ABSTRACT

Title of dissertation: MODELING AND EXPERIMENTAL TECHNIQUES TO DEMONSTRATE NANOMANIPULATION WITH OPTICAL TWEEZERS

Arvind K. Balijepalli, Doctor of Philosophy, 2011

Dissertation directed by: Professor S. K. Gupta
Department of Mechanical Engineering

Dr. Thomas W. LeBrun
National Institute of Standards and Technology

The development of truly three-dimensional nanodevices is currently impeded by the absence of effective prototyping tools at the nanoscale. Optical trapping is well established for flexible three-dimensional manipulation of components at the microscale. However, it has so far not been demonstrated to confine nanoparticles, for long enough time to be useful in nanoassembly applications. Therefore, as part of this work we demonstrate new techniques that successfully extend optical trapping to nanoscale manipulation.

In order to extend optical trapping to the nanoscale, we must overcome certain challenges. For the same incident beam power, the optical binding forces acting on a nanoparticle within an optical trap are very weak, in comparison with forces acting on microscale particles. Consequently, due to Brownian motion, the nanoparticle often exits the trap in a very short period of time. We improve the performance of

optical traps at the nanoscale by using closed-loop control. Furthermore, we show through laboratory experiments that we are able to localize nanoparticles to the trap using control systems, for sufficient time to be useful in nanoassembly applications, conditions under which a static trap set to the same power as the controller is unable to confine a same-sized particle.

Before controlled optical trapping can be demonstrated in the laboratory, key tools must first be developed. We implement Langevin dynamics simulations to model the interaction of nanoparticles with an optical trap. Physically accurate simulations provide a robust platform to test new methods to characterize and improve the performance of optical tweezers at the nanoscale, but depend on accurate trapping force models. Therefore, we have also developed two new laboratory-based force measurement techniques that overcome the drawbacks of conventional force measurements, which do not accurately account for the weak interaction of nanoparticles in an optical trap. Finally, we use numerical simulations to develop new control algorithms that demonstrate significantly enhanced trapping of nanoparticles and implement these techniques in the laboratory.

The algorithms and characterization tools developed as part of this work will allow the development of optical trapping instruments that can confine nanoparticles for longer periods of time than is currently possible, for a given beam power. Furthermore, the low average power achieved by the controller makes this technique especially suitable to manipulate biological specimens, but is also generally beneficial to nanoscale prototyping applications. Therefore, capabilities developed as part of this work, and the technology that results from it may enable the prototyping of three-dimensional nanodevices, critically required in many applications.

MODELING AND EXPERIMENTAL TECHNIQUES TO DEMONSTRATE NANOMANIPULATION WITH OPTICAL TWEEZERS

by

Arvind K. Balijepalli

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2011

Advisory Committee:
Professor Satyandra K. Gupta, Chair/Advisor
Dr. Thomas W. LeBrun, Co-advisor
Professor Abhijit Dasgupta
Professor Michael Zachariah
Assistant Professor Santiago Solares
Professor Amitabh Varshney (Dean's representative)

Acknowledgments

I owe special thanks and gratitude to all the people who have supported me through this dissertation and made my graduate experience significant.

First, I would like to thank my advisor at the National Institute of Standards and Technology, Dr. Thomas LeBrun for his extraordinary support and mentoring. His patient guidance and critical insight when discussing ideas and research directions have played a large part in shaping this dissertation. I cannot say enough about how much I have learned from him and am very appreciative for the opportunity to work with him.

I would also like to thank my thesis advisor, Professor S. K. Gupta for his guidance and unique insight that has helped shape this work. Sincere thanks to Assistant Professor Santiago Solares for his helpful discussions about simulations and agreeing to serve on my committee. I would like to thank Professor Abhijit Dasgupta, Professor Michael Zachariah and Professor Amitabh Varshney for their valuable time reviewing my dissertation and agreeing to serve on my committee.

My colleagues and friends at NIST have made this experience pleasurable for which I will always be grateful. I would like to thank Dr. Jason Gorman for his help in setting up and running experiments in the laboratory, countless discussions and friendship. I also owe a special thanks to colleagues, Cedric, Joe, Richard, Didier, Yannick and Olivier, who over the years have provided helpful discussions, help with experiments and software development.

I owe special appreciation to my friends at the University of Maryland, Dr.

Arvind Ananthanarayanan and Dr. Wojciech Bejgerowski for their camaraderie, motivation and much needed distractions.

I am especially thankful to my parents for their constant encouragement, support and love throughout my life. I am incredibly lucky to have them. I am also grateful for my family and friends who have been very supportive and often provided me with motivation and strength.

Last but not least, I am grateful to my wonderful wife, Cathy for her unwavering backing and encouragement, especially through these last months. Her constant motivation, warmth and affection have played an important part in my life and helped me complete this dissertation.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ADC	Analog to Digital Converter
AFM	Atomic Force Microscope
AOD	Acousto-optic Deflector
API	Application Programming Interface
BD	Brownian Dynamics
BFP	Back Focal Plane
CCD	Charge-coupled Device
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DAC	Digital to Analog Converter
DIC	Differential Interference Contrast
DNA	Deoxyribonucleic Acid
DSP	Digital Signal Processors
EOM	Electro-optic Modulator
FPGA	Field Programmable Gate Array
GLMT	Generalized Lorentz Mie Theory
GPU	Graphics Processing Unit
HDL	Hardware Description Language
LBD	L'Ecuyer Generator with Bays-Durham Shuffle
LUT	Lookup Table
MEMS	Microelectromechanical Systems
MD	Molecular Dynamics
MRI	Magnetic Resonance Imaging
MT	Mersenne Twister
MUX	Multiplex
NA	Numerical Aperture
NEMS	Nanoelectromechanical Systems
NP	Nano-positioner
OT	Optical Tweezers
PD	Photo-diode
PDE	Partial Differential Equations
PDF	Probability Density Function
PGA	Programmable Gain Amplifier
PID	Proportional Integral Derivative
QPD	Quadrant Photo-Diode
RNA	Ribonucleic Acid
RNAP	Ribonucleic Acid Polymerase
ROM	Read Only Memory
RNG	Random Number Generators
SPM	Scanning Probe Methods
TEM	Transverse Electric and Magnetic
USB	Universal Serial Bus
USRP	Universal Software Radio Peripheral
XML	Extensible Markup Language

Chapter 1

Introduction

New techniques have been developed to address key challenges that currently prevent optical tweezers from being widely adopted as a prototype and test tool to fabricate nanodevices. Flexible and fast prototyping tools are an essential component when developing and testing novel designs of nanoscale devices and can accelerate their inclusion in commercial products. Our ability to prototype even basic three-dimensional designs is severely limited due to the rudimentary state of current nanoassembly tools. Optical trapping, which uses light to hold components and move them to their assembly locations, can provide a powerful, yet flexible tool for nanoassembly without some of the drawbacks inherent in other competing techniques that are discussed later in this chapter. At the microscale, optical trapping has been successfully demonstrated, to perform pick and place manipulation of multiple components, in parallel and arrange them in non-trivial three-dimensional patterns – a capability that is currently unparalleled by other assembly techniques.

However, optical trapping has so far not been extended to nanoscale assembly. The behavior of nanoparticles in an optical trap is qualitatively different than larger particles, for a constant incident power and beam geometry. For example, nanoparticles experience significantly weaker optical binding forces in comparison with larger particles. Consequently, as a result of Brownian motion, nanoparticles escape the trap easily. Therefore, confining a nanoparticle to an optical trap requires the use of very high laser powers, which can then result in damage to the specimen.

In this work, we have developed new algorithms and characterization tools that collectively improve the performance of optical traps at the nanoscale. We have implemented a closed loop control system that confines nanoparticles to an optical trap, when a static trap set to the same power is unable to do so. Moreover, the control algorithms we develop are able to confine particles to the trap for tens of seconds, so as to be useful for nanoassembly operations, and for a low average beam power.

Figure 1.1 gives an overview of the techniques and tools developed as part of this work to enable the manipulation of nanoparticles. From the figure, we see that in order to implement enhanced trapping techniques in the laboratory, we must first develop a combination of theoretical (THR), simulation (SIM) and laboratory-based (LAB) tools as described below.

We improve the trapping of nanoparticles by using a control system, which allows us to confine a particle to the trap for long times, so it can be useful in nanoassembly applications. However, developing control algorithms directly in the laboratory is often time-consuming, but can be made more efficient through the use of numerical simulations. Therefore, we have developed physically accurate simulations that model the diffusion of nanoparticles in a fluid under the influence of binding forces from an optical trap. We have designed and implemented a simulation framework that allows us to easily test control algorithms and play a central role

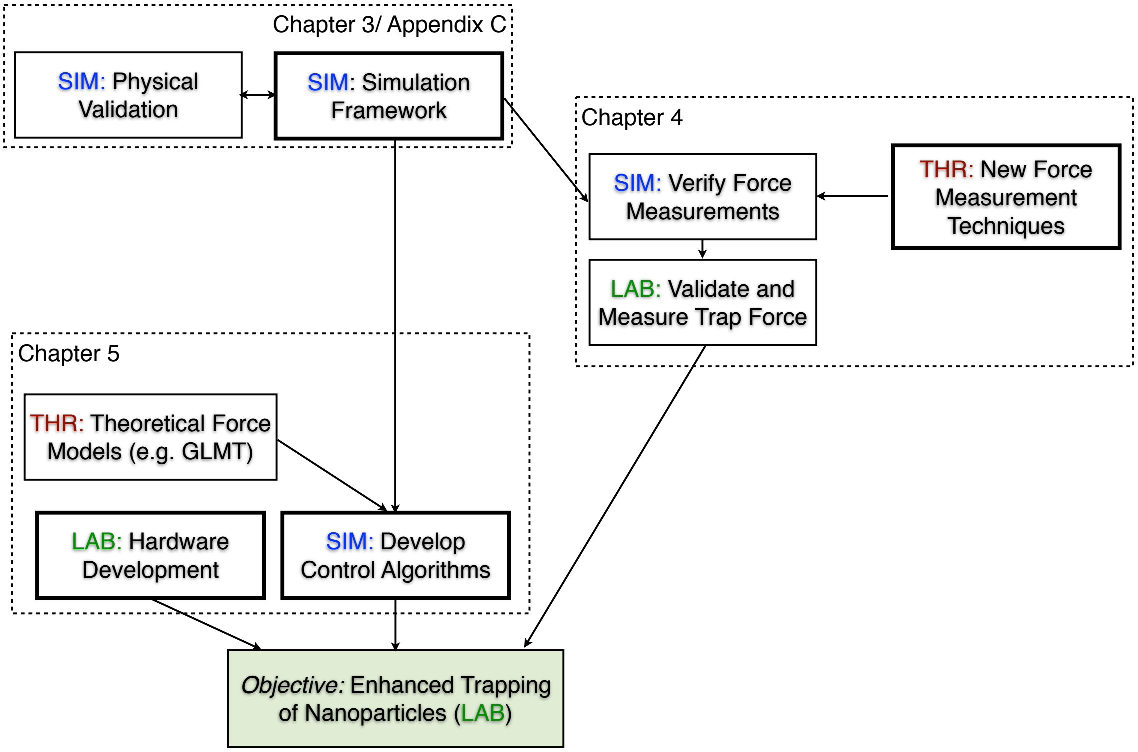


Figure 1.1: Overview of tools and techniques developed for enhanced trapping of nanoparticles. Enhanced nanomanipulation techniques are developed using a combination of theoretical (THR), simulation (SIM) and laboratory-based (LAB) tools.

in key aspects of developing this approach as seen in Figure 1.1. For simulations to be effective, we must also include an accurate force model of the optical trap. Therefore, we have developed methods within our simulation framework to allow the inclusion of tabulated force data from a variety of experimental and theoretical sources.

Existing theoretical models do not sufficiently capture the non-ideal behavior in an optical trap, which can arise from spherical aberrations or diffraction. On the other hand, current experimental techniques, which are designed for larger particles, do not work very well for nanoparticles that experience a very weak interaction with the trap. Therefore, we have developed two new techniques to directly measure the trapping force in the laboratory, which address the drawbacks of current methods in accurately measuring the trapping force experienced by nanoparticles. We test these methods using simulations (see Figure 1.1) and then implement them in the laboratory.

Finally, we develop control algorithms that allow nanoparticles to be confined within an optical trap, for much longer times than is currently possible, using comparably lower power than a static trap. We test new algorithms using numerical simulations and successfully demonstrated them in the laboratory. Consequently, tools developed as part of this research will provide the basic framework required to allow optical trapping to evolve into a practical prototyping and test tool for nanoassembly.

1.1 Motivation

The ability to rapidly prototype novel nanodevices from a wide selection of materials and arbitrarily shaped components is necessary for the rapid and sustained advancement of nanotechnology. Similar to rapid prototyping methods at the macroscale, prototyping tools with quick turn-around times will allow us to test the feasibility of new designs, in order to mass manufacture robust nanodevices economically. Unfortunately, current techniques (described in Section 1.2) that allow us to prototype even moderately complex three-dimensional designs are glaringly inadequate. Therefore, new techniques are required that provide the requisite capabilities to quickly and accurately prototype functional nanodevices.

Optical trapping is currently the only deterministic assembly technology – where we can pick and place multiple components, made from a variety of materials, in three-dimensions. Furthermore, it can be used for the simultaneous transport of multiple components by time-sharing a single optical beam [52]. Components with diverse shapes ranging from microspheres to nanowires and even biological cells and liposomes have been trapped and arranged into non-trivial patterns [3, 21]. Since optical tweezers use light to hold and manipulate individual components, they do not suffer from sticking between the probe and the particle or spatial interference between multiple probes, which can limit the effectiveness of other techniques such as scanning probe methods. Moreover optical tweezers is fully compatible with microfluidics systems and can be parallelized, if necessary, to be used in large throughput systems.

Optical trapping is currently limited to microscale assembly and has not yet been successfully demonstrated for small nanoparticles. Moreover, the flexibility that optical trapping provides at the microscale makes it an attractive candidate for nanoscale assembly. As described previously, it is challenging to keep a nanoparticle confined within an optical trap for extended periods of time. Increasing the power of the trapping beam can, up to a limit, improve the the localization of the nanoparticle close to the center of the trap and keep it confined within the trap for longer periods of time. However, the additional power absorbed by the particle can be sufficient to destroy it. Increased power from the optical beam can also significantly raise the temperature of the surrounding fluid, causing it to boil and thereby destabilize the trap. Therefore new techniques, which are able to take advantage of the flexibility offered by optical trapping, must be developed in order to improve the time nanoparticles can be confined within the trap, while minimizing the power input to the system.

1.2 Background

1.2.1 Applications of Nanotechnology

Nanotechnology holds tremendous promise to significantly impact everyday products. One area where nanotechnology's impact is already evident is in the design and manufacture of novel materials. Nanoscale additives, such as carbon nanotubes and colloidal gold or zinc oxide particles are commonly incorporated into polymer matrix materials to enhance their properties, as shown in Figure 1.2 [16]. Polymer composite materials that include carbon nanotubes have very high strength, while being much lighter than metals of comparable strength, driving their use in structural elements, especially in aerospace and automotive applications [99]. The ability of small quantities of nanoscale additives to radically alter basic physical properties of materials, such as their strength, flame resistance or electrical conductivity, has allowed unprecedented control over their properties. This in turn has led to the development of precisely engineered materials that suit the desired application.

Likewise planar semi-conductor devices, with characteristic dimensions of tens of nanometers, fabricated using well-established photolithography processes, have

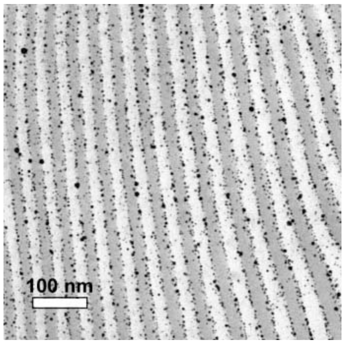


Figure 1.2: Cross-sectional TEM image of a symmetric polystyrene-b-poly(2-vinylpyridine) diblock copolymer containing gold nanoparticles [16].

revolutionized consumer electronics devices. Conventional photolithography techniques have also been used to develop microelectromechanical systems (MEMS), which have found use in everyday products, such as accelerometers and pressure sensors.

Nanotechnology has also started to make a significant impact in medicine, especially in the detection and treatment of cancer. Nanoparticles linked with cancer targeting ligands allow high contrast imaging and detection of cancers earlier than was possible before [5, 73]. Moreover in many cases nanoparticles have been used with existing imaging technologies in order to make these benefits available immediately. One example is the use of iron oxide nanoparticles as a contrast agent in magnetic resonance imaging (MRI), which has resulted in detailed three-dimensional images of tumors [140]. Nanotechnology can also play an important role in drug delivery. Current treatments for cancer are non-specific and significantly damage healthy tissue in addition to targeting cancerous cells. By linking drug nano-complexes with cancer targeting ligands, drugs can be tailored to target only cancerous cells, which greatly reduces the side effects from cancer treatments.

Nanotechnology has also been instrumental in developing new cancer therapies. In one example, gold nanoparticles, conjugated with a ligands which specifically attach to cancer cells are used to deliver nanoparticle selectively. Subsequently infrared radiation, which does not significantly heat biological matter is used to heat the nanoparticles and thereby destroy the cancerous cells, leaving surrounding healthy tissue undamaged [157]. New therapies can also include, specific anti-cancer agents, such as ribonucleic acid (RNA), enclosed in functionalized self-assembled bags made from phospholipid bi-layers called liposomes. The functionalized liposomes attach themselves to cancer cells and are subsequently absorbed into the cell. When the payload delivered by the liposome is a RNA strand, this can result in the expression of specific genes coded by the RNA, which can then induce apoptosis in the cancerous cells [121].

While several nanotechnology-based products are commercially available and many more are under development, almost all existing non-molecular devices are planar with designs that are largely limited by the capabilities of existing fabricating techniques. Next, we outline some of the nanofabrication techniques commonly used in manufacturing nanodevices.

1.2.2 Nanodevices: Manufacturing Methods

There are several available routes to manufacture of nanodevices. While they all offer distinct advantages, to the best of our knowledge, none of them are perfectly suitable for the prototyping and test of three-dimensional devices made from heterogeneous nanocomponents. Nanofabrication techniques can however be classified into bottom-up techniques and top-down techniques. In bottom-up methods, assemblies are built by bringing together individual building blocks to build the device. Bottom-up approaches are commonly used at the molecular level to self-assemble structures, one molecule at a time, but are also applicable for larger components using deterministic assembly techniques. On the other hand, top-down approaches can fabricate complicated structures by removing material from bulk substrates using conventional machining operations. Methods such as micromachining and lithography are prominent examples of top-down approaches. Key competing technologies that promote nanoscale fabrication and assembly are discussed below to highlight their relative strengths and weaknesses.

1.2.2.1 Self-Assembly

Self-assembly is a remarkable example of a bottom-up assembly technique that is commonly found in nature and may some day provide the key to the large scale manufacture of functional three-dimensional nanodevices. Self-assembly occurs routinely in biological cells and is responsible for many cellular functions. Examples of self-assembly in cells include the assembly of lipids to form the cell membrane, the assembly of single deoxyribonucleic acid (DNA) strands into double stranded helical DNA or the self-assembly of cellular proteins into secondary structures such as alpha helices or beta sheets and even tertiary and quaternary structures. However, self assembly has also been demonstrated at the nanoscale and even the macroscale, as shown in Figure 1.3 [165]. Self-assembly operates under the principle that interacting components can be driven to spontaneously form ordered assemblies when they interact under the suitable conditions. Specifically, self-assembly is successful when there is a suitable balance of attractive and repulsive forces between the individual components. [165]. Electrostatic forces are primarily responsible for self-assembly and these forces can arise from Coulomb interactions between the components. Self-assembly can therefore be guided by carefully engineering forces and by leveraging shape complementarity between interacting components.

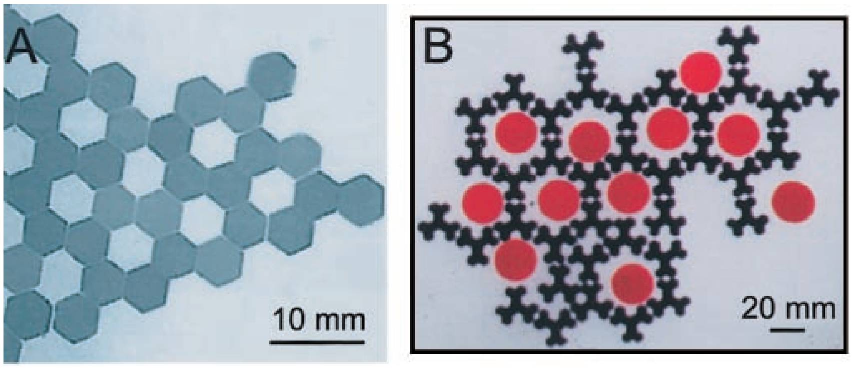


Figure 1.3: Examples of two-dimensional self assembly: (A) Open hexagonal array and (B) hexagonal lattice formed around circular templates [165].

Engineering the electrostatic forces requires the precise control of the chemistry of interacting components and generally involves optimizing many free system parameters. This makes self-assembly particularly challenging and the process has so far only been successful in creating ordered structures of identical components, whereas the assembly of truly three-dimensional nanodevices from heterogenous components has so far been elusive [10]. Finally, while self-assembly is a promising route to the eventual large scale manufacture of nanodevices spanning the nanoscale to the macroscale, the challenges in perfecting the chemistry that drives the process makes it's impact as a prototyping tool, to test and validate new device designs, less evident.

1.2.2.2 Scanning Probe Methods

Scanning probe methods (SPM) have also been used extensively in manipulating nanoscale components. Individual nanocomponents can be pulled or pushed by a single probe or gripped by two or more probes and then transported for assembly into a structure. Probe based methods such as scanning tunneling microscopes have been used to manipulate individual atoms, whereas atomic force microscopes (AFM), which were originally developed for imaging with atomic resolution, are more commonly used to manipulate components that range in size from a few nanometers to several micrometers in 2D. Probe-based manipulation methods offer several advantages over other techniques. Being a bottom-up approach, these techniques can be used to build complicated assemblies using heterogeneous building blocks. AFM has long been used to manipulate spherical nanoparticles, nanowires and other shapes made from a variety of materials into assembled structures, such as the example shown in Figure 1.4 [127]. Moreover since AFM was originally developed as an imaging tool, it is able to combine precise manipulation with extremely high resolution imaging of the workspace. The more recent development of MEMS technology has resulted in the miniaturization of SPM tools, which have several probes in parallel. This has since been extended to include large arrays of probes that can operate simultaneously for large throughput operation [88].

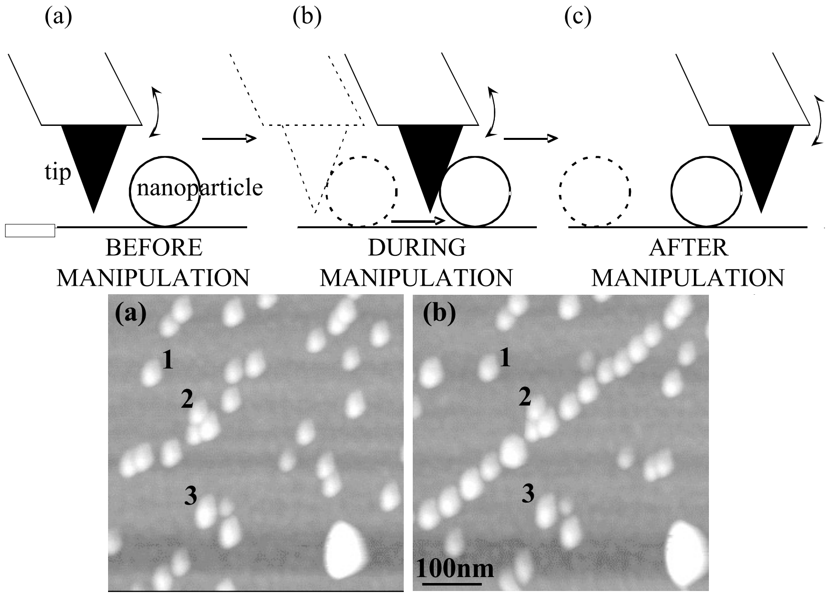


Figure 1.4: Examples of 2D AFM Manipulation: (top) pushing technique to manipulate microspheres and (B) example assemblies created using AFM [127].

However scanning probe methods also suffer from several drawbacks that prevent them from being widely used as a prototyping tool for three-dimensional nanostructures. Tools with only a single probe often rely on pushing components to the desired location, which prevents them from working outside a two-dimensional workspace and makes three-dimensional operation extremely difficult. Even in situations where two or more probes are used to grip components, out-of-plane movement of the components is difficult. Moreover, electrostatic forces at the nanoscale pose formidable challenges to releasing the particle at the desired location. Furthermore the manipulation of non-spherical shapes is also limited due to the need for several probes to simultaneously operate in a confined space to adequately support the specimen. Due to these drawbacks, dynamic three-dimensional assemblies of nanocomponents using AFM has so far not been achieved, rather probe-based methods are used to build stacked planar structures, one layer at a time [129].

1.2.2.3 Photolithography

Photolithography is a top-down approach that has been widely adopted when manufacturing devices with nanoscale features. In the simplest form of photolithographic processes, a substrate such as silicon is first coated with a uniform layer of photoresist, which cures when exposed to light of a certain wavelength, generally in the ultra-violet region of the electromagnetic spectrum. A photomask, which contains a negative of the pattern of the device being manufactured is then placed over the photoresist and then exposed to light, which causes regions in the photoresist that are part of the device to be cured. Any uncured photoresist is then removed, thereby exposing the substrate. Finally an etching process is used to machine the substrate leaving behind raised device features. The photolithography process steps are shown in Figure 1.5. The process is generally repeated several times to create multi-level features for use in functional devices. In addition, metals and other materials can be applied in thin layers at specific points in the process to create heterogenous device components.

Photolithography processes have been developed over several years and are widely used in the large scale manufacture of semi-conductor devices. Advances in photolithography have resulted in electronics with characteristic sizes of only tens of nanometers and this has resulted in ever more powerful computers that

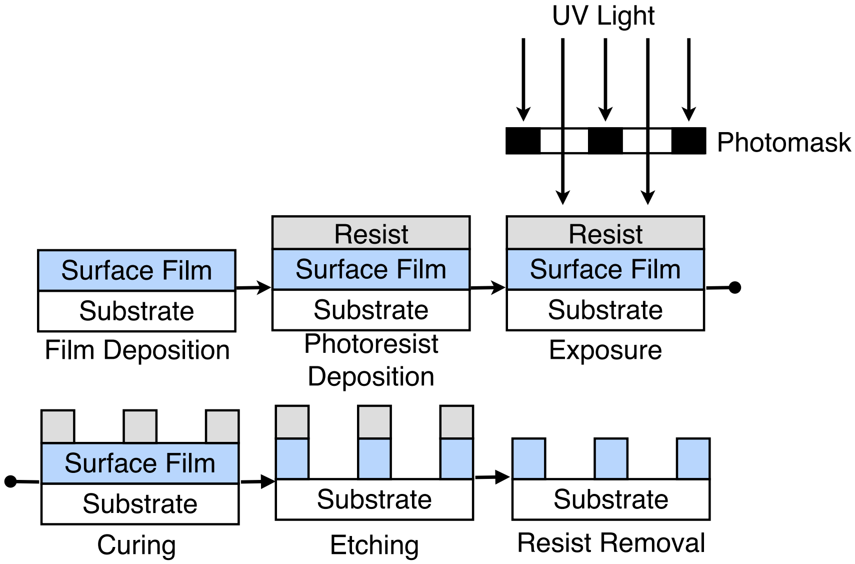


Figure 1.5: Steps in the photolithography process.

are more energy efficient and economical. Photolithography techniques have also been employed to develop MEMS and more recently nanoelectromechanical systems (NEMS). MEMS technology, in particular has resulted in a few commercial products such as accelerometers in automobile and consumer electronics,miniature sensors for pressure and temperature, as well as gyroscopes [89, 144]. Despite their success and widespread adoption, lithography methods are able to manufacture devices in individual planar layers that are stacked on top of each other.

1.3 Objectives and Outline

The primary objective of this work is to develop new tools to advance optical trapping towards nanoscale prototyping and test. One component of this tool set is new control algorithms that improve the amount of time nanoparticles spend in an optical trap, to enable nanoassembly, while minimizing the power. In order to realize such algorithms, we must develop new characterization tools to better understand the interactions of nanoparticles with weak optical traps. Developing new control algorithms directly in the laboratory can be challenging and time-consuming. Therefore we must also develop numerical simulations to assist in the discovery and validation of new laboratory methods. Therefore this work has the following objectives:

(i) Develop and implement new methods, such as control systems, to significantly improve the amount of time we can confine nanoparticles to the trap using low average powers.

(ii) Implement a simulation framework to allow the realization of new tools required to realize nanomanipulation with optical tweezers.

(iii) Develop new characterization tools, such as force measurement methods, to assist the development of enhanced nanomanipulation techniques for optical trapping.

In order to realize these objectives, we have developed several tools that work in conjunction, as seen from the overview in Figure 1.1. In Chapter 3, we describe numerical simulations based on Brownian dynamics to model the diffusive behavior of nanoparticles in a fluid, which allows the inclusion of an external force like that from an optical trap. As part of this work, we have also developed a modular simulation framework (described in Appendix C) that easily allows us to configure the simulation for multiple applications for use throughout this work, as shown in Figure 1.1.

An important component for developing new techniques for nanomanipulation is the inclusion of an accurate trapping force. Existing theoretical models and experimental techniques fall short of accurately describing the weak interactions between a nanoparticle and an optical trap. Therefore, in Chapter 4 we have developed two new methods to measure the optical binding force experienced by nanoparticles in the trap. The force measurement techniques are first validated using numerical simulations and then implemented in the laboratory, as seen from Figure 1.1.

Finally, we have successfully increased the trapping time of nanoparticles long enough to be useful for nanoassembly operations, with an average power at which a static trap is unable to localize particles. As discussed in Chapter 5, numerical simulations in conjunction with theoretical and experimental trapping force models have played a crucial role in guiding the discovery and development of new control techniques. New algorithms developed as part of this work will allow us to extend the trapping time of nanoparticles in an optical trap, thereby overcoming a key challenge to extending optical tweezers to nanoscale assembly. Moreover, the tools and methods developed as part of this work completed the basic toolset required to extend optical trapping for flexible nanoscale prototyping and test.

Chapter 2

Literature Review

2.1 Introduction

The ability of light and other electromagnetic radiation to exert a force on material objects was first predicted by Maxwell in 1874 and later demonstrated experimentally by Nichols and Hull and Lebedev [13]. However, the magnitude of force generated by radiation pressure is too small to significantly influence macro scale objects. With the invention of lasers, the use of radiation pressure to manipulate microscale objects became practical and was first demonstrated by Ashkin in 1970 [11]. The first optical traps were simple experiments, which levitated small objects using radiation pressure, to overcome gravitational forces. The expression for the force generated from the radiation pressure (F_r) is given in Equation 2.1, where n_m is the refractive index of the fluid surrounding the particle, c is the speed of light in vacuum, P is the power of the incident beam and Q is a dimensionless efficiency factor.

$$F_r = \frac{n_m P}{c} Q \tag{2.1}$$

The maximum radiation force is achieved when light is reflected by a perfect mirror opposite to the direction of propagation. In this case the efficiency factor in Equation 2.1, $Q = 2$ and for $1\,W$ of power, the maximum radiation force can be calculated to be approximately $10\,nN$. However, in most practical optical traps Q

is much smaller and the optical trapping forces acting on the particles in the trap are correspondingly smaller than the case of the perfectly reflecting mirror.

