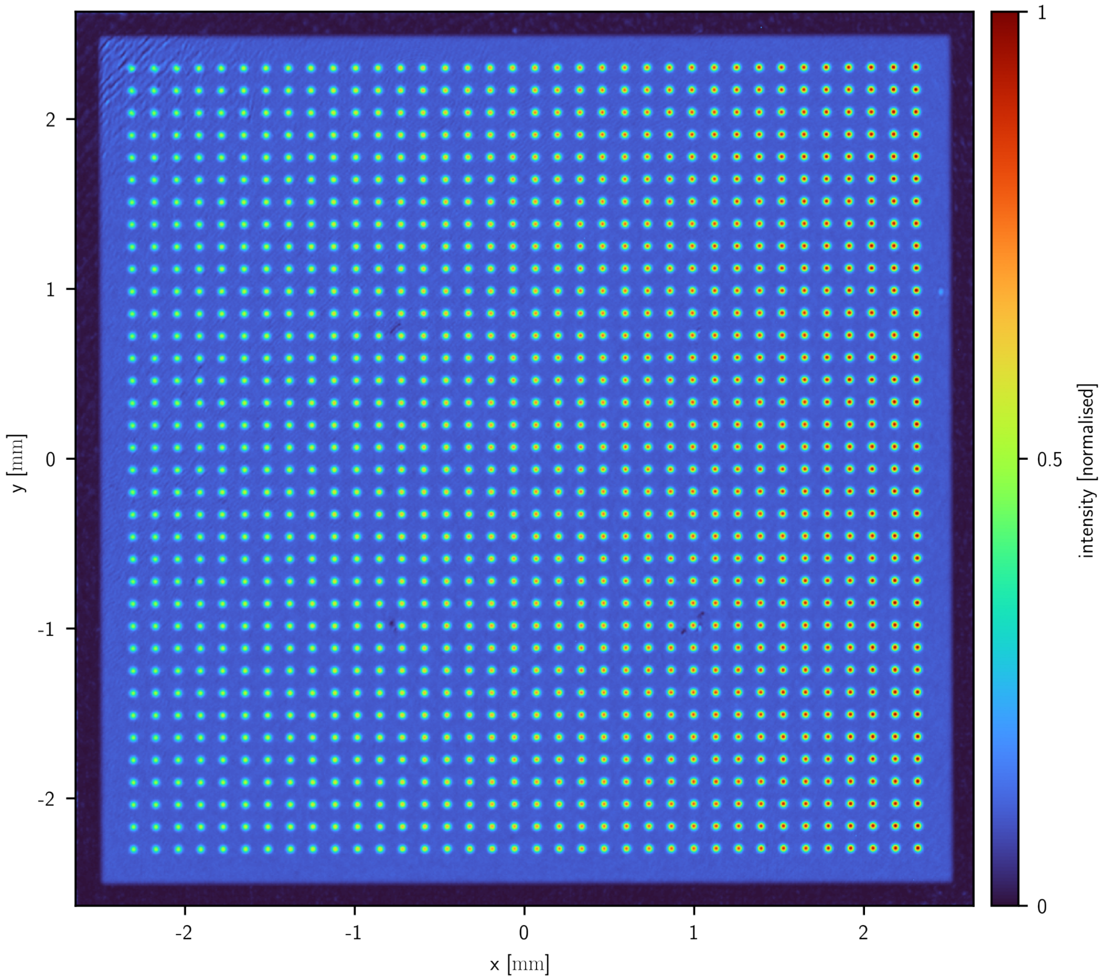


Figure 4.11: Camera image of a large spot array generated using pixel crosstalk model 2 with an upscaling factor of $P = 3$ after 6 feedback iterations [140].

to prevent the formation of optical vortices in the light potential during the feedback process. The large value of R causes a significant loss of optical power to areas outside the signal region. Employing a vortex-removal technique (Section 2.5.4) or adding a phase gradient term to the cost function when solving the phase retrieval problem [7] would allow using smaller values of R in the initial phase guess, increasing the efficiency of the light potential. In previous work [10], an efficiency term was added to the cost function, directly maximising the optical power in the potential.

To show the effect of pixel crosstalk at different sizes of the light potential, we vary the width of the square target potential from $80\,p_{\mathcal{F}}$ to $600\,p_{\mathcal{F}}$ and perform 10 camera feedback iterations for each square without modelling pixel crosstalk. To obtain similar efficiencies for the differently sized potentials, we vary the curvature, R, of the

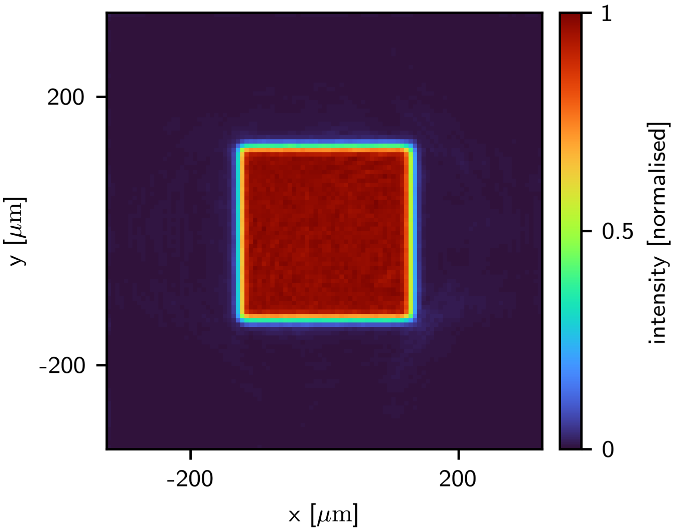


Figure 4.12: Camera image of a square top-hat potential after 8 camera feedback iterations [140]. After the optimisation, the light potential is $\eta_{\mathrm{M}} = 39\%$ efficient with an error of $\varepsilon_{\mathrm{M}} = 0.9\%$.

initial phase guess linearly with the area of the potential, from $R = 0.2\,{}^{\mathrm{mrad}}/_{\mathrm{p_{SLM}}^2}$ to $R = 1.6\,{}^{\mathrm{mrad}}/_{\mathrm{p_{SLM}}^2}$ (Fig. 4.10b). We then repeat this measurement using the crosstalk model 2 with an upscaling factor of $P = 3$ (Fig. 4.10). The RMS error increases in a linear manner as a function of the light potential's area for both sets of measurements, however, with different slopes (Fig. 4.10a). For the smallest square potential with an RMS error of $\sim$ 0.8%, modelling pixel crosstalk does not reduce the RMS error. The RMS error of the largest square potential is reduced by a factor of 5.5 from $\sim$ 11% without modelling pixel crosstalk to $\sim$ 2.0% using crosstalk model 2. Since we image the SLM in the Fourier plane, a larger light potential requires higher spatial frequencies on the SLM, increasing the number of 0 to 2π phase jumps in the SLM phase pattern which are particularly affected by pixel crosstalk. By modelling pixel crosstalk, however, the effect of these phase jumps is taken into account in the simulation and is compensated for by the conjugate gradient minimisation.

To demonstrate that our method can generate light potentials relevant for cold-atom experiments, we generate a square, uniform array of 1296 Gaussian spots on a small, constant background which is $760\,\mathrm{p}_{\mathcal{F}}$ wide. The constant background is surrounded by a zero-intensity region which is $20\,\mathrm{p}_{\mathcal{F}}$ wide. The Gaussian spots are spaced $20\,\mathrm{p}_{\mathcal{F}}$ apart and are $2\,\mathrm{p}_{\mathcal{F}}$ wide with increasing amplitudes along the x-axis. After 6 camera

Pattern	Fig.	A [mm^2]	Crosstalk model	P	ε_{M} [%] n=1	n=3	lowest	η_{M} [%]
Top-hat	4.8 4.9	15.5	-	1	29.5	13.7	10.7	14
			1a	3	32.7	11.8	3.2	11
			1b		32.6	10.8	2.9	11
			1c	3	15.8	3.1	2.2	13
				5	15.8	3.1	2.3	14
			2	3	15.1	3.1	2.4	14
				5	13.9	2.8	2.3	14
	4.10	0.2	-	1	4.9	1	0.8	15
			2	3	7.5	1	0.8	14
		5.1	-	1	17.6	3.9	2.6	14
			2	3	9.6	1.4	1.1	15
		10.6	-	1	23.7	9.2	7.4	14
			2	3	11.7	2.1	1.4	14
		15.5	-	1	29.3	13.4	11	14
			2	3	12.9	2.9	2	14
Spot array	4.11	24.7	2	3	10	2.1	1.6	12
Top-hat	4.12	0.06			8.7	1.7	0.9	39

Table 4.1: The measured efficiency, η_{M}, and the RMS error of the camera image, ε_{M}, of light potentials with various sizes (area A) and shapes are shown at different points in the camera feedback process (feedback iteration n), generated using various pixel crosstalk models for different upscaling factors, P [140].

feedback iterations using crosstalk model 2 with an upscaling factor of $P = 3$, the RMS error of the camera image reached $\varepsilon_{\mathrm{M}} = 1.6\%$. Artefacts are visible in the top left corner of the image (Fig. 4.11) which might stem from the large diffraction angles needed to reach this part of the image plane. A few optical vortices are present in the light potential which can be removed using the vortex removal process described in Section 2.5.4. The low measured efficiency of $\eta_{\mathrm{M}} = 12\%$ can be explained by the large diffraction angles needed as this light potential was displayed off-axis (the optical axis is close to the bottom right corner in Fig. 4.11), resulting in low diffraction efficiency. To demonstrate that our method can produce light potentials of high efficiency required to, for example, shape a laser beam used to excite atoms to the Rydberg state [10], we generated a small square top-hat potential (150 μm wide) and tuned the initial phase guess parameter, R, (Section 2.5.3) to optimise the efficiency. The resulting potential is $\eta_{\mathrm{M}} = 39\%$ efficient with an error of $\varepsilon_{\mathrm{M}} = 0.9\%$ (Fig. 4.12).

4.4 Discussion and outlook

The intensity profile and the wavefront measured using the stochastic approach presented here feature fewer artefacts compared to the results generated by the local sampling method, however, the resulting wavefront is flattened and the intensity profile is widened in comparison. This might be caused by minimising the gradient in the wavefront and the intensity using the smoothness terms C_{φ} and C_{A} which consequently flattens the results. Darker areas towards the edges of the SLM are especially impacted by this since the cost terms C_{φ} and C_{A} can be minimised in these areas without increasing the mean-squared-error term, C_{MSE}, decreasing the total cost, C. We argue that, since the mean-squared-error term is not affected much, the deviations in these areas are not detrimental to the accuracy of the holographic light potential. Optimising for the second derivative of the wavefront and the amplitude instead of the gradient will likely improve this issue. A different approach which does not require any derivative terms in the cost function was used in a recent study [6], where the constant field was modelled using smooth functions like Zernike polynomials for the wavefront and a sum of three Gaussians for the amplitude, however, this restricts the geometry of the wavefront and especially the amplitude. Using the Fourier transform of the Zernike polynomials [147] might provide a better parameterisation of the amplitude than a sum of Gaussians. The wavefront and intensity reconstructed using the calibration method presented here contains fewer artefacts and might be more accurate than the local sampling method within the $1/e^2$ intensity region, however, we cannot verify this as we do not have a more accurate wavefront reference. Methods for absolute characterisation of aberrations without a reference optic exist, however, they require additional equipment such as expensive polarisation optics [148]. The spherical wave diffracted from a sub-diffraction-limited pinhole provides an accurate wavefront reference in point-diffraction interferometers [149]. Placing such a pinhole in the Fourier plane of the lens in our setup might enable an absolute characterisation of the optical system. Using a type-2 non-uniform FFT allowed us to perform the Fourier transform of 20 different SLM phase patterns in parallel on a GPU due to the lower memory footprint compared to a regular FFT. However, the non-uniform FFT is significantly slower than the FFT and is not required when using a lower number of phase patterns, where the memory requirements are lower. Using an FFT further reduces the runtime of the calibration process from ~ 7 minutes to ~ 2 minutes with a number of $N_{\mathrm{F}} = 5$ phase patterns.

Our method to calibrate the pixel crosstalk model allows the use of more sophisticated models with a large number of parameters, reducing the error in large light potentials by $\sim 30\%$ compared to previous work with increased light usage efficiency [111]. Further, the time taken to calibrate pixel crosstalk model 2 is only ~ 2 minutes. However, since our method simply reduces the mean-squared error between the camera image and the simulation, the optimised pixel crosstalk model might correct for experimental effects in the camera image which are not caused by pixel crosstalk and are not accounted for in our simulation, for example, a slightly tilted Fourier lens.

We did not calibrate the phase response of the SLM (gray level to phase lookup table), since the SLM we used in this study was calibrated by the manufacturer to have a linear phase response. However, previous studies have shown that the phase response can vary spatially on the SLM as well as with the diffraction angle [6, 124]. Additionally, there might be alignment errors and aberrations of the Fourier lens which are currently compensated by the calibrated wavefront at the SLM. At larger diffraction angles, however, modelling aberrations at the SLM and at the Fourier lens separately might be beneficial and could further reduce the error in the light potentials. Currently, we use two separate optimisation processes to calibrate the constant field at the SLM and the pixel crosstalk model since we use different kinds of phase patterns at the SLM to calibrate each effect. Combining both of these calibrations into one algorithm would further reduce the time taken to calibrate the SLM which will be the subject of future work.

Chapter 5

Conclusion

To summarise, we demonstrated the experimental generation of holographic light potentials with an RMS error of less than 1% and an experimentally measured efficiency of $\sim 40\%$. We achieved these results by solving the phase retrieval problem using gradient-based optimisation [95] on a GPU with PyTorch [121] using automatic differentiation. This optimisation approach provides great flexibility in modelling the propagation of light from the SLM to the image plane. We actively remove unwanted optical vortices from the light potentials which enables to realise intricate vortex-free potentials and simplifies finding a suitable phase guess to seed the conjugate gradient phase retrieval algorithm. Since we optimise a simulated light potential, the RMS error of the experimentally measured potential is not limited by the convergence of the simulation but rather by how accurately the experimental system is modelled during the optimisation. In quantum simulation and quantum computing experiments, lower errors in the light potentials lead to more precise control over the atomic states, increasing the fidelity of the measurements. In atom array experiments, higher efficiency allows for the generation of more tweezers with the same amount of laser power, enabling larger-scale experiments and potentially increasing the number of qubits available for quantum computing. Additionally, these experiments benefit from highly uniform and efficient top-hat potentials to globally drive the Rydberg excitation across the entire atomic array. A reduced error in the generated top-hat potentials might lead to higher fidelities of entangled gates, an important step towards implementing error correction protocols [150].

By measuring the wavefront and the intensity profile at the SLM and by modelling pixel crosstalk, we reduced the 'reality gap' between the simulation and the experiment, lowering the error in the resulting light potentials. We improved a scheme to calibrate the wavefront at the SLM, making it robust to small pointing fluctuations of the laser beam caused by thermal drifts and by vibrations due to moving air, reducing the recalibration error from $\sim \lambda/40$ to $\sim \lambda/120$. This reduction in recalibration error means that any aberrations in the optical system can be corrected more accurately, enabling diffraction-limited light potentials which are especially significant for atom array experiments requiring tight trapping tweezers. In optical lattice experiments, diffraction-limited light potentials are crucial to achieve site resolved local control since the lattice spacing and therefore the smallest distance between two neighbouring atoms is typically close to the diffraction limit of the optical system.

We developed a method to calibrate the wavefront, the intensity profile and the pixel crosstalk model in ~ 5 minutes, significantly improving on the combined runtime of the previously used calibration methods, which took ~ 3 hours. The reduction in calibration time has a significant impact on the practicality of cold-atom experiments using one or even multiple LCOS SLMs. It allows for more frequent recalibrations, ensuring that the system remains accurately aligned and aberration corrected, reducing the impact of, for example, thermal drifts. Additionally, the faster calibration process enables more rapid iteration and testing of different experimental configurations, accelerating the overall research and development process. Further, our calibration approach is quite general, and it might be possible to extend it to more complicated optical systems involving, for example, high-NA lenses or additional diffractive optical elements.

Using a simple camera feedback algorithm, the RMS error in the light potential is further reduced [112]. We developed a novel chequerboard method to calibrate the coordinate transform between the camera and the computational image plane. The chequerboard calibration we use is insensitive to artefacts caused by pixel crosstalk, making it more robust and enabling to accurately calibrate the camera across its entire field of view. We investigated pixel crosstalk on the SLM and showed that in a Fourier imaging setup, the RMS error of the light potential increases in a linear manner with the area of the light potential. By modelling pixel crosstalk using a novel, computationally efficient method, we greatly reduced the error in large light potentials that occupy a significant fraction of the addressable area in the image plane. For these large light potentials, pixel crosstalk is still a limiting factor and further work is needed to develop more accurate models of pixel crosstalk.

We found that for smaller light potentials which do not require large diffraction angles on the SLM, the sensor uniformity and noise of the feedback camera and the pointing stability of the laser were limiting factors. By shielding the experiment from airflow and by using a low-noise sCMOS camera with high sensor uniformity, we lowered the RMS error of small top-hat potentials to $\sim 0.7\%$. By performing feedback on time-averaged camera images, the RMS error of the time-averaged potential reaches 0.35%. We conclude that the error can be further lowered by eliminating temporal variations in the light potentials which might arise from phase flicker on the SLM and residual pointing instability of the laser beam.

We used a high-NA microscope objective to generate light potentials on a microscopic scale required for cold-atom experiments. The RMS error of the resulting potentials is ~ 3 times larger compared to the macroscopic potentials. This might stem from an increased discrepancy between the simulation and the experimental setup since we approximate the propagation of light with a Fourier transform. A more sophisticated simulation of the propagation of light through the high-NA objective such as the angular spectrum method might reduce the error in the microscopic potentials. Recent progress in this field greatly reduced the computational requirements to run the angular spectrum method for high-NA optics [99, 100], and the implementation of these methods is the subject of future work. Additionally, extending the calibration approach from Chapter 4 to learn aberrations from multiple optical elements in the setup might allow us to further reduce the error of the microscopic potentials.

In addition to cold-atom experiments, the work in this thesis is relevant for a broad range of applications. Holographically generated tweezer arrays can also be used in quantum computation and simulation experiments with ultracold molecules, a field that has seen rapid progress in recent years [151]. To scale up atom and molecular arrays, reducing the error in the tweezer arrays will become increasingly important

[151]. Applications in microscopy can benefit from aberration correction of the optical system and from illuminating the sample with structured light using an SLM, increasing the imaging resolution [152, 153]. Aberration correction in holographic lithography used in biofabrication [154] or to manufacture photonic crystals [155] can decrease the smallest possible feature size of the printed structures. The ability to generate light potentials of arbitrary shape and low error can speed up the lithography process as multiple regions in the target material can be addressed simultaneously. Further, a recent publication demonstrated that holographic near-eye displays for augmented and virtual reality glasses achieve improved image quality by modelling pixel crosstalk [9]. For those applications, the novel pixel crosstalk model proposed in Chapter 4 can lower the computational requirements for holographic displays and improve their image quality.

Appendix A

Angular spectrum method

We implement the ASM to simulate the propagation of light in our CG minimisation. First, the electric field at the SLM plane, $E_{\mathrm{SLM}}(x, y)$, is propagated to the lens plane and multiplied by the aperture, $A_{\mathrm{L}}(x, y)$, and phase, $\phi_{\mathrm{L}}(x, y)$, of the lens using the relation [110]

$$E(x, y, z_{\mathrm{L}}) = \mathcal{F}^{-1}\Big\{\mathcal{F}\left\{E_{\mathrm{SLM}}(x, y)\right\} H\big(\kappa_x', \kappa_y', z_{\mathrm{L}}\big)\Big\} A_{\mathrm{L}}(x, y) \exp\left[i\phi_{\mathrm{L}}(x, y)\right]. \tag{A.1}$$

Here, κ_x' and κ_y' are the spatial frequencies, $E\left(x, y, z_{\mathrm{L}}\right)$ is the electric field in the lens plane just after the lens and z_{L} is the distance between the SLM plane and the lens plane. $A_{\mathrm{L}}(x, y) = \mathrm{circ}(r)$ is the circular aperture of the lens with radius r and $\phi_{\mathrm{L}}(x, y)$ is the phase delay caused by the lens. The transfer function, $H\left(\kappa_x', \kappa_y', \Delta z\right)$, is given by [110]

$$H\big(\kappa_x', \kappa_y', \Delta z\big) = \begin{cases} \exp\left[2\pi i \frac{\Delta z}{\lambda}\sqrt{1 - \left(\lambda\kappa_x'\right)^2 - \left(\lambda\kappa_y'\right)^2}\right] & \text{if } \sqrt{\kappa_x'^2 + \kappa_y'^2} < \frac{1}{\lambda}. \\ 0 & \text{otherwise.} \end{cases} \tag{A.2}$$

with propagation distance, Δz. The resulting electric field, $E(x, y, z_{\mathrm{L}})$, is then propagated to the image plane using [110]

$$E(x, y, z_{\mathrm{I}}) = \mathcal{F}^{-1}\left\{\mathcal{F}\left\{E(x, y, z_{\mathrm{L}})\right\} H\big(\kappa_x', \kappa_y', z_{\mathrm{I}} - z_{\mathrm{L}}\big)\right\}, \tag{A.3}$$

where $E(x, y, z_\mathrm{I})$ is the resulting electric field in the image plane and $\Delta z = z_\mathrm{I} - z_\mathrm{L}$ is the distance between the lens plane and the image plane (Fig. 1a).

Using the ASM instead of the Fourier transform enables to model the lens accurately. Specifically, we use a doublet lens with three spherical surfaces (Thorlabs ACT508-250-B) which causes a phase delay [110]

$$\phi_\mathrm{L}(x,y) = \frac{2\pi}{\lambda}\left[\Delta_{12}(x,y)(n_1-1) + \Delta_{23}(x,y)(n_2-1)\right], \tag{A.4}$$

where Δ_{12} and Δ_{23} are the lens thicknesses and $n_1 = 1.59847$ and $n_2 = 1.76182$ the refractive indices of the crown and the flint glass [156, 157], respectively. The lens thicknesses are given by [110]

$$\Delta_{ab}(x,y) = -R_a\left(1-\sqrt{1-\frac{x^2+y^2}{R_a^2}}\right) + R_b\left(1-\sqrt{1-\frac{x^2+y^2}{R_b^2}}\right) \tag{A.5}$$

with the radii of the spherical surfaces $R_1 = 137.7\,\mathrm{mm}$, $R_2 = -R_1$ and $R_3 = -930.4\,\mathrm{mm}$. The phase of the doublet, $\phi_\mathrm{L}(x,y)$, deviates from the idealised phase of the lens [110]

$$\phi_\mathrm{Q}(x,y) = -\frac{\pi}{\lambda f}\left(x^2+y^2\right), \tag{A.6}$$

with the focal length, $f = 250\,\mathrm{mm}$, by $2.8\,\lambda$ (peak-to-valley) across the aperture of the lens (48.3 mm).

In our numerical implementation, we pad the array representing the SLM field with zeros to match the size of the SLM plane with the aperture of the lens used in our experiment. This increases the computational complexity as the matrix size increases from 2048×2048 to 3864×3864. When using the FFT, the matrix representing the SLM plane of 1024×1024 pixels is zero-padded to 2048×2048 pixels, resulting in a pixel spacing $p_\mathrm{IMG} = \frac{\lambda f}{2Np_\mathrm{SLM}} = 8.32\,\mu\mathrm{m}$ in the image plane, with the number of SLM pixels, N, in each dimension and SLM pixel pitch, p_SLM. With the ASM, the pixel size in the SLM plane equals the pixel size in the image plane. To achieve a similar spatial resolution in the output plane using the ASM, each SLM pixel of $12.5\,\mu\mathrm{m}$ size is sub-resolved computationally into 2×2 pixels which increases the number of pixels to 7728×7728. We use a GPU (Nvidia RTX A5000 24 GB) to accelerate our calculations.

A.1 ASM wavefront correction

Our method to measure the constant phase, φ_{C}, requires the lens to be parabolic and assumes perfect placement of the lens and the camera. Using equation 2.15, the measured phase, $\phi_{\mathrm{M}}(x,y)$, includes the phase difference caused by the distorted wavefront at the SLM and the phase differences caused by a non-parabolic lens and a displacement of the camera along the optical axis. As the ASM is capable of modelling the doublet lens and a displaced camera, it is important to separate these phase differences and the wavefront at the SLM, $\phi_{\mathrm{C}}(x,y)$, from each other.

To implement the ASM, we calculate a corrective phase, $\phi_{\mathrm{ASM}}(x,y)$, which only models the phase caused by the displaced, non-parabolic lens and the displaced camera, assuming a flat wavefront at the SLM. To do so, we calculate the path length of every sample beam between the lens and a fixed point in the image plane as well as the phase delay each sample beam collects when passing through the lens, $\phi_{\mathrm{L}}\big(x_s(x)\,,y_s(y)\big)$.

$$\phi_{\mathrm{ASM}}(x,y) = \frac{2\pi}{\lambda}\sqrt{\big[z_{\mathrm{I}} - z_{\mathrm{L}}\big]^2 + \big[x_s(x) - x_c\big]^2 + \big[y_s(y) - y_c\big]^2} + \phi_{\mathrm{L}}\big(x_s(x)\,,y_s(y)\big), \tag{A.7}$$

with the position of the sample beam on the lens $x_s(x) = x + z_{\mathrm{L}}\tan(\alpha_x)$ and $y_s(y) = y + z_{\mathrm{L}}\tan(\alpha_y)$, where α_x and α_y are the diffraction angles of the linear phase gradient in x- and y-direction, respectively. The phase is sampled at a point in the image plane with co-ordinates $x_c = f\tan(\alpha_x)$ and $y_c = f\tan(\alpha_y)$. We then subtract the corrective phase pattern, $\phi_{\mathrm{ASM}}(x,y)$, from the measured constant phase to obtain the wavefront at the SLM, $\phi_{\mathrm{C}}(x,y) = \phi_{\mathrm{M}}(x,y) - \phi_{\mathrm{ASM}}(x,y)$.

Appendix B

HoloGradPy Python package

The majority of the code used to generate holographic light potentials in Chapter 2 and Chapter 3 has been published on GitHub as part of the Python package *HoloGradPy*, available at `https://github.com/paul-schroff/hologradpy`. This appendix contains the corresponding documentation which has been generated automatically from the code using *Sphinx*. The code documentation can also be found online at `https://hologradpy.readthedocs.io`. *HoloGradPy* and its documentation will be updated in the future to include the code used to generate the results in Chapter 4.

HoloGradPy

Release 1.0

Paul Schroff

Jun 12, 2024

USER GUIDE

HoloGradPy provides functionality to holographically generate light potentials of arbitrary shape using a phase-modulating SLM (see our publication).

HoloGradPy, Release 1.0

CHAPTER ONE

AUTHOR

This package was created by Paul Schroff during his PhD at the University of Strathclyde in the research group of Stefan Kuhr.

Note: For questions or suggestions, email paul.schroff@strath.ac.uk.

CHAPTER

TWO

FEATURES

To calculate the SLM phase pattern for a given target light potential, we implemented a phase-retrieval algorithm based on conjugate gradient minimisation. The gradient used in the conjugate gradient minimisation is calculated using PyTorch's automatic differentiation capabilities.

Functions to characterise the beam profile and the constant phase at the SLM are provided. These measurements are crucial for accurate experimental results.

We employed a feedback algorithm and model pixel crosstalk on the SLM to further reduce experimental errors in the light potentials.

Note: This package works best with a Nvidia GPU to run the phase retrieval algorithm.

Warning: This documentation is work in progress - refer to the *example scripts* to get started.

2.1 Installation

Download this package and `cd` into the folder containing the `setup.py` file. Then, in your virtual conda environment, run the command

```
pip install -e .
```

All required python packages should install automatically. To run PyTorch on a GPU, you have to install CUDA.

2.2 Example scripts

2.2.1 Setting up your hardware

This script sets constant experimental parameters and implements camera and SLM drivers. You will have to do this for your own hardware.

```
import time
import numpy as np
from holograpy import hardware as hw

# These modules are only needed for our camera and SLM drivers
```

(continues on next page)

(continued from previous page)

```
import cv2 as cv
from harvesters.core import Harvester
from screeninfo import get_monitors
import imageio
```

Setting experimental parameters and other constant parameters

To start off, we write our own subclass of `hardware.ParamsBase` which sets some experimental parameters and other constants needed in this script.

```
class Params(hw.ParamsBase):
    wavelength = 670e-9         # Wavelength [m]
    beam_diameter = 7.25e-3     # Diameter of incident beam [m]
    fl = 0.25                   # Focal length [m]

    data_path = '../../holography_data/'

    # Path to measured constant SLM phase and intenstiy
    phi_path = data_path + '23-09-13_13-20-49_measure_slm_wavefront/dphi_uw.npy'
    i_path = data_path + '23-09-13_11-47-42_measure_slm_intensity/i_rec.npy'

    phi_filter_size = 2
    crop = 64


pms_obj = Params()
```

Implementing SLM and camera drivers

This module provides the `hardware.CameraBase` class and the `hardware.SlmDisp` class to interface with the camera and the SLM. You will have to write your own subclasses for the specific devices you are using. Here, we defined the subclasses `Camera` and `SlmDisp` to interface with a MatrixVision BlueFox 3 camera and a Hamamatsu SLM. Make sure you implement all abstract methods of `hw.CameraBase` and `hw.SlmBase` in your own subclasses.

```
class Camera(hw.CameraBase):
    def __init__(self, res, pitch, name='before', roi=None, gain=0, bayer=False):
        if roi is None:
            roi = [1280, 960, 0, 0]
        self.roi = roi
        super().__init__(res, pitch, self.roi)
        self.count = 0
        self.gain = gain
        self.bayer = bayer
        self.name = name
        self.h = None
        self.ia = None
        self.bayer_slope = 0.010487497932442524
        self.bayer_offset = 2.195178550143578
        self.max_frame_count = 2 ** 16 - 1
```

(continues on next page)

(continued from previous page)

```
    def start(self, n=1):
        if n >= self.max_frame_count:
            n = self.max_frame_count

        self.h = Harvester()

        self.h.add_file('C:/Program Files/MATRIX VISION/mvIMPACT Acquire/bin/x64/
↪mvGenTLProducer.cti')

        self.h.update()

        print("start init ia")

        serial_numbers = []
        for info in self.h.device_info_list:
            serial_numbers.append(info.serial_number)
        if self.name == 'before':
            n_cam = serial_numbers.index('F0600075')
        if self.name == 'after':
            n_cam = serial_numbers.index('F0600086')
        self.ia = self.h.create(n_cam)
        self.ia.remote_device.node_map.ExposureAuto.value = 'Off'
        self.ia.remote_device.node_map.mvLowLight.value = 'Off'
        self.ia.remote_device.node_map.ExposureAuto.value = 'Off'
        self.ia.remote_device.node_map.BlackLevelAuto.value = 'Off'
        self.ia.remote_device.node_map.GainAuto.value = 'Off'
        self.ia.remote_device.node_map.ExposureTime.value = 100
        self.ia.remote_device.node_map.PixelFormat.value = 'Mono16'
        self.ia.remote_device.node_map.AcquisitionMode.value = 'MultiFrame'
        self.ia.remote_device.node_map.AcquisitionFrameRateEnable.value = True
        self.ia.remote_device.node_map.AcquisitionFrameRate.value = 12
        if self.name == 'before':
            self.ia.remote_device.node_map.ReverseX.value = True
            self.ia.remote_device.node_map.ReverseY.value = True
        elif self.name == 'after':
            self.ia.remote_device.node_map.ReverseX.value = False
            self.ia.remote_device.node_map.ReverseY.value = True
        self.ia.remote_device.node_map.TriggerMode.value = 'On'
        self.ia.remote_device.node_map.TriggerSource.value = 'Software'
        self.ia.remote_device.node_map.TriggerSelector.value = 'FrameStart'

        w, h, dx, dy = self.roi

        self.ia.remote_device.node_map.AcquisitionFrameCount.value = n
        self.ia.remote_device.node_map.Gain.value = self.gain

        if dx <= self.ia.remote_device.node_map.OffsetX.value:
            self.ia.remote_device.node_map.OffsetX.value = dx
            self.ia.remote_device.node_map.OffsetY.value = dy
            self.ia.remote_device.node_map.Width.value = w
            self.ia.remote_device.node_map.Height.value = h
        else:
```

(continues on next page)

HoloGradPy, Release 1.0

(continued from previous page)

```
            self.ia.remote_device.node_map.Width.value = w
            self.ia.remote_device.node_map.Height.value = h
            self.ia.remote_device.node_map.OffsetX.value = dx
            self.ia.remote_device.node_map.OffsetY.value = dy

        self.ia.start()
        print("start acquisition")

    def get_image(self, exp_time_):
        if self.count >= self.max_frame_count:
            self.stop()
            self.start(n=self.max_frame_count)

        self.ia.remote_device.node_map.ExposureTime.value = exp_time_

        self.ia.remote_device.node_map.TriggerSoftware.execute()

        with self.ia.fetch() as buffer:
            component = buffer.payload.components[0]
            width = component.width
            height = component.height

            im = np.array(component.data.reshape(height, width)).astype(np.double)

        if self.bayer is True:
            im[0::2, 1::2] = im[0::2, 1::2] * (1 + self.bayer_slope) + self.bayer_offset
            im[1::2, 0::2] = im[1::2, 0::2] * (1 + self.bayer_slope) + self.bayer_offset
        self.count += 1
        return im

    def stop(self):
        self.ia.stop()
        print("stopped acquisition")
        self.ia.destroy()
        self.h.reset()


class SlmDisp(hw.SlmBase):
    def __init__(self, res, pitch, calib=None, delay=0.2, dx=0, dy=0):
        super().__init__(res, pitch)
        self.max_phase = 2 * np.pi  # Largest value for phase wrapping
        self.slm_norm = 128         # Gray level on the SLM corresponding to max_phase
        # Gray level vs phase lookup table
        self.lut = np.load(pms_obj.data_path + '23-02-17_13-49-14_calibrate_grey_values/
↪phase.npy')
        self.idx_lut = np.argmin(np.abs(self.lut - self.max_phase))  # Index of max_
↪phase in lut
        self.lut = self.lut[:self.idx_lut]
        # Path to Hamamatsu SLM correction pattern.
        self.cal_path = pms_obj.data_path + 'deformation_correction_pattern/CAL_
↪LSH0802439_' + '{:.0f}'.\
            format(np.around(pms_obj.wavelength * 1e9, decimals=-1)) + 'nm.bmp'
```

(continues on next page)

(continued from previous page)

```
        self.delay = 0.2  # Time to wait after displaying a phase pattern on the SLM [s]

        if calib == 1 or calib is True:
            self.calib_flag = True
            self.calib = imageio.imread(self.cal_path)
            self.calib = np.pad(self.calib, ((0, 0), (0, 8)))
        elif calib == 0 or calib is False or calib is None:
            self.calib_flag = False
            self.calib = np.zeros((self.res[1], self.res[0]))
        self.delay = delay
        self.dx = dx
        self.dy = dy

        monitor = get_monitors()[-1]

        cv.namedWindow('screen', cv.WINDOW_NORMAL)
        cv.resizeWindow('screen', self.res[1], self.res[0])
        cv.moveWindow('screen', monitor.x, monitor.y)
        cv.setWindowProperty('screen', cv.WND_PROP_FULLSCREEN, cv.WINDOW_FULLSCREEN)
        cv.waitKey(100)
        print("SlmDisp initialised")

    def display(self, phi_slm):
        im_res_y, im_res_x = phi_slm.shape
        slm_res_y, slm_res_x = self.res
        slm_pad_x = (slm_res_x - im_res_x) // 2
        slm_pad_y = (slm_res_y - im_res_y) // 2

        slm_norm = self.slm_norm

        if slm_pad_x == 0 and slm_pad_y == 0:
            phi_zeros = slm_norm * phi_slm / (2 * np.pi)
        elif -slm_pad_y - self.dy == 0:
            phi_zeros = np.zeros((slm_res_y, slm_res_x))
            phi_disp = slm_norm * phi_slm / (2 * np.pi)
            phi_zeros[slm_pad_y - self.dy:, slm_pad_x - self.dx:-slm_pad_x - self.dx] =␣
↪phi_disp
        elif -slm_pad_x - self.dx == 0:
            phi_zeros = np.zeros((slm_res_y, slm_res_x))
            phi_disp = slm_norm * phi_slm / (2 * np.pi)
            phi_zeros[slm_pad_y - self.dy:-slm_pad_y - self.dy, slm_pad_x - self.dx:] =␣
↪phi_disp
        else:
            phi_zeros = np.zeros((slm_res_y, slm_res_x))
            phi_disp = slm_norm * phi_slm / (2 * np.pi)
            phi_zeros[slm_pad_y - self.dy:-slm_pad_y - self.dy, slm_pad_x - self.dx:-slm_
↪pad_x - self.dx] = phi_disp

        if self.calib_flag is False:
            phi_zeros = phi_zeros.astype('uint8')
        else:
            phi_zeros = np.remainder(phi_zeros + self.calib, slm_norm).astype('uint8')
```

(continues on next page)

HoloGradPy, Release 1.0

(continued from previous page)

```
        cv.imshow('screen', phi_zeros)
        cv.waitKey(1)
        time.sleep(self.delay)
```

We can now use the classes `Params`, `Camera` and `SlmDisp` in other scripts.