The simplest optical traps still use radiation pressure from a single beam to levitate small particles [11]. The position of the particle can then be controlled along the direction of propagation by varying the beam power. However, a more commonly used setup includes two opposing beams to confine the particle axially. Optical levitation works best with dielectric particles made with materials such as silica. This technique has also been successfully demonstrated using droplets of oil or water. In most experiments a laser power of $1\,W$ is typically used to levitate particles ranging in diameter from $1\,\mu m$ to $40\,\mu m$ [151].

The most common trap geometry (also the trap geometry used in this work) in optical trapping experiments is the single beam gradient force optical trap, which has been used to trap and manipulate a wide variety of particles such as microspheres, nanowires and biological molecules [52, 3]. In gradient optical traps, a tightly focussed optical beam is used to exert a force and effectively grip the particle to form a stable trap close to the focus of the beam. The shape of the beam results in the creation of two components, a gradient force component that pushes the particle towards the focus of the beam and a scattering component, which directs the particle along the direction of propagation of the beam. A stable optical trap is formed when the gradient force overcomes the scattering force. The gradient force acting on the trap varies as a function of the particle's position in the beam – the gradient force is zero at the center and approaches it's maximum value at a point some distance away from the center of the trap, before decreasing again. An example plot of the gradient optical force acting on a $100\,nm$ diameter gold nanoparticle, with $50\,mW$ of incident power using a trapping beam with a wavelength of $1064\,nm$ is shown in Figure 2.1. An ideal optical trap is symmetric about both sides of the center and consequently the trapping force varies symmetrically as well. Deviations from ideal behavior, such as misalignments in the optics can sometimes lead to asymmetries in the trapping force.

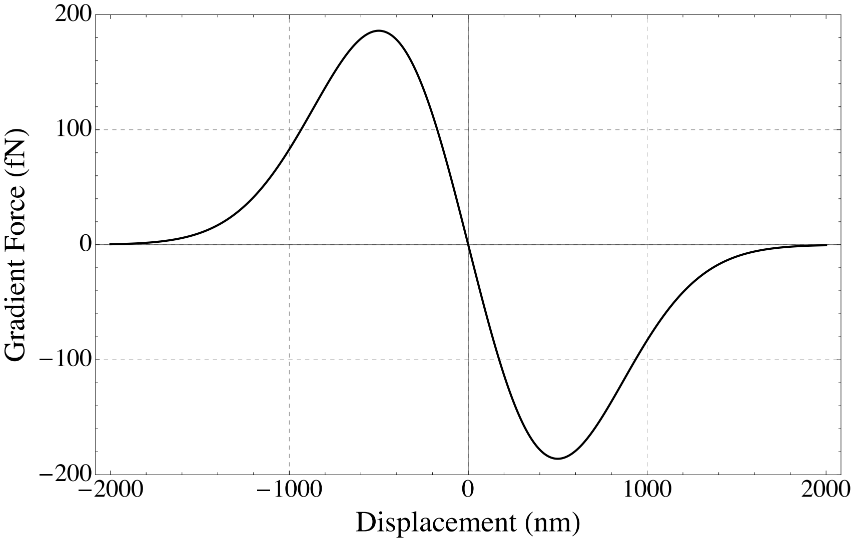


Figure 2.1: The gradient optical force experienced by a $100\,nm$ diameter gold nanoparticle.

The magnitude of the trapping force, which includes the sum of gradient and scattering force components, can be estimated using Equation 2.1 above, if the efficiency factor Q is known. The factor Q is then a function of the particle displacement, but also depends on other parameters such as the size of the particle, the size and geometry of the trapping beam and material properties of the nanoparticle and the surrounding fluid. The maximum Q, which accounts for scattering and gradient trapping efficiencies, for a $216\,nm$ diameter polystyrene particle in an optical trap with wavelength of the incident beam, $\lambda = 1064\,nm$, has been calculated theoretically to be 5×10^{-3}, at a distance of approximately $200\,nm$ from the center of the beam [136]. Therefore the maximum force can then be estimated for $1\,W$ of laser power using Equation 2.1 to be approximately $22\,pN$. Moreover we see from Equation 2.1 that the maximum force varies linearly with the power of the incident beam.

The mechanism by which single beam gradient force optical traps exert a force on microscopic particles is qualitatively described in Figure 2.2, using a geometric optics approximation, valid for particles that are considerably larger than the wavelength of the incident trapping beam [12]. Figure 2.2(a) describes the case where a particle is displaced laterally with respect to the trapping beam. The incident beam, shown in the figure has a Gaussian intensity profile. In the case of geometric optics, the beam is modeled as individual rays, where the thickness of each ray in the figure corresponds to it's intensity. Ignoring reflections at the surface, the refraction of light by the sphere causes a small momentum change in the light, which then induces an opposing momentum change in the sphere due to conservation laws. This in turn exerts a net force on the sphere, which can be decomposed into a scattering component, along the direction of propagation of the light and a gradient component, which moves the particle towards the center of the beam (these force components are not shown in Figure 2.2). A stable trap is formed when the contributions from each ray of light acting on the sphere are summed and the net gradient component of the force exceeds the disruptive scattering component. A similar analysis can be applied to the case where the sphere is displaced axially along the direction of propagation of the beam as shown in Figure 2.2(b). The net result is an optical trap that is stable in three-dimensions. The geometric optics description of optical trapping is only strictly valid for dielectric sphere that are significantly larger than the wavelength of the trapping light. On the other hand, the trapping force acting on particles that are very small in comparison with the wavelength of the incident can be calculated by treating the particle as a simple dipole oscillator [148]. However for most particles with sizes on the order of the wavelength of the incident light, the trapping force is estimated by calculating the momentum change between the incident electromagnetic field and the field obtained when light is scattered by the particle in the trap [23, 134]. One technique which analytically describes the scattering of light by a spherical particle is Mie theory, which has been used to calculate the trapping force for both metallic as well as dielectric particles [67].

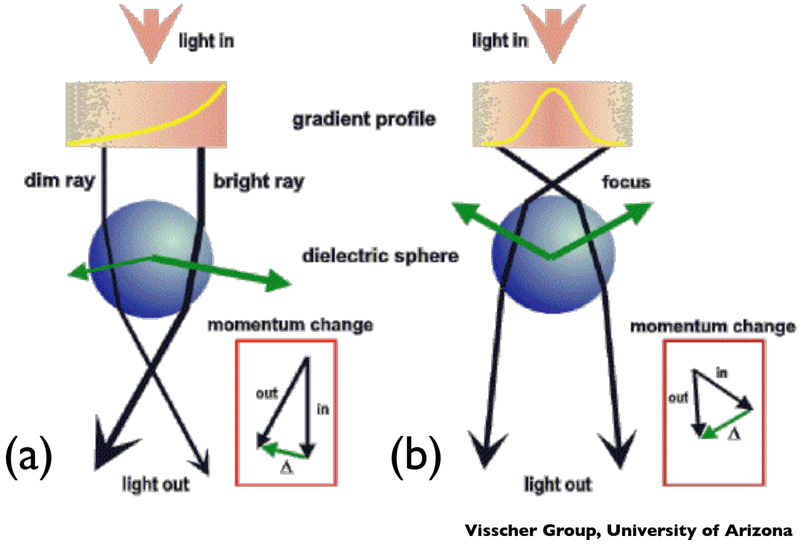


Figure 2.2: A qualitative description of optical trapping in the geometric optics regime, strictly valid for particles that are much larger than the wavelength of the incident light. Sub-figure (a) shows the case where the particle is displaced laterally from the center of the trap and (b) shows the trapping force generated due to the axial displacement of the particle.

Optical tweezers are most commonly operated in liquids such as water, although they can also operate in air. In either case, micro and nano components in the fluid undergo a random statistical fluctuation due to numerous collisions with the surrounding fluid molecules resulting in Brownian motion. Although the random motion of microscopic particles in fluids has long been observed, the discovery of Brownian motion is credited to Scottish botanist Robert Brown in 1827, who observed the random motion of pollen in water. The mathematical formalism of Brownian motion and diffusion was later developed, almost simultaneously, by Einstein in 1905 [50] and Smoluchowski in 1906 [160], who each independently described the relationship between the diffusion constant, D of a particle in a fluid and its mobility μ, which is the ratio of the particle's terminal drift velocity to an externally applied force. This relationship is shown in Equation 2.2, where $D = \frac{k_B T}{6\pi\eta a}$, k_B is Boltzmann's constant, η is the viscosity of water, as a function of it's temperature T and a is the radius of the particle. The Einstein-Smoluchowski relationship is built upon the observation that the random forces exerted by fluid molecules on the particle, are also responsible for a drag force acting on the particle as it is pulled through the fluid.

$$D = \mu\, k_B\, T \tag{2.2}$$

A common example of diffusion is the gradual mixing of a droplet of dye in a liquid. The droplet spreads as molecules of the dye diffuse into the liquid and the size of the droplet increases as the square root of time and can be described using the relationship given in Equation 2.3. The dependence of the spreading of the dye on the square root of time makes diffusion a very slow process. The diffusion process described for the droplet of dye also holds for nanoscale and microscale particles, which undergo Brownian motion in a fluid. For an ensemble of particles, starting the at the same location at time, $t = 0$, the spread of the particle distribution can also be modeled using Equation 2.3. Moreover, since the diffusion constant, D in Equation 2.3 is directly proportional to the temperature, the distribution of particles spreads more rapidly with an increase in temperature.

$$\sigma = \sqrt{2D\,t} \tag{2.3}$$

For a constant time, the size of a particle distribution, also called the diffusion length, increases with decreasing particle radius as shown in Figure 2.3, where the diffusion length is calculated at the end of one second. This implies that in a given time, smaller particles diffuse much farther than larger particles and this has direct consequences for nanoassembly using techniques such as optical trapping, described more below and also in later sections. Moreover, as expected from Equation 2.3, the diffusion length of a particle for a given time scales as $1/\sqrt{r}$, where r is the radius of the particle.

All nanoparticles undergo a random thermal motion inside an optical trap. Large particles, typically with diameters greater than several hundred nanometers experience very strong optical binding forces. Consequently, the force from the optical trap overcomes the random thermal motion experienced by the particle and it is therefore stably trapped. In other words, the optical trapping potential formed by the interaction of light with the microscale particle has a depth that is signifi-

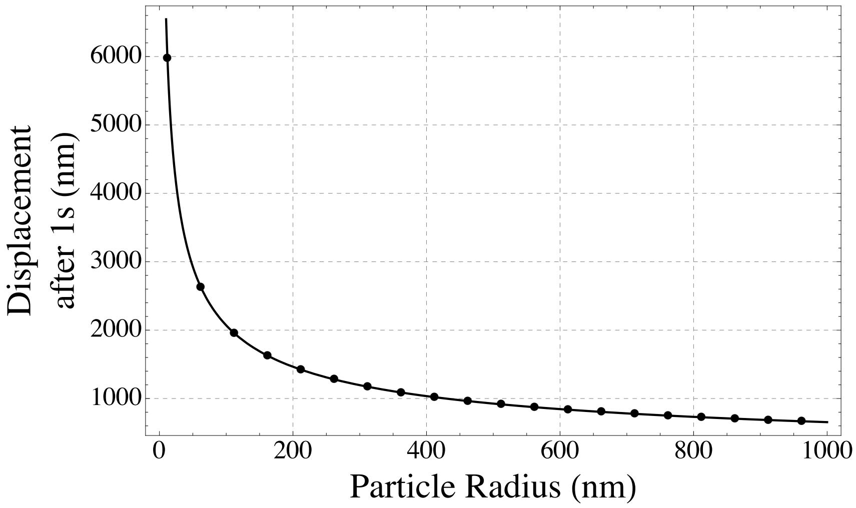


Figure 2.3: The diffusion length at the end of one second, as a function of particle radius is shown. The diffusion length for a constant time is found to scale as the inverse of the square root of particle size.

cantly larger than the thermal energy of the particle. However, the optical binding forces decrease rapidly with particle size and for particles with a diameter of only a few nanometers, the potential well depth can become comparable with the thermal energy of the particle. Moreover as seen in the discussion above, smaller particles diffuse much farther than larger particles in the same amount of time. Therefore weaker optical traps and faster diffusion together makes it significantly harder to trap small nanoparticles and keep them localized in the trap for long periods of time. Therefore techniques that improve the time nanoparticles can be confined in an optical trap are needed to adapt optical trapping for nanoscale manipulation. We demonstrate such techniques in the laboratory using a closed loop control system, discussed in more detail in Chapter 5.

2.2 Overview: Applications of Optical Tweezers

Single beam gradient optical traps are an important tool in many areas of research, but have been particularly important in biophysical experiments. The ability of optical tweezers to perform non-contact and non-invasive manipulation of micrometer sized particles, organelles and even entire cells has opened up new avenues of research in measurement and characterization of biological systems that were previously difficult to achieve. Moreover optical tweezers has enabled the precise measurement and metering of mechanical forces imparted to biological macromolecules so as to allow the careful characterization of the mechanical or kinetic properties of such systems. Visscher et. al. have used optical tweezers to directly measure the mechanical properties of biological systems, such as the force exerted by molecular motors [159]. Optical tweezers have also been used in cellular studies by attaching probes made from dielectric materials to biological cells and using these probes to measure the stiffness of the cell wall in red blood cells [103]. Block *et al.* have studied the mechanics of motion of kinesin, a common molecular motor responsible for vesicular transport along actin microtubules, by attaching a functionalized glass bead, held in an optical trap, to the kinesin molecules [34]. The motion of other molecular motors such as myosin and RNA polymerase, under an applied load have also been studied using optical trapping and used to characterize translocation rates, step sizes and pauses [143, 108, 130]. Optical trapping, in combination with video fluorescence microscopy has been used successfully to study the dynamics of the relaxation of stretched double stranded DNA in the absence of a stretching force

[118]. More recently the unfolding kinetics of folded single stranded RNA and DNA sequences have been studied by pulling on the folded strands using an optical trap [111, 69]. Biophysical experiments using optical trapping require an accurate measurement of the trapping force and often use closed-loop control to precisely control the amount of force applied to the biological systems. The methods employed for force measurements of microscale particles are discussed further in Section 2.4, while commonly used techniques for closed-loop feedback are outlined in Section 2.5.

Colloidal studies have benefitted greatly from the use of optical trapping to study the formation and properties of 3D crystal structures. Novel structures, not found naturally have been constructed using holographic optical tweezers [27]. Figure 2.4 shows an example of a three-dimensional face centered cubic (FCC) colloidal crystal, constructed using $3\,\mu m$ polystyrene spheres and which includes a line defect with a 60° bend used as waveguides in photonic devices. Such arrangements of colloids are significantly harder to reliably manufacture with self-assembly techniques, but are easily fabricated with optical trapping. The dispersion of uniform colloids into a solution under certain conditions results in their spontaneous self-assembly into crystal structures. Pertsinidis and Ling [120] have studied the effect of creating single point defects in 2D crystal structures using optical trapping. The diffusion constant obtained by tracking the motion of the defect through the colloidal crystal using image processing shows good agreement with simulations of this process [43] and can provide valuable insight into the dynamics of two-dimensional crystals. Optical tweezers have also been used for the repair of colloidal crystals manufactured using other techniques. Korda and Grier [83] have shown that it is possible to manufacture large defect free colloidal crystals by annealing a defect region in a crystal using localized heating from an optical trap and subsequently allowing relaxation.

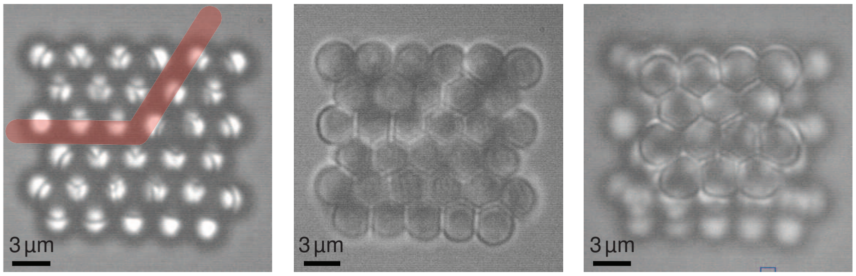


Figure 2.4: Assembly of a FCC colloidal crystal that includes a line defect using optical tweezers [27]. The line defect is shown in red in part (a) and can be seen more clearly at a slightly different focus shown in (b). Part (c) shows the fully assembled crystal.

Integrated photonic circuits [92] and electronic devices using nanowires such as diode junctions (Figure 2.5), as well as nanowire photonic devices such as laser cavities [114] have also been developed using optical trapping. Optical tweezers have also been used successfully in cytometry of colloids as well as biological cells [97]. By flowing an assorted mixture of particles through dynamically generated 3D holographic optical lattice, microsphere as well as cells can are easily sorted by size [47, 46]. Moreover the ability to dynamically reconfigure the optical lattice makes the optical tweezers based cytometric system a particularly powerful and flexible tool.

Optical trapping is a powerful and flexible tool for assembly and test at the microscale. However, in order to advance optical trapping to the nanoscale, new techniques and algorithms are required, which extend the trapping time of nanoparticles in the trap, while minimizing the incident power on the particle. Develop-

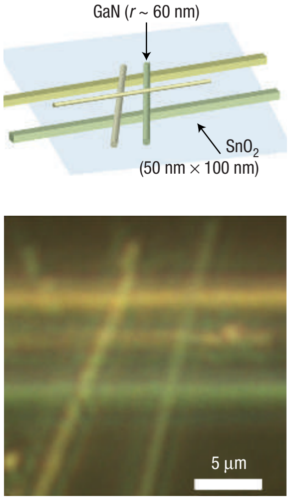


Figure 2.5: Assembly of GaN and SnO_2 nanowires into prototypical diode junctions [114].

ment of such algorithms, in turn require a combination of modeling, simulation and experimental verification tools. Three relevant areas, important to the development of optical tweezers for nanoscale assembly are therefore discussed next. These are:

1. Development of simulation techniques to model the movements of components in an optical tweezers assembly volume (Section 2.3).
2. The accurate experimental measurement of trapping forces imparted to a trapped nanoparticle by an optical trap (Section 2.4).
3. Development of algorithms, including closed-loop control systems to improve the performance of optical tweezers at the nanoscale. (Section 2.5).

2.3 Physically Accurate Simulations

Physically accurate simulations play an important role in the development of new techniques to improve the performance of nanoparticles in an optical trap.

Currently, few techniques exist that maximize the time a nanoparticle spends in an optical trap while, minimizing the amount of power absorbed by either the particle or the surrounding fluid. One route to achieving improved trapping time is through the development of active control systems, which react very quickly to the displacement of the particle away from the center of the trap as described in Chapter 5.

Developing new controller algorithms directly in the laboratory can be challenging. Moreover, many measurements required to quickly characterize the performance of controllers may not be directly accessible, as described in more detail in Chapter 5. A physically accurate simulation, on the other hand, provides a more tractable route to the development of new control algorithms for nanomanipulation. Furthermore, instrument limitations, which can play a limiting role in the feasibility of a particular design are easily included in the simulations and can be used to quickly quantify their effect on controller design.

Physically accurate simulations can also be used to develop new characterization techniques, such as force measurements, to improve the understanding of the weak interactions between nanoparticles and an optical trap. Data from the force measurements, once implemented in the laboratory, can then be used as input to the simulation, further improving the fidelity of the simulation by including information about non-ideal effects such as misalignments and aberrations in the optics.

Small particles undergo a random thermal motion not only in the assembly volume of the optical tweezers, but also within the trap. Therefore, a central component of a physically accurate simulation algorithm is a model for diffusive motion of nanoparticles in a fluid. The simulation must also allow the easy inclusion of external forces, relevant to the experiment being performed, such as the binding force from an optical trap, electrostatic forces from interactions between a nanoparticle and a surface and hydrodynamics forces between two or more particles. In order to be practical, the simulation must also sufficiently account for non-ideal behavior, such as instrumentation limitations including finite bandwidth and time-delays. Two classes of simulation algorithms, molecular dynamics and Brownian dynamics, commonly used to model the trajectories of microscale and nanoscale particles diffusing in fluids are discussed next.

2.3.1 Molecular Dynamics Simulations

Molecular dynamics (MD) is a classical simulation technique that is used to compute the equilibrium properties of many-body systems, whose properties are difficult to compute analytically [56]. In a typical MD simulation, a system of interacting particles is first defined, by specifying their initial positions and velocities. For particle diffusion, MD simulations must explicitly model the particle and the surrounding fluid molecules. Moreover, at each time-step interactions between the fluid and the particle, as well as interactions among the fluid molecules must be accounted for explicitly. This system must then be solved numerically using appropriately sized time-steps, which are often extremely small due to the fast characteristic times of the molecular interactions.

At each time-step, the force between the interacting particles in the system is calculated and then Newton's equations of motion are solved for the system. The output of the MD simulation is a time-series of the positions and momenta for each particle. Physical quantities, relevant to the physical system being studied and that can be expressed in terms of the particle position or momentum, are calculated at the end of each time-step. One example of an observable physical quantity, often calculated using MD, is the temperature of the system. Temperature can be expressed in terms of the equipartition energy per degree of freedom using the relationship shown in Equation 2.4, where k_B is Boltzmann's constant, N is the number of particles in the system, $N_f = 3N - 3$ is the number of degrees of freedom, m_i and p_i are the mass and momentum of the i^{th} particle. The system reaches thermal equilibrium when the net change in the temperature between successive time-steps is negligibly small.

$$T = \sum_{i=1}^{N} \frac{p_i^2}{m_i k_B N_f} \tag{2.4}$$

MD simulations using the technique described above have been used extensively in investigating diffusion driven transport phenomenon. Diffusion driven processes are particularly important in cell biology and are responsible for the inter-cellular and intra-cellular transport of ions, molecules and particles. The transport of organic molecules, not actively controlled by cellular proteins, as is the case for the transport of amino acids and sugars, across cell membranes are currently not very well understood, but directly affect drug delivery and toxicity studies. MD simulations have played an important role to improve our understanding of these phenomenon. Consequently, many groups have studied the diffusion and permeation of small organic molecules across a phospholipid membrane using atomic level MD simulation with good success [169, 26, 25, 100]. However, MD simulations very computationally intensive due to the extremely short time-steps required to capture molecular processes, which have very small characteristic times [40]. In order to overcome these drawbacks, Orsi et al. [112] have developed a coarse grain simulation of the lipid bi-layer and surrounding water molecules, while computing the transport of the organic molecules with atomic resolution. They show that these approximations do not significantly affect the permeation and diffusion coefficients, which agree well with other MD simulations performed at the atomic level, as well as experimental measurements of these coefficients. Another area where MD simulations have played an increasingly important role is in ion transport through gated or non-gated channels [119, 32, 31]. The crystallization of cell membrane pores have revealed new details about their physical structures and subsequently MD simulations have been used to better understand the conformational and chemical changes ion channels undergo during the process of proton transport. Wang and Voth have studied proton transport through an antiporter using MD simulations and reported the reorientation of a protein side chain within the antiporter as the conformational change necessary to facilitate the transport [162]. Nemukhin et al. have performed molecular level simulations of the transport of ions through the Gramicidin A channel and found that their results agree with previous experimental observations [106].

Processes at the molecular scale have very short characteristic times. Consequently, MD simulations run with very short time-steps, typically approaching femto-seconds. Therefore, MD simulations, which typically include hundreds of molecules are very computationally expensive, even after including multi-scale coarse grained models and it can often take several days to gather only a few nanoseconds of simulation data. Diffusion processes for modeling nanoparticles in an optical trap often do not require a description of the system at the molecular level. A more computationally efficient approach to model such systems is described next.

2.3.2 Brownian Dynamics Simulations

Brownian Dynamics (BD) simulations have been developed to model the motion of micro and nanoparticles in a fluid using Langevin's equation, a modified form of Newton's equation of motion that includes a rapidly fluctuating stochastic force term. Unlike MD simulations, where the collisions of the fluid molecules with the nanoparticles are modeled individually, the stochastic force term in Langevin's equation accounts for the average force from numerous molecular collisions. Therefore BD simulations can be computationally more efficient than MD simulations. Equation 2.5 gives Langevin's equation for a particle with mass m and velocity $V(t)$ at time t. The characteristic time-scale of this model is the relaxation time, $\frac{m}{\gamma}$, where $\gamma = 6\pi\eta a$ is the drag coefficient. The scaling constant for the stochastic noise term $\Gamma(t)$ is given by $\zeta = \sqrt{2\gamma K_B T}$ and is obtained by imposing the requirements of the fluctuation-dissipation theorem [84, 68].

$$\frac{dV(t)}{dt} = -\frac{\gamma}{m}V(t) + \frac{\zeta}{m}\Gamma(t) \tag{2.5}$$

BD simulations have been used in colloidal science to describe the self-assembly of dilute polystyrene solutions into colloidal crystals without considering the effect of inter-particle hydrodynamic interactions [91, 2]. However, hydrodynamic interactions between neighboring particles are important when studying the self-assembly of microspheres and were developed and integrated into BD simulations by Ermak and McCammon [51], who describe the diffusive behavior of N interacting microspheres, in addition to the stochastic effects from the fluid surrounding the suspension. This model was later extended to include the behavior of particles of arbitrary shape in the presence of hydrodynamic effects and an external force field [53]. Equation 2.6 gives a version of Langvin's equation that includes hydrodynamic interactions for multi-body systems [51]. In this equation, $\tau_{ij} = \frac{D_{ij} m_j}{k_B T}$ and $\sigma_{ij} = \frac{1}{k_B T} \sum_i D_{ij} \zeta_{ij}$, where D_{ij} is the diffusion tensor, m_i is the mass of the i^{th} particle and ζ_{ij} is a scaling coefficient. Hydrodynamic effects are introduced by deriving expressions for cross terms in the diffusion tensor to model the interaction between particles in the system. Finally, Equation 2.6 is integrated numerically to obtain position trajectories for each particle in the system.

$$\sum_j \tau_{ij} \ddot{x}_j = -\dot{x}_i + \frac{1}{k_B T} \sum_j D_{ij} F_i + \sum_j \sigma_{ij} \Gamma_j \tag{2.6}$$

Brownian Dynamics simulations have been used in colloidal science to model a wide range of interesting physical phenomenon. Simulations of self-assembly processes for heterogeneous components using Brownian dynamics have started to contribute to the understanding of these phenomenon under wide ranging conditions. Lin et al. have shown the self-assembly behavior of rod-like and coil-like copolymer molecules using BD simulations [93]. By modeling the inter-molecular forces, as well as the electrostatic forces between individual copolymer systems, their simulations predict the realignment and packing of the copolymers, in agreement with experimental observations. BD simulations have also been successfully used in templated self-assembly. Haghgooie *et al.* have demonstrated how such simulations can be used to develop the design of efficient templates for use in the self-assembly of magnetorheological fluids in microfluidic channels [71]. Using BD simulations they have shown how specific geometric constraints in template design can not only induce the assembly of these colloids, under a magnetic field, but also transition the system between ordered and disordered states.

Brownian dynamics simulations have also been combined with other simulation methods, most notably molecular dynamics, to result in multi-scale simulations. Allen and Rutledge have demonstrated one such approach, where they use the results from a MD simulation to create an approximate model of the solvent density, which is then input into a Brownian dynamics simulation [6]. This approach is particularly useful in improving the accuracy, while retaining the computational efficiency of BD simulations, when studying solvents whose density varies close to the solute particle, as is often encountered when modeling proteins and other biological systems. Molecular level simulations have also been combined with rotational dynamics simulations to model the behavior and aggreagation of claylike colloids [75], while more recently Sané et al. have studied the hydrodynamic interactions of disk-like colloidal particles using multi-scale simulations [138].

Physcially accurate particle simulations are important in advancing nanoassembly using optical trapping. However, very few simulations exist that model the behavior of nanoparticles in an optical trap. Rohrbach *et al.* have developed a three-dimensional simulation using Brownian Dynamics to model a nanoparticle in an optical trap for use in a thermal motion-based imaging system [133]. They include a physical, but idealized force calculation of the optical trapping force and use their simulations to calculate the spring constant of kinesin, immobilized at one end and attached to a latex bead, held in an optical trap, at the other. Based on the relative position and orientation of the bead with respect to the anchor, they obtain between 5% and 13% agreement their calculated values and the measured value of kinesin's spring constant. More recently, Gong et al have used Monte Carlo simulations to calculate the stiffness of an optical trap using the thermal noise of a particle in the trap [65]. Their simulation results show that insufficient acquisition time can be a significant cause of inaccurate estimation of the trap stiffness. Xu *et al.* have studied the collision dynamics, including hydrodynamic effects, of two particles trapped using an optical tweezers, also using Monte Carlo simulations [170]. They show that multiple particles in an optical trap have little impact on the stiffness of the trap and the frequency of collisions of the particles in the trap decreases, as expected, with the separation distance. More recently Roichman *et al.* have used BD simulations to interpret laboratory results, where they observe that microscale particles undergo a steady-state circulation in an optical trap due to the presence of the forward scattering force [137].

Brownian dynamics simulations play can play an important role in optical tweezers-based assembly and the performance of BD simulations can be significantly improved by porting these algorithms to inexpensive graphics hardware as is discussed next.

2.3.3 Physical Simulations on Graphics Hardware

In recent years graphics intensive programs have greatly benefitted by offloading geometric operations to the inherently parallel and pipelined architcture of the graphics processing unit (GPU) [80]. More recently key architectural [1] and software changes have allowed general purpose calculations to be offloaded to the GPU and the results returned for use in further analysis or computation. The introduction of high level application programming interfaces (API) such as nVidia compute unified device architecture (CUDA) have made it even simpler to implement existing scientific applications to the GPU [74].

Many general purpose calculations have been successfully implemented on the GPU facilitating its use as a cost-effective parallel processor. Geometric modeling applications have heavily leveraged the capabilities of the GPU [125, 79] and methods to solve partial differential equations (PDE) as well as linear systems of equations have also been successfully implemented. This in turn has led to the development of finite element algorithms [48] and spring mass systems. Development of sorting and search methods such as bitwise or a nearest neighbor search have helped the implementation of fluid dynamics problems on graphics hardware. Kipfer et al. [81] have developed a GPU based particle engine that can simulate a million particles with collision detection using a simple sorting algorithm. Kolb and Cuntz adapted a Lagrangian model that directly simulates particle motion where they propose to eliminate the need to sort interacting particles to determine the effect of nearest neighbor interactions [82].

More recently MD simulations have been implemented entirely on the GPU [9, 57, 145] to include the calculation of force fields and integration of the force equation to yield trajectories of the individual molecules, with reported speedups of 150 times over traditional simulation techniques [153]. Auffinger et al. [15] have developed MD simulations using the GPU to study the binding of K+ and Mg^{2+} ions to binding sites on 5 S rRNA. The GPU has also been used successfully in Monte Carlo methods to demonstrate the migration of photons on a surface [4], simulations of ligth modulation [90], as well as the 2D and 3D simulation of the Ising model [123].

A majority of GPU implementations however, emphasize the performance of their respective algorithms on graphics hardware over accuracy. The field of GPU architecture is rapidly evolving. However, most GPU's to-date store data in 32-bit floating point format compared with double-precision 64-bit formats that are common on the CPU, degrading accuracy and potentially resulting in large roundoff errors [19, 102]. The problem is further aggravated by the fact that most GPU vendors (with the exception of nVIDIA Tesla architectures) do not strictly adhere to the IEEE floating point representation standard, causing the round-off errors to vary significantly between GPUs. The availability of 64-bit GPUs in the future is likely to improve the accuracy in some applications, but the characterization of GPU errors will remain an important component in porting scientific calculations to the GPU successfully. The problem of GPU rounding errors is even more important in particle simulations like those proposed for optical trapping, which involve the generation of long trajectories over millions of time-steps, where rounding errors can accrue and thereby pose serious challenges to producing physically accurate results. We discuss the effect of GPU rounding errors on an optical tweezers simulation in more detail in Chapter 3.