2.2.2 Feedback algorithm example

This script calculates phase patterns for a phase-modulating liquid crystal on silicon (LCOS) spatial light modulator (SLM) to create accurate light potentials by modelling pixel crosstalk on the SLM and using conjugate gradient (CG) minimisation with camera feedback (see https://doi.org/10.1038/s41598-023-30296-6).

Using this script, it should be easy to switch between the different patterns from our publication, turn on pixel crosstalk modelling and switch between the fast Fourier transform (FFT) and the angular spectrum method (ASM) to model the propagation of light.

Importing modules

```
import os
import time
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable

from hologradpy import patterns as p
from hologradpy import error_metrics as m
from hologradpy import calibrate_slm as clb
from hologradpy import torch_functions as tfn

from examples.experiment import Params, Camera, SlmDisp
```

Here, we determine which computing hardware to use (CPU or GPU) and create instances from our custom hardware classes.

```
device = tfn.check_device(verbose=True)   # Check for GPU

pms_obj = Params()
cam_obj = Camera(np.array([960, 1280]), 3.75e-6, bayer=True)    # Create instance of␣
↪camera class
slm_disp_obj = SlmDisp(np.array([1024, 1280]), 12.5e-6)         # Create instance of SLM␣
↪class
```

Initializing the camera feedback algorithm

Parameters for the phase-retrieval algorithm:

```
npix = 1024                # Number of pixels on SLM (npix * npix)
propagation_type = 'fft'   # Propagation Type
optimizer = 'cg'           # Optimizer
loss_fn = 'amp'            # Loss function used during optimization
fft_shift = True           # Perform FFT shift?
precision = 'single'       # Computational precision
pixel_crosstalk = False    # Model pixel crosstalk?
pix_res = 1                # Subsampling factor of each SLM pixel
detect_vortices = False    # Detect vortices before the first camera feedback iteration?
threshold_vtx = 0.05       # Vortices are only detected in regions brighter than␣
↪threshold (1 is maximum)

# Path containing a previously calculated affine transform to calibrate the camera.
tf_path = pms_obj.data_path + '23-08-29_18-42-53_torch_camcal/'

calc_transform = True           # Calculate new transform?
measure_slm_intensity = False   # Measure the constant intensity at the SLM (laser beam␣
↪profile)?
measure_slm_phase = False       # Measure the constant phase at the SLM?
```

Parameters for the initial SLM phase guess, the target light potential and the signal region:

```
guess_type = 'guess'                                        # Use analytical phase␣
↪guess
phase_angle = int(-npix // 4)                               # Offset of the target␣
↪light potential to the optical
                                                            # axis in x- and y-
↪direction in Fourier pixels to
                                                            # calculate the gradient␣
↪of linear phase.
linear_phase = np.array([phase_angle + 2, phase_angle - 2])  # Linear term of the␣
↪initial phase guess
quad_phase = np.array([4.7e-4, 0.5])                        # Quadratic term of the␣
↪initial phase guess

# Target Pattern
pattern = 'spot_array'                                      # Name of the target␣
↪light potential
mask_pos = int(phase_angle)                                 # Offset of the target␣
↪light potential to the optical
                                                            # axis in x- and y-
↪direction in Fourier pixels
target_width = int(npix // 2)                               # Size of the target␣
↪light potential
target_blur = 2                                             # Width of the blurring␣
↪kernel for the target light
                                                            # potential
```

Parameters for the camera feedback algorithm:


```
cam_name = 'before'                    # Name of camera
slm_disp_type = 'lut'                  # SLM display mode
iter_fb = 10                           # Number of camera feedback iterations
iter_cg = 50 * np.ones(iter_fb)        # Number of CG iterations per feedback iteration
alpha = np.ones(iter_fb)               # Feedback gain parameter
exp_time = 200                         # Exposure time of camera in microseconds
n_frames_avg = 10                      # Number of camera pictures taken to average
feedback_blur = 0                      # Size of Gaussian blurring for camera feedback
```

Defining the blurring kernel to model pixel crosstalk:

```
if pixel_crosstalk is True:
    extent = 3                         # Extent of crosstalk kernel in SLM pixels
    q = 2.3                            # Crosstalk kernel order
    sigma = 0.92 / slm_disp_obj.pitch  # Crosstalk kernel width
    kernel_ct = p.pixel_ct_kernel(slm_disp_obj.pitch, pix_res, extent, q, sigma)
else:
    kernel_ct = None
```

Inputs for the angular spectrum method:

```
# Number of pixels of zero-padded SLM plane
if propagation_type == 'asm':
    npix_pad = int(pms_obj.lens_aperture // pms_obj.slm_pitch)
else:
    npix_pad = 2 * npix

npix_tot = npix_pad * pix_res                   # Total number of pixels (npix_tot *␣
↪npix_tot)
extent_lens = npix_pad * slm_disp_obj.pitch     # Spatial extent of computational lens␣
↪plane [m]
pd1 = pms_obj.fl                                # Distance from SLM plane to lens plane␣
↪[m]
pd2 = pms_obj.fl                                # Distance from lens plane to camera␣
↪plane [m]
```

Determine which data to save.

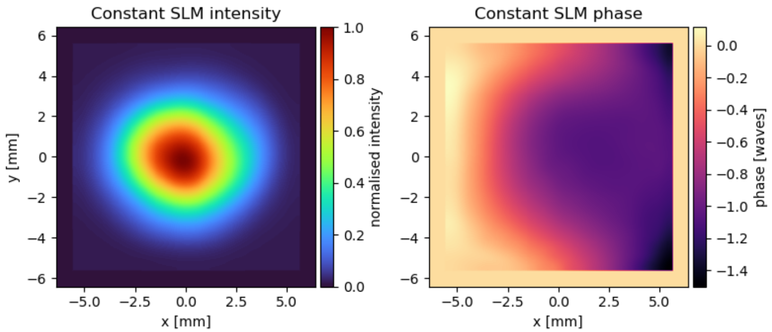
```
save = False                     # Save camera images?
convergence = False              # Save convergence of CG algorithm?
n_save = 5                       # Save every xx th CG iteration
iter_plot = [1, 2, 13, 14, 15]   # List of feedback iterations to save CG convergence

# Create folder to save data
date_saved = time.strftime('%y-%m-%d_%H-%M-%S', time.localtime())
path = pms_obj.data_path + date_saved + '_' + os.path.splitext(os.path.basename(__file__
↪))[0] + '_' + pattern
os.mkdir(path)
```

Measuring the constant intensity and phase at the SLM

Measuring the constant intensity and phase at the SLM is crucial for accurate experimental results - see the supplementary information of our publication for details.

```
if measure_slm_intensity is True:
    i_path = clb.measure_slm_intensity(slm_disp_obj, cam_obj, pms_obj, 30, 32, 10000,␣
↪256, 300)
    pms_obj.i_path = i_path
if measure_slm_phase is True:
    phi_path = clb.measure_slm_wavefront(slm_disp_obj, cam_obj, pms_obj, 30, 16, 64,␣
↪40000, 256, roi_min_x=2,
                                         roi_min_y=2, roi_n=26)
    pms_obj.phi_path = phi_path
```

Using the functions above, this is our constant field at the SLM after upscaling it to the native resolution of the SLM:



Defining the target light potential

The `patterns.Hologram` class contains pre-defined patterns from our publication. It creates

- the upscaled measured constant SLM phase and intensity,
- the initial SLM phase guess,
- the target intensity pattern,
- and the signal region.

Feel free to define the arrays above yourself - using the `patterns.Hologram` class is not mandatory.

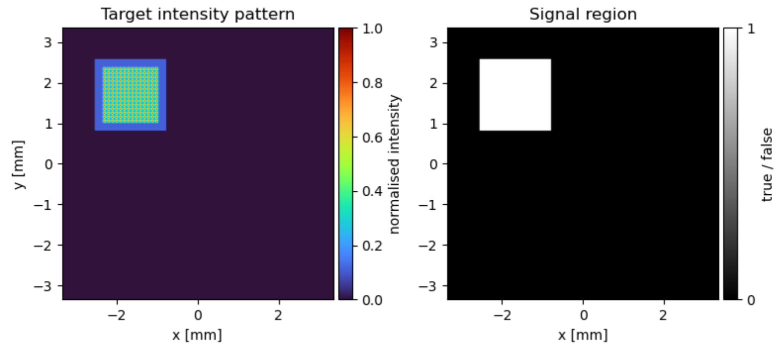
```
holo = p.Hologram(slm_disp_obj, pms_obj, pattern, npix, npix_pad=npix_pad, pix_res=pix_
↪res, phase_guess_type=guess_type,
                  linear_phase=linear_phase, quadratic_phase=quad_phase, slm_field_type=
↪'measured',
```

(continues on next page)

(continued from previous page)

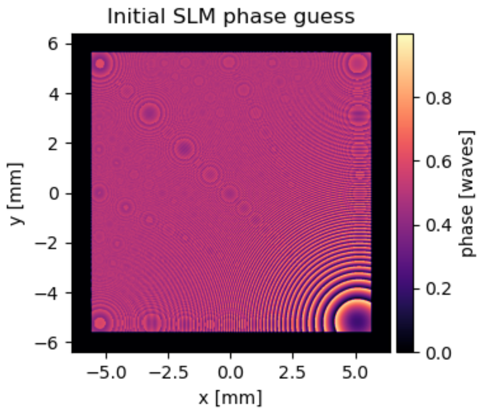
```
                  propagation_type=propagation_type, target_position=mask_pos, target_
↪width=target_width,
                  target_blur=target_blur)
```

Here is our target light potential, a Gaussian spot array, and the signal region:



The target is shifted away from the center to avoid the zeroth order diffration spot. The phase retrieval algorithm only optimises for the intensity inside the signal region.

We use an analytic initial SLM phase guess consisting of a quadratic and a linear phase term. The quadratic phase term depends on the size and the aspect ratio of the target pattern while the linear term depends on the position of the pattern with respect to the optical axis. The initial phase guess defined here looks like this:



Creating a virtual SLM object

This is a digital twin of the experimental Fourier holography setup. The `forward` method of `VirtualSlm` takes an SLM phase pattern, models pixel crosstalk on the SLM and the propagation of light from the SLM to the camera. It returns the electric field at the image plane.

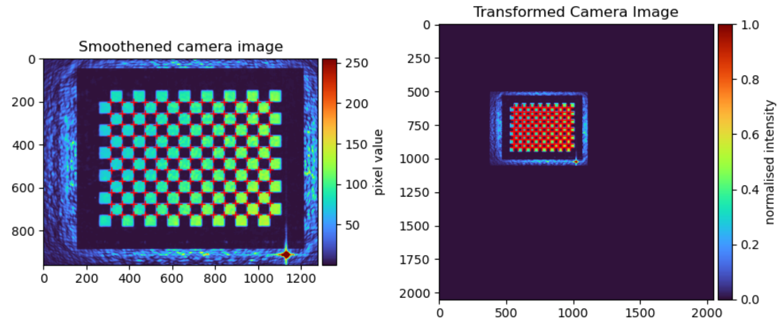
```
# Create SLM mask to set unused pixels to zero
slm_mask = np.zeros((npix, npix))
slm_mask[pms_obj.crop:-pms_obj.crop, pms_obj.crop:-pms_obj.crop] = 1

# Pixel pitch in the Fourier plane (padded) [m]
img_pitch = pms_obj.wavelength * pms_obj.fl / (slm_disp_obj.pitch * slm_disp_obj.res[0]␣
↪* 2)
xf = -256 * img_pitch  # ToDO: Explain this.

# Create virtual SLM object
slm_obj = tfn.VirtualSlm(slm_disp_obj, pms_obj, holo.phi_init, npix_pad, npix=npix, e_
↪slm=holo.e_slm,
                         kernel_ct=kernel_ct, pix_res=pix_res, propagation_
↪type=propagation_type,
                         extent_lens=pms_obj.lens_aperture, pd1=pd1, pd2=pd2, xf=xf,␣
↪device=device, slm_mask=slm_mask,
                         precision=precision, fft_shift=fft_shift).to(device)
```


Camera calibration

Here, we calculate the affine transformation matrix between camera coordinates and image plane coordinates. This is important to compare the simulated light potential to the captured camera image.

```
if calc_transform is False:
    tf = np.load(tf_path + 'tf.npy')
    itf = np.load(tf_path + 'itf.npy')
else:
    # ToDO: Control over checkerboard position missing.
    tf, itf = tfn.camera_calibration(slm_obj, slm_disp_obj, cam_obj, pms_obj, save=True,␣
↪exp_time=1000,
                                     checkerboard_rows=16, checkerboard_columns=12,␣
↪checkerboard_square_size=30)
```

This is the result:



Note that the zeroth-order diffraction spot is now located in the center of the computational image plane on the right hand side.

Running the camera feedback algorithm

First, we create an object from `torch_functions.PhaseRetrieval` which sets the phase retrieval method. By default, `torch_functions.PhaseRetrieval` performs conjugate gradient minimisation using an amplitude-only cost function (see https://doi.org/10.1364/OE.22.026548).

This phase retrieval method is then used iteratively by the camera feedback algorithm (see https://dx.doi.org/10.1088/0953-4075/48/11/115303).

Before running the camera feedback algorithm, we set the phase of the virtual SLM , `slm_obj`, with the initial phase guess. The phase pattern of `slm_obj` might have been modified by the `torch_functions.camera_calibration` function.

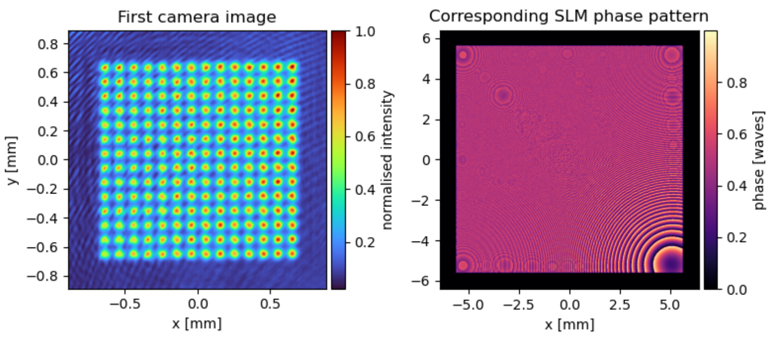
```
phase_retrieval_obj = tfn.PhaseRetrieval(slm_obj, n_iter=int(iter_cg[0]), i_tar=holo.i_
↪tar, signal_region=holo.sig_mask,
                                         save=convergence, n_save=n_save)
```

(continues on next page)

(continued from previous page)

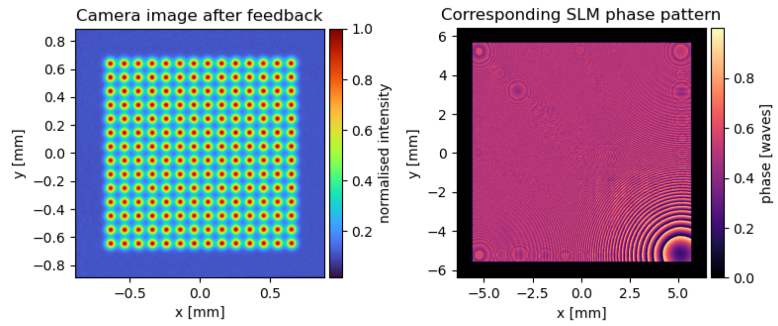
```
if propagation_type == 'asm':
    # Modify the initial phase pattern on our virtual SLM if the ASM is used.
    slm_obj.set_phi(holo.phi_init - slm_obj.asm_obj.phi_q_native)
else:
    phase_retrieval_obj.slm_obj.set_phi(holo.phi_init)

# Run camera feedback algorithm
output = tfn.camera_feedback(phase_retrieval_obj, slm_disp_obj, cam_obj, tf, itf, iter_
↪fb=iter_fb, iter_cg=iter_cg,
                             detect_vortices=detect_vortices, threshold_vtx=threshold_
↪vtx, n_save=n_save,
                             n_avg=n_frames_avg, exp_time=exp_time, fb_blur=feedback_
↪blur, alpha=alpha,
                             convergence=convergence, iter_convergence=iter_plot,␣
↪path=path)

phi, img, M, T, [rmse, psnr], [rmse_conv_sv, rmse_pred_conv_sv, eff_conv_sv, n_conv_sv]␣
↪= output
```

After the first 50 CG iterations, the optimised SLM phase pattern is displayed on the device and a camera image is taken:



Here, we only show the signal region on the camera. The experimental errors in the camera image are greatly reduced after 10 camera feedback iterations with 50 CG iterations each:

HoloGradPy, Release 1.0

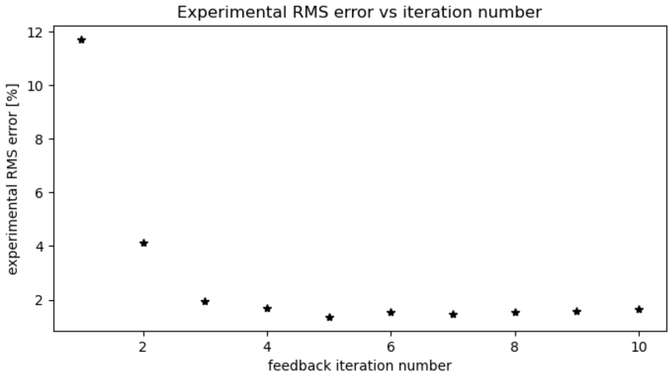



```
# Transfer electric field in the image plane to CPU
e_out = tfn.gpu_to_numpy(slm_obj())

# Calculate intensity pattern of the simulated light potential
i_out = np.abs(e_out) ** 2

# Calculate phase pattern of simulated light potential
phi_out = np.angle(e_out)

# Calculate efficiency
eff = m.eff(holo.sig_mask, i_out)
print('Efficiency of the simulation:', eff * 100, '%')
```

Plotting

```
px = 1 / plt.rcParams['figure.dpi']
fig0, axs0 = plt.subplots(figsize=(800*px, 400*px))
plt.plot(np.arange(1, iter_fb + 1), rmse * 100, 'k*', label='RMS error')
plt.title('Experimental RMS error vs iteration number')
plt.xlabel('feedback iteration number')
plt.ylabel('experimental RMS error [%]')

plt.figure()
plt.plot(psnr, 'go', label='PSNR')
plt.title('Experimental PSNR vs iteration number')
plt.xlabel('experimental iteration number')
plt.ylabel('PSNR [dB]')
```

We can now plot the rms error of the camera images after each feedback iteration:



The feedback algorithm lowered rms error from ~12 % to ~1.6 %.

```
plt.figure()
plt.imshow(i_out, cmap='turbo')
plt.title('Simulated light potential')

plt.figure()
plt.imshow(phi_out, cmap='magma')
plt.title('Phase of simulated light potential')

plt.figure()
plt.imshow(img[..., -1].squeeze(), cmap='turbo')
plt.title('Camera image')
plt.savefig(path + '//img.pdf', dpi=1200)

target_norm = T[..., 0].squeeze() * tfn.camera_feedback.sig_mask_tf
mask_target = target_norm > 0.1 * np.max(target_norm)
target_norm = target_norm / np.sum(target_norm[mask_target])

img_norm = img[..., 3].squeeze() * mask_target
img_norm = img_norm / np.sum(img_norm)
diff_img = (img_norm - target_norm) * mask_target

plt.figure()
plt.imshow(diff_img, cmap='seismic', vmin=-np.max(np.abs(diff_img)), vmax=np.max(np..
↪abs(diff_img)))
```

We can investigate the convergence of the phase retrieval algorithm in-between feedback iterations by saving intermediate phase patterns, displaying them on the SLM and capturing the resulting camera image. This allows us to see when the rms error of the camera image converges to determine the number of CG iterations needed per feedback iteration.

```
# Plot and save convergence graphs
if convergence is True:
```

(continues on next page)

(continued from previous page)

```
    plt.figure('rmse')
    x = min(iter_plot)
    for i in range(len(iter_plot)):
        plt.figure('rmse')
        x = np.linspace(iter_plot[i] - 1 + 1 / iter_cg[iter_plot[i] - 1],
                        iter_plot[i] - 1 + n_conv_sv[i] * n_save / iter_cg[iter_plot[i] -
↪ 1], n_conv_sv[i])
        line_exp, = plt.plot(x, rmse_conv_sv[i] * 100, '-', color='C0')
        line_pred, = plt.plot(x, rmse_pred_conv_sv[i] * 100, '--', color='r')

        plt.figure('eff')
        plt.plot(x, eff_conv_sv[i] * 100, '-', color='C1')

    line_exp.set_label('experiment')
    line_pred.set_label('predicted')

    plt.figure('rmse')
    plt.plot(np.arange(1, iter_fb + 1), rmse * 100, 'k*', label='RMS within 50% of max.␣
↪intensity')
    plt.legend()
    plt.xlabel('CG iterations')
    plt.ylabel('RMS error [%]')
    plt.grid()
    plt.savefig(path + '//rmse.pdf', bbox_inches='tight', dpi=600)

    plt.figure('eff')
    plt.xlabel('CG iterations')
    plt.ylabel('Predicted efficiency [%]')
    plt.grid()
    plt.savefig(path + '//efficiency.pdf', bbox_inches='tight', dpi=600)
```

Saving data

Saving data to the hard drive.

```
if save is True:
    np.save(path + '//lin_phase', linear_phase)
    np.save(path + '//quad_phase', quad_phase)
    np.save(path + '//tf', tf)
    np.save(path + '//itf', itf)
    np.save(path + '//T', T)
    np.save(path + '//npix', npix)
    np.save(path + '//npix_pad', npix_pad)
    np.save(path + '//pix_res', pix_res)
    np.save(path + '//M', M)
    np.save(path + '//img', img)
    np.save(path + '//phi', phi)
    np.save(path + '//prop', propagation_type)
    np.save(path + '//exp_time', exp_time)
    np.save(path + '//kernel_ct', kernel_ct)
```

(continues on next page)

(continued from previous page)

```
    np.save(path + '//rmse', rmse)
    np.save(path + '//eff', eff)
    np.save(path + '//psnr', psnr)

    np.save(path + '//rmse_conv_sv', rmse_conv_sv)
    np.save(path + '//rmse_pred_conv_sv', rmse_pred_conv_sv)
    np.save(path + '//eff_conv_sv', eff_conv_sv)
    np.save(path + '//n_conv_sv', n_conv_sv)
    np.save(path + '//iter_plot', iter_plot)

    np.save(path + '//a_tar', holo.a_tar)
    np.save(path + '//sig_mask', holo.sig_mask)
    np.save(path + '//n_save', n_save)
    np.save(path + '//iter_fb', iter_fb)
```

2.3 API Reference

This page contains auto-generated API reference documentation[1].

2.3.1 hologradpy

Submodules

hologradpy.calibrate_slm

Module to measure the constant amplitude and phase at the SLM.

Functions

find_camera_position(slm_disp_obj, cam_obj, pms_obj, ...)	This function generates a spot on the camera by displaying a circular aperture on the SLM containing a linear phase
get_aperture_indices(nx, ny, x_start, x_stop, y_start, ...)	This function calculates a grid of `nx * ny` rectangular regions in an array and returns the start and end indices
measure_slm_intensity(slm_disp_obj, cam_obj, pms_obj, ...)	This function measures the intensity profile of the laser beam incident onto the SLM by displaying a sequence of
measure_slm_wavefront(slm_disp_obj, cam_obj, pms_obj, ...)	This function measures the constant phase at the SLM by displaying a sequence of rectangular phase masks on the SLM.

[1] Created with sphinx-autoapi

Module Contents

`hologradpy.calibrate_slm.`**`find_camera_position`**(*slm_disp_obj*, *cam_obj*, *pms_obj*, *lin_phase*, *exp_time=100*, *aperture_diameter=25*, *roi=[500, 500]*)

This function generates a spot on the camera by displaying a circular aperture on the SLM containing a linear phase gradient. The position of the spot is found by fitting a Gaussian to the camera image.

Parameters

- **`slm_disp_obj`** – Instance of your own subclass of `hardware.SlmBase`
- **`cam_obj`** –
- **`pms_obj`** –
- **`npix`** – Number of used SLM pixels
- **`lin_phase`** – x and y gradient of the linear phase
- **`cam_name`** – Name of the camera to be used
- **`exp_time`** – Exposure time
- **`aperture_diameter`** – Diameter of the circular aperture
- **`roi`** – Width and height of the region of interest on the camera to remove the zeroth-order diffraction spot

Returns

x and y coordinates of the spot on the camera

`hologradpy.calibrate_slm.`**`get_aperture_indices`**(*nx*, *ny*, *x_start*, *x_stop*, *y_start*, *y_stop*, *aperture_width*, *aperture_height*)

This function calculates a grid of `nx * ny` rectangular regions in an array and returns the start and end indices of each region. All units are in pixels.

Parameters

- **`nx`** – Number of rectangles along x.
- **`ny`** – Number of rectangles along y.
- **`x_start`** – Start index for first rectangle along x.
- **`x_stop`** – End index for last rectangle along x.
- **`y_start`** – Start index for first rectangle along y.
- **`y_stop`** – End index for last rectangle along y.
- **`aperture_width`** – Width of rectangle.
- **`aperture_height`** – Height of rectangle.

Returns

List with four entries for the start and end index along x and y: [idx_start_y, idx_end_y, idx_start_x, idx_end_x]. Each list entry is a vector of length `nx * ny` containing the start/end index for each rectangle along x/y.

`hologradpy.calibrate_slm.`**`measure_slm_intensity`**(*slm_disp_obj*, *cam_obj*, *pms_obj*, *aperture_number*, *aperture_width*, *exp_time*, *spot_pos*, *roi_width*)

This function measures the intensity profile of the laser beam incident onto the SLM by displaying a sequence of rectangular phase masks on the SLM. The phase mask contains a linear phase which creates a diffraction spot on the camera. The position of the phase mask is varied across the entire area of the SLM and the intensity of each diffraction spot is measured using the camera. Read the SI of https://doi.org/10.1038/s41598-023-30296-6 for details.

Parameters

- **slm_disp_obj** – Instance of your own subclass of `hardware.SlmBase`.
- **cam_obj** – Instance of your own subclass of `hardware.CameraBase`.
- **aperture_number** – Number of square regions along x/ y.
- **aperture_width** – Width of square regions [px].
- **exp_time** – Exposure time.
- **spot_pos** – x/y position of the diffraction spot in th computational Fourier plane [Fourier pixels].
- **roi_width** – Width of the region of interest on the camera [camera pixels].

Returns

`hologradpy.calibrate_slm.`**`measure_slm_wavefront`**(*slm_disp_obj*, *cam_obj*, *pms_obj*, *aperture_number*, *aperture_width*, *img_size*, *exp_time*, *spot_pos*, *n_avg_frames=10*, *benchmark=False*, *phi_load_path=None*, *roi_min_x=16*, *roi_min_y=16*, *roi_n=8*)

This function measures the constant phase at the SLM by displaying a sequence of rectangular phase masks on the SLM. This scheme was adapted from this Phillip Zupancic's work (https://doi.org/10.1364/OE.24.013881). For details of our implementation, read the SI of https://doi.org/10.1038/s41598-023-30296-6.

Parameters

- **slm_disp_obj** – Instance of your own subclass of `hardware.SlmBase`.
- **cam_obj** – Instance of your own subclass of `hardware.CameraBase`.
- **pms_obj** – Instance of your own subclass of `hardware.Parameters`.
- **aperture_number** – Number of square regions along x/ y.
- **aperture_width** – Width of square regions [px].
- **img_size** – Width of the roi in the camera image [camera pixels].
- **exp_time** – Exposure time.
- **spot_pos** – x/y position of the diffraction spot in th computational Fourier plane [Fourier pixels].
- **n_avg_frames** – Number of camera frames to average per shot.
- **benchmark** (*bool*) – Load previously measured constant phase and display it on the SLM to check for flatness.
- **phi_load_path** – Path to previously measured constant phase.
- **roi_min_x** – Aperture column number to display the first phase mask.
- **roi_min_y** – Aperture row number to display the first phase mask.
- **roi_n** – Number of apertures to display (roi_n * roi_n), starting at roi_min_x, roi_min_y.

Returns

Path to measured constant phase at the SLM.

hologradpy.error_metrics

This module contains functions to characterise light potentials.

Functions

normalize(img, roi[, thres])	Normalises an image by dividing it by the pixel sum in a region of interest. Only pixels brighter than
fidelity(signal, a_tar, phi_tar, a_out, phi_out)	Calculate fidelity between two electric fields in a region of interest (signal region).
rms(signal, i_target, i_out[, frac])	Calculate normalised root-mean-squared error between two images inside a region of interest. Only pixels which are brighter
rms_phase(phi)	Calculates the root-mean-squared error of an image.
psnr(signal, i_target, i_out)	Calculates the peak signal-to-noise ratio between two images in a region of interest according to
eff(signal, i_out)	Calculates the predicted efficiency of a light potential by dividing the pixel sum in the signal region by

Module Contents

`hologradpy.error_metrics.`**`normalize`**(*img*, *roi*, *thres=0.5*)

Normalises an image by dividing it by the pixel sum in a region of interest. Only pixels brighter than thres * max(roi * img) are taken into account.

Parameters

- **img** – Input image.
- **roi** – Binary mask containing region of interest.
- **thres** – Pixel value threshold (see above).

Returns

Normalised image.

`hologradpy.error_metrics.`**`fidelity`**(*signal*, *a_tar*, *phi_tar*, *a_out*, *phi_out*)

Calculate fidelity between two electric fields in a region of interest (signal region).

Parameters

- **signal** – Binary mask containing the region of interest.
- **a_tar** – Target amplitude pattern.
- **phi_tar** – Target phase pattern.
- **a_out** – Amplitude of light potential.
- **phi_out** – Phase of light potential.

Returns

Fidelity.

holoGradpy.error_metrics.**rms**(*signal*, *i_target*, *i_out*, *frac=0.5*)

Calculate normalised root-mean-squared error between two images inside a region of interest. Only pixels which are brighter than `frac * max(i_target_norm)` are taken into account, where `i_target_norm` is the normalised target intensity pattern.

Parameters

- **signal** – Binary mask containing region of interest (signal region).
- **i_target** – Target intensity pattern.
- **i_out** – Intensity pattern of light potential.
- **frac** – Threshold as explained above.

Returns
Normalised rms error.

hologradpy.error_metrics.**rms_phase**(*phi*)

Calculates the root-mean-squared error of an image.

Parameters
phi – Phase pattern.

Returns
Root-mean-squared error.

hologradpy.error_metrics.**psnr**(*signal*, *i_target*, *i_out*)

Calculates the peak signal-to-noise ratio between two images in a region of interest according to https://doi.org/10.1364/OE.24.006249.

Parameters

- **signal** – Binary mask containing region of interest (signal region).
- **i_target** – Target intensity pattern.
- **i_out** – Intensity pattern of light potential.

Returns
Peak signal-to-noise ratio [dB].

hologradpy.error_metrics.**eff**(*signal*, *i_out*)

Calculates the predicted efficiency of a light potential by dividing the pixel sum in the signal region by the pixel sum in the entire pattern.

Parameters

- **signal** – Binary mask containing the signal region.
- **i_out** – Intensity pattern of the light potential.

Returns
Efficiency.

hologradpy.fitting

This module contains functions for curve fitting.

Classes

FitSine	This class is used in the wavefront measurement to fit a 2D sine to the interference pattern on the camera. The

Functions

tilt(xy, *args[, mask])	Fit function containing the first three Zernike polynomials of the form z = c0 + c1 * x + c2 * y + c3 * 2xy.
remove_tilt(img[, mask])	This function removes fits the first three Zernike polynomials (Piston and tilt) to an image and subtracts the
gaussian(xy, *args)	Gaussian fit function.
fit_gaussian(img[, dx, dy, sig_x, sig_y, a, c, ...])	Fits a 2D Gaussian to an image. The image s blurred using a Gaussian filer before fitting.

Module Contents

hologradpy.fitting.**tilt**(*xy*, **args*, *mask=None*)

Fit function containing the first three Zernike polynomials of the form z = c0 + c1 * x + c2 * y + c3 * 2xy.

Parameters

- **xy** – x, y coordinate vectors.
- **args** – Vector of length 4, containing Zernike coefficients.

Returns
First 3 Zernike polynomials.

hologradpy.fitting.**remove_tilt**(*img*, *mask=None*)

This function removes fits the first three Zernike polynomials (Piston and tilt) to an image and subtracts the fitted function from the original image. :param ndarray img: Input image. :param ndarray mask: Binary mask in which to remove tilt. :return: Image without tilt.

hologradpy.fitting.**gaussian**(*xy*, **args*)

Gaussian fit function.

Parameters

- **xy** – x, y coordinate vectors.
- **args** – Fitting parameters passed to patterns.gaussian.

Returns
Gaussian.

hologradpy.fitting.fit_gaussian(*img*, *dx=None*, *dy=None*, *sig_x=15*, *sig_y=15*, *a=None*, *c=0*, *blur_width=10*, *xy=None*)

Fits a 2D Gaussian to an image. The image s blurred using a Gaussian filer before fitting.

Parameters

- **img** – Input image.
- **dx** – X-offset of Gaussian [px].
- **dy** – Y-offset of Gaussian [px].
- **sig_x** – X-width of Gaussian [px].
- **sig_y** – -width of Gaussian [px].
- **a** – Amplitude.
- **c** – Offset.
- **blur_width** – Width of Gaussian blurring kernel [px].
- **xy** – X, Y meshgrid. If not specified, pixel coordinates are used.

Returns
Fitting parameters, parameter errors.

class hologradpy.fitting.**FitSine**(*fl*, *k*, *dx=None*, *dy=None*)

This class is used in the wavefront measurement to fit a 2D sine to the interference pattern on the camera. The distance between the sample and reference patch can be set by calling the method set_dx_dy.

set_dx_dy(*dx*, *dy*)

Method to set parameters dx and dy. :param dx: New dx. :param dy: New dy.

fit_sine(*xy*, **args*)

Method to perform 2D sine fit. :param xy: x, y coordinate vectors. :param args: Args passed to patterns.fringes_wavefront :return: 2D sine.

hologradpy.hardware

This module provides to interface with the camera and the SLM.

Classes

ParamsBase	Class storing experimental parameters and constant properties.
CameraBase	Helper class that provides a standard way to create an ABC using
SlmBase	Helper class that provides a standard way to create an ABC using

Functions

get_image_avg(cam_obj, exp_time, n_avg)	This function captures multiple camera images and calculates the average.