2.4 Measurement of Optical Trapping Forces

The accurate measurement of the optical trapping forces with sufficient sensitivity has been greatly beneficial in many applications of optical trapping, especially in studying the mechanics of biological systems such as biological motors. In most experiments, large microspheres with diameters of several micrometers, are attached to the biological systems. The optical trapping force is then estimated for these large particles, held in the optical trap. Large microspheres are very strongly bound to an optical trap and undergo very little movement away from the center of the trap due to Brownian motion. For very small displacements close to the center, the optical trap can be approximated by a harmonic oscillator and the optical trapping force, under this harmonic approximation, varies linearly with the displacement of the particle.

Several techniques, developed to measure the trapping force on a strongly bound particle in an optical trap using a harmonic approximation of the trapping force are described below. The harmonic approximation, which results in a linear spring stiffness is appropriate and works well when measuring the trapping force experienced by particles larger than several hundred nanometers. Force measurement techniques employing the harmonic approximation generally require that the displacement of the particle be measured with nanometer precision and are sensitive to very small changes in the trapping force. Accurate measurements of the force exerted by an optical trap are critical in several biological experiments. Probe particles such as silica microspheres ranging in diameter from several hundred nanometers to several micrometers, held in an optical trap, are often attached to biomolecules. Therefore force measurement techniques so far have centered around calibrating the linear stiffness of optical traps, close to the center. However, the optical trapping force experienced by a particle reduces signicantly for smaller particles and the trapped particle is no longer strongly confined in the trap. Consequently, the particle exhibits larger excursions away from the trap center, where the harmonic approximation for the trapping force is no longer valid. New force measurement methods that estimate the trapping force without any prior assumptions about the shape of trapping potential are described in detail in Chapter 4.

We start by discussing particle detection techniques commonly used in measuring the optical trapping forces (Section 2.4.1) followed by current techniques to measure the optical trapping force on microspheres. The power spectrum method is discussed in Section 2.4.2, followed by the drag force method in Section 2.4.3 and the equipartition method in Section 2.4.4. A summary of the discussed techniques is then presented in Section 2.4.5.

2.4.1 Particle Detection Methods

The accurate measurement of the trapping force often depends on sensitive position measurement techniques. Techniques to accurately measure the position of a nanoparticle in an optical trap include video microscopy and laser interferometry and their use is determined by the requirements of the particular experiment being performed.

Digital video microscopy is a versatile sensing and measurement tool that is appropriate for low bandwidth applications, where sampling rates between $30\,Hz$ and $1\,kHz$ are sufficient [60, 42]. More recently high speed video microscopy has been developed to track gold nanoparticles, tagged with fluorescent dyes, with a bandwidth of up to $40\,kHz$ [59]. Position information is generally recovered with sub-pixel resolution using image processing algorithms, such as image centroiding to locate the center of a spherical particle, which results in an accuracy of approximately $10\,nm$ [24]. Leveraging fast image centroiding algorithms, video microscopy has been used extensively in tracking suspensions of colloids to determine their structure as well as dynamics [70] and to study the hydrodynamic interactions of two spherical particles close to a plate [49]. Digital image processing techniques have also been used successfully to recover Boltzmann's constant from a dilute suspension of microspheres [105]. More recently digital image processing techniques have been developed for the three-dimensional tracking of microspheres [115] and nanowires [116, 117] in a fluid suspension for use in path planning algorithms using optical tweezers. Video microscopy methods, even with the use of high-speed charge-coupled device (CCD) cameras are not generally used in real-time, due to the high computational time of processing the acquired image, in comparison with the acquisition time. Therefore their use in online application such as closed-loop feedback control is limited.

Laser based detection techniques have been developed for translucent particles as an alternative to image processing, especially in real-time applications where high detection bandwidth is desired. One common detection technique relies on the interference between orthogonally polarized light, analogous to differential interference contrast (DIC) microscopy [107]. In this technique, an incoming linearly polarized beam is first split by a Wollaston prism into two orthogonally polarized components that are spatially shifted relative to each other [45, 149, 158]. Each of these beams form an overlapping spot at the specimen plane, separated by a few nanometers [38]. After passing through the specimen plane, the two polarized components of light are recombined by a second Wollaston prism and the resulting polarization of the recombined beam is measured. For a particle, at equilibrium and aligned with the center of the trap, the path length and thereby the phase shift in each arm of the interferometer is the same and the recombined beam retains a linear polarization. However, as the particle moves away from the center of the trap, the recombined beam becomes elliptically polarized (on account of uneven phase shifts in each path), with the degree of the polarization varying as a function of the displacement. The polarization state of the recombined beam is measured by splitting the beam into two orthogonal components using a polarizing beam splitter and measuring the power in each beam path with a photodiode. This technique is extremely sensitive to particle displacements in the trap, however it's main drawback is that it is limited to one-dimensional measurements aligned with the axis of the Wollaston prism [45].

Another laser based detection method for translucent particles is shown in Figure 2.6. This technique monitors a detection beam, which passes through a microsphere and is imaged at the back focal plane (BFP) of the microscope condenser [63, 8]. The detection beam is then projected onto a quadrant photo-diode (QPD) located in a plane, conjugate to the BFP. The QPD is a four element detector, which uses differential intensity signals from its four detectors to generate a two-dimensional position signal, proportional to the displacement of the beam. In this system a lateral movement by the sphere in the specimen plane, causes a corresponding deflection at the BFP and thereby also on the QPD. Back focal plane detection is commonly used in optical trapping and provides a sensitive measurement of the motion of the particle relative to the trap. However, in certain applications, such as when closed-loop control of the trapped particle is desired, a separate low-power detection beam is incorporated [86] into the experiment and aligned with the trapping beam to provide independent position measurements. The power and wavelength of the detection beam are important parameters to be considered when designing an independent detection system. The detection beam must have sufficiently low power, so as to not generate significant optical trapping forces at the specimen plane. This is achieved by limiting the beam power to a few milli-watts and choosing a laser line to which the detector is very sensitive. A common combination used in optical trapping experiments is a visible red laser with a silicon photo-diode.

The position detection methods discussed above only measure the lateral position of a particle in a trap, but not axial displacements, which are important for three-dimensional optical trapping. Axial position measurements are typically obtained by measuring the intensity of light at the back focal plane of the condenser (or a plane conjugate to it) using a photo-diode [122]. In the axial direction, light passing through the focal plane of the objective lens acquires a phase shift of π called the Gouy phase [135]. However, light scattered by a particle located at the specimen plane (close to the focus) does not acquire the full Gouy phase shift, but instead acquires a phase shift proportional to it's axial displacement [107]. This results in intensity fluctuations as a function of the particle's axial displacement, which are recorded using a photo-diode. Lateral and axial position measurement techniques have been combined to provide three-dimensional tracking of a particle in an optical trap with high resolution and bandwidth [135].

The bandwidth obtained using laser-based detection methods is sensitive to the wavelength of the detection beam, as well as the power. Typical silicon detectors used in position measurement have poor absorption of light at wavelengths above $850\,nm$ and consequently a poor response to the particle position, which in turn leads to unintended filtering and lower bandwidth [29, 30]. Typical bandwidths for silicon detectors at $1064\,nm$ (a popular laser line for optical trapping and detection) have been reported to be approximately $8 - 9\,kHz$ [107]. Improved performance is obtained by using an independent detection beam with wavelength in the visible region or by using photodiodes built with a different material, such as indium gallium

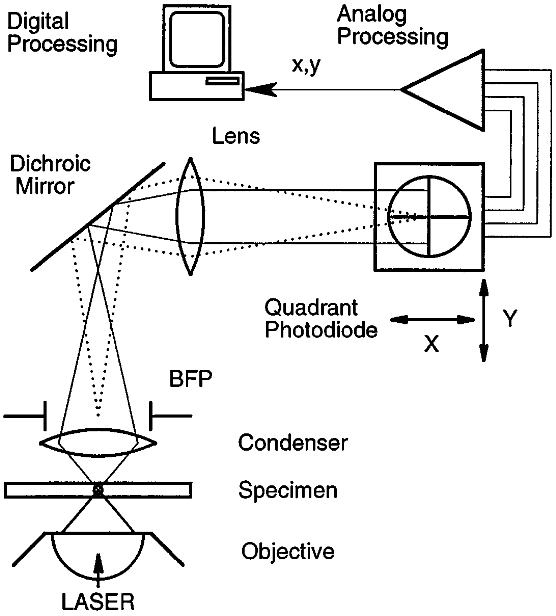


Figure 2.6: Back focal plane detection setup used to measure lateral displacements of trapped microspheres [63].

arsenide (InGaAs) which have better response than silicon in the infrared spectrum.

The response of detectors such as quadrant photo-diodes to a change in the particle position is generally unknown and must therefore be calibrated prior to use in a force measurement. Calibration is typically performed by immobilizing a bead on a coverglass and using a nanopositioning stage to move the particle across the field of view of the detector in discrete steps [142]. The response of the detector to the particle position is then recorded as a function of the particle position, shown in Figure 2.7 [66]. In many experiments, it is advantageous to limit the position measurement of the particle to a region close to the center of the detector, where the sensitivity is the largest. The plot shown in Figure 2.7 is for one-dimensional position detection, however Lang *et al.* have also performed the QPD calibration in two-dimensions [86]. Force measurement methods for strongly trapped particles, which rely on accurate position measurements are described next.

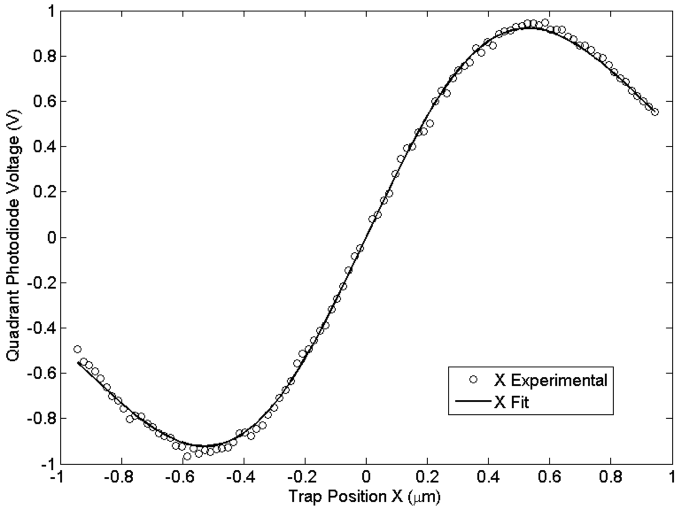


Figure 2.7: Calibration of a quadrant-photodiode response to the position of an immobilized bead [66, 142].

2.4.2 Power Spectrum Method

All trapped particles, regardless of their size undergo some random motion in an optical trap. Leveraging this principle, power spectrum methods have been developed to recover the trap stiffness for strong optical traps by analyzing the motion of a particle in the trap in the frequency domain [159, 147, 8]. In order to calculate the power spectrum of a particle in a trap, we start with the motion of a particle in a harmonic potential, given by the following Langevin equation [39].

$$m\,\ddot{x}(t) \;+\; \gamma\,\dot{x}(t) \;+\; \kappa\,x(t) \;=\; \sqrt{2k_BT\gamma}\,\Gamma(t) \tag{2.7}$$

In Equation 2.7, m is the mass of the particle, $\gamma = 6\pi\eta\epsilon a$ is the Stokes drag coefficient for a spherical particle with radius a in a liquid with viscosity η at temperature T and $x(t)$ is the trajectory of the particle. The stiffness of the harmonic trap is given by κ, while $\dot{x}$ and $\ddot{x}$ represent the first and second time derivatives of the position trajectory of the particle. The factor ϵ in the drag coefficient γ, which takes into account the correction to the viscosity of the fluid due to particle motion parallel to the walls of the sample cell, at a distance h, is given by Equation 2.8 [36].

$$\epsilon = \frac{1}{1 - \frac{9}{16}(a/h) + \frac{1}{8}(a/h)^3 - \frac{45}{256}(a/h)^4 - \frac{1}{16}(a/h)^5} \tag{2.8}$$

For over-damped systems (such as trapping microscale or nanoscale particles in water), the second order term in Equation 2.7 is ignored and the power spectrum P_f is obtained as a function of frequency f by first taking the Fourier transform of Equation 2.7 and then calculating it's magnitude to obtain the power spectral density. Under an assumption of an over-damped system, the power spectrum is known to have the Lornetzian form given in Equation 2.9, where f_c is the corner frequency of the trap. The corner frequency is recovered experimentally by calculating the power spectrum from a time-series of a particle's position in the optical trap and then fitting this data to the Lorentzian function shown in Equation 2.9 [37]. The corner frequency (f_c) recovered from the power spectrum of the particle's position in a trap is then related to the stiffness using the relationship given in Equation 2.10.

$$P_f \; = \; \frac{k_B T}{2\gamma\pi^2(f_c^2 \; + \; f^2)} \tag{2.9}$$

$$\kappa = 2\pi\gamma f_c \tag{2.10}$$

A typical power spectrum for a trapped spherical particle is shown in Figure 2.8. As seen above, the corner frequency (at the point where there is knee in the

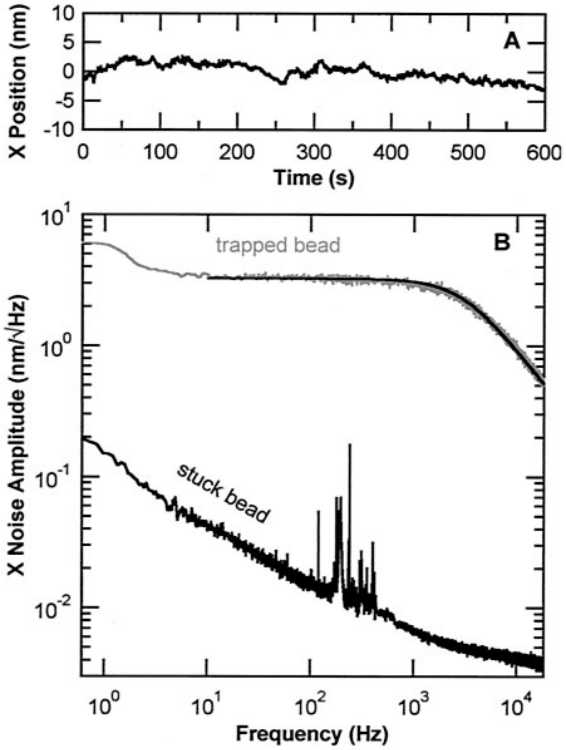


Figure 2.8: A typical power spectrum for a bead stuck to the surface of the sample cell and for one trapped above the surface. The power spectrum of the bead trapped above the surface has Lorentzian form [86].

plot) of the trapped bead in the figure corresponds to the stiffness of the optical trap. As the particle size decreases, the optical trapping force and therefore the trapping stiffness decreases and the corner frequency will shift to the left. Similarly, as the power input to the particle is increased, the corner frequency will shift to the right leading to an optical trap with a higher stiffness. The power spectrum method has been extensively studied and has been used to measure trap stiffness for transverse displacements [38], axial displacements [58] and both the transverse and axial displacements of the particle simultaneously [37]. This method is sensitive to extraneous sources of noise in the instrumentation (which appear as peaks in the power spectrum) as well as any misalignment or instability in the optical beam path (which result in a distortion of the Lorentzian shape of the power spectrum). However, the accuracy of the measured stiffness is directly influenced by detector bandwidth. Therefore for optimal performance, the bandwidth of the detection system must be at least an order magnitude greater than the expected bandwidth of a particle in the trap [107]. Inadequate detection bandwidth (even with the inclusion of a low pass filter) leads to underestimating the corner frequency and thereby the stiffness of the trap. On the other hand, since the stiffness of the trap depends only on the corner frequency, detector calibration is not required for this method to provide accurate results.

The power spectrum method has been improved to take into account the effects of finite sampling frequency on trap stiffness, the effect of electronic filters and the unintended filtering arising from slow response of silicon photo-diodes to infrared beams [30, 29, 110]. These improvements to the power spectrum method have resulted in lower uncertainties in the measurements. Lukić et. al [96, 95] have carefully characterized the influence of hydrodynamic memory effects arising from the inertia of the fluid surrounding a trapped particle. The inclusion of corrections from such memory effects greatly improves the accuracy of stiffness measurements. While the influence of hydrodynamic memory effects generally diminish with particle size, they find that this decrease is non-trivial and varies significantly with detector bandwidth and trap strength. Moreover particles with a diameter as large as $270\,nm$ were found to be influenced by this phenomenon. The power spectrum method can yield moderately accurate results when the Stokes drag coefficient γ is accurately known [38]. In many cases accurate measurement of the Stokes coefficient, which depends on the fluid viscosity is difficult. In order to overcome this, the power spectrum method has been combined with an online drag force measurement (described next), which eliminates the need to know the fluid viscosity beforehand [37, 152].

2.4.3 Drag Force Method

A direct measure of the optical trapping force can be obtained by moving a particle through a fluid with a constant velocity [139, 61], generally performed by moving the sample stage relative to a stationary trap. In this case, the force exerted by the trap on the particle is balanced by a viscous drag force from the fluid, given by Stokes law. As the particle moves relative to the fluid surrounding it, it is displaced from its equilibrium position in the trap by an amount proportional to it's velocity. By accurately measuring the velocity (v_p) of a particle, with radius a, the force acting on the particle can be calculated using Equation 2.11, by ignoring inertia and thereby assuming that the system is over-dampled. The viscosity of the fluid surrounding the particle is η (as a function of the temperature T) and ϵ accounts for corrections due to particle's proximity to the surface of the sample cell, given by Equation 2.8.

$$F_{drag} = 6\pi\eta\epsilon a v_p \tag{2.11}$$

The experiment is performed several times and with each iteration, the velocity of the particle v_p is increased and the displacement of the particle from it's equilibrium position is recorded, until the particle escapes the trap past some critical velocity known as the escape velocity. The drag force calculated from the velocity v_p using Equation 2.11 is then plotted against the displacement of the particle from it's equilibrium position as seen in Figure 2.9 for the transverse case [85]. The linear stiffness of the optical trap (κ) is then simply the slope of the plot in Figure 2.9.

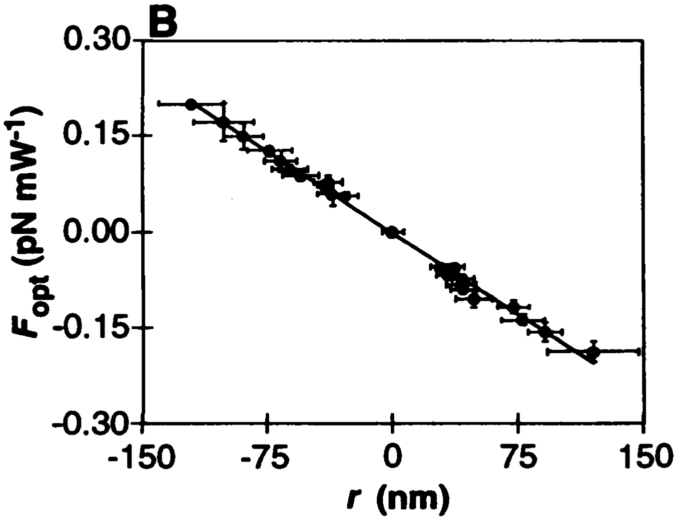


Figure 2.9: The optical trapping force F_{drag} plotted against the displacement of a spherical particle from it's equilibrium position in the trap [85]. The experiment is performed with positive and negative velocities in order to measure the trapping force on either side of the equilibrium position.

The drag force method has been successfully used to calculate the transverse [85, 98, 166] as well as the axial trapping stiffness [61] and has also been used to map out the linear region of the trapping potential close to the center of the trap. Since the measurement is slow compared with techniques such as the power spectrum method or the equipartition theorem method, the bandwidth requirements for detectors used in this technique are not very stringent. Moreover since this technique depends on accurate measurement of the displacement of the particle from its equilibrium position, detector calibration plays a crucial role in achieving high accuracy. However as in the case of the power spectrum method and the equipartition method, an accurate measurement of the viscosity of the fluid is necessary.

2.4.4 Equipartition Method

The equipartition theorem relates the average energy of a particle to it's temperature (T). It has been used to calibrate the stiffness of an optical trap under the assumption of harmonic motion of the particle within the trap [55, 107]. As seen in Equation 2.12, the potential energy for an optical trap under a harmonic approximation is equated with the thermal energy experienced by the particle in the optical trap, where the stiffness of the optical trap is given by κ, k_B is Boltzmann's constant and $\langle(x-x_0)^2\rangle$ is the mean-squared displacement of the particle from the center of the trap (x_0).

$$\frac{1}{2}k_BT = \frac{1}{2}\kappa\left\langle(x-x_0)^2\right\rangle \tag{2.12}$$

The equipartition method depends on the position of the particle in the trap, but requires a calibrated position detector with sufficiently high bandwidth [158] since insufficient bandwidth often results in underestimating the trap stiffness. This method is explicitly developed under the harmonic approximation of the trapping potential and is therefore suitable for large particles that only explore a small region close to the trap center. Like the power spectrum method, this force measurement technique is also susceptible to unintended filtering effects [30, 29, 110] as well as the influence of hydrodynamic memory effects [96, 95].

2.4.5 Comparison of Force Measurement Techniques

Several techniques to measure the stiffness of an optical trap formed by a tightly focussed laser beam have been described above. These techniques have been compared with each other experimentally and show moderate agreement [95]. However, very little data exists that compares experimental force measurement methods with existing theory. In one case, where theoretical calculations for dielectric particles have been compared with the stiffness of the optical trap, the errors between the theoretical and experimental values vary significantly with particle size [132]. In these measurements, smaller particles showed moderate agreement with theoretically predicted values (varying from approximately 3 % to 50 %), while for micrometer sized particles the errors were even larger. This highlights the continuing difficulty in obtaining agreement between laboratory results and current theoretical models over a wide range of particle sizes and materials. These results also imply that new techniques, developed using numerical simulations, to localize nanoscale particles within an optical trap cannot fully rely on theoretical models. Instead, we develop new laboratory-based force measurements for nanoparticles (described in Chapter 4) and use this data as input to numerical simulations to drive the discovery of new control algorithms to enhance optical trapping at the nanoscale.

2.5 Closed-Loop Control Systems for Optical Tweezers

Particles confined in an optical trap undergo a constant random thermal motion, which in some cases can limit the accuracy with which nanostructures are assembled. This uncertainty in the particle position due to Brownian motion can be reduced using a closed-loop control system, which continuously monitors the position of the particle in the trap and uses it's displacement away from the center, to servo the trap and return the particle to equilibrium. The first controlled optical trapping experiments were performed by Ashkin and Dziedzic using a levitating optical trap, shortly after the optical tweezers were invented [14]. In these experiments (performed in air as well as in vacuum), the displacement of the particle away from equilibrium, along the direction of propagation of the beam, was recorded using a photodiode and then used to control the power to the optical beam, to bring the particle back to it's equilibrium position. A simple proportional controller was found to be sufficient to hold the particle steady, when running the experiment in air, since the air molecules provided natural damping to suppress oscillations. However, a derivative term was added to the controller when trapping particles in vacuum, to suppress high frequency oscillations of the particle in the trap.

Current closed-loop controllers employ proportional-integral-derivative (PID) control to suppress the Brownian motion of microscale particles. Two techniques are commonly utilized to change the magnitude of the trapping force acting on the particle: intensity control, and scan control. In intensity control, the intensity of the beam is varied, typically using an electro-optic modulator (EOM), in response to a displacement of the particle from the center of the trap, thereby directly increasing the amount of force acting on the particle. On the other hand, in scan control, the intensity of the beam is kept constant, but the trap is moved rapidly, using an actuator such as an acousto-optic deflector (AOD), to oppose the motion of the particle from its equilibrium position in the trap.

Experiments that study biological systems have benefitted from both intensity and scan control. Such systems often use a large dielectric particle held in an optical trap and tethered to the biological molecule being studied. Wang et al. use intensity control techniques to study the micromechanical properties of RNA polymerase (RNAP), a motor protein complex, critical in the transcription of nascent RNA with a template DNA strand, capable of exerting large translocation forces during the transcription process [163]. They first held a polystyrene bead, tethered to the DNA strand, at a fixed position in an optical trap, by controlling the intensity of the beam and thereby changing the force exerted on the bead. The force from the optical trap, transmitted to the RNAP motor through the DNA tether, was steadily increased, eventually causing the RNAP motor to stall. The force applied to the RNAP as a function of time was then used to calculate the efficiency of this molecular motor. A similar experimental setup was also used to study the force required to stretch DNA and thereby compare the measured mechanical properties of DNA with theoretically predicted values [164].

Scan control has also been used successfully in the control of tethered microscale particles. Block *et al.* have used scan control to study the mechanics of the power stroke, responsible for the motility of kinesin molecules [33, 86]. A microscale spherical bead attached to a kinesin molecule was confined to an optical trap and scan control in two-dimensions was employed to maintain a constant force on the particle, as the kinesin motor tugged on the it. They found that the velocity of kinesin motion was heavily dependent on the directionality of the applied force and while forward and sideways forces had little impact on the motility of the molecule, small backward forces caused it to stall. This experiment provided new insights into the operation of kinesin, however the exact mechanics of the kinesin power stroke are still uncertain. Scan control methods have also been used by Finer, Simmons and Spudich to measure the forces generated by myosin, when taking discrete steps along microtubules in muscles [54].

Simmons et al. first demonstrated closed-loop scan control on freely diffusing particles in two-dimensions using a single laser for trapping and detection [142]. Using proportional and integral control, they demonstrated an increase in the stiffness of the optical trap by a factor of 400. This measurement has been repeated more recently by Wallin *et al.*, who use separate trapping and detection beams in their experiment [161]. Using a proportional controller, implemented on a field programmable gate array (FPGA), they demonstrate a several fold increase in the trap stiffness. Recent efforts have focussed on developing a more rigorous treatment of controlled optical trapping by applying the principles of control theory. Ranaweera and Bamieh have modeled the optical trapping instrument and applied online system identification techniques to obtain controller parameters in real-time [128]. They then demonstrate, through simulations, the benefit of using feeback control algorithms, such as proportional control, linear-quadratic-Gaussian or non-linear control, on the suppression of Brownian motion as well as increasing the trap stiffness. Wulff, Cole and Clark have developed techniques to select parameters that improve the performance of closed-loop controllers used in optical trapping [168, 167]. Much like Ranaweera and Bamieh, they model the optical tweezers instrument and develop adaptive online identification methods to determine the parameters of the controller. They also derive appropriate relationships that help in selecting the controller gain for the suppresion of Brownian motion. However, their experimental system is only capable of suppresing low frequency motion, due to limitations in their actuation bandwidth. Moreover, they use a single beam for trapping and detection, which has the drawback of potentially increasing the uncertainty in the absolute particle position significantly. The modeling of control systems for Brownian motion suppression has been further extended by Gorman *et al.*, who not only provide a detailed treatment of how to select optimum controller gains, but also discuss the effect of actual instumentation limitations, such as time-delays and instrumentation bandwidth, on the performance of the controller [66].

The development of efficient and optimal control systems for microscale particles is still a nascent field, however the application of control systems to localize nanoscale particles is virtually non-existent. Developing closed-loop control systems to control nanoparticles in an optical trap have several advantages. Feedback controllers using either scan control or intensity control can be used to corrall nanoparticles and keep them confined in the trap for sufficient time, for use in practical nanoassemblies. Control systems can also be used to reduce the uncertainty with which a nanoparticle is assembled into a structure and reduce the amount of power required to trap and localize the particle. The development of control systems for nanoparticles is however more challenging in comparison with microscale particles, since nanoparticles have weaker interactions with the optical trap and consequently diffuse out of the trap rapidly. Furthermore, limitations in instrumentation, such as time-delays and bandwidth, can also play a more decisive role in limiting the performance of such control systems and must therefore be overcome.

2.6 Summary

Optical tweezers have evolved to be a powerful prototype and test tool for microscale assembly. However, they have so far not been extended to the nanoscale. Extending optical trapping for nanoscale prototyping requires overcoming some key challenges. As seen earlier, the optical trapping force decrease drastically for nanoscale particles and consequently it is significantly harder to keep nanoparticles stably trapped within an optical trap. Therefore new techniques must be developed that increase the amount of time nanoparticles remain trapped, without also increasing the amount of power. One route to improved optical tweezers performance at the nanoscale is through the use of closed-loop control systems, but the development of efficient control algorithms also requires development of supporting tools such as physically accurate simulations and trap characterization techniques.

Physically accurate simulations of diffusion processes are discussed in Section 2.3. Existing techniques include using molecular dynamics simulations or Brownian dynamics simulations to model diffusion. Since we are more interested in modeling the diffusive behavior of microscale and nanoscale particles, rather than molecular systems, Brownian dynamics simulations offer the more efficient route to developing physically accurate simulations. However, these methods require small time-steps and therefore can take a long time to complete. The computational efficiency of such simulations can be greatly improved using commodity graphics hardware and the details of this development and the benefit to optical trapping are discussed in more detail in Chapter 3.

The interactions of nanoparticles with optical traps are currently not very well understood The behavior of nanoparticles in an optical trap is significantly differently than that of microscale particles and new methods that quantify the optical binding force acting on nanoparticles in an optical trap, can greatly improve our understanding of these interactions. However, accurately measuring the optical trapping potential in the laboratory and using that data to improve the fidelity of our simulations, will also yield to better algorithms to improve the performance of optical tweezers at the nanoscale. Existing methods used to detect microscale particles and measure the force exerted on them by the optical trap were discussed in Section 2.4. However these techniques are inadequate at the nanoscale and therefore two new force measurement methods, developed and tested through simulations and laboratory experiments, are discussed in Chapter 4.

Finally, closed-loop control systems, which play an important role at the microscale, especially in biophysical experiments are discussed in Section 2.5. Control systems for microspheres, which have only recently started to be carefully characterized, can play a central role in the extension of optical trapping to the nanoscale. However, control systems are currently not used for nanoscale manipulation with optical tweezers. In Chapter 5, we demonstrate new methods that improve the trapping time of nanoparticles in an optical trap using closed loop control systems.

Chapter 3

Development and Characterization of Physically Accurate Simulations for Diffusion of Nanoparticles in a Fluid

In this chapter[1], we develop a Brownian dynamics simulation to model particle diffusion in a fluid, under the influence of an external force. This model forms the foundation of a simulation environment for nanoassembly using optical tweezers. The simulation algorithms are implemented on the central processing unit (CPU) of the computer, as well as the graphics processing unit (GPU), choosing parts of the graphics pipeline that provide the most computational benefit. The simulation is tested against the well-known theory of diffusion in order to characterize it's physical accuracy. We also quantify the accuracy of the GPU simulation in relation with a double precision CPU implementation of the algorithm. Finally, we show that the GPU implementation provides appreciable speedup in comparison with a CPU implementation with the necessary accuracy.

Physically accurate simulations provide a versatile platform and serve as a powerful discovery tool to develop and test new techniques, which will allow us to extend optical trapping to nanoscale assembly. We have already used the numerical simulation techniques, developed in this chapter, to test the feasibility of new characterization techniques for use in the laboratory, such as the enhanced force

---

[1] The work in this chapter is derived from the published work in [19].

measurement methods outlined in Chapter 4. Physically accurate simulations have also played an important role in developing the control algorithms demonstrated in Chapter 5, which improve the time a nanoparticle remains confined to an optical trap. When developing control systems to enhance the performance of optical tweezers, simulations are one of the only practical tools that allow us to quickly prototype and test new algorithms.

Numerical simulations also have other important uses, outside the scope of this work, which make them an invaluable tool for optical trapping. Simulations can be used in a batch mode to test the effectiveness of new assembly algorithms, by repeating an assembly operation several times in order to collect statistical information about the repeatability and reliability of the process. Such operation is desirable, when access to the physical instrument may be restricted or the desired assembly operation is time or cost intensive to prototype directly in the laboratory. Moreover , in some cases simulations may be able to quantify trapping behavior in ways that is not practical in the laboratory. In these situation, numerical simulations combined with select experiments run in the laboratory can be used to provide quantitive information about the performance of optical traps. Numerical simulation algorithms, which run in real-time can be included within a closed-loop control loop for feed forward operation. Similarly, real-time numerical simulations can be used for operator training, where the simulation completely replaces the physical optical tweezers instrument.

3.1 Motivation

All particles in a fluid undergo random thermal motion, the result of millions of collisions of the surrounding fluid molecules with the particle. For large particles in a strong optical trap, the magnitude of the thermal motion away from the trap center is small. However, for a constant beam power, the trapping force acting on a particle decreases significantly as it's size decreases. Therefore small nanoparticles are able to make large excursions away from the trap center, eventually escaping the trap in times that are too short to allow their use in practical nanoassemblies. Closed-loop control algorithms, discussed in Chapter 5, have been developed to increase the time particles spend in an optical trap. Physically accurate simulations provide a flexible and perhaps the only route to rapidly prototype and test such control algorithms. Control algorithms, developed to improve the performance of optical traps acting on nanoparticles, can be sensitive to practical limitations, such as time-delays and finite bandwidth of instrumentation. It is difficult to account for these effects in analytical calculations that model controller behavior. However, such limitations are easily imposed in simulations and therefore physically accurate simulations are an invaluable tool to develop effective control algorithms for nanomanipulation with optical tweezers.

We develop a physically accurate model of particle diffusion in a fluid using Brownian dynamics, which forms the foundation of a simulation platform for optical tweezers based nanoassembly. Brownian dynamics simulation algorithms developed to model diffusion of nanoparticles in a fluid require the use of very small time-steps (on the order of pico-seconds for particles smaller than a few hundred nanometers) and therefore millions of iterations are required to generate sufficiently long runs of simulation data. Often, data from several hundred simulation runs are combined to calculate equilibrium averages or gather additional statistical properties of the physical process, which take a very long time to complete. However, the independent nature of the simulations, which do not consider interactions between particles, lends itself very well to parallelization.

Current GPUs have considerable parallel computing power that make them an attractive option to speed up scientific calculations. One drawback of many current GPUs is that they only support single precision floating-point numbers and do not follow the IEEE floating-point standard, which has been universally implemented on the CPU. Moreover, the implementation of floating-point numbers varies across GPU vendors, which causes the rounding errors on individual GPUs to significantly vary. Recently there has been a surge in the adoption of GPUs for general purpose physical models but little work has addressed the issues of accuracy and precision for deterministic systems [48]. Such characterization has not been attempted, to the best of our knowledge for iterative systems that generate long trajectories and may accumulate errors over millions of steps. The focus of this chapter, therefore is to:

a) Develop Brownian dynamics simulations of particle diffusion in a fluid for use in optical tweezers based nanoassembly.

b) Characterize simulation results against established theory to verify the accuracy of results represented by physical quantities such as energy conservation or the value of the diffusion constant.

c) Implement the simulation model on the GPU, using parts of the architecture that provide most benefit and evaluate the accuracy of GPU simulations against the results from the CPU.

The simulation model for a freely diffusing particle system (Section 3.2) is first described followed by an overview of the GPU architecture (Section 3.3). Special attention is given to how the stages of the graphics pipeline map onto our simulation model and which stages of the graphics hardware provide the most benefit for a given operation. Finally, results (Section 3.4) highlighting the accuracy achieved by running the simulation on the GPU and the measured speedup are presented, followed by a discussion of the results and conclusions (Section 3.5).

3.2 Brownian Dynamics Model for Particle Diffusion in a Fluid

A large particle in a fluid is bombarded by the surrounding molecules, which gives rise to a randomly fluctuating force on the particle that consequently imparts some velocity to it. As the particle moves through the liquid with finite velocity, it also feels the effect of a systematic drag force. The randomly varying force acting on the particle is intimately related to the drag force through the fluctuation-dissipation theorem [84]. Langevin's equation, which is simply Newton's equation of motion coupled with a properly scaled randomly fluctuating force, is used to formally describe this phenomenon [87, 39]. Langvein's equation is shown in Equation 3.1 for a particle with mass, m, radius a and velocity $V(t)$ at time t in a fluid with viscosity η, which is a function of temperature T. The characteristic time-scale of this model is given by the relaxation time, $\frac{m}{\gamma}$, where $\gamma = 6\pi\eta a$ is the drag coefficient (from Stoke's law for a spherical particle), and is the time taken by the particle to reach thermal equilibrium with the fluid, by gaining or losing velocity.

$$\frac{dV(t)}{dt} = -\frac{\gamma}{m}V(t) + \frac{\zeta}{m}\Gamma(t) \tag{3.1}$$

The scaling constant, $\zeta = \sqrt{2\gamma k_B T}$, in Equation 3.1 is obtained by imposing the requirements of the fluctuation-dissipation theorem [68], where k_B is Boltzmann's constant. The presence of the stochastic force term, $\zeta\Gamma(t)$ prevents the direct analytical solution of Langevin's equation, which does not have a unique solution. Instead, solving Langvein's equation results in an infinite number of trajectories from which we can calculate the average behavior of the system. Therefore, in order to calculate a single random trajectory, we first express Langevin's equation in a finite difference form and then proceed to integrate it numerically using an appropriate integration algorithm. Assuming a uniform time-step δt and constant acceleration over δt, we can write the acceleration of the particle at the end of a time-step in the finite difference form shown in Equation 3.2. A force term (F_{ext}) has been added to allow the inclusion of an external force, such as an optical trapping force, but is set to zero for the free particle. Simulations that include an optical trapping force are implemented in Chapter 4.

$$A(t) = -\frac{\gamma}{m}V(t) + \frac{\zeta}{m}\Gamma(t) + \frac{F_{ext}}{m} \tag{3.2}$$

As stated before, a particle moving with velocity, $V(t)$ in a fluid bath at temperature T undergoes many collisions with the molecules in the liquid [62]. In the finite difference form, the stochastic force in Equation 3.2 is the result of these collisions over a finite period of observation, δt. Furthermore, the stochastic force that arises from these collisions is found to be normally distributed with a mean of 0 and a standard deviation proportional to $\frac{1}{\delta t}$ [68]. Therefore we can replace $\Gamma(t)$ in Equation 3.2 with a properly scaled normal distribution $N(0, \frac{1}{\delta t})$. Using the property of the normal random variable, $N(\alpha\mu, \alpha^2\sigma^2) = \alpha N(\mu, \sigma^2)$, we can absorb $(\frac{1}{\delta t})^{\frac{1}{2}}$ into the scaling constant, ζ. The final form of the acceleration at the end of the time-step is then given by Equation 3.3, where $\chi = \sqrt{\frac{2\gamma k_B T}{\delta t}}$.

$$A(t) = -\frac{\gamma}{m}V(t) + \frac{\chi}{m}N(0,1) + \frac{F_{ext}}{m} \tag{3.3}$$

The acceleration of the particle must be numerically integrated in order to obtain the new position and velocity of the particle. Several numerical integration algorithms exist to perform the integration, however simulations based on Brownian dynamics models use explicit finite difference methods, which are simple to implement and provide good energy conservation [35]. The most popular explicit finite difference methods are the Gear predictor-corrector method [7] and the Verlet method [154, 155]. The Gear predictor-corrector, as the name suggests, consists of two steps – an initial prediction step that provides the initial estimates of the quantities being calculated, based on their values at previous time-steps, followed by a correction step. Often, the process is repeated to reduce the error in the trajectory. However, since each additional correction step requires another evaluation of the force function, this technique can become computationally expensive. Moreover, the error in the equilibrium energy increases very rapidly for large time-steps. The

Verlet algorithm, on the other hand, is easy to implement, compact and computationally efficient. The Verlet algorithm provides good energy conservation at long times, with little drift [113, 7]. Eventually, an integration method must be selected that a) satisfies conservation laws with acceptable accuracy, b) is computationally cost effective and c) permits the use of the largest possible time-step. For our application, we choose the velocity form of the Verlet algorithm (which explicitly includes a velocity term in the calculation of the position in contrast with the original Verlet algorithm) due to its ease of implementation, the possibility to use larger time-steps and excellent energy conservation at long times. Next, we obtain expressions for the position ($R(t)$) and velocity ($V(t)$) of the particle for the velocity form of the Verlet integration algorithm.

$$R^{'}(t) = V(t) \approx \frac{R(t+\delta t) - R(t-\delta t)}{2\delta t} \tag{3.4}$$

$$R^{''}(t) = A(t) \approx \frac{R(t+\delta t) - 2R(t) + R(t-\delta t)}{\delta t^2} \tag{3.5}$$

In order to obtain the expressions for the position and velocity of a nanoparticle using the velocity form of the Verlet algorithm, we first define approximate expressions for the first and second derivatives of the position $R(t)$, with respect to time, as shown in Equation 3.4 and Equation 3.5 [150].

$$R(t+\delta t) = A(t)\delta t^2 + 2R(t) - R(t-\delta t) \tag{3.6}$$

We then obtain an expression for the position of the particle at the next time-step $(t+\delta t)$ by treating Equation 3.4 as an equality and solving for $R(t+\delta t)$ as shown in Equation 3.6. Finally, we substitute the expression for $R(t-\delta t)$ from Equation 3.4

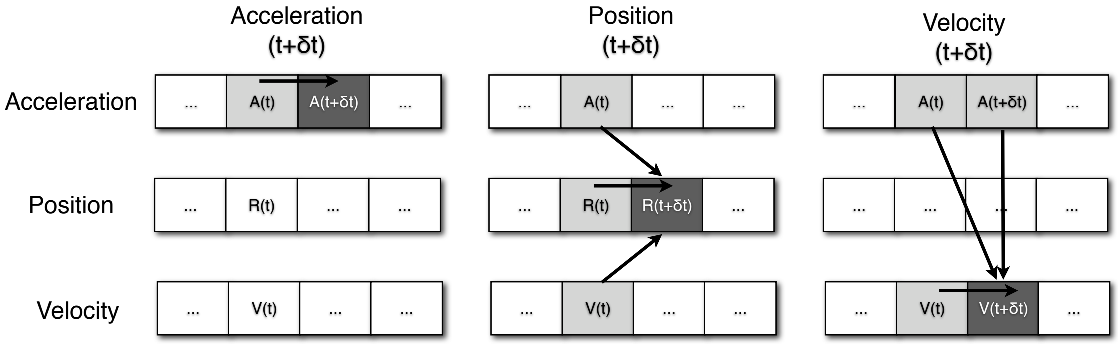


Figure 3.1: Steps involved in calculating the acceleration, position and velocity of the free particle to advance the simulation by one time step. The light gray boxes are the quantities used to calculate the acceleration, position and velocity in the dark gray box at time, $t + \delta t$ (adapted from Allen and Tildesley [7])

into Equation 3.6 to obtain the final expression for the particle at the next time-step ($R(t + \delta t)$), shown in Equation 3.7.

$$R(t + \delta t) = R(t) + V(t)\delta t + \frac{1}{2}A(t)\delta t^2 \tag{3.7}$$

We obtain an expression for the new velocity of the particle as shown in Equation 3.8 below by substituting Equation 3.7 into Equation 3.4 and simplifying. The expressions for the velocity and acceleration terms, calculated from a position time-series are shown graphically in Figure 3.1.

$$V(t + \delta t) = V(t) + \frac{A(t) + A(t + \delta t)}{2}\delta t \tag{3.8}$$

At each simulation time-step, we advance the position, velocity and acceleration of each particle in the system. We first calculate the acceleration of the particle at the next time-step ($t+\delta t$) given by Equation 3.3. Next, we update the position of the particle using the previous velocity and acceleration, shown in Equation 3.7. Finally, we calculate the velocity of the particle at $t+\delta t$ using the average acceleration from the current and previous time-steps and the previous velocity, shown in Equation 3.8. Therefore, the velocity Verlet generates a list of positions, velocities and accelerations in one pass. Since the algorithm requires only one evaluation of any external force at each time-step, it is generally more efficient than other methods. The overall errors in velocity for this algorithm are proportional to δt^2, which makes the velocity Verlet integrator a second order integrator [7]. The result of a single simulation run is a list the particle's position, velocity and acceleration vectors at each time step.

3.3 GPU Architecture and Implementation of the Particle Diffusion Simulation

The GPU is used to transform a set of geometric objects, referred to as the graphics scene, into a two-dimensional bitmap that is output to a display device. The process, which transforms the input geometry into the final bitmap consists of several steps, performed sequentially by the graphics pipeline. An outline of the steps in the rendering process is first presented conceptually, followed by a detailed look at the physical embodiment of the rendering pipeline commonly found in most modern GPUs. We then describe how we leverage stages of the graphics pipeline in the GPU to speedup our Brownian dynamics simulation.

Objects in a graphics scene, defined using arbitrary world coordinates, require several pieces of data – geometric information that defines the shape of the object, lighting, point of view and other state information. The geometry of each object is

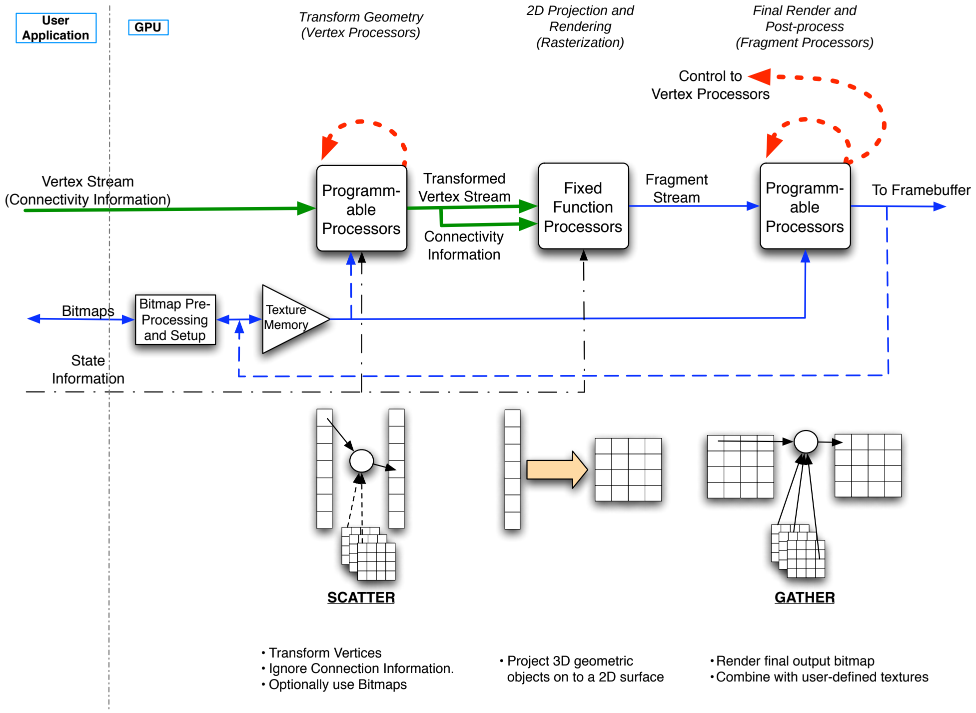


Figure 3.2: Conceptual map of the GPU rendering pipeline. The three main stages are physically represented in the center of the figure while examples of their memory models (scatter and gather) are shown at the bottom.

described using a set of vertices, together with connection information. However, in order to properly display object properties like lighting and color, we also require additional information associated with the surfaces that form the geometry, such as normal vectors, which in turn add additional storage and processing overhead to the GPU. This problem is overcome in the GPU by associating additional data attributes to each vertex that defines the geometric object, such as a Red, Green, Blue (RGB) color, normal vector and other user-defined information. This information is interpolated to generate the correct display attributes for the geometry by the appropriate stages of the pipeline. Therefore a list of vertices, grouped by geometry forms the input to the graphics pipeline. Connectivity information between the vertices of a geometry object, e.g. which vertices are connected by edges, is also stored implicitly within this list.

The rendering process then consists of three stages: geometric transformation, rasterization and post-processing. The first stage of the rendering process manipulates incoming vertices by ignoring the embedded connectivity information and treating each vertex independently. Operations such as translation, rotation, scaling and lighting are applied to every vertex and the net result of these transformations is to align the user defined coordinate system with the coordinates of the viewing plane. This stage of the graphics pipeline is fully programmable and allows user-defined manipulation of individual vertices, as well as any attributes associated with them. The output of this stage is the *transformed* vertex list which is sent to the next stage in the pipeline.

In the rasterization stage, the three-dimensional vertices are first mapped onto the final screen coordinate system and connection information which was implicitly stored in the vertex list is used to render the scene, starting with polygons that are farthest from the viewpoint. Pixels between connected vertices are then filled-in by interpolating the color values between them. Next, pixels that are obscured by polygons lying on top of them and pixels that fall outside the screen viewing area are discarded. The final two-dimensional array is called a fragment stream and contains a RGB color and transparency value for each pixel as well as additional information which encodes knowledge of the geometry each fragment represents. This allows a bitmap to be applied to the fragments in the next stage for photorealistic rendering. Operations in this stage are not user programmable, however specific operations in this pipeline stage, such as the order of the interpolation to fill-in connected vertices, or information that defines the perspective view can be specified beforehand.

The post-processing stage is the last step in the rendering process where the incoming fragment stream is transformed into the final bitmap before being sent to the display device. Optionally user-supplied bitmaps are loaded, transformed and applied to individual fragments in this stage. Special operations to control the appearance of individual pixels on the screen can also be applied before the final bitmap is rendered. Bitmaps are sometimes called a pixel stream when referring to the output of the post-processing stage.

The geometric transformation and post-processing stage are user programmable. Data passing through either of these stages can therefore be manipulated by default system instructions (fixed function mode) or by loading a custom program. More than one program can also be chained together sequentially to perform a series of operations on the data list. Both programmable stages are capable of looping as well as writing their output to graphics memory to be re-used in a subsequent pass.

Recent GPU hardware, such as the nVIDIA 8800 use a unified architecture, where rather than having separate physical stages in the pipeline, operations such as geometric transformation, rasterization and post-processing are executed on processor cores that can be dynamically configured at runtime to handle any of these operations [1]. At the end of each operation, the output is written to a register and looped back as input to the processor core for the next operation. This architecture results in a far more flexible system, especially for general purpose calculations, which for example, often do not use the geometric transformation stages. GPU hardware is fast evolving, causing the specific details of the pipeline stages to change rapidly, even if the functions of the graphics pipeline remain constant. Furthermore, high level APIs such as nVIDIA's CUDA or the more recently released OpenCL have reduced the necessity to understand the intricate details of GPU architecture. However carefully studying the GPUs pipeline functions and memory model are invaluable to extract the most performance from these devices. Therefore, the memory architecture and individual processing stages of the GPU are described next.

The memory model of the GPU differs considerably from that of the CPU. Important memory access restrictions within the individual stages of the pipeline, which are designed to improve the performance of graphics applications, can impact the utility of the GPU in general purpose applications. When sequentially processing the elements of a list, the GPU employs two memory models called *scatter* and *gather*. Given an input and output list, a scatter operation allows the result of a calculation on an input element to be mapped to an arbitrary location in the output list. A gather operation on the other hand, allows the results of a calculation on an input element to be mapped only to its corresponding location in the output list. Both the scatter and gather operations can read an arbitrary memory location in the input data for use in a calculation. The geometric transformation stage allows both scatter and gather operations, while the post-processing stage allows only gather operations. These memory restrictions are consistent with the use of the GPU in graphics applications but may pose a challenge when porting general purpose calculations to graphics hardware.

3.3.1 Implementing Stochastic Simulations on the GPU Stages

The power of the GPU can be harnessed to run simulations of particle diffusion, due to the GPUs ability to advance the simulation simultaneously for several particles. The GPU architecture contains several computational blocks or stages that embody the conceptual rendering pipeline (see Figure 3.2) described previously onto which the simulation algorithm can be mapped. There are however three main stages that are used the most, i) programmable vertex processors, ii) fixed function rasterization and iii) programmable fragment processors.

The particle diffusion simulations are performed entirely using the last stage of the pipeline, called the fragment processors. However, initial stages of the pipeline must first be configured properly to obtain the desired results. The fragment processors simultaneously process a bitmap with dimensions $M \times N$, where, each element of the bitmap can hold a vector, such as the position of a particle. However in order to correctly setup this bitmap in the last stage of the GPU pipeline, we first define a simple rectangular geometry, together with options to configure the environment, so that the first two stages of the GPU – the vertex processors and the rasterization stage – correctly transform this arbitrary geometry into a two-dimensional bitmap with $M \times N$ elements. The fragment processors then run multiple algorithms to execute the simulation, before returning the results to the CPU. Next, we describe each stage of the pipeline, highlighting how we the GPU is used in stochastic particle simulations.

Geometric Transformation (Vertex Processors)

The geometry on which the GPU operates is defined by a list of vertices, as was described earlier. The vertex processors transform the vertices in this incoming vertex list appropriately to align them with the coordinates of the viewing plane. The processors in this stage support the Single Instruction Multiple Data (SIMD) and the Multiple Instruction Multiple Data (MIMD) computational models and are therefore capable of handling branching and looping. In current GPU architectures, any available processing element can be dynamically configured to perform the task of the vertex processors. However, most high-level general purpose GPU APIs, such as nVIDIA's CUDA do not directly provide access to this stage and consequently, this stage is not heavily used in general purpose computing. In some applications, the geometry that is defined in this stage can be designed cleverly so as to leverage the capabilities of the rasterization stage, described next.

Simulations of free particles for optical trapping are designed to simultaneously perform operations on $M \times N$ particles arranged in a rectangular array, each clock cycle. All of the processing however is performed using the fragment processors, described later in this section. Therefore, in order to correctly configure the simulation, we simply setup a rectangular geometry in this stage using arbitrary coordinates and some information about the environment, which when processed through this stage and the rasterization stage, results in a $M \times N$ rectangular array for use in the fragment processors. Each element in this array represents one freely diffusing particle in three-dimensions.

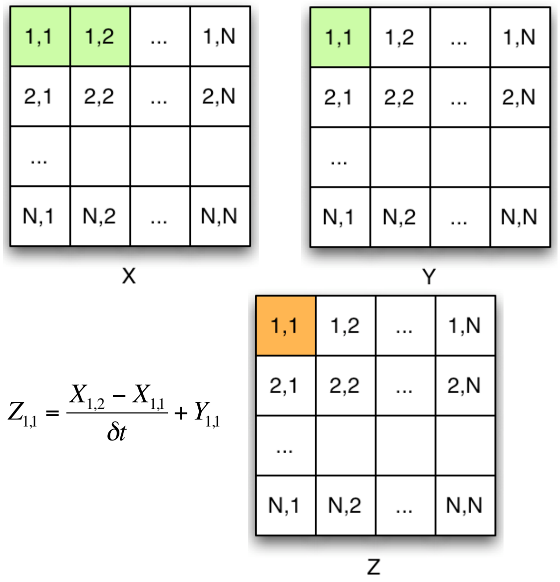
Rasterization

The transformed vertex list is first mapped onto a surface in screen coordinates and the regions between connected vertices are filled-in starting with polygons that are farthest from the viewpoint. As described previously, attributes associated with each vertex are interpolated and polygons and pixels that lie outside the viewing volume are discarded before the results are output as a fragment stream. Since the rasterization step can be used to interpolate any user-defined attribute efficiently, we can use this capability in a calculation by defining an appropriate geometry and attributes in the previous step. The resulting fragment stream will then contain interpolated values of all the defined attributes. Sumanaweera and Liu [146] exploit this functionality to perform Fast Fourier Transforms on large image data sets. Interpolations can also be performed in the post processing stage, but since the rasterizer is designed specifically to perform this operation, utilizing this feature can result in significant performance benefits.

In our simulations the rasterization stage converts the square polygon we defined previously into a 2-D fragment stream with dimensions $M \times N$ for use in the next stage of the pipeline.

Post-Processing (Fragment Processors)

The post-processing stage transforms an incoming fragment stream to the final bitmap that is output to the display device and is the most useful stage in the pipeline for general purpose simulations. The program (user-defined or default



$$Z_{1,1} = \frac{X_{1,2} - X_{1,1}}{\delta t} + Y_{1,1}$$

Figure 3.3: Calculate $z = \frac{dx}{dt} + y$ using the post-processing stage. The green squares in the figure are inputs to the program and the orange squares are the outputs.

system instructions) that manipulates the data can access any attributes that are embedded in the fragment stream as well as additional user-defined bitmaps (2-D data arrays) to calculate the final output. Each location of the user-defined bitmap contains a vector with RGB or RGBA values where each color component can be configured to occupy 16, 24 or 32-bits. The PEs in this stage have the ability to gather information from several sources such as user-defined bitmaps or interpolated attributes embedded within the fragment stream, but they cannot scatter the output of any calculation. Therefore in the post-processing stage, each processed fragment corresponds to a pixel in the output bitmap.

A very useful extension to the graphics instruction set allows the output of the fragment processors to be retained on the graphics card instead of being written to the display device. This provides two important advantages a) it allows the result of a calculation to be returned to the calling program and b) it allows multiple passes or iterations over a data stream using the output of one pass as input to the next. As with any parallel processing hardware, it is advantageous to have an application that iterates multiple times over a data set before it needs to be refreshed or return results. Figure 3.3 shows an example operation in the post-processing stage on a square array. In this example, the gradient calculated using adjacent cells in bitmap X is offset by a constant from bitmap Y to generate the final output. The output can be written to a third bitmap - Z as in this example or can be output to the display device. Bitmap Z is then either stored and used as input in future iterations or returned to the calling CPU program.

The parallel architecture of this GPU stage allows us to simultaneously advance several particles through one simulation time-step δt. The majority of the free particle simulation runs in this stage of the pipeline. Several bitmaps are created to hold the random deviates, position, velocity and acceleration information for $M \times N$ particles. The incoming fragment stream and bitmaps are combined using three distinct steps that implement Equation 3.3, Equation 3.8 and Equation 3.7. This program, shown in Figure 3.4 give the layout of the program in the fragment processors that produces the final acceleration, velocity and position for each δt. The boxes on the left of the figure represent the input textures which hold the initial position, velocity and accelerations. The small boxes on the bottom of the figure are the input constants used in the calculations. The three processing stages combine the inputs to produce the final results shown on the right of the figure. Step 1 calculates the final acceleration at the end of the time-step using the input acceleration, pre-computed random deviates and the scalars for particle mass

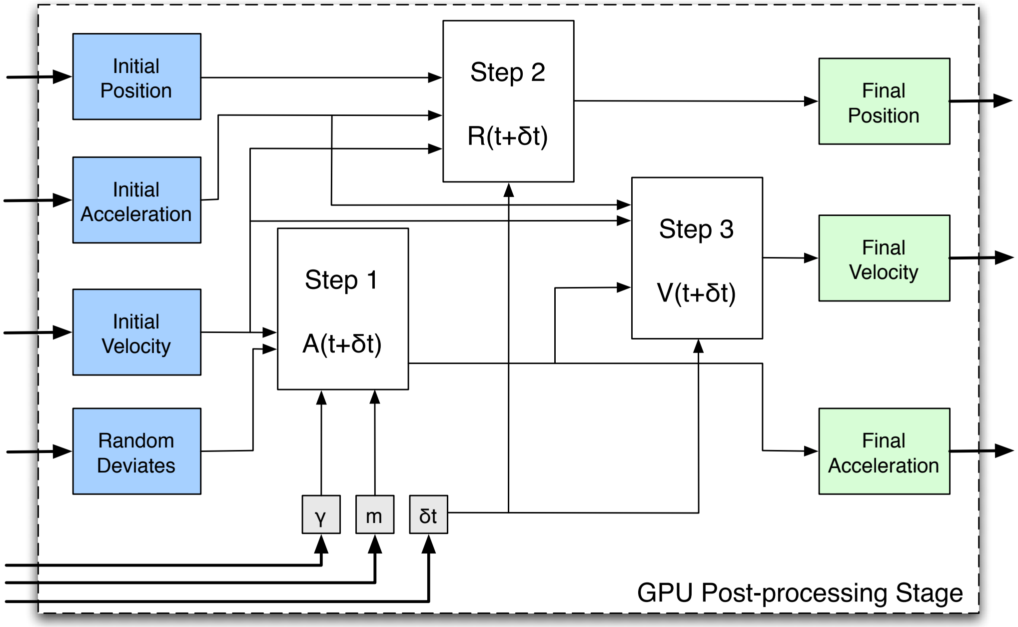


Figure 3.4: Three user-defined programs chained together to implement the velocity Verlet integrator in the post-processing stage of the GPU.

m, viscous drag, γ and time-step, δt. Similarly steps 2 and 3 use the appropriate input bitmaps and scalars shown in Figure 3.4 to calculate the final positions and velocities for each of the $M \times N$ particles.

A 128-bit representation is used to store the X, Y and Z components of a particle's velocity, position or acceleration within the RGB components of a single pixel of the bitmap, while the alpha component is left empty. The PEs in this stage simultaneously process the four color elements and depending on the specification of the graphics card, four to sixteen pixels (and therefore 3-D particles), in one clock cycle. At the end of each simulation pass, the $M \times N$ particle simulation is advanced by one time-step and the output is written to a bitmap, held in the graphics memory. The simulation is allowed to run recursively for several passes before the results are read back to the CPU for storage. This strategy allows us to run a tight simulation loop, with a physically relevant time-step, while sampling the simulation output at longer intervals that are consistent with laboratory experiments. In the case of the GPU simulation, we also get the additional benefit of reducing the overall data communication time, thereby improving performance. At the end of the simulation, a time-series of position, velocity and acceleration for each of the $M \times N$ particles is obtained for analysis.

3.4 Simulation Results and Discussion

The simulation returns the position, velocity and acceleration for each particle at regular time intervals, similar to the data obtained from a laboratory experiment when tracking an ensemble of freely diffusing particles. Several simulation experiments were setup to test the physical accuracy of the CPU and GPU implementations and also to measure the speedup achieved when using the parallel GPU architecture. In each experiment the particles were assumed to be spherical, made of glass and suspended in a water bath at room temperature (293 K) in the absence of any external potential field, such as gravity. The simulation was then run using three representative particle radii of $50\,nm$, $500\,nm$ and $5\,\mu m$. In this section, first we outline the methods used to pick the parameters of the simulations, including the time-step. Next, we discuss the physical results output by the simulation using the CPU and the GPU, followed by an analysis of the rounding error in the GPU results. Finally, we discuss the performance of the GPU.

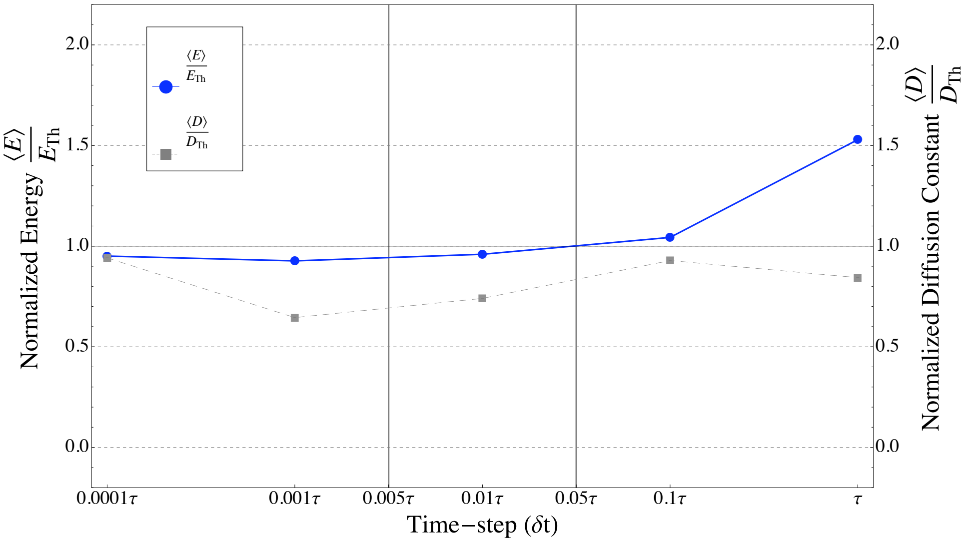


Figure 3.5: Log linear plot of normalized energy ($\frac{E_{Simulated}}{E_{Theoretical}}$) and normalized diffusion constant ($\frac{D_{Simulated}}{D_{Theoreitcal}}$) against time-step ($\delta t$) for an ensemble of 30 glass particles with radius 500 nm suspended in a water bath at 293 K. This data in this plot is used to pick the parameters for the full simulation wiht 900 particles.