Module Contents

class hologradpy.hardware.**ParamsBase**

Bases: abc.ABC

Class storing experimental parameters and constant properties.

abstract property wavelength

Wavelength [m].

property k
Wavenumber [rad/m].

abstract property beam_diameter

Diameter of incident Gaussian beam [m].

abstract property fl

Focal length [m].

n1 = 1.59847

n2 = 1.76182

r1 = 0.1377

r2

r3

lens_aperture = 0.0483

abstract property phi_path

abstract property i_path

abstract property data_path

Path to store data.

crop = 32

phi_filter_size = 5

i_filter_size = 3

class hologradpy.hardware.**CameraBase**(*res*, *pitch*, *roi*)

Bases: abc.ABC

Helper class that provides a standard way to create an ABC using inheritance.

property cam_size

Calculates the physical size of the camera.

Returns
x, y dimensions of the camera [m].

abstract start(*n=1*)

You have to implement this yourself. Starts the acquisition.

Parameters
n – Number of frames to be captured.

abstract get_image(*exp_time*)

You have to implement this yourself. Acquires and returns a camera image of the shape determined by `self.roi`.

Parameters
exp_time – Exposure time.

Returns
Camera image of shape as defined by `self.roi`

abstract stop()

You have to implement this yourself. Stops the acquisition.

holagradpy.hardware.**get_image_avg**(*cam_obj*, *exp_time*, *n_avg*)

This function captures multiple camera images and calculates the average.

Parameters

- **cam_obj** – Instance of your own camera class which is a subclass of `CameraBase`.
- **exp_time** – Exposure time.
- **n_avg** – Number of frames to be averaged.

Returns
Averaged image.

class holagradpy.hardware.**SlmBase**(*res*, *pitch*)

Bases: `abc.ABC`

Helper class that provides a standard way to create an ABC using inheritance.

property slm_size

Calculates the physical size of the SLM.

Returns
x, y dimensions of the SLM [m].

abstract display(*phi*)

This function displays a phase pattern on the SLM. You have to implement this yourself.

Parameters
phi – SLM phase pattern [radians].

Returns

holagradpy.patterns

This module contains utility functions for array manipulation and functions to create binary masks, intensity and phase patterns of various shapes.

The *Hologram* class provides arrays needed for the CG minimisation: The target light potential and the signal region, the measured constant SLM phase and intensity at the required resolution, and the initial SLM phase guess to start the CG minimisation.

Classes

Hologram	This class provides arrays needed for the CG minimisation:

Functions

`make_grid`(im[, scale])	Return a xy meshgrid based in an input array, im, ranging from -scal * im.shape[0] // 2 to scal * im.shape[0] // 2.
`pixel_corr`(img, x, y)	Replace a pixel value with coordinates x and y by the mean value of its 3x3 neighbourhood.
`unwrap_2d`(img, **kwargs)	Unwraps an image along the x- and y-axis.
`unwrap_2d_mask`(img, mask, **kwargs)	Unwraps an image within a region of interest defined by a binary mask.
`crop`(img, n_crop)	Crops an image around all four edges by n_crop pixels.
`crop_to_mask`(img, mask)	Crops an image to the smallest size taken up by a binary mask.
`load_filter_upscale`(path, npx, pix_res[, crop, ...])	Loads a 2D numpy array and crops its edges. A uniform filter is applied to the cropped image before it is upscaled
`rect_mask`(im, dx, dy, w, h)	Rectangular mask using pixel coordinates of an input image.
`rect_mask_xy`(x, y, dx, dy, w, h)	Rectangular mask using XY meshgrid coordinates.
`circ_mask`(im, dx, dy, r)	Circular mask using pixel coordinates of an input image.
`circ_mask_xy`(x, y, dx, dy, r[, sparse])	Circular mask using XY meshgrid coordinates.
`gaussian`(x, y, dx, dy, sig_x[, sig_y, a, c])	2D Gaussian.
`super_gaussian`(x, y, dx, dy, nx, ny, sig_x, sig_y[, a, c])	2D super-Gaussian.
`gauss_array`(im, nx, ny, dx, dy, d, sigma)	Gaussian spot array using coordinates of input image.
`ring_gauss`(x, y, dx, dy, r, w[, a])	Ring with Gaussian profile.
`checkerboard`(npx, dx, dy, rows, columns, square_size)	Creates a checkerboard on a canvas of *(npx, npx)* pixels.
`fringes_wavefront`(x, y, dx, dy, k, f, phi, a, b)	Standing wave interference pattern on the camera caused by two patches on the SLM seperated by dx and dy.
`init_phase`(img, slm_disp_obj, pms_obj[, lin_phase, ...])	SLM phase guess to initialise phase-retrieval algorithm (see https://doi.org/10.1364/OE.16.002176).
`lens`(x, y, k, f)	Phase of a parabolic lens.
`doublet`(x, y, k, n1, n2, r1, r2, r3[, dx, dy])	Phase of a doublet lens.
`slm_phase_doublet`(dx, dy, k, xf, yf, z1, z2, fl, n1, ...)	Models the phase difference in the wavefront measurement caused by the doublet lens and an out-of-focus camera
`pixel_ct_kernel`(slm_pitch, pix_res, extent, m, sigma)	2D blurring kernel to model pixel crosstalk on the SLM (see https://doi.org/10.1186/s41476-021-00174-7).
`vortex_field`(img, m, x, y)	Creates the phase of a vortex field of charge `m` at positions `x` and `y` . The origin of the coordinate
`detect_vortices`(n_pix, e_holo, i_tar[, threshold])	This function detects the positions and charges of optical vortices in an electric field.

Module Contents

`hologradpy.patterns.`**`make_grid`**(*im*, *scale=None*)

Return a xy meshgrid based in an input array, im, ranging from -scal * im.shape[0] // 2 to scal * im.shape[0] // 2.

Parameters

- **`im`** – Input array.
- **`scale`** – Optional scaling factor.

Returns

x and y meshgrid arrays.

`hologradpy.patterns.`**`pixel_corr`**(*img*, *x*, *y*)

Replace a pixel value with coordinates x and y by the mean value of its 3x3 neighbourhood.

Parameters

- **`img`** – Input image.
- **`x`** – x-coordinate of pixel.
- **`y`** – y-coordinate of pixel.

Returns

Corrected image.

`hologradpy.patterns.`**`unwrap_2d`**(*img*, ***kwargs*)

Unwraps an image along the x- and y-axis.

Parameters

- **`img`** – Input image.
- **`kwargs`** – kwargs for `np.unwrap()` function.

Returns

Unwrapped image.

`hologradpy.patterns.`**`unwrap_2d_mask`**(*img*, *mask*, ***kwargs*)

Unwraps an image within a region of interest defined by a binary mask.

Parameters

- **`img`** – Input image.
- **`mask`** – Binary mask with region of interest.
- **`kwargs`** – kwargs for np.unwrap() function.

Returns

Unwrapped image.

`hologradpy.patterns.`**`crop`**(*img*, *n_crop*)

Crops an image around all four edges by n_crop pixels.

Parameters

- **`img`** – Input image.
- **`n_crop`** – Number of pixels to crop at both end of each dimension.

Returns

Cropped image.

hologradpy.patterns.**crop_to_mask**(*img*, *mask*)

Crops an image to the smallest size taken up by a binary mask.

Parameters

- **img** – Input image.
- **mask** – Binary mask.

Returns
Cropped image.

hologradpy.patterns.**load_filter_upscale**(*path*, *npx*, *pix_res*, *crop=None*, *filter_size=None*)

Loads a 2D numpy array and crops its edges. A uniform filter is applied to the cropped image before it is upscaled using Lanczos interpolation.

Parameters

- **path** – Numpy array or path to numpy array.
- **npx** – Number of SLM pixels.
- **pix_res** – Number of pixels per SLM pixel.
- **crop** – Number of unused pixels [SLM pixels].
- **filter_size** – Size of the uniform filter.

Returns
Upscaled image.

hologradpy.patterns.**rect_mask**(*im*, *dx*, *dy*, *w*, *h*)

Rectangular mask using pixel coordinates of an input image.

Parameters

- **im** – Input image
- **dx** – X-offset of rectangle from the centre of the image.
- **dy** – Y-offset of rectangle from the centre of the image.
- **w** – Width of rectangle.
- **h** – Height of rectangle.

Returns
Binary mask.

hologradpy.patterns.**rect_mask_xy**(*x*, *y*, *dx*, *dy*, *w*, *h*)

Rectangular mask using XY meshgrid coordinates.

Parameters

- **x** – X meshgrid
- **y** – Y meshgrid
- **dx** – X-offset of rectangle from the centre of the image.
- **dy** – Y-offset of rectangle from the centre of the image.
- **w** – Width of rectangle.
- **h** – Height of rectangle.

Returns
Binary mask.

hologradpy.patterns.**circ_mask**(*im*, *dx*, *dy*, *r*)

Circular mask using pixel coordinates of an input image.

Parameters

- **im** – Input image
- **dx** – X-offset of circle.
- **dy** – Y-offset of circle.
- **r** – Radius of circle.

Returns
Binary mask.

hologradpy.patterns.**circ_mask_xy**(*x*, *y*, *dx*, *dy*, *r*, *sparse=None*)

Circular mask using XY meshgrid coordinates.

Parameters

- **x** – X meshgrid.
- **y** – Y meshgrid.
- **dx** – X-offset of circle.
- **dy** – Y-offset of circle.
- **r** – Radius of circle.

Returns
Binary mask.

hologradpy.patterns.**gaussian**(*x*, *y*, *dx*, *dy*, *sig_x*, *sig_y=None*, *a=1*, *c=0*)

2D Gaussian.

Parameters

- **x** – X meshgrid.
- **y** – Y meshgrid.
- **dx** – X-offset of Gaussian.
- **dy** – Y-offset of Gaussian.
- **sig_x** – X width of Gaussian.
- **sig_y** – Y width of Gaussian
- **a** – Amplitude.
- **c** – Offset.

Returns
2D Gaussian.

hologradpy.patterns.**super_gaussian**(*x*, *y*, *dx*, *dy*, *nx*, *ny*, *sig_x*, *sig_y*, *a=1*, *c=0*)

2D super-Gaussian.

Parameters

- **x** – X meshgrid.
- **y** – Y meshgrid.
- **dx** – X-offset of Gaussian.

- **dy** – Y-offset of Gaussian.
- **nx** – X-order.
- **ny** – Y-order.
- **sig_x** – X-width.
- **sig_y** – Y-width.
- **a** – Amplitude.
- **c** – Offset.

Returns
2D super-Gaussian.

`hologradpy.patterns.`**`gauss_array`**(*im*, *nx*, *ny*, *dx*, *dy*, *d*, *sigma*)

Gaussian spot array using coordinates of input image.

Parameters

- **im** – Input image.
- **nx** – Number of array columns.
- **ny** – Number of array rows.
- **dx** – X-offset of array.
- **dy** – Y-offset of array.
- **d** – Separation between neighbouring spots.
- **sigma** – Width of Gaussian spots.

Returns
Spot array.

`hologradpy.patterns.`**`ring_gauss`**(*x*, *y*, *dx*, *dy*, *r*, *w*, *a=1*)

Ring with Gaussian profile.

Parameters

- **x** – X meshgrid.
- **y** – Y meshgrid.
- **dx** – X-offset of ring.
- **dy** – Y-offset of ring.
- **r** – Radius of ring.
- **w** – Width of Gaussian profile.
- **a** – Amplitude.

Returns
Ring with Gaussian profile.

`hologradpy.patterns.`**`checkerboard`**(*npx*, *dx*, *dy*, *rows*, *columns*, *square_size*)

Creates a checkerboard on a canvas of *(npx, npx)* pixels.

Parameters

- **npx** – Size of canvas.
- **dx** – X-offset of checkerboard.

HoloGradPy, Release 1.0

- **dy** – Y-Offset of checkerboard.
- **rows** – Checkerboard rows.
- **columns** – Checkerboard columns.
- **square_size** – Size of a square in pixels

Returns
Checkerboard.

`hologradpy.patterns.`**`fringes_wavefront`**(*x*, *y*, *dx*, *dy*, *k*, *f*, *phi*, *a*, *b*)

Standing wave interference pattern on the camera caused by two patches on the SLM seperated by dx and dy. Equation adapted from https://doi.org/10.1364/OE.24.013881.

Parameters

- **x** – X meshgrid.
- **y** – Y meshgrid.
- **dx** – Separation between reference and sample patch along x [m].
- **dy** – Separation between reference and sample patch along y [m].
- **k** – Wavenumber [rad/m].
- **f** – Focal length of Fourier lens [m].
- **phi** – Phase difference between reference and sample patches (see paper above) [rad].
- **a** – Amplitude on reference patch.
- **b** – Amplitude on sample patch.

Returns
Interference pattern.

`hologradpy.patterns.`**`init_phase`**(*img*, *slm_disp_obj*, *pms_obj*, *lin_phase=None*, *quad_phase=None*, *lin_method=None*)

SLM phase guess to initialise phase-retrieval algorithm (see https://doi.org/10.1364/OE.16.002176).

Parameters

- **img** (*ndarray*) – 2D array with size of desired output.
- **slm_disp_obj** – Instance of Params class
- **lin_phase** (*ndarray*) – Vector of length 2, containing parameters for the linear phase term
- **quad_phase** (*ndarray*) – Vector of length 2, containing parameters for the quadratic phase term
- **lin_method** (*str*) – Determines how the linear phase term is parameterised. The options are:

  -'pixel'
  Defines the linear phase in terms of Fourier pixels [px].

  -'angles'
  Defines the linear phase in terms of angles [rad].

Returns
Phase pattern of shape `img.shape`

hologradpy.patterns.**lens**(*x*, *y*, *k*, *f*)

Phase of a parabolic lens.

Parameters

- **x** – X-meshgrid [m].
- **y** – Y-meshgrid [m].
- **k** – Wavenumber [rad/m].
- **f** – Focal length [m].

Returns

Phase of the lens [rad].

hologradpy.patterns.**doublet**(*x*, *y*, *k*, *n1*, *n2*, *r1*, *r2*, *r3*, *dx=None*, *dy=None*)

Phase of a doublet lens.

Parameters

- **x** – X-meshgrid [m].
- **y** – Y-meshgrid [m].
- **k** – Wavenumber [rad/m].
- **n1** – Refractive index of flint.
- **n2** – Refractive index of crown.
- **r1** – Radius of curvature of the first crown surface [m].
- **r2** – Radius of curvature of the second crown surface/ first flint surface [m].
- **r3** – Radius of curvature of the second flint surface [m].
- **dx** – X-offset of lens [m].
- **dy** – Y-offset of lens [m].

Returns

Phase of the doublet lens [rad].

hologradpy.patterns.**slm_phase_doublet**(*dx*, *dy*, *k*, *xf*, *yf*, *z1*, *z2*, *fl*, *n1*, *n2*, *r1*, *r2*, *r3*)

Models the phase difference in the wavefront measurement caused by the doublet lens and an out-of-focus camera placement (see equation S8 in the supplementary information of https://doi.org/10.1038/s41598-023-30296-6).

Parameters

- **dx** – X-position of sample patch [m].
- **dy** – Y-position of sample patch [m].
- **k** – Wavenumber [rad/m].
- **xf** – X-position of phase measurement in the image plane [m].
- **yf** – Y-position of phase measurement in the image plane [m].
- **z1** – Distance between SLM and lens [m].
- **z2** – Distance between lens and camera [m].
- **fl** – Focal length of doublet lens [m].
- **n1** – Refractive index of flint.
- **n2** – Refractive index of crown.

- **r1** – Radius of curvature of the first crown surface [m].
- **r2** – Radius of curvature of the second crown surface/ first flint surface [m].
- **r3** – Radius of curvature of the second flint surface [m].

Returns

Corrective phase pattern [rad].

hologradpy.patterns.**pixel_ct_kernel**(*slm_pitch*, *pix_res*, *extent*, *m*, *sigma*)

2D blurring kernel to model pixel crosstalk on the SLM (see https://doi.org/10.1186/s41476-021-00174-7).

Parameters

- **slm_pitch** – Pixel pitch of SLM [m].
- **pix_res** – Up-scaling factor (computational pixels per SLM pixel).
- **extent** – Spatial extent of kernel in SLM pixels.
- **m** – Order of the kernel.
- **sigma** – Width of the kernel.

Returns

2D blurring kernel.

hologradpy.patterns.**vortex_field**(*img*, *m*, *x*, *y*)

Creates the phase of a vortex field of charge `m` at positions `x` and `y` . The origin of the coordinate system is in the top-left corner of `img` .

Parameters

- **img** – 2D array with size of desired output.
- **m** – Vector of vortex charge (1 or -1).
- **x** – Vector of vortex x-coordinate [px].
- **y** – Vector of vortex y-coordinate [px].

Returns

Phase of vortex field with size `img.shape`.

hologradpy.patterns.**detect_vortices**(*n_pix*, *e_holo*, *i_tar*, *threshold=None*)

This function detects the positions and charges of optical vortices in an electric field. Todo: Tidy up this function and improve documentation.

Parameters

- **n_pix** – Number of pixels.
- **e_holo** – Electric field.
- **i_tar** – Target intensity pattern.
- **threshold** – Only look for vortices in areas which are brighter than `theshold * max(abs(i_tar) ** 2)`. Vortices in low-intensity regions are hard to detect.

Returns

Charge of vortices and their xy coordinates.

class holograd.patterns.**Hologram**(*slm_disp_obj*, *pms_obj*, *name*, *npix*, *npix_pad=None*, *pix_res=1*, *phase_guess_type='random'*, *linear_phase=None*, *quadratic_phase=None*, *slm_field_type='guess'*, *propagation_type='fft'*, *target_position=None*, *target_width=None*, *target_blur=None*, *checkerboard_rows=8*, *checkerboard_columns=10*, *checkerboard_square_size=32*)

This class provides arrays needed for the CG minimisation:

- The target light potential
- the signal region
- the measured constant SLM phase and intensity at the required resolution
- and the initial SLM phase guess to start the phase retrieval.