3.4.1 Selection of an Appropriate Simulation Time-step

The characteristic time-scale of models that describe free particle diffusion in a fluid medium is given by the velocity relaxation time $\frac{m}{\gamma}$, which is a function of the particle radius, a and the friction coefficient, γ from Stokes law. Choosing a time-step that is much smaller than the characteristic time-scale allows us to capture interesting non-equilibrium behavior that occurs at time-scales shorter than the relaxation time, but is also important to maintain energy conservation in the simulation. For the sake of convenience in determining the time-step, we pick a number τ, which is the closest multiple of 10 smaller than $\frac{m}{\gamma}$. We then pick the simulation time-step to be some fraction of τ.

Figure 3.5 shows a plot of the mean kinetic energy of an ensemble of 30 particles with radius 500 nm, normalized by the thermal energy ($\frac{1}{2}k_BT$), as a function of time-step (δt). The figure also shows, in gray, the calculated value of the diffusion constant for the $500\,nm$ diameter particle, as a function of time-step. The selection of parameters to test the physical validity of the simulation, such as the mean energy and the diffusion constant, are discussed in detail in Section 3.4.2.

This small ensemble is only used to determine suitable parameters for the full simulation with 2700 particles and 10^6 time-steps, discussed later in this section. We find that the relative error in the normalized energy increases for longer time-steps – approaching $60\,\%$ for a time-step of one τ. For 30 particles and time-steps shorter than $0.1\,\tau$, the relative error in the normalized energy is better than 5 %. As we shall see later in this section, this error reduces to below $1\,\%$ for the larger ensemble. We also find that the relative error does not decrease significantly when δt is reduced further.

From the figure, we see that in order accurately estimate the equilibrium ensemble energy of the system, the time-step selected for the simulations must be much smaller than the relaxation time. Therefore, we test the simulations for time-steps ranging from $0.005\,\tau$ to $0.05\,\tau$. As the size of the time-step is reduced even further, the Brownian motion model has been observed to fail [95]. At very short observation times, the effects of fluid inertia become significant and the model for free particle diffusion must therefore account for these corrections. However, this does not affect our measurements. We pick δt close to the largest allowable value where the error in the equilibrium averages is small. The simulation is then repeated for each particle radius with three representative time-step values $0.005\,\tau$, $0.01\,\tau$ and $0.05\,\tau$. Table

Table 3.1: Simulation Parameters (SI Units)

Sphere Material: Glass	
Fluid Medium: Water	Fluid Temperature (T): 293 K
Number of Trajectories (N): 2700	Data-points per trajectory (n): 10,000

a	m	$\frac{m}{\gamma}$	δt ($Simulation\ Time = \delta t \times n \times 100^*$)					
			0.005τ		0.01τ		0.05τ	
50 nm	1.3614 ag	1.442 ns	5 ps	(5 μs)	10 ps	(10 μs)	50 ps	(50 μs)
500 nm	1.3614 fg	144.2 ns	500 ps	(500 μs)	1 ns	(1 ms)	5 ns	(5 ms)
5 μm	1.3614 pg	14.42 μs	50 ns	(50 ms)	100 ns	(100 ms)	500 ns	(500 ms)

* Since only every 100^{th} point is recorded, the total simulation time is 100 times longer

3.1 gives the simulation parameters for the nine distinct simulation cases.

Each of the nine simulation experiments was carried out with 900 independent particles using a three-dimensional Cartesian coordinate system. The simulation algorithm was run for a total of 1 000 000 time-steps, but only every 100^{th} data point was recorded resulting in a total of 10 000 data points per particle trajectory. Although the simulation experiments were run for a fixed number of iterations, the choice of time-step ensures that each simulation run is longer than several hundred relaxation times thereby giving the system sufficient time to equilibrate. All particles in the simulation were started with initial positions at the origin of the coordinate system and with velocities sampled from an appropriate thermal distribution. Each experiment was run on the CPU (using single, CPU32 and double, CPU64 precision floating point representation) and on the GPU (using single precision, GPU32 floating point representation). The simulations are run for 2700 non-interacting particles to generate trajectories with 10 000 time-steps per particle.

The physical accuracy of the recorded data was verified against quantities such as the thermal energy and the diffusion constant obtained from the theory of Brownian motion. The effects of time-step, particle radius, round-off error and statistical error on the accuracy of the simulation results are then discussed. The time required for the simulation to complete was also recorded for the CPU32, CPU64 and GPU32 implementations. This information was subsequently used to calculate the speedup obtained by running the simulation on the GPU. Finally, the scaling of speedup as a function of the number of particles is discussed.

3.4.2 Physical Validation

A complete description of the physical parameters of a collection of non-interacting freely diffusing particles can be obtained from the joint probability distribution of their positions and velocities, also called the phase space distribution. At times longer than the characteristic time, when the system is in thermal equilibrium with the surrounding fluid, the velocity distribution becomes the time-invariant Boltzmann distribution. Therefore, the position and velocity distributions can be studied independently and the position and velocity outputs of the simulation are both tested against the theory of diffusion.

The velocity output by the simulation, can be used to calculate the energy distribution of the particles, given by the time-invariant Boltzmann distribution. Therefore, in order to fully test the simulation, we test energy conservation by calculating the mean energy, which must equal the thermal energy from the equipartition theorem, as well as the energy distribution of the particles, which must be consistent with the Boltzmann distribution. The position distribution constructed from the simulation output, however is not time-invariant. Instead, the size of the position distribution increases proportional to the square-root of time. The proportionality constant, which must equal twice the diffusion constant, can then be recovered and compared with the theoretically predicted value to test the output of the simulation. The techniques developed to analyze the simulation data are described next.

3.4.2.1 Energy Conservation

In order to verify energy conservation, we i) calculate the mean kinetic energy and the energy distribution and then compare them with theoretical values and ii) verify that on average the ensemble of particles does not gain or lose energy over time. The energy distribution for an ensemble of freely diffusing non-interacting particles is given by the Boltzmann distribution, which takes the form $A\,exp\,(-E/k_BT)$, where A is a normalization constant obtained by setting the area under the distribution to unity, E is the energy of the particle, k_B is Botzmann's constant and T is the temperature. For freely diffusing particles, the energy is simply the kinetic energy, $\frac{1}{2}mV^2$ and the Boltzmann's distribution can be written in the normalized form shown in Equation 3.9.

$$\rho = \sqrt{\frac{m}{2\pi k_BT}}\,exp\left(-\frac{mV^2}{2k_BT}\right) \tag{3.9}$$

Comparing Equation 3.9 to a normal distribution, we obtain the variance of the velocity to be $\langle V^2\rangle = \frac{k_bT}{m}$ and the mean is zero. From the expression for the variance of the distribution, we directly obtain the relationship given by Equation 3.10, which relates the thermal energy of the particle to it's kinetic energy. Moreover, Equation 3.10 is a statement of the equipartition theorem.

$$\frac{1}{2}m\left\langle V^2\right\rangle = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}mV_i^2 = \frac{1}{2}k_BT \tag{3.10}$$

Using the velocity data output by the simulation we calculate the energy of all the particles at each time-step and then verify that the mean of the calculated kinetic energies, $\langle E\rangle$, satisfies Equation 3.10. The standard deviation of the particle's energy is then calculated by first calculating the energy of all N particles at every time-step (E_i) using the left side of Equation 3.10. The standard deviation (σ) and the standard error (SE) are then calculated using Equation 3.11.

$$\begin{aligned} \sigma &= \left(\frac{1}{N}\sum_{i=1}^{N}(E_i - \langle E\rangle)^2\right)^{1/2} \\ SE &= \frac{\sigma}{\sqrt{N}} \end{aligned} \tag{3.11}$$

The mean energy, normalized by the theoretical value of the thermal energy for each simulation case are shown in Table 3.2. The data is then compared between CPU and GPU implementations. Finally, in order to satisfy energy conservation, the net change in the mean energy of the simulation must be statistically consistent with zero.

Table 3.2 gives the average energy ($\langle E\rangle$), standard deviation (σ), standard error (SE) and normalized energy, $E_{Norm} = \frac{\langle E\rangle}{E_{Theoretical}}$ calculated for varying particle size, time-step and for when the simulation is run on the CPU32, CPU64 and GPU32 platforms. The relative error in the mean energy, calculated as the difference between the values calculated from the simulation data and the theoretical values, is better than 0.5 % when $\delta t = 0.005\tau$ and $\delta t = 0.01\tau$ and reaches a maximum of 1.77 % for $\delta t = 0.05\tau$ on the GPU. The error in the calculated mean energy of the simulation is always better than the 2.5 % reported in the literature [28, 7] for the velocity Verlet integration algorithm. From Table 3.2, we see that variations in the magnitude of the mean energy ($\langle E \rangle$) across simulation cases is consistent with the reproducibility (SE). The magnitude of the calculated error also increases with the size of the time-step as expected and it is observed that particle size has no systematic effect on the equilibrium average energy. The difference between the normalized energy calculated using the CPU32/CPU64 generated data and the GPU32 generated data is also small for each simulated case.

The Boltzmann energy distribution, which has the form of an exponential energy distribution, has the same mean and standard deviation. Therefore from the results in Table 3.2, we expect the columns for the mean energy ($\langle E \rangle$) and the standard deviation (σ) to be the same. However, this is not the case and the standard deviation is larger than the mean energy. The error is most likely introduced from the velocity Verlet algorithm, which is known for excellent long term energy conservation, but only moderate energy conservation at shorter times [56]. This necessarily implies that the fluctuations in the energies for this numerical integration method will be larger than expected. The larger fluctuations however are only expected to contribute very small errors to the particle position in optical tweezers applications and therefore do not pose a significant problem. Finally, energy drift for the CPU and GPU implementations is found to be excellent and the energy in each case shows no appreciable gain or loss over time.

Table 3.2: Ensemble Average Energy

		$\delta t = 0.005\tau$				$\delta t = 0.01\tau$				$\delta t = 0.05\tau$			
		$\langle E\rangle$ (J) $\times 10^{-21}$	σ (J) $\times 10^{-21}$	SE (J) $\times 10^{-25}$	E_{Norm}	$\langle E\rangle$ (J) $\times 10^{-21}$	σ (J) $\times 10^{-21}$	SE (J) $\times 10^{-25}$	E_{Norm}	$\langle E\rangle$ (J) $\times 10^{-21}$	σ (J) $\times 10^{-21}$	SE (J) $\times 10^{-25}$	E_{Norm}
$CPU32$	50 nm	2.0247	2.8636	5.5110	1.0010	2.0300	2.8712	5.5256	1.0036	2.0584	2.9117	5.6035	1.0177
	500 nm	2.0264	2.8649	5.5135	1.0019	2.0308	2.8728	5.5288	1.0040	2.0578	2.9101	5.6005	1.0174
	5 μm	2.0259	2.8664	5.5164	1.0016	2.0299	2.8719	5.5269	1.0036	2.0572	2.9105	5.6013	1.0171
$GPU32$	50 nm	2.0261	2.8653	5.5142	1.0017	2.0285	2.8691	5.5216	1.0029	2.0585	2.9105	5.6012	1.0177
	500 nm	2.0252	2.8640	5.5118	1.0013	2.0294	2.8694	5.5222	1.0033	2.0572	2.9088	5.5979	1.0171
	5 μm	2.0246	2.8641	5.5120	1.0010	2.0311	2.8724	5.5280	1.0042	2.0585	2.9112	5.6025	1.0177
$CPU64$	50 nm	2.0253	2.8634	5.5107	1.0013	2.0288	2.8680	5.5195	1.0030	2.0585	2.9105	5.6012	1.0177
	500 nm	2.0264	2.8655	5.5146	1.0019	2.0296	2.8704	5.5241	1.0034	2.0577	2.9094	5.5991	1.0173
	5 μm	2.0263	2.8650	5.5137	1.0018	2.0297	2.8698	5.5230	1.0035	2.0578	2.9099	5.6001	1.0174

$E_{Theoretical} = 2.0227 \times 10^{-21}$ J at T= 293 K

Table 3.3: Diffusion Constant

			$\delta t = 0.005\tau$			$\delta t = 0.01\tau$			$\delta t = 0.05\tau$		
		D_{Th} $(\frac{m^2}{sec})$ $\times 10^{-13}$	$\langle D\rangle$ $(\frac{m^2}{sec})$ $\times 10^{-13}$	σ $(\frac{m^2}{sec})$ $\times 10^{-14}$	D_{Norm}	$\langle D\rangle$ $(\frac{m^2}{sec})$ $\times 10^{-13}$	σ $(\frac{m^2}{sec})$ $\times 10^{-14}$	D_{Norm}	$\langle D\rangle$ $(\frac{m^2}{sec})$ $\times 10^{-13}$	σ $(\frac{m^2}{sec})$ $\times 10^{-14}$	D_{Norm}
$CPU32$	50 nm	42.8363	42.6539	4.5405	0.9957	42.2938	7.9684	0.9873	42.5465	5.2338	0.9932
	500 nm	4.2836	4.4637	0.4792	1.0420	4.2871	0.4093	1.0008	4.4640	0.4792	1.0420
	5 μm	0.4284	0.4295	0.0568	1.0027	0.4248	0.0781	0.9917	0.4309	0.0414	1.0060
$GPU32$	50 nm	42.8363	40.9561	9.0678	0.9561	42.5527	4.9765	0.9934	43.7657	3.6273	1.0217
	500 nm	4.2836	4.0610	0.6134	0.9480	3.9420	0.7599	0.9202	4.0610	0.6134	0.9480
	5 μm	0.4284	0.3981	0.01210	0.9294	0.4070	0.0347	0.9501	0.4247	0.0734	0.9916
$CPU64$	50 nm	42.8363	42.2823	7.4140	0.9871	43.8533	4.4342	1.0237	43.2453	4.3501	1.0095
	500 nm	4.2836	4.2563	0.4649	0.9936	4.3589	0.5775	1.0176	4.2563	0.4649	0.9936
	5 μm	0.4284	0.4371	0.0844	1.0204	0.4368	0.0631	1.0196	0.4234	0.0350	0.9884

Standard Error (SE) = $\sigma \times 10^{-2}$ $\frac{m^2}{sec}$ for all cases

3.4.2.2 Diffusion Constant

The mean squared displacement of a freely diffusing particle in one-dimension is given by the diffusion relation in Equation 3.12, where $\frac{k_B T}{\gamma}$ is the diffusion constant, D [62].

$$\langle R^2 \rangle = 2\frac{k_B T}{\gamma} t \tag{3.12}$$

The procedure for obtaining the diffusion constant from the simulation data is shown in Listing 3.1 below. The simulation outputs the particle positions as a time-series for each particle. All of the simulation output can be represented as a two-dimensional matrix, $R_{N\times M}$ with rows representing N particles and the columns representing M time-steps per particle. First we transpose this matrix, by interchanging the rows and columns. The new matrix obtained, $R_{M\times N}$, has columns that represent the position of all the particles in the simulation at a single time-step and the rows represent the time-step. Next, we calculate the variance of the particle positions in each column to obtain the variance of the particle positions as a function of time. Finally, we fit the variance to a straight line of the form $a\,t\ +\ b$, where the constant a must equal twice the diffusion constant (D) and b must be consistent with zero.

```
 // Transpose the position matrix to group the particle
 // positions by time-step
 $R_{M\times N}$ = TRANSPOSE( $R_{N\times M}$ )

// Calculate the variance of each column in the
// new matrix $R_{M\times N}$
 $\sigma^2_M$ = VARIANCE of each column in $R_{M\times N}$

 // Fit $\sigma^2_M$ to a line of the form $(a\,t + b)$
```

```
$\{a, b\}$ = LEAST_SQUARES_FIT $\sigma_M^2$ to function $(a t + b)$
```

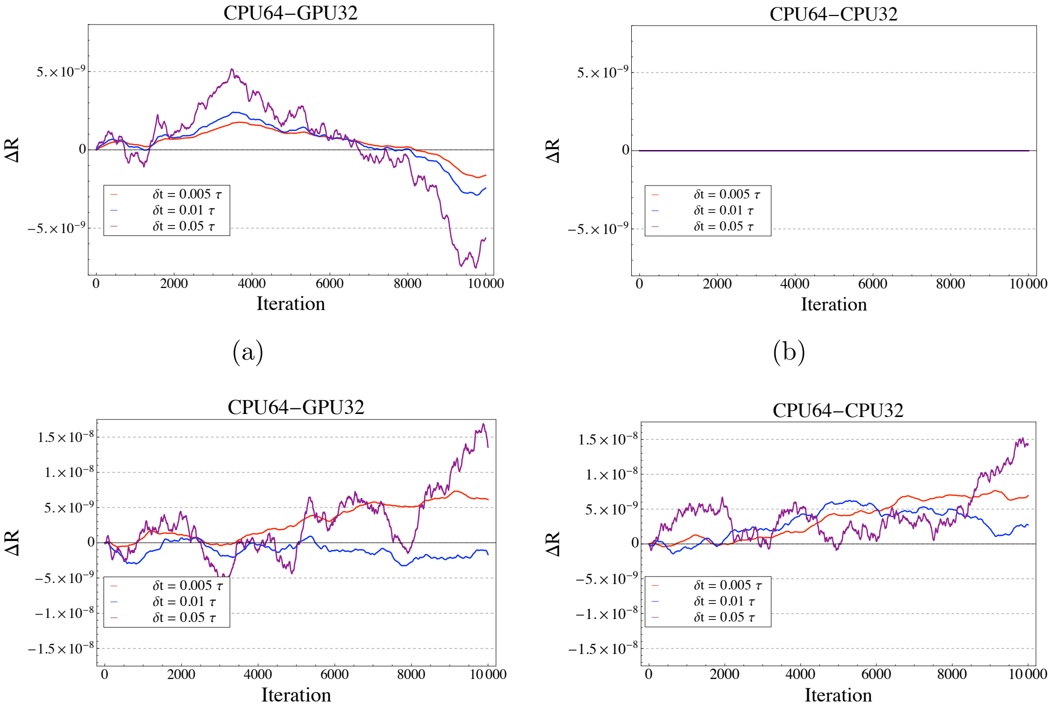
Listing 3.1: Pseudo-code to calculate the diffusion constant

Table 3.3 shows the average diffusion constant ($\langle D \rangle$), the standard deviation (σ), standard error (SE) and the normalized diffusion constant, $D_{Norm} = \frac{\langle D \rangle}{D_{Theoreitcal}}$ for each simulation case. The columns show the data for each implementation and the three representative particle sizes and the rows show the results for different time-steps. The effect of particle size and time-step on the error in the calculated value of the diffusion constant, D_{Norm} is small. The calculated diffusion constant is found to be on average within 1.4 % of the the theoretical value for the CPU32 and CPU64 implementations, with maximum errors within 4.2 % and 2.4 % respectively. However, the average error in the diffusion constant for the GPU32 runs was found to be within 4.3 %, with the maximum error within 8 % of the theoretically expected values. From the table, we can observe that the normalized diffusion constant (D_{Norm}) for the CPU simulations is consistent with reproducibility, shown by the standard error in the table. However, on the GPU the errors in the normalized diffusion constant are slightly larger. This small increase for the GPU32 implementation is caused by rounding errors.

3.4.2.3 Analysis of GPU Rounding Error

The accuracy of results from the particle simulation depend mainly on errors introduced by the simulation algorithm, random number generator and roundoff error. In our simulations, algorithmic and random number errors exhibit similar behavior since the same algorithm is used across the CPU and GPU implementations and the random numbers are always generated on the CPU in double precision. Therefore the difference in simulation results on the GPU and CPU, arises primarily from varying rounding errors on each architecture. In order to investigate the GPU rounding error, relative to the CPU, we generate a position trajectory for a single $500\,nm$ particle on the CPU (32 and 64-bit) and on the GPU using the same list of random deviates and simulation conditions. Therefore, we expect the trajectories generated on the CPU and the GPU to be identical, however this is not the case. Using the CPU64 trajectory as a reference, we plot the relative error in the CPU32 and GPU32 trajectories (by subtracting them from the CPU64 reference) for the three separate time-steps in Figure 3.6. The figure shows two cases, one where the particle is initially located at the origin (top row of the figure) and the second, where the particle starts at $100\,nm$ from the origin, where we expect the resolution of the single precision floating point representation in adding numbers of different magnitudes in Equation 3.7 to be poor (bottom row of figure).

The case where the particle starts at the origin, shown in Figure 3.6(a) and Figure 3.6(b), exactly duplicate the conditions for the simulations tested in Table 3.2 and Table 3.3. From these figures, we see that the plots look qualitatively different. In fact, the error for the GPU in Figure 3.6(a) diverges very rapidly from the CPU64 reference trajectory to result in large relative errors. On the other hand, from Figure 3.6(b), we see that the relative errors in the CPU32 plot are negligible. The differences in the two plots can be explained due to differences in the method



(a) (b)

(c) (d)

Figure 3.6: Relative error observed by subtracting a reference CPU64 position trajectory for a single 500 nm particle from those of the CPU32 and GPU32 implementations. All trajectories are generated using the same list of random deviates. In figures (a) and (b), the particles is initially located at zero, whereas in figures (c) and (d), the particle is initially located 100 nm from the origin.

in which the CPU and GPU store the result of floating point operations. The CPU, which follows the IEEE-754 floating point convention always performs exact rounding in all its operations, similar to double-precision CPU operation. Therefore, the relative errors in the generated trajectories are very small in comparison to the CPU64 reference trajectory. The GPU, on the other hand, depending on the vendor, performs exact rounding in some operations and chops the result of other operations, which can lead to errors that vary as much as a factor of 2 per time-step [64, 76]. However, since individual trajectories matter less, in optical trapping simulations, than statistical averages, the use of the GPU does not adversely impact the results.

On the other, when the particle is initially located $100\,nm$ from the origin, shown in Figure 3.6(c) and Figure 3.6(d), the relative errors obtained from the GPU appear qualitatively comparable to those on the CPU for single-precision operations. The relative errors on the CPU and GPU are of the same magnitude, but increase as expected when larger time-steps are used to generate the trajectories. By moving the particle away from the origin, both the single-precision CPU and the GPU simulation experience larger relative errors in comparison with the double precision CPU reference, due to a loss in precision, which arises from adding numbers of significantly different magnitudes in Equation 3.7. This is not problematic since the simulation is never run long enough, on the GPU or in single precision CPU simulations, to trigger this condition. Also, this situation is easily avoided in longer simulations by shifting the particle back and recording the offset when it diffuses a certain distance away from the center.

Errors on the GPU, in stochastic simulations that are run for millions of time-steps, can arise from rounding errors or a loss of precision, when the particle diffuses far away from the origin. However, the errors in individual trajectories are less important in optical trapping simulations than statistical errors, calculated by aggregating a large number of trajectories. Moreover, the statistical averages calculated using GPU data, yield only slightly larger errors than their CPU counterparts, as seen in results earlier in this section.

3.4.3 Speedup

The total simulation time is divided into the computation time, the communication time (the time spent for moving data to and from memory to the on-chip cache) and the file I/O time (the time spent when writing the results to disk for analysis). The speedup of the GPU over the CPU is then calculated as the ratio of the sum of the computation and communication time for the entire simulation run. For both the CPU and GPU implementations, the random deviates are precalculated for use in the simulation as the algorithms to generate random numbers works best on the CPU. However, we have also implemented other random number generators, such as the Mersenne-Twister, to run entirely on the GPU. Moreover, using the same random number generator on the CPU and the GPU allows us to directly compare trajectories on both platforms, as seen in the previous section, in order to characterize the performance and accuracy of running the simulations on the GPU. Speedup was calculated for an ensemble of particles radius of 500 nm and a time-step, $\delta t = 0.05\tau$. The experiments were repeated for ensemble sizes of 100,

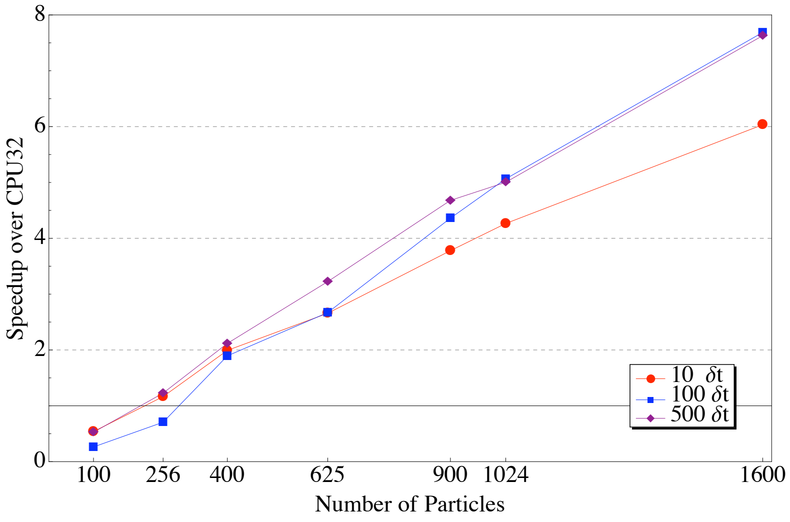


Figure 3.7: Speedup (calculated as the ratio of the GPU simulation time over single precision CPU simulation time) as a function of the ensemble size. The three curves show the results when the output was sampled after every 10, 100 and 500 simulation time-steps (δt).

256, 400, 625, 900, 1024 and 1600 particles. For each particle size, the simulation was run for a constant time and the output sampled every 10, 100 and 500 steps. All experiments were run on a Dell desktop with a dual core Intel Pentium, D processor with a clock speed of 2.8 GHz, 1 GB DDR2 RAM and a NVIDIA GeForce 7950GT PCI Express graphics card with 512 MB GDDR3 Video RAM running the OpenSuSe 10.2 Linux operating system.

Figure 3.7 shows the speedup obtained by the GPU over a single-precision CPU implementation. The three curves in the figure show the speedup when the output is sampled every 10, 100 and 500 time-steps. For ensembles with fewer than 256 particles, the CPU is faster because the time required to transfer data to the GPU memory is large compared to the computation time. As the ensemble size grows, the computation and communication times required by the GPU to run one simulation experiment remains constant while the computation time required by the CPU to run the same experiment grows steadily. This results in large performance gains by the GPU. We also see some performance benefits when the simulation is allowed to loop on the GPU for several cycles before the data is recorded. Figure 3.7 plots the speedup against the ensemble size in three cases where the output is sampled every $10\delta t$, $100\delta t$ and $500\delta t$. We observe that sampling the output every 100 time-steps yields better performance over the $10\delta t$ case but increasing the sampling interval to $500\delta t$ does not significantly change the observed speedup. The GPU hardware used supports simulations with more than one million particles, which would significantly increase the speedup. However, while the computational time on the CPU scales linearly, the computational time on the GPU is not expected to be linear due to the time required to transfer data to and back form the GPU. Therefore, the speedup as a function of particle size is not expected to increase linearly for simulations that have thousands of particles.

3.4.4 The GPU as a Coprocessor in Stochastic Simulations

The GPU, when used to run a free particle simulation yields significant speedup, especially with large particle ensembles. However the benefits of using the GPU can be leveraged for more general simulations of the optical tweezers system as well as other stochastic systems. Most simulations of micro and nanoscale assembly tasks include an external force, which is calculated periodically. These forces are often pre-computed and stored in a parametric form on the CPU. When running the stochastic simulation on the GPU, the number of times the simulation loops on the GPU is tuned to closely match the timescale of the problem. For example in the case of the optical tweezers simulation, we allow the simulation to loop for 100 time-steps on the GPU before we read data back to the CPU. This corresponds to a time in which the particle undergoes very little movement within the trap and therefore the optical trapping force on the particle can then assumed to be constant and still yield physically accurate results.

3.5 Summary

A Brownian dynamics simulation to model the diffusive motion of nanoparticles in a fluid has been developed and tested. The simulation algorithms were implemented on the CPU with 32-bit and 64-bit floating-point precision and on the GPU with 32-bit precision. Equilibrium averages for energy conservation and the diffusion constant were calculated from the simulation results, for an ensemble of 2700 particles with diameters of $50\,nm$, $500\,nm$ and $50\,\mu m$. For the benchmark 64-bit CPU case, the maximum relative errors in the equilibrium averages, compared with the well-established theory of diffusion, were found to be approximately 1% for energy conservation and approximately 2% for the diffusion constant.

The simulation for particle diffusion was also implemented on the GPU and the results were compared with the reference double precision CPU case. The maximum error in energy conservation was found to be approximately 2%, however the maximum relative error in the diffusion constant was 8%, which is slightly higher than that calculated on the CPU. The larger error is attributed to lower precision on the GPU (since the GPU uses single precision floating-point numbers), as well as higher rounding errors, which affect the particle trajectories used in calculating the diffusion constant. The slightly larger errors on the GPU, however account for a uncertainty in the particle's position that is only a small fraction of it's diameter over the course of the simulation. Moreover the accuracy of individual trajectories for a freely diffusing particle does not adversely affect a nanomanipulation operation with optical tweezers.

Finally, the computation time for the CPU and GPU simulations, recorded as a function of the ensemble size is used to calculate the speedup obtained by using the GPU. For ensembles that contain fewer than 256 particles, the GPU performs worse than the CPU, as expected since the time to transfer data between the CPU and GPU memory is much greater than the computation time. However for an ensemble of 1600 particles, which is comparable to the ensemble size we would need to test optical tweezers operations, we obtain a speedup of 8.

The simulation algorithms allow the inclusion of force models to include the binding force from an optical trap. Moreover the algorithms transparently support trapping forces calculated from approximate models, physics based force models or even from laboratory data. This is demonstrated in Chapter 4, where two new force measurement methods are developed and tested using the simulation. Data collected after implementing these methods in the laboratory can then be input back into the simulations to further improve their fidelity in replicating laboratory conditions.

Chapter 4

Novel Measurement Techniques to Characterize the Optical Trapping Force Experienced by Nanoparticles in an Optical Trap

Two new force measurement techniques are developed in this chapter[1] for use in the laboratory. Accurate models of the optical binding force experienced by a nanoparticle in an optical trap form an important component of physically accurate simulations of optical trapping. When combined with an accurate trapping force model, the simulations developed previously in Chapter 3, play a vital role in the discovery of new techniques to enhance optical trapping at the nanoscale. In particular, we have used simulations to drive the development of novel control algorithms, such as those demonstrated in Chapter 5 that have significantly improved the performance of the optical tweezers instrument when interacting nanoparticles.

One route to developing an accurate trapping force model, is through a calculation of the optical trapping forces experienced by the nanoparticle from first principles. An optical trap is created in the laboratory by using a high numerical aperture objective, which strongly focusses the incident light in order to maximize the binding force acting on the particle. However, optical traps formed by tightly focussing the incident light can in some configurations suffer from spherical aberrations, where light passing through the edges of the lens comes to a focus at a

---

[1] The work in this chapter is derived from the published work in [17, 20, 18].

different axial location than light passing through the center. When combined with diffraction from finite sized apertures in the optics and other optical misalignments, this can lead to a deviation from an ideal optical trap, which must then be accounted for in the model. Currently, no single theoretical force model exists for tightly focussed beams or which works for a range of particle sizes and materials and can simultaneously account for all of the phenomenon that cause an optical trap to deviate from ideal behavior. Moreover, the development of such a force calculation is very challenging, as demonstrated by the lack of agreement between theory and experiment.

A more direct approach to developing a force model, for inclusion in numerical simulations, is to experimentally measure the optical binding force, experienced by a nanoparticle in an optical trap. This laboratory based technique has the added advantage that it will include all the phenomenon that cause a deviation from the ideal behavior of an optical trap. However, existing force measurement techniques are not well equipped to accurately measure the force experienced by a weakly bound nanoparticle in an optical trap, for reasons outlined in more detail in Section 4.1.

Therefore, two new force measurement techniques are developed in this chapter in order to measure the binding forces acting on nanoparticles in an optical trap. A significant advantage, over existing methods, offered by the new force measurement techniques developed here is that we are able to measure the complete force profile of an optical trap, making no assumptions about the nature of the underlying force. These new techniques are validated using the numerical simulations algorithms developed previously. Finally we present results obtained by implementing these techniques in the laboratory and in Chapter 5 show how these force measurement techniques are used to estimate the parameters of a controller.

4.1 Motivation

Particles larger than a few hundred nanometers experience a sufficiently strong binding force within an optical trap to keep them confined to a spatial region very close to the center of the trap. This is illustrated in Figure 4.1, which shows the effect of particle size on the depth of the optical trapping potential for a constant beam power. The figure shows the optical trapping potentials for two representative nanoparticles on the left and the corresponding trapping forces on the right. The trapping force acting on the larger particle, shown by the blue curve in the right side of the figure, is much stronger than the force from the thermal energy of the particle. Therefore, the particle is confined to a very small region, close to the center of the trap, shown by the shaded region on the left side of the plot. In this region, the optical trapping force can be assumed to vary linearly with the displacement of the particle and the trap can be approximated by a linear spring. Consequently, existing force measurement techniques, such as those discussed in Section 2.4, approximate the optical trap as a harmonic oscillator. Under the harmonic approximation, the optical trap can be parameterized using only a linear stiffness and any higher order corrections to the optical force are ignored.

As the size of the particle decreases, the optical binding force confining the nanoparticle within an optical trap decreases significantly, for constant beam power, as seen from the purple curve in Figure 4.1. Consequently, the thermal energy of the particle can equal or even exceed the potential energy of the trap, allowing it to make larger excursions away from the trap center and even quickly escape the trap, shown by the shaded region for the purple curve of the figure. Therefore, under conditions where the nanoparticle is no longer localized close to the trap center, the assumption that the optical trap can be approximated by a harmonic oscillator fails. Existing force measurement techniques can then no longer provide an accurate measure of the trapping force.

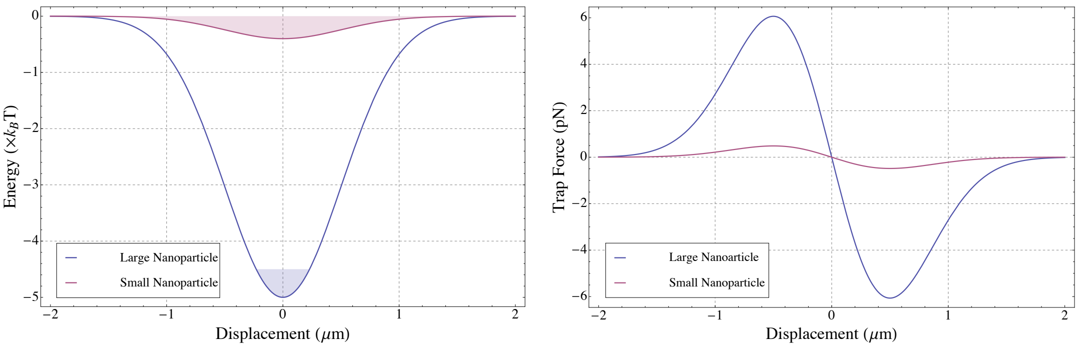


Figure 4.1: Relative scaling of the depth of the optical trapping potential and force with particle size for a constant beam power. The weaker optical trap for smaller particles, results in larger excursions of the particle away from the trap center, in comparison with the larger particles, shown by the shaded regions in the plots.

Therefore, we develop new measurement techniques, which do not make any prior assumptions about the nature of the trapping force. Two new methods are developed, i) the step input, which estimates the trapping force by measuring the change in position of a nanoparticle, in response to a rapid movement of the optical trap and ii) the stochastic noise method, which recovers the optical trapping force from a measurement of the random thermal motion experienced by a nanoparticle in a stationary optical trap. Numerical simulations play a crucial role in developing and validating the new force measurement techniques. The simulation algorithms closely mimic the measurements of the optical force made in the laboratory and yield a time-series of the particle's position as output. The analysis techniques used to recover the profile of the optical trapping force are then exactly the same for both simulation and laboratory data. Therefore, in order to test the effectiveness of the force measurement methods, we first input an idealized approximate trapping force model to the numerical simulations. We then run simulations repeatedly to obtain several trajectories of the nanoparticle over the course of the measurement, as we would in the laboratory. The analysis of this data is then used to recover the trapping force. Based on the results of the simulations, we then implement the new force measurement techniques in the laboratory.

The two new methods, developed to measure the optical trapping potential are described next. The step input method, which relies on the response of a nanoparticle to a step input provided to the optical trap is described first in Section 4.2, followed by results from numerical simulations, as well as laboratory experiments in Section 4.3. The stochastic noise method, which relies on the thermal fluctuations of a nanoparticle in an optical trap is described in Section 4.4, followed by results for this technique in Section 4.5. Finally, a summary is presented in Section 4.6.

4.2 Step Input Method

In the step input method, the optical trapping force is estimated by measuring the trajectory of a nanoparticle in response to being rapidly displaced from the center of the trap. The process is illustrated more clearly for the one-dimensional case in Figure 4.2. In the figure, a nanoparticle, initially trapped close to the center of an optical trap, shown in Figure 4.2(a), is subjected to a step input by rapidly moving the trap relative to the particle. At the end of the step, the nanoparticle is displaced from the center of the trap by the step-size and the optical trap then exerts a restoring force, which drives the particle back to the center (see Figure 4.2(b)). The particle returns to the center of the trap, some time after the initial step input, depending on the size of the particle, the size of the step and magnitude of the force exerted by the trap. The process is repeated by moving the trap in the opposite direction, in order to measure the optical trapping force on either side of the center (shown in Figure 4.2(d)).

In the step input method, we measure the trajectory of the nanoparticle, as a function of time, as it responds to the step input and then use it to calculate the optical trapping force. Ignoring random thermal noise, the motion of the nanoparticle as it returns to equilibrium is governed by Newton's equation of motion, given in Equation 4.1, where $\vec{r}(t)$ is the position of the particle in Cartesian coordinates, m is the mass and γ is the Stokes coefficient for a spherical particle. For larger micrometer sized particles in a strong optical trap, the thermal force acting on the particles is overwhelmed by the force from the optical trap and has a negligible effect on the trapping force measurement. Smaller particles, which experience a weaker optical binding force, take longer to return to equilibrium and also experience a higher thermal force, which increases the noise in the position trajectory. However,

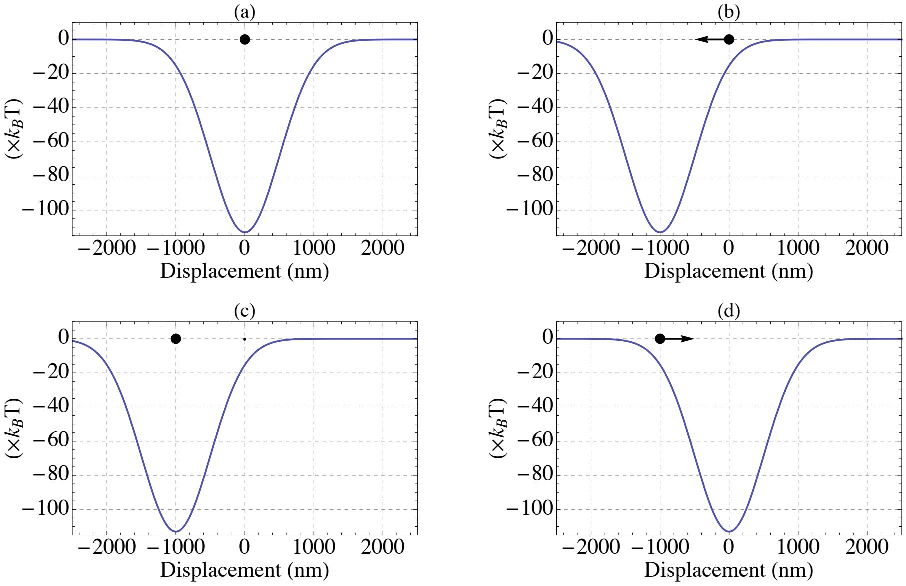


Figure 4.2: A conceptual overview of the step input method in one-dimension, where the particle is initially located at the center of the trap (subfigure (a)). The trap is then rapidly moved away from the center, causing the particle to be suddenly shifted from the trap center (subfigure (b)). The particle then feels a strong restoring force, which drives it back to the center of the trap, shown in subfigure (c). Finally the method is repeated by moving the trap in the opposite direction as shown in subfigure (d).

since we always repeat the experiment several times and average the trajectories, the random thermal motion of the particle cancels out and does not significantly affect the force measurement. Therefore, we can ignore the effect of thermal noise on this measurement and directly use Newton's equation of motion to calculate the optical trapping force.

$$m\,\frac{d^2\vec{r}(t)}{dt^2} = f_T(t) - \gamma\,\frac{d\vec{r}(t)}{dt} \tag{4.1}$$

The outline of the analysis is then given by an example in Figure 4.3. The particle is initially subjected to a step input by moving the center of the trap (this occurs in the left half of Figure 4.3(a), $5\,ms$ after the start of the experiment). The second step (shown in the figure, $10\,ms$ after the start of the experiment) represents a symmetric movement of the trap in the opposite direction in order to measure both halves of the trap and is not discussed here. Next, we measure the trajectory of the nanoparticle as it responds to this step input, shown in Figure 4.3(b). As we see from this figure, the particle moves quickly at first, where it feels the strongest restoring force from the trap, but then slows down as it approaches the center of the trap at it's new position, where the trapping force becomes zero. Using the first time derivative of the position of the particle ($\vec{r}(t)$), we calculate the damping force (see Figure 4.3(c)), given by the second term on the right hand side of Equation 4.1. Similarly, using the second time derivative of the position, we calculate the resultant force (see Figure 4.3(d)), given by the left hand side of Equation 4.1. We then substitute these values into Equation 4.1 to obtain the optical trapping force, $\vec{f_T}(t)$ as a function of time. The optical trapping force obtained is a vector quantity, which includes force components along each axis. We can then calculate the trapping force as a function of displacement, by first inverting $\vec{r}(t)$ and then using it to map the time dependent trapping force into a position dependent force. From Figure 4.3, we observe that the resultant force is several orders of magnitude smaller than the damping force and does not contribute significantly in the force measurement. Therefore, we can assume that as the particle returns to the center of the trap, the restoring force from the optical trap is always opposed only by the damping force, which arises as the particle moves through the liquid. The force measurement described above is valid for continuous functions of time. However, in practice since we always obtain discrete position measurements, we perform the

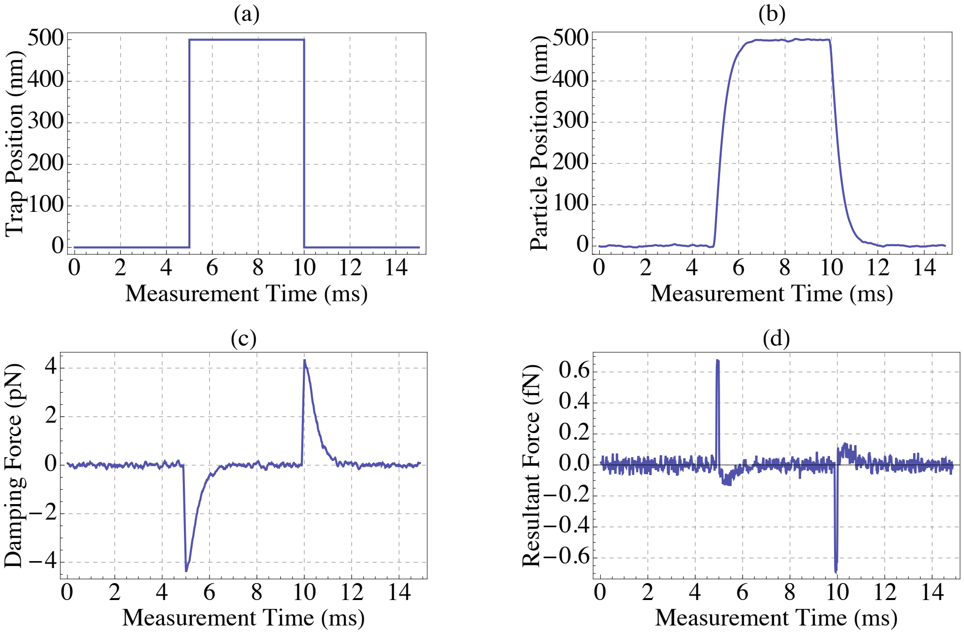


Figure 4.3: Conceptual outline of the step input analysis technique. The position of the particle in response to a rapid step of the optical trap is used to calculate the damping force and the resultant force and eventually used to recover the optical trapping force acting on the particle.

analysis numerically.

The optical trap is inherently three dimensional, however in this chapter we develop the step input method for a two-dimensional system. The measurement is performed in the two-dimensional equatorial plane about the center of the trap, which is also the origin of the coordinate system used in the measurements. The particle's response is measured by stepping the trap back and forth (in order to measure both sides of the trapping potential) along a line, with step size s. The line that describes the movement of the trap is then rotated from 0 to π in discrete steps, described by the azimuthal angle θ. At each angle, we record the trajectory of the particle, in response to a step input and use it to calculate the optical trapping force using the technique described above. However, data obtained either from numerical simulations or digitally measured in the laboratory is discretized and we must therefore calculate the optical trapping force numerically.

The position of the particle, in Cartesian coordinates for two dimensions is given by the vector, $\vec{r}(t) = \{x(t), y(t)\}$, as a function of time. the particle position is recorded at each angle (θ) as it responds to the rapid trap movement of the trap. In order to simplify the notation, it is assumed that the particle position $\vec{r}(t)$ is always relative to the trap center. Symmetric steps of size s in the positive and negative directions are used in order to allow the optical trapping potential to be measured on both sides of the trap center. The restoring force from the optical trap, which drives a particle to an equilibrium position, is calculated by rearranging Newton's equations, given in Equation 4.1, as described previously. We first numerically calculate the trapping force, as a function of time by calculating the velocity and acceleration of the particle at each time-step using Equation 4.2 and then substitute the results from these expressions into Equation 4.1. We must ensure that the position, velocity and acceleration properly align when calculating the trapping force, $\vec{f_T}(t)$ for a given time-step.

$$\begin{aligned} \dot{\vec{r}} &\approx \frac{\vec{r}(t+\delta t) - \vec{r}(t-\delta t)}{2\,\delta t} \\ \ddot{\vec{r}} &\approx \frac{\vec{r}(t+\delta t) - 2\vec{r}(t) + \vec{r}(t-\delta t)}{\delta t^2} \end{aligned} \tag{4.2}$$

We are interested in calculating the force as a function of displacement from the trap center. This transformation can be performed by first calculating the inverse of the position function, $\vec{r}(t)$ and using it to map the time-dependent trapping force to a position dependent force. However, the measured particle position obtained either from simulations or from laboratory experiments is numerical data, uniformly sampled in time with intervals of δt. The trapping force as a function of position, $\vec{F_T}(\vec{r})$ is then constructed by pairing $\vec{r}(t)$ at each time-step with its corresponding trapping force, $\vec{f_T}(t)$ as shown in Equation 4.3. It is important to note that while the trapping force as a function of time, $\vec{f_T}(t)$ is uniformly sampled in time, the force as a function of distance from the trap center, $\vec{F_T}(r)$ is not uniformly sampled in $\vec{r}$.

$$\begin{aligned}\vec{F_T}(\vec{r}) &= \{(\vec{r}(2\delta t),\, \vec{f_T}(2\delta t)),\; (\vec{r}(3\delta t),\, \vec{f_T}(3\delta t)),\\ &\qquad \ldots\;\ldots,\; (\vec{r}((n-2)\delta t),\, \vec{f_T}((n-2)\delta t))\}\end{aligned} \tag{4.3}$$

Finally, the optical trapping potential is constructed by numerically integrating the trapping force $\vec{F_T}(\vec{r})$, using Equation 4.4. The constant U_0 in the equation is used to set the potential energy to zero at infinity.

$$U(\vec{r}) = -\int \vec{F_T}(\vec{r})\, d\vec{r} + U_0 \tag{4.4}$$

The force measurement technique described above is tested using the Brownian dynamics simulations, as well as through laboratory experiments. The method is presented for a two-dimensional system, a constraint easily imposed in simulations. However, the optical trap used in laboratory experiments is inherently three-dimensional. Therefore, after a step input, a particle in the trap will invariably wander out of plane due to Brownian motion. The effects of this particle motion are investigated using simulations in Section 4.3.1 and shown to induce negligible errors to the two-dimensional force measurement and can therefore be ignored. The results from numerical simulations and laboratory experiments for this technique are discussed in Section 4.3.

4.3 Step Input Method: Results

The force measurement technique described in Section 4.2 is validated using data generated from numerical simulations, as well as through laboratory experiments. The results of these measurements are described next.

4.3.1 Simulation Results

A numerical simulation, based on algorithms developed in Chapter 3 is used to test the step input method. A three-dimensional model, which approximates the optical trapping force is included in the numerical simulations. However, the force measurements are performed in the xy-plane, while the trap is kept stationary along the z-axis. Therefore, the particle is free to diffuse outside the plane of the measurement. However, as seen later in this section, the out-of-plane motion of the particle has a minimal impact on the force measurement since the particle remains confined close to the plane of the force measurement. Similar to a measurement made in the laboratory, the output of the simulation is a time-series of the particle's position as it responds to the movement of the trap. Therefore, the optical trapping force is recovered by applying the analysis technique to the position time-series of the particle, as described in the previous section.

The simulations are tested for two distinct particles – 100 nm diameter gold nanoparticles and 640 nm glass microspheres. The nanoparticles are suspended in water at room temperature ($T = 293\,K$), ignoring gravity and buoyancy, which have a negligible effect on the particle. In this section, we first describe the parameters of the optical trapping force model used in the simulations. Next, we present results for the 100 nm diameter gold particles, followed finally by results from the 640 nm glass microspheres.

The optical trapping force ($\vec{F_T}(\vec{r})$) included in the numerical simulations in order to test the step input methods is shown in Equation 4.5. The trapping force when measured in the laboratory includes a gradient component, which is conservative and a scattering component, which directs the particle along the direction of propagation of light. However, in the simulations performed to test the force measurement methods, the scattering components is small and therefore neglected. We can then use a trapping potential, such as a Gaussian potential to derive the trapping force $\vec{F_T}(\vec{r})$, given for a three dimensional system in Equation 4.5. In this equation, the position of the particle within the optical trap is given by the vector $\vec{r} = \{x, y, z\}$ and the trapping force is parameterized by the scalar quantity α, which characterizes the depth of the trapping potential and the trap width $\vec{\sigma} = \{\sigma_x, \sigma_y, \sigma_z\}$. The position of the trap in three-dimensions can be controlled by changing the vector $\vec{\mu} = \{\mu_x, \mu_y, \mu_z\}$.

$$\vec{F_T}(\vec{r}) = \left\{ \begin{array}{c} -\frac{\alpha}{\sigma_x^2}(x-\mu_x) \\ \\ -\frac{\alpha}{\sigma_y^2}(y-\mu_y) \\ \\ -\frac{\alpha}{\sigma_z^2}(z-\mu_z) \end{array} \right\} \times exp\left(-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2} - \frac{(z-\mu_z)^2}{2\sigma_z^2}\right) \qquad (4.5)$$

The parameters selected for each simulation case are summarized in Table

4.1, for particles with diameter, d. In the table, ρ is the density of the particle's material, m is it's mass and γ is the Stoke's constant for spherical particles. The trap parameters, $\vec{\alpha}$ and $\vec{\sigma}$, for the $100\,nm$ gold nanoparticles are calculated using the force model outlined in Appendix D, whereas the trap parameters for the $640\,nm$ glass microsphere are selected from published literature. The results of the simulation experiments for these two cases are discussed next.

Table 4.1: Simulation Parameters for 2D Step Input Method

	$d = 100\,nm$	$d = 640\,nm$
Particle Parameters		
Material	Gold	Glass
ρ (kg/m^3)	19300	2600
m (ag)	10.1	356.9
γ $(\mu g/s)$	0.94	6.04
Trap Parameters		
α (k_BT)	113.6	500
σ_x (nm)	500	462.6
σ_y (nm)	500	400.7
σ_z (nm)	1000	1131.4
Simulation Parameters		
Δt (ns)	0.5	1
δt (μs)	10	10
$T = 293\,K$ for all simulation cases. Number of particles in the ensemble = 100		

Case 1: 100 nm gold nanoparticles

The parameters of simulations for the step input force measurement method for $100\,nm$ diameter gold nanoparticles are summarized in Table 4.1. The parameters for the optical trapping force are selected using the force calculation, under a dipole approximation, described in Appendix D. The dipole approximation, which treats the nanoparticle as a simple dipole oscillator, is only strictly valid for particles that are much smaller than the wavelength of the incident light. However, this force calculation is used only to provide physically valid parameters for the numerical simulations to validate the force measurements. The trapping potential, ignoring polarization effects, is symmetric in the xy-plane and is given in one-dimension by Equation D.8 (shown again in Equation 4.6 below), where α_p is the polarizability of the nanoparticle, P is the incident beam power, w_0 is the half-width of the incident beam waist, r is the radial coordinate, n_m is the refractive index of the surrounding medium and c is the speed of light in vacuum. The polarizability, α_p depends on the effective volume, V', which in turn depends on the skin depth of the particle, δ. The derivation of the force model to arrive at Equation D.8 is described in more detail in Appendix D.

$$U = -\frac{P|\alpha_p|n_m}{2\pi c w_0^2}\, exp\left(-\frac{r^2}{2w_0^2}\right) \tag{4.6}$$

In order to calculate the parameters of the trapping force used in the simulations, we first select a beam with a Gaussian intensity distribution, a wavelength $\lambda = 1050\, nm$ and beam power, $P = 150\, mW$. We then select the half-width of the beam waist, which in this case also defines the size of the trap, using a conservative estimate of the optical trapping beam sizes used in the laboratory to be $\vec{w_0} = \vec{\sigma} = \{500, 500, 1000\}\, nm$ along the x, y and z-axes. The polarizability of the particle, α_p, given by Equation D.3, depends on the effective volume of the particle V', the dielectric constant of the particle, $\epsilon_g = -54 + i\, 5.9$ at a wavelength of $1050\, nm$ and the dielectric constant of the surrounding fluid, $\epsilon_w = n_w^2$, where $n_w = 1.333$ is the refractive index of water. The skin depth of the gold nanoparticle is calculated to be $22.7\,nm$ (Equation D.5) and the effective volume, V' is then calculated to be $3.3 \times 10^5\,nm^3$ (Equation D.4). The trap force parameter α, which is simply the depth of the trapping potential, is then calculated by evaluating Equation 4.6 at $r = 0$ and found to be $113.6\,k_BT$. Since α is scalar quantity, which sets the potential depth in three-dimensions for the optical trap, we use $\alpha = 113.6\,k_BT$ along all three axes. Based on results from Chapter 3, the simulation time-step, Δt is chosen to be $0.047 \times \frac{m}{\gamma}$ or $0.5\,ns$ for the $100\,nm$ diameter nanoparticle in order to optimize the simulation performance. Finally, the output of the simulation is sampled uniformly $(\frac{1}{\delta t})$ at $100\,kHz$.

The simulation starts with the particle confined close to the center of the optical trap. The trap is then moved a finite distance $s = 700\,nm$ along the positive x-axis, with $\theta = 0$ and the trajectory of the particle as it returns to the trap center is recorded. A second step of equal size but in the opposite direction along the x-axis allows both sides of the trapping potential to be measured and brings the trap back to the starting point. The measurement is repeated by rotating the direction of trap motion by a finite angular step $d\theta$ and a total of nine separate force measurements are made over the range 0 to π (see Table 4.2 below) in order to sample the optical trapping force in the xy-plane about the center of the optical trap. Two representative particle trajectories for trap steps at $\theta = 30\,deg$ and $\theta = 150\,deg$ are shown in Figure 4.4. 100 trajectories of the particle's response to the step input are recorded, separated into groups of 10 and averaged, before being shown in Figure 4.4. The error bars in the plots show the standard deviation of the particle's position.

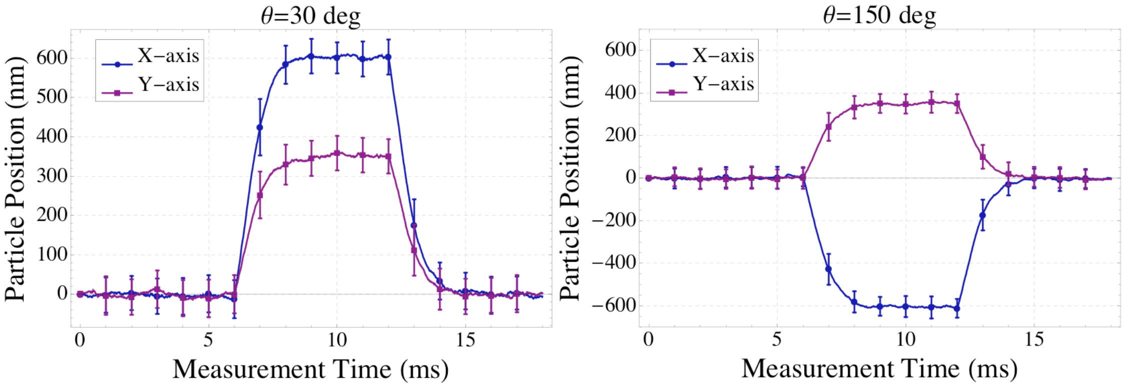


Figure 4.4: Simulation data of a $100\,nm$ diameter gold nanoparticle's position in the xy-plane as it responds to a step input along a line that makes an angle θ with the positive x-axis. The figure shows two representative cases for $\theta = 30\,deg$ and $\theta = 150\,deg$.

For $\theta = 30\,deg$, the size of the $700\,nm$ trap step projected onto the x and y-axes is $606\,nm$ and $350\,nm$. In the plot on the left side of Figure 4.4, for the $30\,deg$ case, the trap first moves at $t = 6\,ms$. However, while the trap moves instantly to its final location, the particle takes a finite time to respond to the motion of the trap and therefore, the plot shows a sharp change in the particle position at $t = 6\,ms$. The position curve, along both axes, then appears slightly concave as the particle accelerates with increasing restoring force from the trap and then assumes a much more gradual increase as it approaches the trap center at approximately $t = 9\,ms$, where the restoring force from the trap approaches zero. From the left half of Figure 4.4, we see that the particle takes approximately $3\,ms$ to respond to the projection of the step input to the trap along the x-axis and approximately $4\,ms$ along the y-axis. The right half of Figure 4.4 shows the particle trajectory in two-dimensions for the 150 deg case. However, in this case, the direction of the motion of the trap along the x-axis proceeds in the opposite direction as the previous case.

The trapping force is then recovered by first calculating the first and second time derivatives of each trajectory numerically using Equation 4.2 to obtain the particle's velocity and acceleration as a function of time. By substituting the values obtained into Equation 4.1, we then obtain the trapping force $\vec{f_T}(t)$ and finally the trapping force as a function of the particle displacement ($\vec{F_T}(\vec{r})$) using the procedure shown in Equation 4.3. The resulting optical trapping force in two-dimensions is a vector and therefore calculating the optical trapping force for each angle θ results in a vector field, which is shown on the left hand side of Figure 4.5. In the figure, trapping force vectors calculated for trap motion along lines for discrete angles are shown in the foreground, color coded by magnitude and the light gray field in the background is obtained by interpolating the calculated force vectors to obtain the trapping force at discrete locations on a two-dimensional grid. Finally, the optical trapping potential is calculated by numerically integrating the trapping force in the two-dimensional space and is shown in the right half of Figure 4.5.

The trapping potential used in the simulations is purely conservative, however optical trapping forces in the laboratory include non conservative scattering forces, which are not accounted for in this analysis or in the experimental results in the next section. We see from the left side of Figure 4.5 that the optical trapping force is very small close to the center of the force field and increases radially away from the center, before decreasing near the edges. Since the input trap model in this case

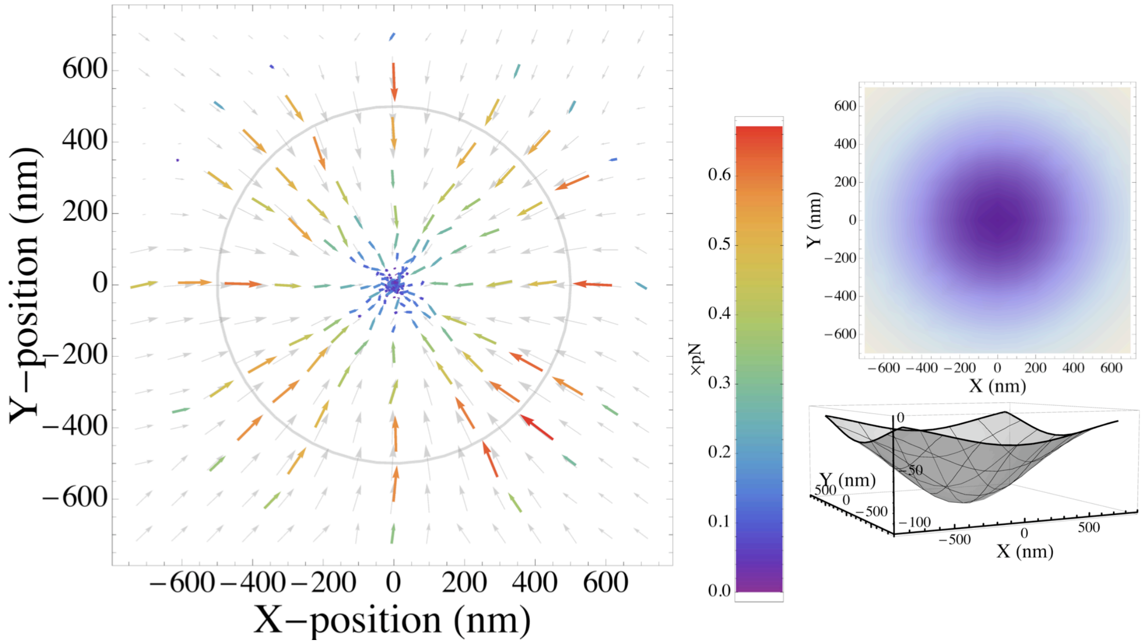


Figure 4.5: The optical trapping force, calculated for a 100 *nm* gold nanoparticle, is shown in two-dimensions on the left side for each θ. The solid line represents the width of the trap. The right side of the figure shows two representations of the trapping potential obtained by integrating the force field.

is circularly symmetric, the trapping force vectors are oriented roughly along the direction of the trap step. The noise in the magnitude and direction of the trapping force is due to the thermal noise experienced by the particle as it returns to it's equilibrium location in the trap. However, despite the noise in the force field, we are able to recover the parameters of the optical trap with good accuracy as seen below in Table 4.2.