Some patterns, including the patterns from our publication (https://doi.org/10.1038/s41598-023-30296-6), are pre-defined here. Feel free to define your own patterns, you don't have to use this class to do this.

Todo: Tidy up this class and improve documentation.

holograd.torch_functions

Module containing PyTorch-specific functions to perform conjugate gradient minimisation.

Classes

ASM	This class models the propagation of light from the SLM to the Fourier lens and from the Fourier lens to the image
VirtualSlm	This class models pixel crosstalk on the SLM and the propagation of light from the SLM to the camera.
PhaseRetrieval	This function calculates the SLM phase pattern for a given target light potential in the image plane using conjugate

Functions

check_device([verbose])	Check if GPU is available.
gpu_to_numpy(gpu_tensor)	
fft(e_in[, shift, norm])	Performs the FFT.
ifft(e_in[, shift, norm])	Performs the IFFT.
asm(e_in, e_lens, theta1[, theta2, shift])	Performs the angular spectrum method (ASM) twice, from the SLM to the Fourier lens and from the Fourier lens to the
rms(signal, i_target, i_out, frac)	Calculate normalised root-mean-squared error between two images inside a region of interest. Only pixels which are
eff(signal, i_out)	Calculates the predicted efficiency of a light potential by dividing the pixel sum in the signal region by
loss_fn_fid(e_out, i_tar, phi_tar, signal)	Phase and amplitude cost function from https://doi.org/10.1364/OE.25.011692.
loss_fn_amp(e_out, i_tar, signal)	Amplitude-only cost function from https://doi.org/10.1364/OE.22.026548.
camera_calibration(slm_obj, slm_disp_obj, cam_obj, pms_obj)	This function performs the camera calibration to obtain the coordinate transform between
camera_feedback(phase_retrieval_obj, slm_disp_obj, ...)	This function implements a camera feedback algorithm to reduce experimental errors in the light potentials

Module Contents

holograd.torch_functions.**check_device**(*verbose=None*)

Check if GPU is available.

Parameters

verbose (*bool*) – Verbose output?

Returns

'cuda' if GPU available, otherwise 'cpu'.

holograd.torch_functions.**gpu_to_numpy**(*gpu_tensor*)

holograd.torch_functions.**fft**(*e_in*, *shift=True*, *norm=None*)

Performs the FFT.

Parameters

- **e_in** – Input electric field.
- **shift** (*bool*) – Perform FFT shift?
- **norm** – Normalisation of FFT.

Returns

FFT of `e_in`.

holograd.torch_functions.**ifft**(*e_in*, *shift=True*, *norm=None*)

Performs the IFFT.

Parameters

- **e_in** – Input electric field.
- **shift** (*bool*) – Perform IFFT shift?
- **norm** – Normalisation of IFFT.

Returns
IFFT of `e_in`.

`hologradpy.torch_functions.`**`asm`**(*e_in*, *e_lens*, *theta1*, *theta2=None*, *shift=True*)

Performs the angular spectrum method (ASM) twice, from the SLM to the Fourier lens and from the Fourier lens to the camera.

Parameters

- **e_in** – Electric field at the SLM.
- **e_lens** – Electric field of the lens (phase and aperture).
- **theta1** – Propagation phase from SLM to lens.
- **theta2** – Propagation phase from lens to camera. If not provided it is assumed `theta2 = theta1`.
- **shift** (*bool*) – Perform FFT shift?

Returns
Electric field at the camera.

`class hologradpy.torch_functions.`**`ASM`**(*slm_disp_obj*, *pms_obj*, *pix_res*, *npix_tot*, *pd1*, *pd2*, *extent_lens*, *xf*, *shift=False*, *precision='single'*, *device='cuda'*)

This class models the propagation of light from the SLM to the Fourier lens and from the Fourier lens to the image plane using the angular spectrum method. The lens is modelled as a doublet. The ASM wavefront correction is also calculated in this class.

`forward`(*e_in*)

This function performs the simulation.

Parameters
e_in – Electric field at the SLM plane.

Returns
Electric field at the image plane.

`class hologradpy.torch_functions.`**`VirtualSlm`**(*slm_disp_obj*, *pms_obj*, *phi*, *npix_pad*, *npix=None*, *e_slm=None*, *kernel_ct=None*, *pix_res=None*, *propagation_type='fft'*, *extent_lens=None*, *pd1=None*, *pd2=None*, *xf=None*, *device='cpu'*, *slm_mask=None*, *precision=None*, *fft_shift=True*)

Bases: `torch.nn.Module`

This class models pixel crosstalk on the SLM and the propagation of light from the SLM to the camera.

`set_phi`(*new_phi*)

Set SLM phase from numpy array.

Parameters
new_phi (*ndarray*) – SLM phase [rad].

`forward`()

Model the SLM and simulate the propagation of light from the SLM plane to the image plane. This method is used by gradient-based optimizers.

Returns
Electric field in the image plane.

`hologradpy.torch_functions.`**`rms`**(*signal*, *i_target*, *i_out*, *frac*)

Calculate normalised root-mean-squared error between two images inside a region of interest. Only pixels which are brighter than `frac * np.max(i_target_norm)` are taken into account, where `i_target_norm` is the normalised target intensity pattern.

Parameters

- **signal** – Binary mask containing region of interest (signal region).
- **i_target** – Target intensity pattern.
- **i_out** – Intensity pattern of light potential.
- **frac** – Threshold as explained above.

Returns
Normalised rms error.

`hologradpy.torch_functions.`**`eff`**(*signal*, *i_out*)

Calculates the predicted efficiency of a light potential by dividing the pixel sum in the signal region by the pixel sum in the entire pattern.

Parameters

- **signal** – Binary mask containing the signal region.
- **i_out** – Intensity pattern of the light potential.

Returns
Predicted efficiency.

`hologradpy.torch_functions.`**`loss_fn_fid`**(*e_out*, *i_tar*, *phi_tar*, *signal*)

Phase and amplitude cost function from https://doi.org/10.1364/OE.25.011692.

Parameters

- **e_out** – Electric field at the image plane.
- **i_tar** – Target intensity pattern.
- **phi_tar** – Target phase pattern.
- **signal** – Binary mask containing signal region.

Returns
Cost.

`hologradpy.torch_functions.`**`loss_fn_amp`**(*e_out*, *i_tar*, *signal*)

Amplitude-only cost function from https://doi.org/10.1364/OE.22.026548.

Parameters

- **e_out** – Electric field at the image plane.
- **i_tar** – Target intensity pattern.
- **signal** – Binary mask containing signal region.

Returns
Cost.

`class holodradpy.torch_functions.PhaseRetrieval(`*slm_obj*, *n_iter=10*, *i_tar=None*, *phi_tar=None*, *signal_region=None*, *save=False*, *n_save=10*, *loss_type='amp'*, *optim_type='cg'*`)`

This function calculates the SLM phase pattern for a given target light potential in the image plane using conjugate gradient minimisation or stochastic gradient descent (Adam).

set_target(*target*)

Sets the target light potential.

Parameters
target – Target light potential.

set_optimizer()

Sets the optimisation algorithm based on `self.optim_type`.

loss_fn(*e_out*)

Defines the loss function based on `self.loss_type`.

Parameters
e_out – Electric field at the image plane.

Returns
Loss value.

callback(*x*)

This function is called after every iteration of the optimisation. It saves intermediate SLM phase patterns and the electric field in the image plane if `save=True`. The progress of the optimisation is printed after every iteration.

retrieve_phase()

Performs phase retrieval algorithm.

Returns
Optimised SLM phase(s), (RMS error and efficiency if `save=True`)

`holodradpy.torch_functions.camera_calibration(`*slm_obj*, *slm_disp_obj*, *cam_obj*, *pms_obj*, *save=False*, *exp_time=None*, *checkerboard_rows=None*, *checkerboard_columns=None*, *checkerboard_square_size=None*, *linear_phase=None*`)`

This function performs the camera calibration to obtain the coordinate transform between the camera image and the computational image plane. To do this, an SLM phase pattern is calculated for a checkerboard-shaped target light potential using CG minimisation and displayed on the SLM. The corners of the checkerboard in the resulting camera image are detected and fitted to the corners of the checkerboard in the computational image plane using an affine transformation.

Parameters

- **slm_obj** – Virtual SLM object created by *VirtualSlm*.
- **slm_disp_obj** – Object created by a subclass of `hardware.SlmBase`.
- **cam_obj** – Object created by a subclass of `hardware.CameraBase`.
- **pms_obj** – Object created by a subclass of `hardware.ParamsBase`.
- **save** (*bool*) – Save data?
- **exp_time** – Exposure time.
- **checkerboard_rows** –
- **checkerboard_columns** –

- **checkerboard_square_size** –
- **linear_phase** –

Returns

`holodradpy.torch_functions.camera_feedback(`*phase_retrieval_obj*, *slm_disp_obj*, *cam_obj*, *tf*, *itf*, *iter_fb=1*, *iter_cg=None*, *detect_vortices=False*, *threshold_vtx=0.2*, *n_save=10*, *n_avg=10*, *exp_time=1000*, *fb_blur=0*, *alpha=None*, *convergence=False*, *iter_convergence=None*, *path=None*`)`

This function implements a camera feedback algorithm to reduce experimental errors in the light potentials (see https://dx.doi.org/10.1088/0953-4075/48/11/115303). Before applying any camera feedback, optical vortices in the light potential are detected using the patterns.detect_vortices() function and removed if required.

After vortices are removed, the optimised phase pattern is displayed on the SLM and a camera image, M, is recorded. To create the target light potential for the next feedback iteration, T[. . . , i + 1], a discrepancy, D, between the camera image and the original target light potential, T[. . . , 0], is calculated and added to the previous light potential, T[. . . , i].

At the end of each feedback iteration, the root-mean-squared error (RMSE) and the peak signal-to-noise ratio (PSNR) of the camera image are calculated and saved. To find the experimental convergence of the CG minimisation, intermediate SLM phase patterns are saved and displayed on the SLM. A camera image is taken for each pattern and the RMSE is calculated.

Parameters

- **phase_retrieval_obj** – Instance of the class *PhaseRetrieval*.
- **slm_disp_obj** – Object created by a subclass of `hardware.SlmBase`.
- **cam_obj** – Object created by a subclass of `hardware.CameraBase`.
- **tf** – Affine transform matrix.
- **itf** – Inverse affine transform matrix.
- **iter_fb** – Number of feedback iterations.
- **iter_cg** – Number of conjugate gradient iterations per feedback iteration.
- **detect_vortices** (*bool*) – Detect vortices?
- **threshold_vtx** – See `patterns.detect_vortices()`
- **n_save** – Save data for every `n_save` th CG iteration.
- **n_avg** – Number of camera frames to capture and average per feedback iteration.
- **exp_time** – Exposure time.
- **fb_blur** – Width of blurring kernel for camera image [px].
- **alpha** – Feedback gain parameter for each feedback iteration.
- **convergence** (*bool*) – Save intermediate phase patterns during CG minimisation?
- **iter_convergence** – During which feedback iterations to save intermediate phase patterns.
- **path** – Save path.

Returns
See code.

CHAPTER

THREE

CITE AS

If you are using this code, please cite our publication:

P. Schroff, A. La Rooij, E. Haller, S. Kuhr, Accurate holographic light potentials using pixel crosstalk modelling. *Sci Rep* **13**, 3252 (2023). https://doi.org/10.1038/s41598-023-30296-6

3.1 Indices and tables

- genindex
- modindex
- search

PYTHON MODULE INDEX

h

INDEX

Bibliography

1. Adesnik, H. & Abdeladim, L. Probing neural codes with two-photon holographic optogenetics. *Nat. Neurosci.* **24,** 1356–1366. doi:10.1038/s41593-021-00902-9 (2021).

2. May, M. A., Barré, N., Kummer, K. K., Kress, M., Ritsch-Marte, M. & Jesacher, A. Fast holographic scattering compensation for deep tissue biological imaging. *Nat. Commun.* **12,** 4340. doi:10.1038/s41467-021-24666-9 (2021).

3. Wu, Y., Sharma, M. K. & Veeraraghavan, A. WISH: wavefront imaging sensor with high resolution. *Light Sci. Appl.* **8,** 44. doi:10.1038/s41377-019-0154-x (2019).

4. Huang, L., Xu, K., Yuan, D., Hu, J., Wang, X. & Xu, S. Sub-wavelength patterned pulse laser lithography for efficient fabrication of large-area metasurfaces. *Nat. Commun.* **13,** 5823. doi:10.1038/s41467-022-33644-8 (2022).

5. Moon, W., Jeon, H., Na, S., Kim, H. & Hahn, J. Accommodation-capable holographic waveguide head-up display with extended field of view. *Displays* **81,** 102617. doi:10.1016/j.displa.2023.102617 (2024).

6. Peng, Y., Choi, S., Padmanaban, N. & Wetzstein, G. Neural holography with camera-in-the-loop training. *ACM Trans. Graph.* **39,** 1–14. doi:10.1145/3414685.3417802 (2020).

7. Choi, S., Gopakumar, M., Peng, Y., Kim, J. & Wetzstein, G. Neural 3D holography: learning accurate wave propagation models for 3D holographic virtual and augmented reality displays. *ACM Trans. Graph.* **40,** 1–12. doi:10.1145/3478513.3480542 (2021).

8. Choi, S., Gopakumar, M., Peng, Y., Kim, J., O'Toole, M. & Wetzstein, G. *Time-multiplexed Neural Holography: A Flexible Framework for Holographic Near-eye Displays with Fast Heavily-quantized Spatial Light Modulators* in *ACM SIGGRAPH 2022 Conference Proceedings* (Association for Computing Machinery, New York, NY, USA, 2022), 1–9. doi:10.1145/3528233.3530734.

9. Jang, C., Bang, K., Chae, M., Lee, B. & Lanman, D. Waveguide holography for 3D augmented reality glasses. *Nat. Commun.* **15,** 66. doi:10.1038/s41467-023-44032-1 (2024).

10. Ebadi, S., Wang, T. T., Levine, H., Keesling, A., Semeghini, G., Omran, A., Bluvstein, D., Samajdar, R., Pichler, H., Ho, W. W., Choi, S., Sachdev, S., Greiner, M., Vuletić, V. & Lukin, M. D. Quantum phases of matter on a 256-atom programmable quantum simulator. *Nature* **595,** 227–232. doi:10.1038/s41586-021-03582-4 (2021).

11. Kwon, W. J., Pace, G. D., Panza, R., Inguscio, M., Zwerger, W., Zaccanti, M., Scazza, F. & Roati, G. Strongly correlated superfluid order parameters from dc Josephson supercurrents. *Science* **369,** 84–88. doi:10.1126/science.aaz2463 (2020).

12. Luick, N., Sobirey, L., Bohlen, M., Singh, V. P., Mathey, L., Lompe, T. & Moritz, H. An ideal Josephson junction in an ultracold two-dimensional Fermi gas. *Science* **369,** 89–91. doi:10.1126/science.aaz2342 (2020).

13. Guo, Y., Kroeze, R. M., Marsh, B. P., Gopalakrishnan, S., Keeling, J. & Lev, B. L. An optical lattice with sound. *Nature* **599,** 211–215. doi:10.1038/s41586-021-03945-x (2021).

14. Sompet, P., Hirthe, S., Bourgund, D., Chalopin, T., Bibo, J., Koepsell, J., Bojović, P., Verresen, R., Pollmann, F., Salomon, G., Gross, C., Hilker, T. A. & Bloch, I. Realizing the symmetry-protected Haldane phase in Fermi–Hubbard ladders. *Nature* **606,** 484–488. doi:10.1038/s41586-022-04688-z (2022).

15. Mazurenko, A., Chiu, C. S., Ji, G., Parsons, M. F., Kanász-Nagy, M., Schmidt, R., Grusdt, F., Demler, E., Greif, D. & Greiner, M. A cold-atom Fermi–Hubbard antiferromagnet. *Nature* **545,** 462–466. doi:10.1038/nature22362 (2017).

16. Navon, N., Gaunt, A. L., Smith, R. P. & Hadzibabic, Z. Emergence of a turbulent cascade in a quantum gas. *Nature* **539,** 72–75. doi:`10.1038/nature20114` (2016).

17. Barredo, D., Léséleuc, S. D., Lienhard, V., Lahaye, T. & Browaeys, A. An atom-by-atom assembler of defect-free arbitrary two-dimensional atomic arrays. *Science* **354,** 1021–1023. doi:`10.1126/science.aah3778` (2016).

18. Amico, L. *et al.* Roadmap on Atomtronics: State of the art and perspective. *AVS Quantum Sci.* **3,** 039201. doi:`10.1116/5.0026178` (2021).

19. Singh, K., Anand, S., Pocklington, A., Kemp, J. T. & Bernien, H. Dual-Element, Two-Dimensional Atom Array with Continuous-Mode Operation. *Phys. Rev. X* **12,** 011040. doi:`10.1103/PhysRevX.12.011040` (2022).

20. Barredo, D., Lienhard, V., de Léséleuc, S., Lahaye, T. & Browaeys, A. Synthetic three-dimensional atomic structures assembled atom by atom. *Nature* **561,** 79–82. doi:`10.1038/s41586-018-0450-2` (2018).

21. Buccheri, F., Bruce, G. D., Trombettoni, A., Cassettari, D., Babujian, H., Korepin, V. E. & Sodano, P. Holographic optical traps for atom-based topological Kondo devices. *New J. Phys.* **18,** 075012. doi:`10.1088/1367-2630/18/7/075012` (2016).

22. Fatemi, F. K., Bashkansky, M. & Dutton, Z. Dynamic high-speed spatial manipulation of cold atoms using acousto-optic and spatial light modulation. *Opt. Express* **15,** 3589–3596. doi:`10.1364/OE.15.003589` (2007).

23. Lau, J. W. Z., Gan, K. S., Dumke, R., Amico, L., Kwek, L.-C. & Haug, T. Atomtronic multiterminal Aharonov-Bohm interferometer. *Phys. Rev. A* **107,** L051303. doi:`10.1103/PhysRevA.107.L051303` (2023).