Table 4.2 quantifies the results of the step input force measurement results for the 100 *nm* gold particles. The table shows the results from a least squares fit of the data to a function of the form shown in Equation 4.5, with the axial position of the particle, $z = 0$. This essentially results in a two-component force vector, parameterized by the potential depth α and the width of the trap along the x and y-axes, σ_x and σ_y. The table shows the recovered fit parameter together with the repeatability (standard error) of these parameters and the percent error, relative to the nominal values input to the simulation, for each azimuthal angle θ. As described earlier, the analysis is performed by first dividing the simulations into 10 groups of 10 particles each and then calculating the trap parameters by performing the trapping force analysis separately for each group. The columns in Table 4.2 therefore display the mean value of each parameter, along with the standard error. The analysis must however also consider three special cases when θ is either 0, $\pi/2$ or π. In each case, the particle returns to the center of the trap along either the x or the y-axis and undergoes very little motion in the axis orthogonal to it's direction of motion. For example, in the case when $\theta = 0$, the particle travels along the x-axis and undergoes very little motion along the y-axis. Consequently, it experiences very little restoring force from the trap along the y-axis. Therefore, for $\theta = 0$, we are unable to obtain a good estimate of the the trap parameter σ_y. Similarly, for $\theta = \pi/2$ we only calculate the trap width along the y-axis and for $\theta = \pi$, along the x-axis. We observe that fit parameter, α, which quantifies the potential depth shows agreement to within 11 % of the value input to the simulation, while the parameters for the trap width are within approximately 10 % of the input value.

Effect of the Out of Plane Motion of a Trapped Particle on the 2D Force Measurement The numerical simulations employed in verifying the step input method calculates and records the particle's trajectory along all three axes. Therefore when the trapping force is measured in the equatorial xy-plane of the

Table 4.2: Simulation Results for 100nm gold nanoparticles: 2D Force Measurements with a 3D Gaussian Trapping Potential

Angle rad	$\alpha \pm SE$ $(Pct.Err)$ k_BT (%)	$\sigma_x \pm SE$ $(Pct.Err)$ nm (%)	$\sigma_y \pm SE(Pct.Err)$ nm (%)
0	116 ± 4 (1.7)	524 ± 12 (4.8)	–
$\pi/6$	102 ± 6 (−9.9)	518 ± 12 (3.5)	538 ± 24 (7.5)
$\pi/4$	101 ± 4 (−10.7)	523 ± 14 (4.6)	530 ± 14 (6.0)
$\pi/3$	103 ± 3 (−9.4)	505 ± 20 (1.0)	541 ± 11 (8.2)
$\pi/2$	108 ± 4 (−5.0)	–	498 ± 15 (−0.5)
$2\pi/3$	109 ± 3 (−3.6)	549 ± 18 (9.8)	526 ± 14 (5.3)
$3\pi/4$	110 ± 6 (−3.6)	518 ± 15 (3.6)	508 ± 16 (1.6)
$5\pi/6$	102 ± 4 (−10.4)	507 ± 11 (1.5)	550 ± 15 (10.1)
π	109 ± 2 (−4.4)	500 ± 7 (0.0)	–

$\alpha = 113.64\, k_BT$ for all cases.
$\sigma_x = \sigma_y = 500\, nm$ for all cases.

optical trap, the particle is free to diffuse axially (z-axis), out of the measurement plane due to Brownian motion. As the particle diffuses farther away from the center of the trap, it starts to feel an axial restoring force, which forces it back towards the center, however when the mean axial displacement is sufficiently large, ignoring it in the analysis of the trapping force may sometimes lead to errors. We show that for the nanoparticle considered in this experiment, the axial motion outside the xy-plane in fact does not add significant errors to the force measurement.

The distribution of the particle's position along the z-axis, calculated from the output of the simulation, has a mean at $0 \pm 4\, nm$ and a standard deviation of $95 \pm 4\, nm$. The diffusion length of a freely diffusing $100\, nm$ diameter gold nanoparticle over $3\, ms$, the average response time of the microsphere to a $700\, nm$, is calculated using the relation $\langle d_{free} \rangle^2 = 2 \frac{K_B T}{\gamma}\, t$ and found to be $160\, nm$. Therefore since the particle is confined close to the equatorial plane, it may experience little restoring force from the trap.

However, movement of the particle outside the plane of measurement may in some cases significantly impact the parameters of the optical trap recovered from the analysis of the measured data. In the analysis, the parameters of the trapping force are recovered by fitting the calculated optical trapping force to the model shown in Equation 4.5, in the equatorial plane with $z = 0$. However, as the particle drifts outside the measurement plane, the z-component of the trapping force may no longer be neglected and may impact the accuracy of the force measurement. In order to test the effect of the out-of-plane motion of the particle, we fit the calculated force vectors to the model in Equation 4.5 for different values of z above the equatorial plane. The effect of the particle's motion, outside the measurement plane, on the trap depth parameter α, in one case ($\theta = 0$) is shown in Figure 4.6, which shows the relative error in α as a function of z. We see from the figure that the change in the relative error in α is small – only 0.5% for a z-displacement of $95\,nm$ (the measured standard deviation of the microsphere's out of plane motion). Therefore this result indicates that ignoring the out of plane movement of the particle when performing the step input force method in the equatorial plane has a minimal impact on the results.

Case 2: 640 nm glass microspheres

The step input method was also tested for glass microspheres, $640\,nm$ in diameter, using the system parameters that are summarized in Table 4.1. The parameters of the trapping force $\vec{F_T}(\vec{r})$ for this case were obtained using experimental stiffness

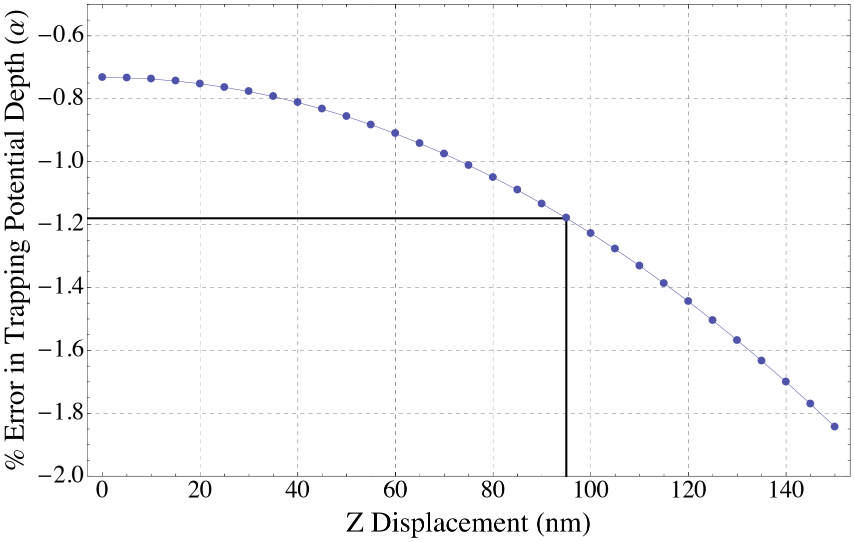


Figure 4.6: Change in relative error of the measured trap depth parameter α as a function of the particle's displacement along the z-axis.

measurements, $\vec{\kappa} = \{9.45,\ 12.6,\ 1.58\}\ pN/\mu m$, published in the literature, for an optical trap with incident light polarized in the x-direction [132]. The shape of the three-dimensional force, derived from a Gaussian potential is determined by the well-depth parameter of the potential, α and the trap width $\vec{\sigma} = \{\sigma_x, \sigma_y, \sigma_z\}$. One of these parameters must be selected beforehand in order to maintain a stiffness value of $\vec{\kappa}$ close to the center of the trap. Therefore, we assume a well-depth, $\alpha = 500\, K_B T$ and then calculate the width of the trap along the three-axes using Equation 4.7. This equation is obtained by re-arranging the first order term from the power series expansion of the trapping force from Equation 4.5 about the origin. For the stiffness $\vec{\kappa}$ and trap depth α, the trap width is then found to be $\vec{\sigma} = \{462.64, 400.66, 1131.44\}\ nm$.

$$\vec{\sigma} = \sqrt{\frac{\alpha}{\vec{\kappa}}} \tag{4.7}$$

The trap half-width ($\vec{\sigma}$) in the simulations is defined as the distance of one standard deviation of the Gaussian potential. However, the trap width in an optical trapping instrument is less clearly defined and is found to change as a function of the particle and beam size. Furthermore, the width of the beam is not necessarily the same along the x and y-axis due to the polarization of the incident light, which causes the trap to be stronger along the axis orthogonal to the polarization. The trap position, controlled by the parameter μ is assumed to be 0 at the end of the trap motion. The simulation time-step, Δt is chosen to be $0.017 \times \frac{m}{\gamma}$ or $1\,ns$ for the $640\,nm$ diameter microsphere in order to optimize the simulation performance. Finally, the output of the simulation is sampled uniformly $(\frac{1}{\delta t})$ at $100\,kHz$.

The simulations proceed exactly as described previously for the $100\,nm$ gold nanoparticles. Two representative particle trajectories for trap steps at $\theta = 30\,deg$ and $\theta = 150\,deg$ are shown in Figure 4.7. 100 trajectories of the particle's response to the step input are recorded, separated into groups of 10 and averaged, before being shown in Figure 4.7 and the error bars in the plots show the standard deviation of the particle's position. As expected, for $\theta = 150\,deg$ (shown on the right side of the figure), the particle moves in the opposite direction along the x-axis, when compared with the $\theta = 30\,deg$ plot. As seen from the figure, the trap is first moved at $t = 6\,ms$ and the particle takes approximately $4\,ms$ to return to the center of the trap. A second step at $t = 10\,ms$ is performed to sample both sides of the optical trap.

The trapping force is then recovered exactly as before using Equation 4.1, Equation 4.2 and Equation 4.3. The resulting vector field is shown in the left hand

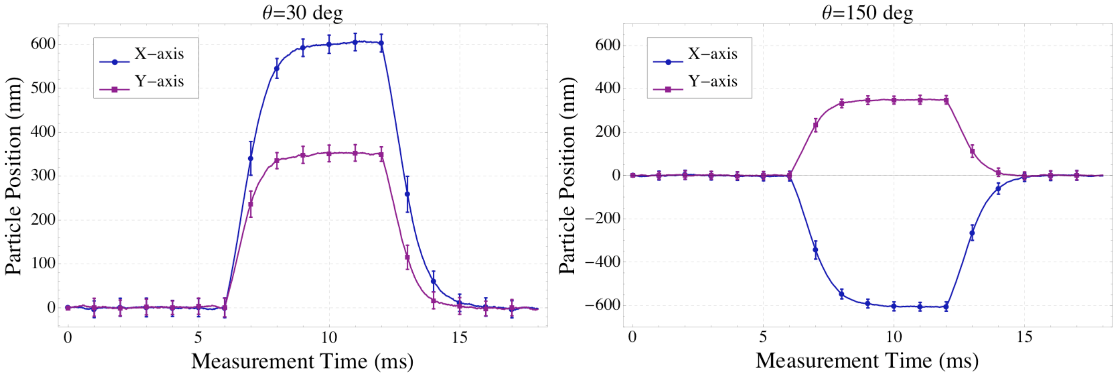


Figure 4.7: Simulation data of a $640\,nm$ diameter glass microsphere's position in the xy-plane as it responds to a step input along a line that makes an angle θ with the positive x-axis. The figure shows two representative cases for $\theta = 30\,deg$ and $\theta = 150\,deg$.

side of Figure 4.8, for each measurement angle θ in the xy-plane. The color coded force vectors in the foreground are calculated for each simulated cases, while the light gray field in the background is obtained by interpolating the calculated force vectors to obtain a new field that uniformly samples the two-dimensional grid. The asymmetries in the trapping force due to polarization effects can also be seen in this figure. Unlike, Figure 4.5 where the vectors are oriented approximately along the line which represents the movement of the trap, in this case the force vectors appear slightly curved towards the y-axis, which is orthogonal to the axis of polarization. This effect can also be seen when the force field is integrated to obtain the potential energy as a function of the displacement of the particle from the center of the trap, shown in the right side of Figure 4.8.

Asymmetries in the optical trap, discussed above can be seen very clearly by plotting the trace of the particle's trajectory in response to a step input along a straight line (on either side of the origin). One such example is shown in Figure

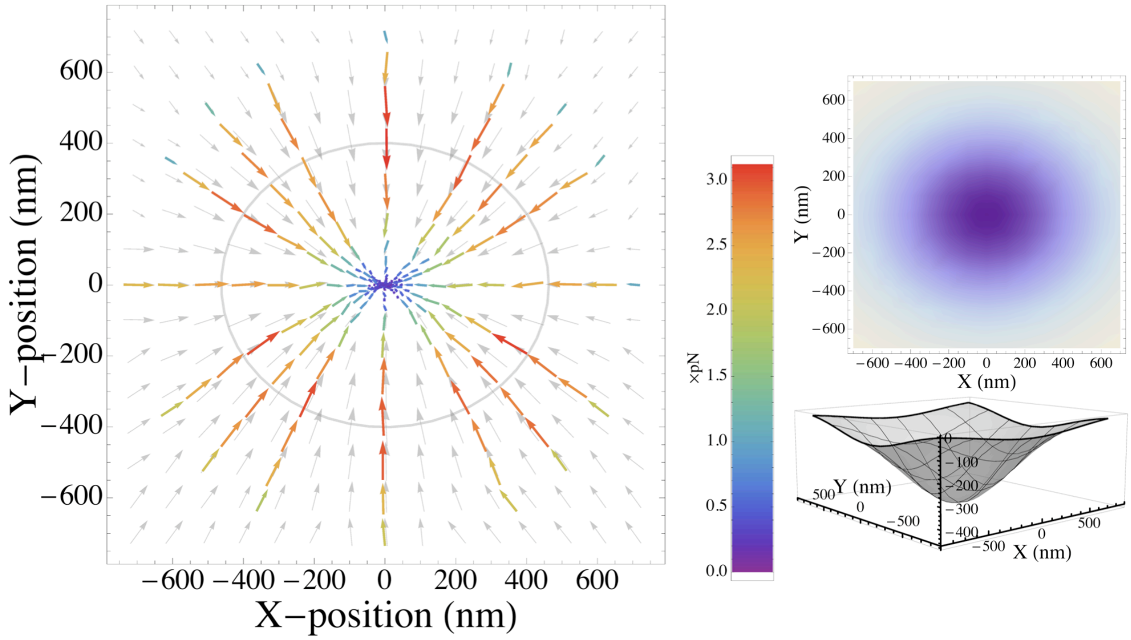


Figure 4.8: The optical trapping force and the trapping potential as a function of distance from the trap center are shown for a step size of 700 nm, while the inset in the bottom half of the figure shows a 3D-representation of the trapping potential. The solid line represents the width of the trap.

4.9 for the $\theta = \pi/4$ direction, where the red lines indicate the path followed by the particle in 100 independent experiments and the black line is the mean trajectory of the particle. The projection of the particle's motion in the xz-plane is found to be a straight line along the x-axis as expected, while the projection of the particle's trajectory in the yz-plane mirrors that in the xz-plane and is not shown in the figure. In the xy-plane, the particle follows a slightly curved path as it returns to it's equilibrium position close to the center of the trap due to the asymmetry in the trap geometry arising from polarization effects, which causes the particle to be attracted more strongly in the x-direction. The curvature of the trajectory is a function of the asymmetry in the trap, with a straight line expected for a circularly symmetric trap, as seen for the 100 nm diameter gold nanoparticle.

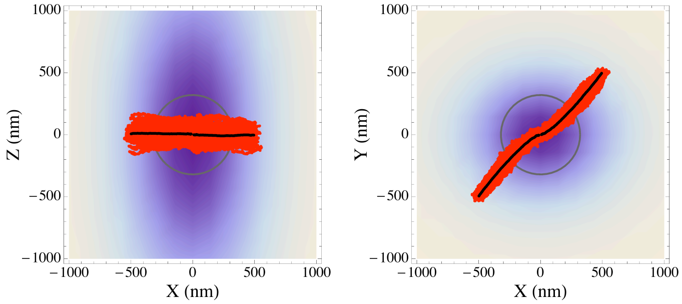


Figure 4.9: Trace of a particle's trajectory in response to a step input along a line at θ=45 *deg*. The red lines show the individual particle trajectories, while the black line is the mean path followed by the particle. The trap is shown in the background of the figure, while the gray circle represents the particle.

Table 4.3 summarizes the simulation results and estimates the trap parameters α, σ_x and σ_y, by fitting the trapping force field to the model shown in Equation 4.5 using a least squares estimator. As in the previous simulation results, the fit is performed at the equatorial plane by setting $z = 0$ in Equation 4.5. The maximum error for the well depth α was found to be 5.5 %, while the maximum error for the trap widths $\vec{\sigma}$ was $\{6.8\,\%, 1.8\,\%\}$. Like in the previous results, only one trap width parameter, either σ_x or σ_y is calculated in three cases, $\theta = 0$, $\theta = \pi/2$ and $\theta = \pi$, in Table 4.3. As expected, the effect of the particle's out of plane motion during the measurement is even less pronounced for the 640 *nm* glass particle. The distribution of the particle's position along the z-axis has a mean of $0 \pm 3\,nm$ and has a standard deviation of $49 \pm 1\,nm$. The effect of the out of plane motion on the trapping parameters, calculated exactly as before for Figure 4.6, is found to induce an error of only 0.1%.

Table 4.3: Simulation Results for 2D Force Measurements with a 3D Gaussian Trapping Potential, 640nm glass microspheres

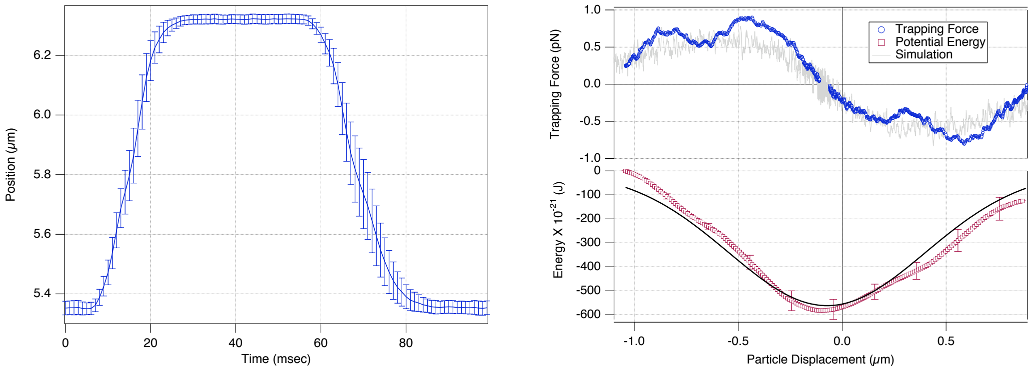
Angle rad	$\alpha \pm SE\ (Pct.Rel.Err)$ $k_BT\,(\%)$	$\sigma_x \pm SE\ (Pct.Rel.Err)$ $nm\ (\%)$	$\sigma_y \pm SE(Pct.Rel.Err)$ $nm\ (\%)$
0	$494 \pm 4\ (-1.3)$	$458 \pm 4\ (-1.0)$	–
$\pi/6$	$472 \pm 9\ (-5.5)$	$472 \pm 4\ (2.0)$	$398 \pm 8\ (-0.6)$
$\pi/4$	$511 \pm 16\ (2.2)$	$494 \pm 8\ (6.8)$	$395 \pm 6\ (-1.3)$
$\pi/3$	$479 \pm 16\ (-4.1)$	$486 \pm 10\ (5.1)$	$395 \pm 4\ (-1.3)$
$\pi/2$	$493 \pm 2\ (-1.4)$	–	$406 \pm 2\ (1.3)$
$2\pi/3$	$479 \pm 12\ (-4.2)$	$488 \pm 9\ (5.5)$	$405 \pm 4\ (1.2)$
$3\pi/4$	$478 \pm 12\ (-4.3)$	$470 \pm 8\ (1.7)$	$404 \pm 5\ (0.9)$
$5\pi/6$	$479 \pm 10\ (-4.3)$	$465 \pm 5\ (0.5)$	$393 \pm 7\ (-1.8)$
π	$499 \pm 4\ (-0.1)$	$464 \pm 3\ (0.2)$	–

$\alpha = 500\,k_BT$ for all cases.

$\sigma_x = 462.64\,nm$ and $\sigma_y = 400.66\,nm$ for all cases.

4.3.2 Laboratory Results

4.3.2.1 $1\,\mu m$ Glass Microspheres



(a) (b)

Figure 4.10: (a) Calibrated response of a particle to a step input of 980 nm. The plot is the result of averaging over 100 data samples and the error bars show one standard deviation in the particle's position at a given time. (b) The optical trapping force and the trapping potential are shown as a function of the particles distance from trap center. The thick black line shows a fit to the data assuming a Gaussian trap.

The step response method was experimentally verified in the laboratory using the optical tweezers instrument, described in Appendix A, in one-dimension using a dilute suspension of $1\,\mu m$ glass beads (Duke Scientific, P/N: 8100), in de-ionized water. A sample cell, approximately $30\,\mu m$ thick, was prepared using a microscope slide and cover glass and sealed on all sides using vacuum grease. The glass bead was always trapped and lifted approximately $15\,\mu m$ above the bottom surface (Z-axis) to reduce surface effects in the viscous drag, as discussed in Chapter 2. Furthermore, between each calibration step and also between individual force measurements, the particle is relocated by several micrometers (in the xy-plane by moving the sample stage) to reduce the effect of local heating due to the illumination source and minimize changes in the fluid viscosity within the cell.

Once the calibration steps outlined in Appendix B are completed, the trapped glass bead is scanned using a $10\,Hz$ square wave input, with a peak-to-peak amplitude of $2.5\,V$ or $980\,nm$, which represents the distance of the particle from the center of the trap. The position response of the particle to the square wave input is recorded using the QPD with a sampling frequency of $50\,kHz$. The measurement is repeated 100 times and then averaged. The resulting position data is smoothed using a 11 point moving average to further reduce high frequency measurement noise and the QPD calibration curve from Figure B.2(b) is then used to calculate the position of the particle as a function of time as shown in Figure 4.10(a). We observe from Figure 4.10(a) that the onset of the particle's response to the step input near $t = 10\,ms$ is more gradual than in the simulation results. This is expected since the optical trap used in the experiments is much weaker than previous examples. Next, the same analysis technique applied to the simulation data is used to recover the optical trapping force shown in the top half of Figure 4.10(b). From this plot, we see that the maximum trapping force is approximately $0.6\,pN$ and occurs at a distance of $\pm 500\,nm$ from the trap center. The light gray plot in the top half of the figure are the force data calculated using a numerical simulation, using the trap parameters measured in the laboratory. The trapping force is then integrated to give the trapping potential, seen in the bottom half of Figure 4.10(b) and the solid line in the plot is from a least squares fit to the potential energy data. Figure 4.10(b) shows that this method yields the trapping force well beyond the region close to the trap center. Asymmetry in the optical trap arising probably due to misalignments in the optics train can also be seen in Figure 4.10(b). It is also observed that the minimum of the trapping potential does not coincide exactly with the origin of the coordinate system suggesting that the trapping and detection beams are not perfectly coincident. This is caused most likely due to drift in the detection system over the course of the experiment. Furthermore, as discussed previously, we perform a one-dimensional measurement using an inherently three-dimensional optical tweezers instrument, which may give rise to small measurement errors due to off-axis particle motion from Brownian motion. However, this error is expected to be small, based on earlier simulation results.

4.3.2.2 $350\,nm$ Glass Nanoparticles

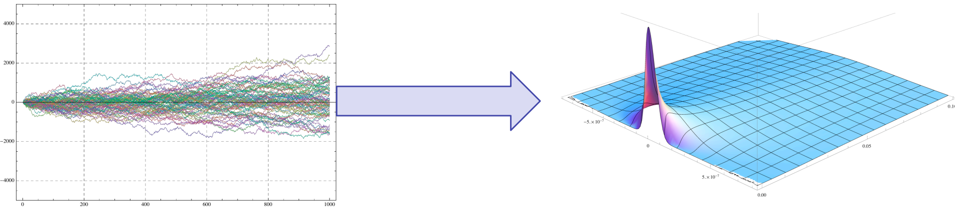
We have also used the step input method to estimate the maximum force point of the optical trap when designing the closed loop controller described in Section

5.4. We use the step input method to measure the trapping force on a $350\,nm$ glass nanoparticle. For this measurement, the trapped nanoparticle is scanned using a $1\,Hz$ square wave input, with a peak-to-peak amplitude of $350\,nm$, which represents the distance of the particle from the center of the trap. The position response of the particle to the square wave input is recorded using the QPD with a sampling frequency of $40\,kHz$. The measurement is repeated 150 times and then averaged. Finally, we estimate the location of the maximum force point to be $236 \pm 18\,nm$.

4.4 Stochastic Noise Method

A nanoparticle undergoes a constant diffusive motion, away from the center of the trap. This diffusive motion is opposed by the restoring force generated by the optical trap, which acts towards the center, in order to keep the particle localized. For small particles, that are weakly bound to the optical trap, the restoring force provided by the trap is insufficient and the particle eventually escapes. In the stochastic noise force measurement method, the particle acts like a probe and samples the optical trap, effectively encoding information about the trap within it's position trajectories. The trapping force can then be recovered from the position trajectories of the particle inside an optical trap. Since the nanoparticle acts as the probe which samples the optical trap, the estimate of the optical trapping force improves with the time the particle spends inside the trap. However, we can also improve the force measurement technique by recording the position of an ensemble of particles in the optical trap and then combining this data to estimate the trapping force.

Position trajectories from an ensemble of particles can be naturally represented using a probability density function (PDF), $W(x;t)$, which in one-dimension, is a function of the position x, but also parameterized with time (t). The technique to construct the PDF from position trajectories, measured either in the laboratory or calculated using simulations, is shown in Figure 4.11. The position trajectories of individual particles (shown in the left half of the figure), as a function of time, are first re-arranged to group the particles by the measurement time. The PDF is then constructed by calculating the histogram of the particle positions, normalized by the bin size, at each measurement time. The resulting PDF is shown in the right half of the figure. As seen from the figure, the nanoparticles are initially located very close to the center of the trap, represented by the sharp peak in the PDF and then eventually escape the trap over time.



Individual Particle Trajectories as a Function of Time

Probability Density Function (pdf): function of position and parameterized by time

Figure 4.11: The probability density function (PDF) can be constructed from the trajectories, as a function of time, of several non-interacting nanoparticles in an optical trap. The trajectories are first transposed and then the PDF is constructed by calculating the histogram, normalized by the bin width, at each time-step.

The evolution of the measured PDF in time, t can then be calculated using a Fokker-Planck equation, which is a form of the diffusion equation. A Fokker-

Planck equation for systems with large coefficients of friction is shown in Equation 4.8 [131]. The right hand side of the Fokker-Planck equation is expressed in terms of the probability current, $J = D\left(\frac{1}{k_BT}W(x;t)\ F(x) - \frac{\partial}{\partial x}W(x;t)\right)$. In this equation, $D = \frac{k_BT}{\gamma}$ is the diffusion constant, $\gamma = 6\pi\eta a$ is the Stokes drag coefficient for a spherical particle with radius a in a fluid with viscosity η, k_B is Bolztmann's constant and T is the temperature of the fluid. The force from the optical trap can be given by $F(x) = -\frac{\partial}{\partial x}U(x)$, where $U(x)$ is an optical trapping potential and therefore $F(x)$ would be a conservative force. Non-conservative components of the trapping force, such as the force from scattering are ignored when calculating trapping potentials.

$$\frac{\partial}{\partial t}W(x;t) = -\frac{\partial}{\partial x}\ J = -\frac{\partial}{\partial x}D\left(\frac{1}{k_bT}\left[W(x;t)\ F(x)\right] - \frac{\partial}{\partial x}W(x;t)\right) \tag{4.8}$$

As discussed previously, the probability density function ($W(x;t)$) can be constructed by combining serveral particle trajectories, either generated from numerical simulations or measured in the laboratory. Once $W(x;t)$ is known, Equation 4.8 can be rearranged to obtain a first order differential equation for the trapping force, $F(x)$. We can then derive the expression for the optical trapping force $F(x)$, shown in Equation 4.9, by integrating both sides over position and rearranging the terms. The constant of integration is set to zero by defining $x = 0$ to be the minimum of the trapping potential.

$$F(x) = \frac{k_BT\frac{\partial}{\partial x}W(x;t) - \frac{k_BT}{D}\int\frac{\partial}{\partial t}W(x;t)dx}{W(x;t)} \tag{4.9}$$

We are interested in calculating the trapping force at long times, where the nanoparticle has sufficiently explored the optical trap and therefore improving the measurement of the trapping force. At sufficiently long times, the PDF does not change appreciably with time and therefore the second term in the numerator of Equation 4.9 can be ignored, resulting in the simplified expression for the trapping force shown in Equation 4.10. Moreover, this expression is consistent with the Einstein-Smoluchowski relationship.

$$F(x) = k_B T \; \frac{\frac{\partial}{\partial x} W(x;t)}{W(x;t)} \tag{4.10}$$

Since the PDF is calculated from discrete measurements of the particle position, either from simulation data or from laboratory experiments, the derivative of the PDF with respect to position in Equation 4.10 is calculated numerically. This is performed using the expression in Equation 4.11, for a probability density function with a bin size, δx. The expression for the numerical derivative is obtained exactly as shown earlier in Equation 3.4. The resulting function is then substituted back into Equation 4.10 to obtain the optical trapping force as a function of the position, x. Based on the form of the numerical derivative used in Equation 4.11, we observe that the resulting derivative will have two points less than the number of points used to tabulate the function $W(x;t)$. Therefore it is important to keep the results of the numerical derivative aligned with $W(x;t)$.

$$\frac{\partial}{\partial x} W(x;t) \approx \frac{W(x+\delta x;t) \; - \; W(x-\delta x;t)}{2\,\delta x} \tag{4.11}$$

4.5 Stochastic Noise Method: Results

Results of the stochastic noise force measurement technique, outlined in Section 4.4, are described next. Section 4.5.1 describes the results of numerical simulation used to validate the stochastic noise method. Preliminary experiments are also performed to demonstrate this technique in the laboratory and are discussed in Section 4.5.2.

4.5.1 Simulation Results

We demonstrate the stochastic noise force measurement method using data calculated from a one-dimensional Brownian dynamics simulation (described in Chapter 3), for an ensemble of independent non-interacting particles initially located at the center of an optical trap. Simulations are performed for gold and glass nanoparticles with diameters of $100\,nm$, $150\,nm$, $200\,nm$ and $300\,nm$ using the trapping force ($F(x) = -\frac{\partial}{\partial x}U(x) = -\frac{\alpha\,x}{\sigma^2}e^{-\frac{x^2}{2\sigma^2}}$) derived from a Gaussian potential with depth (α) and potential half-width, σ. The parameters for each numerical simulation are shown in Table 4.4. The trapping force parameters for the $100\,nm$ gold nanoparticle are chosen using the physical force calculation outlined in Appendix D for an incident beam power of $1\,mW$. The half width of the incident beam (σ) is set to $500\,nm$, which represents a typical optical trapping beam and the measurement is performed for a total of 7 seconds to allow the particle sufficient time to sample the optical trap. The trapping parameters for the glass nanoparticles are set empirically, by first choosing the potential depth of the $200\,nm$ particle to be $\frac{1}{2}k_BT$ and then scaling the trap depths for the $150\,nm$ and $300\,nm$ particles proportional to their volumes [44]. While this approximation is only rigorously valid for very small particles, it provides sufficient accuracy for our current measurements. The simulation runs at a time-step of $50\,ps$ or $100\,ps$, depending on particle size and material, and the output is sampled uniformly at $1\,\mu s$ to collect a time-series of position for each particle. The simulation is repeated for 600 000 particles and we perform the analysis by grouping the data into groups of 50 000, resulting in 12 independent force measurements for each case.