24. Barredo, D., Lienhard, V., Scholl, P., Léséleuc, S. D., Boulier, T., Browaeys, A. & Lahaye, T. Three-dimensional trapping of individual Rydberg atoms in ponderomotive bottle beam traps. *Phys. Rev. Lett.* **124,** 023201. doi:`10.1103/PhysRevLett.124.023201` (2020).

25. Gauthier, G., Lenton, I., Parry, N. M., Baker, M., Davis, M. J., Rubinsztein-Dunlop, H. & Neely, T. W. Direct imaging of a digital-micromirror device for

configurable microscopic optical potentials. *Optica* **3,** 1136–1143. doi:10.1364/OPTICA.3.001136 (2016).

26. Navon, N., Smith, R. P. & Hadzibabic, Z. Quantum gases in optical boxes. *Nat. Phys.* **17,** 1334–1341. doi:10.1038/s41567-021-01403-z (2021).

27. Eigen, C., Gaunt, A. L., Suleymanzade, A., Navon, N., Hadzibabic, Z. & Smith, R. P. Observation of Weak Collapse in a Bose-Einstein Condensate. *Phys. Rev. X* **6,** 041058. doi:10.1103/PhysRevX.6.041058 (2016).

28. Martirosyan, G., Ho, C. J., Etrych, J., Zhang, Y., Cao, A., Hadzibabic, Z. & Eigen, C. Observation of Subdiffusive Dynamic Scaling in a Driven and Disordered Bose Gas. *Phys. Rev. Lett.* **132,** 113401. doi:10.1103/PhysRevLett.132.113401 (2024).

29. Gotlibovych, I., Schmidutz, T. F., Gaunt, A. L., Navon, N., Smith, R. P. & Hadzibabic, Z. Observing properties of an interacting homogeneous Bose-Einstein condensate: Heisenberg-limited momentum spread, interaction energy, and free-expansion dynamics. *Phys. Rev. A* **89,** 061604. doi:10.1103/PhysRevA.89.061604 (2014).

30. Gaunt, A. L., Schmidutz, T. F., Gotlibovych, I., Smith, R. P. & Hadzibabic, Z. Bose-Einstein Condensation of Atoms in a Uniform Potential. *Phys. Rev. Lett.* **110,** 200406. doi:10.1103/PhysRevLett.110.200406 (2013).

31. Wang, S., Wang, J., Cheng, M., Zhang, J., Yang, X., Xu, J. & Wang, L.-G. Three transmission properties of the perfect vortex beam. *Optics Express* **32,** 34141–34152. doi:10.1364/OE.536960 (2024).

32. Kwon, W. J., Del Pace, G., Xhani, K., Galantucci, L., Muzi Falconi, A., Inguscio, M., Scazza, F. & Roati, G. Sound emission and annihilations in a programmable quantum vortex collider. *Nature* **600,** 64–69. doi:10.1038/s41586-021-04047-4 (2021).

33. Del Pace, G., Xhani, K., Muzi Falconi, A., Fedrizzi, M., Grani, N., Hernandez Rajkov, D., Inguscio, M., Scazza, F., Kwon, W. J. & Roati, G. Imprinting Persistent Currents in Tunable Fermionic Rings. *Phys. Rev. X* **12,** 041037. doi:10.1103/PhysRevX.12.041037 (2022).

34. Reeves, M. T., Goddard-Lee, K., Gauthier, G., Stockdale, O. R., Salman, H., Edmonds, T., Yu, X., Bradley, A. S., Baker, M., Rubinsztein-Dunlop, H., Davis, M. J. & Neely, T. W. Turbulent Relaxation to Equilibrium in a Two-Dimensional Quantum Vortex Gas. *Phys. Rev. X* **12,** 011031. doi:10.1103/PhysRevX.12.011031 (2022).

35. Samson, E. C., Wilson, K. E., Newman, Z. L. & Anderson, B. P. Deterministic creation, pinning, and manipulation of quantized vortices in a Bose-Einstein condensate. *Physical Review A* **93,** 023603. doi:10.1103/PhysRevA.93.023603 (2016).

36. Kim, S., Lee, K., Kim, J. & Shin, Y. Flattening a trapped atomic gas using a programmable optical potential in a feedback loop. *Phys. Rev. A* **110,** 033326. doi:10.1103/PhysRevA.110.033326 (2024).

37. Lee, K., Kim, S., Kim, T. & Shin, Y.-i. *Observation of universal Kibble-Zurek scaling in an atomic Fermi superfluid* 2023. doi:10.48550/arXiv.2310.05437.

38. Häusler, S., Nakajima, S., Lebrat, M., Husmann, D., Krinner, S., Esslinger, T. & Brantut, J. P. Scanning Gate Microscope for Cold Atomic Gases. *Phys. Rev. Lett.* **119,** 030403. doi:10.1103/physrevlett.119.030403 (2017).

39. Henderson, K., Ryu, C., MacCormick, C. & Boshier, M. G. Experimental demonstration of painting arbitrary and dynamic potentials for Bose–Einstein condensates. *New J. Phys.* **11,** 043030. doi:10.1088/1367-2630/11/4/043030 (2009).

40. Wright, K. C., Blakestad, R. B., Lobb, C. J., Phillips, W. D. & Campbell, G. K. Driving phase slips in a superfluid atom circuit with a rotating weak link. *Phys. Rev. Lett.* **110,** 025302. doi:http://dx.doi.org/10.1103/PhysRevLett.110.025302 (2013).

41. Gauthier, G., Szigeti, S. S., Reeves, M. T., Baker, M., Bell, T. A., Rubinsztein-Dunlop, H., Davis, M. J. & Neely, T. W. Quantitative acoustic models for superfluid circuits. *Phys. Rev. Lett.* **123,** 260402. doi:10.1103/PhysRevLett.123.260402 (2019).

42. Schlömer, H., Lange, H., Franz, T., Chalopin, T., Bojović, P., Wang, S., Bloch, I., Hilker, T. A., Grusdt, F. & Bohrdt, A. Local Control and Mixed Dimensions: Exploring High-Temperature Superconductivity in Optical Lattices. *PRX Quantum* **5,** 040341. doi:`10.1103/PRXQuantum.5.040341` (2024).

43. Yamamoto, D., Fukuhara, T. & Danshita, I. Frustrated quantum magnetism with Bose gases in triangular optical lattices at negative absolute temperatures. *Communications Physics* **3,** 1–10. doi:`10.1038/s42005-020-0323-5` (2020).

44. Zheng, Y.-G., Luo, A., Shen, Y.-C., He, M.-G., Zhu, Z.-H., Liu, Y., Zhang, W.-Y., Sun, H., Deng, Y., Yuan, Z.-S. & Pan, J.-W. Observation of counterflow superfluidity in a two-component Mott insulator. Preprint at `http://arxiv.org/abs/2403.03479` (2024).

45. Ji, G., Xu, M., Kendrick, L. H., Chiu, C. S., Brüggenjürgen, J. C., Greif, D., Bohrdt, A., Grusdt, F., Demler, E., Lebrat, M. & Greiner, M. Coupling a Mobile Hole to an Antiferromagnetic Spin Background: Transient Dynamics of a Magnetic Polaron. *Phys. Rev. X* **11,** 021022. doi:`10.1103/PhysRevX.11.021022` (2021).

46. Zupancic, P., Preiss, P. M., Ruichao Ma, A. L., Tai, M. E., Rispoli, M., Islam, R. & Greiner, M. Ultra-precise holographic beam shaping for microscopic quantum control. *Opt. Express* **24,** 13881–13893. doi:`10.1364/OE.24.013881` (2016).

47. Hirthe, S., Chalopin, T., Bourgund, D., Bojović, P., Bohrdt, A., Demler, E., Grusdt, F., Bloch, I. & Hilker, T. A. Magnetically mediated hole pairing in fermionic ladders of ultracold atoms. *Nature* **613,** 463–467. doi:`10.1038/s41586-022-05437-y` (2023).

48. Scalapino, D. Superconductivity and Spin Fluctuations. *J. Low Temp. Phys.* **117,** 179–188. doi:`10.1023/A:1022559920049` (1999).

49. Qiu, X., Zou, J., Qi, X. & Li, X. Precise programmable quantum simulations with optical lattices. *Npj Quantum Inf.* **6,** 1–8. doi:`10.1038/s41534-020-00315-9` (2020).

50. Choi, J.-y., Hild, S., Zeiher, J., Schauß, P., Rubio-Abadal, A., Yefsah, T., Khemani, V., Huse, D. A., Bloch, I. & Gross, C. Exploring the many-body localization transition in two dimensions. *Science* **352,** 1547–1552. doi:10.1126/science.aaf8834 (2016).

51. Trisnadi, J., Zhang, M., Weiss, L. & Chin, C. Design and construction of a quantum matter synthesizer. *Rev. Sci. Instrum.* **93,** 083203. doi:10.1063/5.0100088 (2022).

52. Di Carli, A., Parsonage, C., La Rooij, A., Koehn, L., Ulm, C., Duncan, C. W., Daley, A. J., Haller, E. & Kuhr, S. Commensurate and incommensurate 1D interacting quantum systems. *Nat. Commun.* **15,** 474. doi:10.1038/s41467-023-44610-3 (2024).

53. Amico, L., Aghamalyan, D., Auksztol, F., Crepaz, H., Dumke, R. & Kwek, L. C. Superfluid qubit systems with ring shaped optical lattices. *Sci. Rep.* **4,** 4298. doi:10.1038/srep04298 (2014).

54. Legrand, T., Winkelmann, F.-R., Alt, W., Meschede, D., Alberti, A. & Weidner, C. A. Three-dimensional imaging of single atoms in an optical lattice via helical point-spread-function engineering. *Phys. Rev. A* **109,** 033304. doi:10.1103/PhysRevA.109.033304 (2024).

55. Manetsch, H. J., Nomura, G., Bataille, E., Leung, K. H., Lv, X. & Endres, M. A tweezer array with 6100 highly coherent atomic qubits. Preprint at http://arxiv.org/abs/2403.12021 (2024).

56. Ren, Z., Yan, X., Wen, K., Chen, H., Hajiyev, E., He, C. & Jo, G.-B. Creation of a tweezer array for cold atoms utilizing a generative neural network. Preprint at http://arxiv.org/abs/2401.06014 (2024).

57. Nogrette, F., Labuhn, H., Ravets, S., Barredo, D., Béguin, L., Vernier, A., Lahaye, T. & Browaeys, A. Single-Atom Trapping in Holographic 2D Arrays of Microtraps with Arbitrary Geometries. *Phys. Rev. X* **4,** 021034. doi:10.1103/PhysRevX.4.021034 (2014).

58. Cesa, F. & Pichler, H. Universal Quantum Computation in Globally Driven Rydberg Atom Arrays. *Phys. Rev. Lett.* **131,** 170601. doi:10.1103/PhysRevLett.131.170601 (2023).

59. Bluvstein, D., Evered, S. J., Geim, A. A., Li, S. H., Zhou, H., Manovitz, T., Ebadi, S., Cain, M., Kalinowski, M., Hangleiter, D., Bonilla Ataides, J. P., Maskara, N., Cong, I., Gao, X., Sales Rodriguez, P., Karolyshyn, T., Semeghini, G., Gullans, M. J., Greiner, M., Vuletić, V. & Lukin, M. D. Logical quantum processor based on reconfigurable atom arrays. *Nature* **626,** 58–65. doi:10.1038/s41586-023-06927-3 (2024).

60. Gyger, F., Ammenwerth, M., Tao, R., Timme, H., Snigirev, S., Bloch, I. & Zeiher, J. Continuous operation of large-scale atom arrays in optical lattices. Preprint at http://arxiv.org/abs/2402.04994 (2024).

61. Pause, L., Sturm, L., Mittenbühler, M., Amann, S., Preuschoff, T., Schäffner, D., Schlosser, M. & Birkl, G. Supercharged two-dimensional tweezer array with more than 1000 atomic qubits. *Optica* **11,** 222–226. doi:10.1364/OPTICA.513551 (2024).

62. Menon, S. G., Glachman, N., Pompili, M., Dibos, A. & Bernien, H. An integrated atom array – nanophotonic chip platform with background-free imaging. Preprint at http://arxiv.org/abs/2311.02153 (2023).

63. Ravon, B., Méhaignerie, P., Machu, Y., Hernández, A. D., Favier, M., Raimond, J.-M., Brune, M. & Sayrin, C. Array of Individual Circular Rydberg Atoms Trapped in Optical Tweezers. *Phys. Rev. Lett.* **131,** 093401. doi:10.1103/PhysRevLett.131.093401 (2023).

64. De Oliveira, A. G., Diamond-Hitchcock, E., Walker, D. M., Wells-Pestell, M. T., Pelegrí, G., Picken, C. J., Malcolm, G. P. A., Daley, A. J., Bass, J. & Pritchard, J. D. Demonstration of Weighted-Graph Optimization on a Rydberg-Atom Array Using Local Light Shifts. *PRX Quantum* **6,** 010301. doi:10.1103/PRXQuantum.6.010301 (2025).

65. Saffman, M., Walker, T. G. & Mølmer, K. Quantum information with Rydberg atoms. *Rev. Mod. Phys.* **82,** 2313–2363. doi:10.1103/RevModPhys.82.2313 (2010).

66. Grimm, R., Weidemüller, M. & Ovchinnikov, Y. B. in *Advances In Atomic, Molecular, and Optical Physics* (eds Bederson, B. & Walther, H.) 95–170 (Academic Press, 2000). doi:10.1016/S1049-250X(08)60186-X.

67. Roichman, Y., Sun, B., Roichman, Y., Amato-Grill, J. & Grier, D. G. Optical Forces Arising from Phase Gradients. *Phys. Rev. Lett.* **100,** 013602. doi:10.1103/PhysRevLett.100.013602 (2008).

68. Kepler, J. *De Cometis - Libelli Tres* First edition (Aperger, A., 1619).

69. Jones, P. H., Marago, O. M. & Volpe, G. *Optical tweezers - principles and applications* (Cambridge University Press, 2015).

70. Chu, S., Bjorkholm, J. E., Ashkin, A. & Cable, A. Experimental Observation of Optically Trapped Atoms. *Phys. Rev. Lett.* **57,** 314–317. doi:10.1103/PhysRevLett.57.314 (1986).

71. Foot, C. J. & Foot, C. J. *Atomic Physics Oxford Master Series in Physics* (Oxford University Press, 2004).

72. Albaladejo, S., Marqués, M. I., Laroche, M. & Sáenz, J. J. Scattering Forces from the Curl of the Spin Angular Momentum of a Light Field. *Physical Review Letters* **102,** 113602. doi:10.1103/PhysRevLett.102.113602 (2009).

73. Walker, G., Arnold, A. S. & Franke-Arnold, S. Trans-spectral orbital angular momentum transfer via four-wave mixing in Rb vapor. *Phys. Rev. Lett.* **108,** 243601. doi:10.1103/PhysRevLett.108.243601 (2012).

74. Brachmann, J. F. S., Bakr, W. S., Gillen, J., Peng, A. & Greiner, M. Inducing vortices in a Bose-Einstein condensate using holographically produced light beams. *Opt. Express* **19,** 12984–12991. doi:10.1364/OE.19.012984 (2011).

75. *Hamamatsu Photonics has developed an LCOS-SLM with the world's highest power handling capability that will streamline production of laser metal machining.* https://www.hamamatsu.com/eu/en/news/products-and-technologies/2023/20230620000000.html (Hamamatsu Photonics, 2023).

76. *LCOS-SLM X13267/X13138 series.* Datasheet (2019).

77. Douglass, M. *DMD reliability: a MEMS success story* in *Reliability, Testing, and Characterization of MEMS/MOEMS II* **4980** (SPIE, 2003), 1–11. doi:10.1117/12.478212.

78. *The Digital Micromirror Device.* Technical report. https://www.asme.org/wwwasmeorg/media/resourcefiles/aboutasme/who%20we%20are/engineering%20history/landmarks/dmd_brochure.pdf (Texas Instruments, 2008).

79. *DLP Technology and Products.* Technical report. https://www.ti.com/lit/ml/dlpb010e/dlpb010e.pdf (Texas Instruments, 2016).

80. Bregazzi, A., Janin, P., Dyer, S., McGilligan, J. P., Burrow, O., Riis, E., Uttamchandani, D., Bauer, R. & Griffin, P. F. Cold-atom shaping with MEMS scanning mirrors. *Opt. Lett.* **48,** 37–40. doi:10.1364/OL.475353 (2023).

81. Panuski, C. L., Christen, I., Minkov, M., Brabec, C. J., Trajtenberg-Mills, S., Griffiths, A. D., McKendry, J. J. D., Leake, G. L., Coleman, D. J., Tran, C., St Louis, J., Mucci, J., Horvath, C., Westwood-Bachman, J. N., Preble, S. F., Dawson, M. D., Strain, M. J., Fanto, M. L. & Englund, D. R. A full degree-of-freedom spatiotemporal light modulator. *Nat. Photonics* **16,** 834–842. doi:10.1038/s41566-022-01086-9 (2022).

82. Trajtenberg-Mills, S., ElKabbash, M., Brabec, C. J., Panuski, C. L., Christen, I. & Englund, D. LNoS: Lithium Niobate on Silicon Spatial Light Modulator. Preprint at http://arxiv.org/abs/2402.14608 (2024).

83. *DLP670S data sheet.* https://www.ti.com/lit/ds/symlink/dlp670s.pdf (Texas Instruments, 2022).

84. *DLP7000 data sheet.* https://www.ti.com/lit/ds/symlink/dlp7000.pdf (Texas Instruments, 2023).

85. Deng, M.-J., Zhao, Y.-Y., Liang, Z.-X., Chen, J.-T., Zhang, Y. & Duan, X.-M. Maximizing energy utilization in DMD-based projection lithography. *Opt. Express* **30,** 4692–4705. doi:10.1364/OE.448231 (2022).

86. Chao, B., Gopakumar, M., Choi, S. & Wetzstein, G. High-brightness holographic projection. *Opt. Lett.* **48,** 4041–4044. doi:10.1364/OL.489617 (2023).

87. Zhang, Z., You, Z. & Chu, D. Fundamentals of phase-only liquid crystal on silicon (LCOS) devices. *Light Sci. Appl.* **3,** e213. doi:10.1038/lsa.2014.94 (2014).

88. *LCOS-SLM X15213 series.* Datasheet. https://www.hamamatsu.com/content/dam/hamamatsu-photonics/sites/documents/99_SALES_LIBRARY/lpd/x15213_E.pdf (Hamamatsu Photonics, 2024).

89. *GAEA-2 10 Megapixel Phase Only LCOS-SLM.* https://holoeye.com/products/spatial-light-modulators/gaea-2-phase-only/ (Holoeye Photonics, 2023).

90. *Spatial Light Modulator - 1024 x 1024.* Datasheet. https://photonics.laser2000.co.uk/wp-content/uploads/2022/10/SLM-HSP-UHSP-1024-x-1024-Data-Sheet-Rev.0822.pdf (Meadowlark Optics, 2022).

91. Mansha, S., Moitra, P., Xu, X., Mass, T. W. W., Veetil, R. M., Liang, X., Li, S.-Q., Paniagua-Domínguez, R. & Kuznetsov, A. I. High resolution multispectral spatial light modulators based on tunable Fabry-Perot nanocavities. *Light Sci. Appl.* **11,** 141. doi:10.1038/s41377-022-00832-6 (2022).

92. Bruce, G. D., Mayoh, J., Smirne, G., Torralbo-Campo, L. & Cassettari, D. A smooth, holographically generated ring trap for the investigation of superfluidity in ultracold atoms. *Phys. Scr.* **2011,** 014008. doi:https://dx.doi.org/10.1088/0031-8949/2011/T143/014008 (2011).

93. *2K 0.94" M180 FLCoS Display.* https://www.kopin.com/technologies-products/commercially-available-products/spatial-light-modulators/2k-0-94-m180-flcos-display/ (Kopin, 2024).

94. Gaunt, A. L. & Hadzibabic, Z. Robust digital holography for ultracold atom trapping. *Sci. Rep.* **2,** 721. doi:10.1038/srep00721 (2012).

95. Harte, T., Bruce, G. D., Keeling, J. & Cassettari, D. Conjugate gradient minimisation approach to generating holographic traps for ultracold atoms. *Opt. Express* **22,** 26548–26558. doi:10.1364/OE.22.026548 (2014).

96. Pasienski, M. & DeMarco, B. A high-accuracy algorithm for designing arbitrary holographic atom traps. *Opt. Express* **16,** 2176–2190. doi:10.1364/OE.16.002176 (2008).

97. Bowman, D., Harte, T. L., Chardonnet, V., Groot, C. D., Denny, S. J., Goc, G. L., Anderson, M., Ireland, P., Cassettari, D. & Bruce, G. D. High-fidelity phase and amplitude control of phase-only computer generated holograms using conjugate gradient minimisation. *Opt. Express* **25,** 11692–11700. doi:10.1364/OE.25.011692 (2017).

98. Ronzitti, E., Guillon, M., de Sars, V. & Emiliani, V. LCoS nematic SLM characterization and modeling for diffraction efficiency optimization, zero and ghost orders suppression. *Opt. Express* **20,** 17843–17855. doi:10.1364/OE.20.017843 (2012).

99. Wang, Z., Zhang, S., Baladron-Zorita, O., Hellmann, C. & Wyrowski, F. Application of the semi-analytical Fourier transform to electromagnetic modeling. *Opt. Express* **27,** 15335–15350. doi:10.1364/OE.27.015335 (2019).

100. Heintzmann, R., Loetgering, L. & Wechsler, F. Scalable angular spectrum propagation. *Optica* **10,** 1407–1416. doi:10.1364/OPTICA.497809 (2023).

101. Zhang, H., Xie, J., Liu, J. & Wang, Y. Elimination of a zero-order beam induced by a pixelated spatial light modulator for holographic projection. *Appl. Opt.* **48,** 5834–5841. doi:10.1364/AO.48.005834 (2009).

102. Moser, S., Ritsch-Marte, M. & Thalhammer, G. Model-based compensation of pixel crosstalk in liquid crystal spatial light modulators. *Opt. Express* **27,** 25046–25063. doi:10.1364/OE.27.025046 (2019).

103. Martínez, J. L., Moreno, I., Sánchez-López, M. d. M., Vargas, A. & García-Martínez, P. Analysis of multiple internal reflections in a parallel aligned liquid crystal on silicon SLM. *Opt. Express* **22,** 25866–25879. doi:10.1364/OE.22.025866 (2014).

104. Pushkina, A. A., Costa-Filho, J. I., Maltese, G. & Lvovsky, A. I. Comprehensive model and performance optimization of phase-only spatial light modulators. *Meas. Sci. Technol.* **31,** 125202. doi:https://dx.doi.org/10.1088/1361-6501/aba56b (2020).

105. Buske, P., Hofmann, O., Bonnhoff, A., Stollenwerk, J. & Holly, C. High fidelity laser beam shaping using liquid crystal on silicon spatial light modulators as diffractive neural networks. *Opt. Express* **32,** 7064–7078. doi:10.1364/OE.507630 (2024).

106. Wang, X.-L., Ding, J., Ni, W.-J., Guo, C.-S. & Wang, H.-T. Generation of arbitrary vector beams with a spatial light modulator and a common path interferometric arrangement. *Opt. Lett.* **32,** 3549–3551. doi:10.1364/OL.32.003549 (2007).

107. Zhu, W., Gao, F., Fu, Q., Zhou, X., Xie, Y., Zhang, B. & Kumar, S. Multi-Mode Vector Light Field Generation Using Modified Off-Axis Interferometric Holography and Liquid Crystal Spatial Light Modulators. *Photonics* **11,** 33. doi:10.3390/photonics11010033 (2024).

108. Sánchez-López, M. M., Davis, J. A., Moreno, I., Cofré, A. & Cottrell, D. M. Gouy phase effects on propagation of pure and hybrid vector beams. *Opt. Express* **27,** 2374–2386. doi:10.1364/OE.27.002374 (2019).

109. Saleh, B. E. A. & Teich, M. C. *Fundamentals of photonics* Third edition. *Wiley series in pure and applied optics* (Wiley, 2019).

110. Goodman, J. W. *Introduction to Fourier optics* (Macmillan Learning, 2017).

111. Schroff, P., La Rooij, A., Haller, E. & Kuhr, S. Accurate holographic light potentials using pixel crosstalk modelling. *Sci. Rep.* **13,** 3252. doi:10.1038/s41598-023-30296-6 (2023).

112. Bruce, G. D., Johnson, M. Y., Cormack, E., Richards, D. A., Mayoh, J. & Cassettari, D. Feedback-enhanced algorithm for aberration correction of holographic atom traps. *J. Phys. B: At. Mol. Opt. Phys.* **48,** 115303. doi:https://dx.doi.org/10.1088/0953-4075/48/11/115303 (2015).

113. Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* **13,** 600–612 (2004).

114. Clark, T. W., Offer, R. F., Franke-Arnold, S., Arnold, A. S. & Radwell, N. Comparison of beam generation techniques using a phase only spatial light modulator. *Opt. Express* **24,** 6249–6264. doi:`10.1364/OE.24.006249` (2016).

115. Wang, Z. & Bovik, A. C. Mean squared error: Love it or leave it? A new look at Signal Fidelity Measures. *IEEE Signal Process. Mag.* **26,** 98–117. doi:`10.1109/MSP.2008.930649` (2009).

116. Bowman, D. *Ultracold Atoms in Flexible Holographic Traps.* PhD thesis (University of St. Andrews, 2018).

117. Gerchberg, R. W. & Saxton, W. O. Practical Algorithm for the Determination of Phase From Image and Diffraction Plane Pictures. *Optik* **35,** 237–246 (1972).

118. Kim, D., Keesling, A., Omran, A., Levine, H., Bernien, H., Greiner, M., Lukin, M. D. & Englund, D. R. Large-scale uniform optical focus array generation with a phase spatial light modulator. *Opt. Lett.* **44,** 3178–3181. doi:`10.1364/OL.44.003178` (2019).

119. Polak, E. & Ribiere, G. Note sur la convergence de méthodes de directions conjuguées. *Rev. Fr. Inform. Rech. Opér.* **3,** 35–43. doi:`10.1051/m2an/196903R100351` (1969).

120. Feinman, R. Pytorch-minimize: a library for numerical optimization with autograd. `https://github.com/rfeinman/pytorch-minimize` (2021).

121. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. & Chintala, S. *PyTorch: An Imperative Style, High-Performance Deep Learning Library* in *Advances in Neural Information Processing Systems* (eds Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d., Fox, E. & Garnett, R.) **32** (Curran Associates, Inc., 2019).

122. Senthilkumaran, P. *Singularities in Physics and Engineering* doi:`https://dx.doi.org/10.1088/978-0-7503-1698-9` (IOP Publishing, 2018).

123. Senthilkumaran, P., Wyrowski, F. & Schimmel, H. Vortex stagnation problem in iterative Fourier transform algorithms. *Opt. Lasers Eng.* **43,** 43–56. doi:10.1016/j.optlaseng.2004.06.002 (2005).

124. Supikov, A., Nentedem, M. M., Grunnet-Jepsen, A. & Azuma, R. T. TWINS: improved spatial and angular phase calibration for holography. *Appl. Opt.* **62,** 7860–7867. doi:10.1364/AO.499387 (2023).

125. Fuentes, J. L. M., Fernández, E. J., Prieto, P. M. & Artal, P. Interferometric method for phase calibration in liquid crystal spatial light modulators using a self-generated diffraction-grating. *Opt. Express* **24,** 14159–14171. doi:10.1364/OE.24.014159 (2016).

126. Liu, Z.-G. & Mattina, M. *Learning Low-precision Neural Networks without Straight-Through Estimator (STE)* in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* (2019), 3066–3072. doi:10.24963/ijcai.2019/425.

127. *Timing spec of LCOS-SLM.* Unpublished confidential document (Hamamatsu, 2022).

128. Lanczos, C. Trigonometric Interpolation of Empirical and Analytical Functions. *J. Math. Phys.* **17,** 123–199. doi:10.1002/sapm1938171123 (1938).

129. Ronneberger, O., Fischer, P. & Brox, T. *U-Net: Convolutional Networks for Biomedical Image Segmentation* in *Medical Image Computing and Computer-Assisted Intervention* (eds Navab, N., Hornegger, J., Wells, W. M. & Frangi, A. F.) (Springer International Publishing, 2015), 234–241. doi:10.1007/978-3-319-24574-4_28.

130. Swan, H., Torchylo, A., Graaff, M. J. V. d., Rudolph, J. & Hogan, J. M. High-fidelity holographic beam shaping with optimal transport and phase diversity. *Optics Express* **33,** 6290–6303. doi:10.1364/OE.540901 (2025).

131. Van Bijnen, R. *Quantum engineering with ultracold atoms.* PhD thesis (Technische Universiteit Eindhoven, 2013). doi:10.6100/IR754785.

132. Geiger, A., Moosmann, F., Car, Ö. & Schuster, B. *Automatic camera and range sensor calibration using a single shot* in *2012 IEEE International Conference on Robotics and Automation* (2012), 3936–3943. doi:10.1109/ICRA.2012.6224570.

133. Milewski, G., Engström, D. & Bengtsson, J. Diffractive optical elements designed for highly precise far-field generation in the presence of artifacts typical for pixelated spatial light modulators. *Appl. Opt.* **46,** 95–105. doi:10.1364/AO.46.000095 (2007).

134. Persson, M., Engström, D. & Goksör, M. Reducing the effect of pixel crosstalk in phase only spatial light modulators. *Opt. Express* **20,** 22334–22343. doi:10.1364/OE.20.022334 (2012).

135. Guesmi, M. & Žídek, K. Calibration of the pixel crosstalk in spatial light modulators for 4f pulse shaping. *Appl. Opt.* **60,** 7648–7652. doi:10.1364/AO.434309 (2021).

136. Moreno, I., Sánchez-López, M. D. M., Davis, J. A. & Cottrell, D. M. Simple method to evaluate the pixel crosstalk caused by fringing field effect in liquid-crystal spatial light modulators. *J. Eur. Opt. Soc.: Rapid Publ.* **17,** 27. doi:10.1186/s41476-021-00174-7 (2021).

137. Kim, H., Kim, M., Lee, W. & Ahn, J. Gerchberg-Saxton algorithm for fast and efficient atom rearrangement in optical tweezer traps. *Opt. Express* **27,** 2184–2196. doi:10.1364/OE.27.002184 (2019).

138. Schroff, P., La Rooij, A., Haller, E. & Kuhr, S. *Accurate Holographic Light Potentials For Cold-Atom Experiments* in *Optica Imaging Congress (3D, COSI, DH, FLatOptics, IS, pcAOP)* (Optica Publishing Group, 2023), HM1D.5. doi:10.1364/DH.2023.HM1D.5.

139. Etchepareborda, P., Connolly, P., Henneberger, J. & Crosier, J. *Semi-analytical angular spectrum method in lensless digital inline holography: effects on resolution and detection of microparticles* in *Unconventional Imaging and Adaptive Optics* **12239** (SPIE, 2022), 173–178. doi:10.1117/12.2642760.

140. Schroff, P., Haller, E., Kuhr, S. & Rooij, A. L. Rapid stochastic spatial light modulator calibration and pixel crosstalk optimization. *Opt. Express* **32,** 48957–48971. doi:10.1364/OE.539548 (2024).

141. Li, J., Du, Y., Fan, C., Zhao, R., Hu, X., Wan, J., Yang, X., Cheng, H., Hu, Z., Zhao, Z. & Zhao, H. Fast and robust calibration method of liquid-crystal spatial light modulator based on polarization multiplexing. *Opt. Express* **31,** 25635–25647. doi:10.1364/OE.496392 (2023).

142. Lee, M., Koo, D. & Kim, J. Simple and fast calibration method for phase-only spatial light modulators. *Opt. Lett.* **48,** 5–8. doi:10.1364/OL.473703 (2023).

143. Zhao, Z., Xiao, Z., Zhuang, Y., Zhang, H. & Zhao, H. An interferometric method for local phase modulation calibration of LC-SLM using self-generated phase grating. *Rev. Sci. Instrum.* **89,** 083116. doi:10.1063/1.5031938 (2018).

144. Zhao, Y., Yan, W., Gao, Y., Yuan, Z., Ren, Z.-C., Wang, X.-L., Ding, J. & Wang, H.-T. High-Precision Calibration of Phase-Only Spatial Light Modulators. *IEEE Photonics J.* **14,** 1–8. doi:10.1109/JPHOT.2021.3129082 (2022).

145. Li, R. & Cao, L. Progress in Phase Calibration for Liquid Crystal Spatial Light Modulators. *Appl. Sci.* **9,** 2012. doi:10.3390/app9102012 (2019).

146. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. Preprint at http://arxiv.org/abs/1412.6980 (2017).

147. Dai, G.-m. Zernike aberration coefficients transformed to and from Fourier series coefficients for wavefront representation. *Opt. Lett.* **31,** 501–503. doi:10.1364/OL.31.000501 (2006).

148. Eismann, J. S., Neugebauer, M., Mantel, K. & Banzer, P. Absolute characterization of high numerical aperture microscope objectives utilizing a dipole scatterer. *Light Sci. Appl.* **10,** 223. doi:10.1038/s41377-021-00663-x (2021).

149. Smartt, R. N. & Steel, W. H. Theory and Application of Point-Diffraction Interferometers. *Jpn. J. Appl. Phys.* **14,** 351. doi:10.7567/JJAPS.14S1.351 (1975).

150. Evered, S. J., Bluvstein, D., Kalinowski, M., Ebadi, S., Manovitz, T., Zhou, H., Li, S. H., Geim, A. A., Wang, T. T., Maskara, N., Levine, H., Semeghini, G., Greiner, M., Vuletić, V. & Lukin, M. D. High-fidelity parallel entangling gates on a neutral-atom quantum computer. *Nature* **622,** 268–272. doi:10.1038/s41586-023-06481-y (2023).

151. Cornish, S. L., Tarbutt, M. R. & Hazzard, K. R. A. Quantum Computation and Quantum Simulation with Ultracold Molecules. Preprint at http://arxiv.org/abs/2401.05086 (2024).

152. Hannebelle, M. T. M., Raeth, E., Leitao, S. M., Lukeš, T., Pospíšil, J., Toniolo, C., Venzin, O. F., Chrisnandy, A., Swain, P. P., Ronceray, N., Lütolf, M. P., Oates, A. C., Hagen, G. M., Lasser, T., Radenovic, A., McKinney, J. D. & Fantner, G. E. Open-source microscope add-on for structured illumination microscopy. *Nat. Commun.* **15,** 1550. doi:10.1038/s41467-024-45567-7 (2024).

153. Zhang, C., Yu, B., Lin, F., Samanta, S., Yu, H., Zhang, W., Jing, Y., Shang, C., Lin, D., Si, K., Gong, W. & Qu, J. Deep tissue super-resolution imaging with adaptive optical two-photon multifocal structured illumination microscopy. *PhotoniX* **4,** 38. doi:10.1186/s43074-023-00115-2 (2023).

154. Harley, W. S., Li, C. C., Toombs, J., O'Connell, C. D., Taylor, H. K., Heath, D. E. & Collins, D. J. Advances in biofabrication techniques towards functional bioprinted heterogeneous engineered tissues: A comprehensive review. *Bioprinting* **23,** e00147. doi:10.1016/j.bprint.2021.e00147 (2021).

155. Li, J., Yan, J., Jiang, L., Yu, J., Guo, H. & Qu, L. Nanoscale multi-beam lithography of photonic crystals with ultrafast laser. *Light Sci. Appl.* **12,** 164. doi:10.1038/s41377-023-01178-3 (2023).

156. *N-SF11* Datasheet. https://www.schott.com/shop/advanced-optics/en/Optical-Glass/N-SF11/c/glass-N-SF11 (Schott, 2022).

157. *N-SK2* Datasheet. https://www.schott.com/shop/advanced-optics/en/Optical-Glass/N-SK2/c/glass-N-SK2 (Schott, 2022).