Table 4.4: Stochastic Noise Method: Simulation Parameters

Dia. (d) nm	Mat.	Pot. Depth (α) (k_BT)	Pot. Width (σ) (nm)	Sim. Time-step (ps)	Meas. Time (s)
100	Gold	0.758	500	100	7.0
150	Glass	0.211	400	50	2.0
200	Glass	0.500	400	50	2.0
200	Glass	0.500	400	50	7.0
300	Glass	1.687	400	100	2.0
Fluid medium: water at $T = 293\,K$					

In order to test the stochastic noise technique, the simulation must be initialized with a distribution of particle positions and then run up to a time when the particles in the ensemble completely fill the trap and have substantially leaked out. The system then approaches a quasi-steady state, which for a weak optical trap can be several seconds after the start of the experiment. However, running the simulation with a large ensemble of non-interacting nanoparticles (required to obtain an accurate force measurement) for long times can be very computationally intensive. Therefore, in order to reduce the amount of computation time required to complete the simulation, we first calculate the quasi-steady state PDF by numerically solving the Fokker-Planck equation, given in Equation 4.8 and use this distribution to initialize the simulation. Using the Fokker-Planck equation, to initialize the simulation, in this manner does not affect the outcome of the calculation. The simulations allow us to quantify the effect of stochastic noise from individual particle trajectories, on the measurement – an important effect that must be taken into account to extend this technique to the laboratory.

The output of the simulation trajectories are then used to construct the PDF of the particle ensemble, as discussed earlier. At the start of each simulation, the particles in the ensemble are located at the center of the optical trap and the PDF is a sharp peak. As the simulation progresses, we find that the ensemble of particles fill the trap in a very short time, typically a few milliseconds, and then slowly leaks outside the trap over a longer time-scale. A snapshot of the PDF, at $t = 7.0\,s$, is shown in Figure 4.12 for a $100\,nm$ diameter gold particle, calculated over a spatial range of $\pm 10\sigma$ using a bin width of $125\,nm$. From the plot, we see a central peak close to zero, which represents the particle distribution inside the optical trap (shown in the bottom half of the figure), with a standard deviation of approximately $500\,nm$ and long tails in the region where the particles have left the trap and undergo nearly free diffusive motion.

The PDF calculated using the trajectories output by the simulation are then used to recover the optical trapping force, as a function of the particle's displacement, using Equation 4.10. The trapping force and standard error for the two glass and one gold nanoparticle are shown in Figure 4.13. The solid lines in the figure are the input trapping force model for each case. We see from the figure, that we are able to obtain the trapping force beyond the inflection point of the trapping

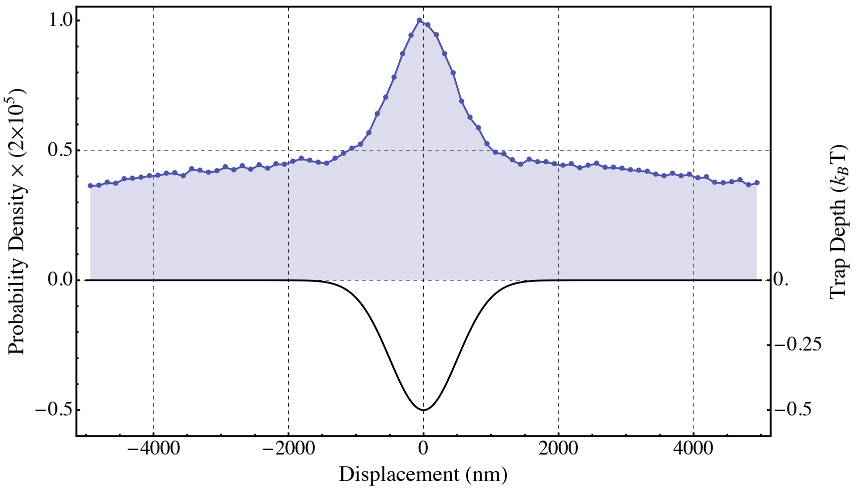


Figure 4.12: The probability density function of a 100 nm gold nanoparticle in a Gaussian optical trap, 7 s after the start of the measurement.

potential for all three particles.

The trapping force, as a function of particle displacement, is calculated for all the simulation cases shown in Table 4.4. The optical trapping force profile obtained from the simulations is then evaluated in terms of two trap parameters – the well depth (α) and the beam size (σ). For each simulation case, the numerical force profiles, calculated from the simulations are fit to a force function of the form $-\frac{\alpha x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$ to recover the trap parameters α and σ, where x is the displacement of the particle. The trapping force is calculated in each case to a distance 700 nm from the trap center and the results are then shown in Table 4.5, which summarizes the relative errors and reproducibility of the method, obtained by calculating the standard deviation of each fit parameter and normalizing that value by the square root of the number of observations, for each simulation case.

From Table 4.5, we see that for glass particles and a measurement time of 2 s,

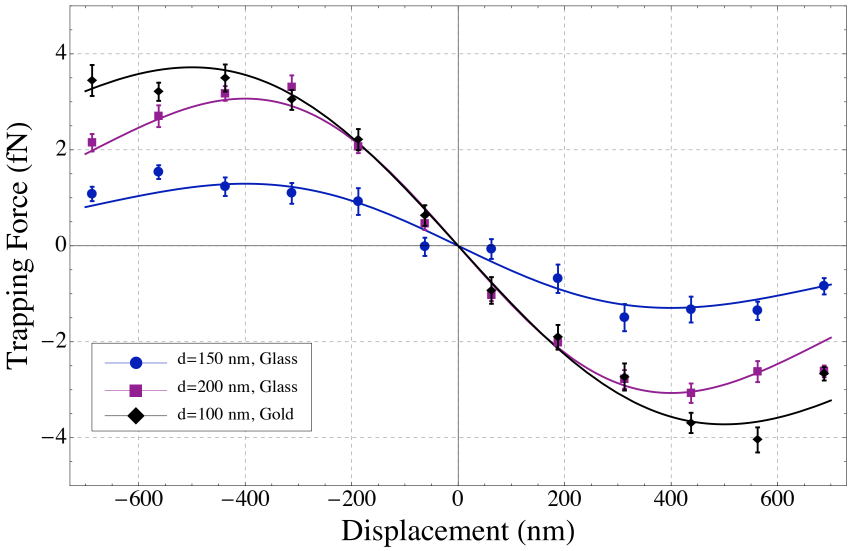


Figure 4.13: The trapping force as a function of displacement from the trap center, calculated using numerical simulations of gold and glass nanoparticles in an optical trap. The solid lines are the input force to the simulation and data points indicate the trapping force calculated from the simulation output.

the error in the trap parameters α and σ is larger for the $150\,nm$ particle while the errors between the $200\,nm$ and $300\,nm$ diameter particles are comparable. The results highlight the importance of the total measurement time on the fit parameters, especially the errors in σ. The first three entries in the table present results for simulations performed with glass nanoparticles, for a total measurement time of $2\,s$. The error in the trap parameter α is approximately $7\,\%$ for the $200\,nm$ diameter glass particle. However, when the simulations are repeated for a total measurement time of $7\,s$, the error in σ does not change systematically, however, the error in the parameter α reduces significantly to 1% and is discussed next.

The dependence of errors in estimating the parameters of the trapping force on measurement time is due to the fact that at short times, the particles remain close to the center of the trap and therefore do not sufficiently sample the optical trapping

Table 4.5: Stochastic Noise Method: Simulation Results

d	Mat.	Meas.	α		σ	
		Time	Actual	Sim±SE (Pct. Err.)	Actual	Sim±SE (Pct. Err.)
nm		s	k_BT	k_BT (%)	nm	nm (%)
150	Glass	2.0	0.21	0.24 ± 0.01 (16.2)	400	455 ± 28 (13.8)
200	Glass	2.0	0.50	0.54 ± 0.01 (7.2)	400	421 ± 13 (5.3)
300	Glass	2.0	1.69	1.81 ± 0.01 (7.2)	400	434 ± 3 (8.5)
200	Glass	7.0	0.50	0.50 ± 0.02 (1.1)	400	416 ± 16 (4.1)
100	Gold	7.0	0.76	0.73 ± 0.03 (-3.6)	500	500 ± 21(0.0)
Fluid medium: water at $T = 293\,K$						

potential. Since the particles themselves act as probes in this process, at very short measurement times we can recover the trapping force only very close to the center of the traps. The estimate of the trapping force improves as the particles explore regions of the trap that are away from the center. This phenomenon can also be seen in the structure of Equation 4.10, used to calculate the optical trapping force profile from a PDF. The force calculation involves a division by $W(x, t_s)$, therefore at short times, when the probability of finding the particle far away from the center of the trap is small, the force calculation is only numerically stable close to the trap center and can estimate the trapping force close to the center of the trap. On the other hand, as the measurement time increases, more particles exit the trap and the estimate of the trapping force away from the center of the trap improves. Therefore at very short times, we expect to estimate only the trap stiffness (the first term in a power series expansion of the force about the trap center) and progressively estimate higher order terms at longer times. Eventually, at very long measurement times we can expect to recover the trapping force profile beyond the inflection points of the potential. Therefore, at $t = 2.0\,s$, we are able to recover the trapping force beyond the inflection points of the potential, but we see relatively large errors in the fit parameter σ, which reduce even further by extending the measurement time to $7\,s$.

The bottom of Table 4.5 shows the results of the force measurement for the $100\,nm$ gold nanoparticle for a measurement time of $7\,s$ to demonstrate that this technique works for nanoparticles of different size and materials. We see that the value of the trap parameter α, recovered from the simulation data has a lower error than the $150\,nm$ glass nanoparticle. This can be explained due to the stronger optical forces that act on the gold particle in comparison with a glass nanoparticle of similar size. The stronger interactions confine the particle to the trap for longer times and therefore improve the sampling of the trap.

4.5.2 Laboratory Results

We report results obtained by implementing the stochastic noise force measurement technique in the laboratory using $100\,nm$ gold nanoparticles. The procedure used to obtain these results is similar to the simulation results discussed previously. We begin the experiments with a nanoparticle stably confined in a strong optical trap. Next, we reduce the power of the trap sufficiently so that the particles escapes the trap in a short time and record the trajectories of nanoparticles as they exit the trap. We then construct probability density function from the recorded trajectories and are able to recover the trapping force, as discussed in Section 4.4. Moreover, as we will discuss in Section 5.2.1, nanoparticles preferentially exit the trap in the longitudinal direction due to the effect of the scattering force. Hence we perform this force measurement along the Z-axis. Consequently, we observe that this technique allows us to recover the axial force profile of the optical trap and even gives us good agreement with existing theory.

We first prepare a sample cell, approximately $100\,\mu m$ thick using a glass slide and cover slip and sealed on all sides using double sided adhesive tape. We then fill this sample cell with a dilute suspension of $100\,nm$ gold nanoparticles (Ted Pella, P/N: 15711-20). Similar to the step input method, we calibrate the detection system, used to measure the XYZ-position of the particle in the trap using the procedure described in Section B.2. Next, we create a stationary trap by moving the AOD to the center of its range and ensure that it is properly aligned with the detection beam.

We first adjust the voltage bias of an electro-optic modulator (EOM) (Model 350-80/Power Supply: M302RM, ConOptics Inc.), which controls the trap intensity, to set the trap power to approximately $150\,mW$, measured using the online power measurement system described in Appendix A. Next, we locate and then trap a nanoparticle in the strong static trap. The measurement is then performed using a LabView script, which changes the EOM bias voltage to reduce the power of the trap to approximately $60\,mW$ and then records the X, Y and Z position of the particle for $5\,s$, as it exits the trap. The particle trajectories were recorded with a bandwidth of $40\,kHz$ and repeated with 65 independent particles.

The trajectories of the particles are then used to recover the trapping force using the analysis technique outlined in Section 4.4 and demonstrated previously for simulation results. However, as we will show in Section 5.2.1, nanoparticles in a weak optical trap escape preferentially along the Z-axis due to the presence of a non-conservative scattering component in the total optical force. Therefore, this technique naturally provides a good estimate of the longitudinal trapping force, using the Z-component of the measured position, as seen below.

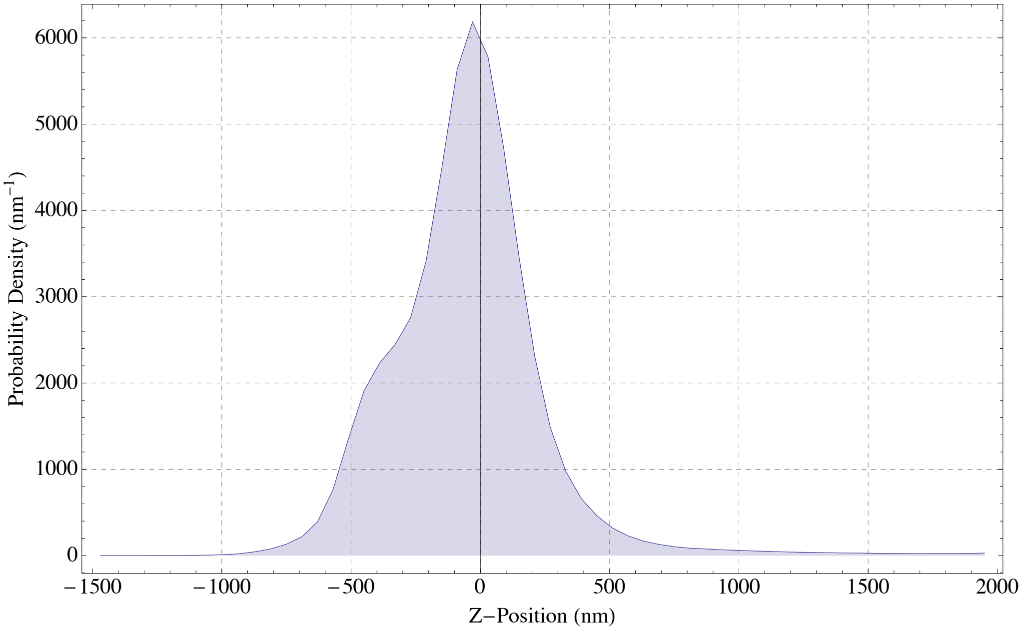


Figure 4.14: Probability density function of the particle's position in the trap.

The quasi-steady state probability density function (PDF) constructed, using techniques described in Section 4.4, from measured trajectories of $100\,nm$ gold nanoparticles in a weak optical trap is shown in Figure 4.14. The trap power in the measurements is low enough so that the particles first diffuse to fill the trap volume before proceeding to exit the trap, well before the measurement period of $5\,s$ and typically between $1 - 2\,s$. Therefore to calculate the quasi-steady state PDF, we discard the first $25\,ms$ of data from each trajectory, which is much larger than the time required by a 100 nm gold nanoparticle to fill the trap volume, before we calculate the PDF.

As we see from the figure, the quasi-steady state probability density plot exhibits qualitatively different behavior on either side of the origin. As we describe in more detail in Section 5.2.1, nanoparticles escape the trap preferentially along the positive Z-axis. The PDF in Figure 4.14 supports this phenomenon and we observe that the density of particles falls quickly as we move along the positive Z-axis, but remains non-zero right up to the point where the plot is truncated at 2000 nm. On the other hand, particles do not escape the trap along the negative Z-axis. The PDF appears wider on the negative side of the origin and becoming negligibly small before $-1000\,nm$. Next, we use the measured PDF to estimate the trapping force exerted on the particle using the analysis techniques described previously.

Figure 4.15 shows the trapping force measured for 100 nm gold nanoparticles in the laboratory using the stochastic noise method. In the figure, the blue plot markers show the trapping force measured in the laboratory and the error bars represent the standard error of the measurement. Furthermore, the black plot in the figure shows the trapping force calculated using the generalized Lorentz Mie theory (GLMT) model, for a comparable beam power as the measured data [104]. The GLMT force field provides a realistic estimate of the optical trapping force and includes the gradient and scattering components of the total force. The details of the calculation can be found in Ref. [67]. The black curve in Figure 4.15 shows the axial trapping force obtained from the GLMT model, for a 100 nm gold nanoparticle

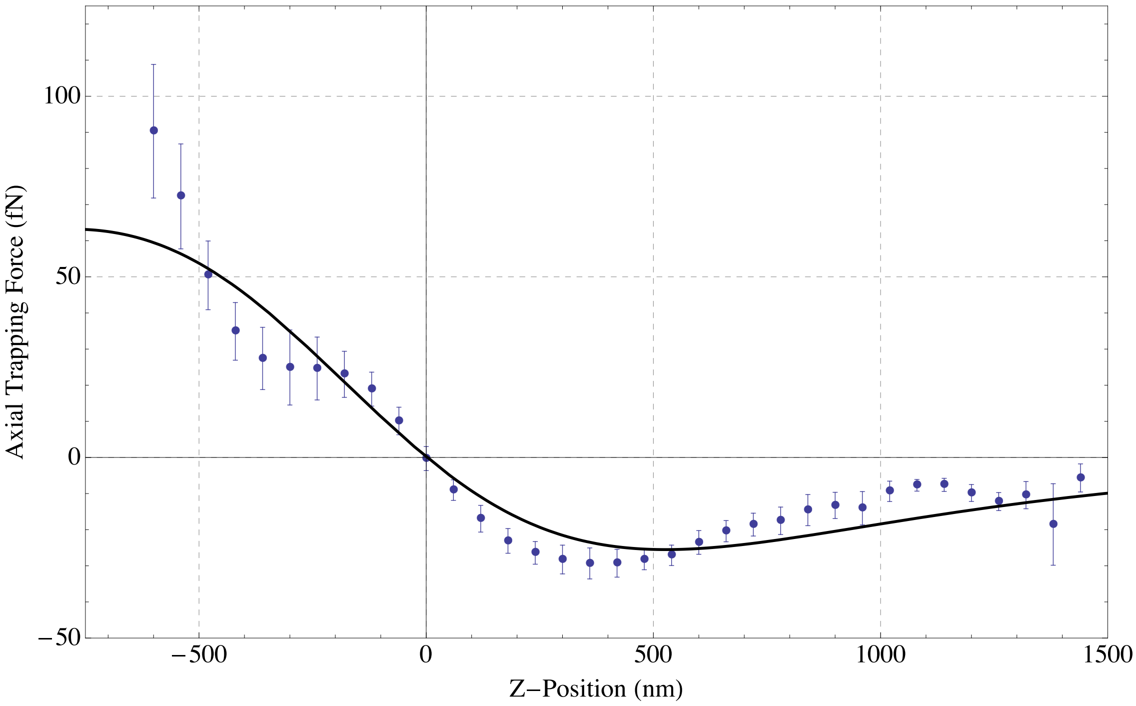


Figure 4.15: Axial trapping force calculated for a trapped $100\,nm$ gold nanoparticle in a weak optical trap. The plot markers in the figure represent the measured force and the error bars show the standard error of the measurement. The black solid line shows the theoretical GLMT force model for a comparable beam power. We observe that there is very good agreement between theory and the measured force.

in an optical trap with the wavelength of the incident light, $\lambda = 1064\,nm$. We set the size of the focal spot of the beam to $550\,nm$ and assume the surrounding medium is water, with a refractive index of 1.33.

From the figure, we observe that we are able to measure the trapping force well beyond the maximum force point along the positive Z-axis. Moreover, the trapping force is stronger on the negative side of the origin, consistent with the results of the GLMT calculation. Overall, we see good agreement between the GLMT force model and the measured force data, both with the magnitude of the trapping force, as well as the overall shape of the curve. However, unlike the GLMT plot, the measured force data is sensitive to optical misalignments and other deviations of the trap from ideal behavior. One example of such a deviation from ideal trapping behavior is seen from the kink the trapping force at approximately $-250\,nm$, which could arise from a combination of factors, such as distortions of the trapping beam wavefront from the AOD or other misalignments in the optical system. Nonetheless, the reasonable good agreement between experiment and theory, so far a relatively rare occurrence in the field of optical trapping, serves to improve our confidence in this force measurement technique.

4.6 Summary

We have developed two new force measurement methods to measure the optical trapping potential, without making any prior assumptions about it's shape. The techniques developed are validated using simulations and also implemented in the laboratory. Laboratory measurements of these techniques can allow us to improve force models that are incorporated into the simulations, thereby improving their utility in extending optical trapping to nanomanufacturing.

The first force measurement technique developed is the step input method, where an optical trap is subjected to a step input and the response of the particle is measured and used to recover the optical trapping force. This method is validated, in two-dimensions, using numerical simulations of $100\,nm$ gold nanoparticles and $640\,nm$ glass microspheres. The relative error in the recovered trapping parameters was found to be small and the error due motion of the particle outside the measurement plane was found to have a negligible impact on the recovered trapping force. The step input method has also been demonstrated in the laboratory, in one-dimension for a 1 μm glass microsphere and for a 350 nm glass nanoparticles as described in Chapter 5.

The optical trapping force has also been measured using the thermal motion of a nanoparticle in an optical trap. This method has been tested using numerical simulations, with gold and glass nanoparticles, ranging in size from 100 nm to 200 nm, with good results. Furthermore, we have implemented this technique in the laboratory using 100 nm gold nanoparticles to measure the axial trapping force and found reasonably good agreement between the measured data and the theoretical GLMT force field.

The optical trapping force techniques we have developed will benefit the extension of optical trapping to the nanoscale. As we have shown, theoretical models of the optical trapping force do not always account for all optical effects, such as tightly focussed beams or spherical aberrations. Therefore, data collected by implementing these methods in the laboratory is valuable, when used in conjunction with numerical simulations, to develop new control methods like those demonstrated in Chapter 5.

Chapter 5

Enhanced Nanomanipulation Through Controlled Optical Trapping

In this chapter, we demonstrate enhanced trapping methods using control systems, which are able to confine nanoparticles to an optical trap, when a corresponding static trap at the same power is unable to localize a same-sized particle. Furthermore, these techniques significantly reduce the average power required by a static trap to confine a particle. Therefore, we believe we can mitigate damage to particles in the trap and this technique can be particularly beneficial when trapping nanoscale biological particles. Moreover, the development of enhanced trapping of nanoparticles may lead to better nanomanipulation using optical tweezers and also lead to the development of this instrument as a flexible tool for prototyping novel three-dimensional nanodevices.

Numerical simulations, developed in Chapter 3, have played an important role in developing the new control algorithms used in this work. We have implemented a realistic model of the optical trapping force, which includes both conserved and non-conserved components of the trapping force, based on the generalized Lorentz Mie theory (GLMT) [104, 67]. Simulations of nanoparticles in an optical trap modeled by the GLMT force field have resulted in a better understanding of the escape mechanism of nanoparticles in weak traps, which we discuss in more detail in Section 5.2.1. This in turn has significantly impacted the design of the controller. For example, we have determined that modifying the intensity of the trap has a far greater impact on lifetime than scanning the position of the trap in response to particle movement. We discuss the design of our intensity controller in Section 5.2.3, followed by simulation results that describe the performance of the control algorithms.

We have implemented our representative controller design in the laboratory and tested it using $350\,nm$ glass nanoparticles and $100\,nm$ gold nanoparticles. Instead of designing analog electronics to implement the controller, we use low-cost field programmable gate arrays (FPGA), which provide excellent performance and flexibility. The implementation details of the controller on the FPGA are discussed in Section 5.3.1. Finally, we discuss laboratory results for the two particle sizes using the FPGA-based control in Section 5.4.

5.1 Motivation

Manipulation and assembly of microscale particles in fluids using optical tweezers is widely used in many areas of research. However, the extension of optical trapping to nanoscale assembly is less developed. One challenge is that for a constant beam power, the optical binding forces acting on a nanoparticle decrease significantly in comparison with larger particles. Consequently, it is more challenging to keep a particle confined to the trap without significantly increasing the laser power. Recent work has demonstrated that even when using laser powers as high as $855\,mW$ (which results in power densities of thousands of megawatts per square centimeter for diffraction limited traps), gold nanoparticles with a diameter of 18 nm, dispersed in water, can only be confined to an optical trap for a few seconds at a time, which limits their utility in nanoassembly applications [72]. Furthermore, particles smaller than 18 nm could not be localized when using even higher laser powers. Moreover using high laser powers to trap nanoparticles may not be practical since increasing the intensity of the trap may cause significant heating and damage to the particles. In fact, Seol *et al.* have studied the effect of particle mediated heating for 100 nm gold nanoparticles due to absorption and found significant heating of up to 266 $^{\circ}C/W$ [141].

We have successfully developed a new approach that uses closed loop control systems to improve the lifetime of nanoparticles in an optical trap, in comparison with a static trap. Moreover this technique does not increase the average power input to the system. The lifetime of the optical trap is defined in our work as the inverse of the rate at which nanoparticles first cross the boundary of the optical trap, as described in more detail in Section 5.2.2. Since the edge of an optical trap is not clearly defined and changes with the parameters of the system, we fix the trap boundary, for each case in the analysis. As part of this work, we have developed prototypical controller designs to accomplish these goals. The controller design has been developed using numerical simulations developed previously, which actuate the trap in response to a change in the particle position within the trap. This controller design has finally been successfully demonstrated in the laboratory using 350 nm glass and 100 nm gold nanoparticles. Next, we discuss the design of the prototype control that is used in this work.

5.2 Controller Design

A closed-loop controller changes the parameters of the optical trap in response to the position of a nanoparticle confined within the trap. The controller can directly control only two trap parameters, (i) trap position, where the controller can move the optical trap in response to the motion of the particle and (ii) trap intensity, where the controller can adjust the incident power in response to the nanoparticle's movement. Sophisticated control algorithms can then be constructed by combining the control of these two trap parameters.

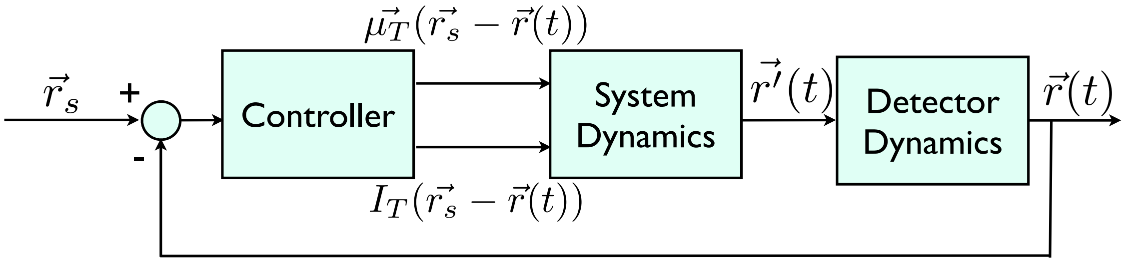


Figure 5.1: Controller Block Diagram. In the figure, $\vec{r_s}$ is the controller set-point, $\vec{r}(t)$ is the ideal particle position and $\vec{r'}(t)$ is the measured position of the particle.

The block diagram for a control system for use with optical trapping is shown in Figure 5.1. In the figure, the controller set-point is given by the vector $\vec{r_s}$ and the ideal position of the particle in the trap is given by $\vec{r'}(t)$. Furthermore, the error signal input to the controller is obtained by subtracting the measured position of the particle in the trap ($\vec{r}(t)$) from the set-point. Since only the position of the trap and it's intensity can be actively controlled, the controller transfer function must generate two output functions, (i) a vector function describing the target position of the optical trap ($\vec{\mu_T}(\vec{r_s} - \vec{r}(t))$) and (ii) a scalar function that sets the target intensity of the trap ($I_T(\vec{r_s} - \vec{r}(t))$. Moreover the two output control signals are independent of each other and the controller can be fully described by two arbitrary functions which operate on the error signal as a function of time.

Controller functions can be parameterized in several ways depending on the performance objectives and constraints on the system. In the simplest case, they can be Proportional Integral Derivative (PID) controllers for both position and intensity. Moreover, the limitations imposed by analog electronics, often define the type of controllers that are used in many optical trapping experiments. We have developed new methods to use a Field Programmable Gate Array (FPGA) to implement our controllers in the laboratory, which gives us the flexibility to implement arbitrary controller functions (see Section 5.3.1 for implementation details).

The primary objective of our control algorithms is to increase the lifetime of nanoparticles in the trap without significantly increasing the average trap power. Therefore, we choose to parameterize our controller using two features, (i) *tracking*, which can improve the lifetime by maintaining a constant force on the particle over a longer spatial range than a static trap and (ii) *blanking*, which can reduce the average power by reducing the beam power when the particle is close to the center of the trap. These two key controller parameters are described next.

Tracking: The primary objective of tracking control is to maintain a constant force on the particle in order to restore it to the center of the trap. In tracking control, we keep the trap stationary until the particle reaches a pre-determined point away from the center, such as the maximum force point of the optical trap (see Section 4.3). When the particle moves past this point, the controller moves the center of the trap to follow the particle in order to maintain a maximum force on it. The controller stops tracking the particle when either a) the particle goes past the detection limit of the laser based tracking system or b) the particle returns close to the trap center. By tracking the particle position, the tracking controller effectively confines the particle to the trap for an extended period of time and thereby increases the lifetime of the nanoparticle.

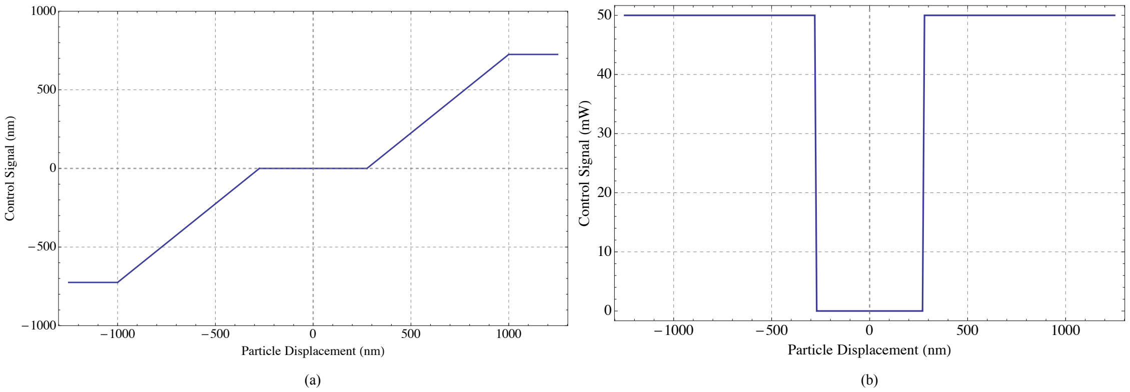


Figure 5.2: (a) An example tracking control signal and (b) and example blanking control signal designed for a $350\,nm SiO_2$ nanopoarticle.

A typical control signal for tracking control for a one-dimensional system with a $350\,nm$ glass nanoparticle is shown in Figure 5.2(a). In the figure, the particle position is shown along the horizontal axis and the control signal that sets the trap position is shown along the vertical axis. As seen from the figure, when the particle is within $\pm 250\,nm$ of the center, the trap remains stationary. As the particle crosses this pre-determined hold point, the trap follows the particle until it reaches the detector limit, set at $\pm 1000\,nm$. Moreover this method is easily extended to two or more dimensions. However due to the hardware configuration of our experiment, we can only fast scan the trap along the X and Y axes, transverse to the direction of propagation of light. Therefore, tracking control is currently limited to two dimensions in our experiments.

Blanking: The second controller parameterization we use is blanking control, which is designed to reduce the power of the trapping beam when the particle is close to the trap center. An optical trap, such as the one used in our experiment has a beam intensity profile that is maximum at the center and decreases away from the origin. However the optical trapping force is zero at the center of the trap and increases until it reaches the maximum force point. Therefore by blanking the beam close to the center of the trap, we are able to reduce the average power incident on the particle and the surrounding fluid.

A typical blanking control signal is shown in Figure 5.2(b), for a $350\,nm$ glass nanoparticle in one dimension. As we see from the figure, the control signal is symmetric about the origin. The controller is designed to turn off the trap when the particle is within $\pm 250\,nm$ of the center and then quickly increases the beam intensity to a maximum value of $50\,mW$ outside of this region.

For the one-dimensional case shown in Figure 5.2, tracking control can be combined with blanking control to apply a constant force on the particle over a wide spatial range and reduce the average incident power by blanking the beam near the center. However, as we show in subsequent sections, 3D traps that include the scattering force, moving the trap position in response to the particle movement plays a much more limited role and intensity control provides the most benefit.

5.2.1 Escape Mechanism of Nanoparticles From Weak Optical Traps

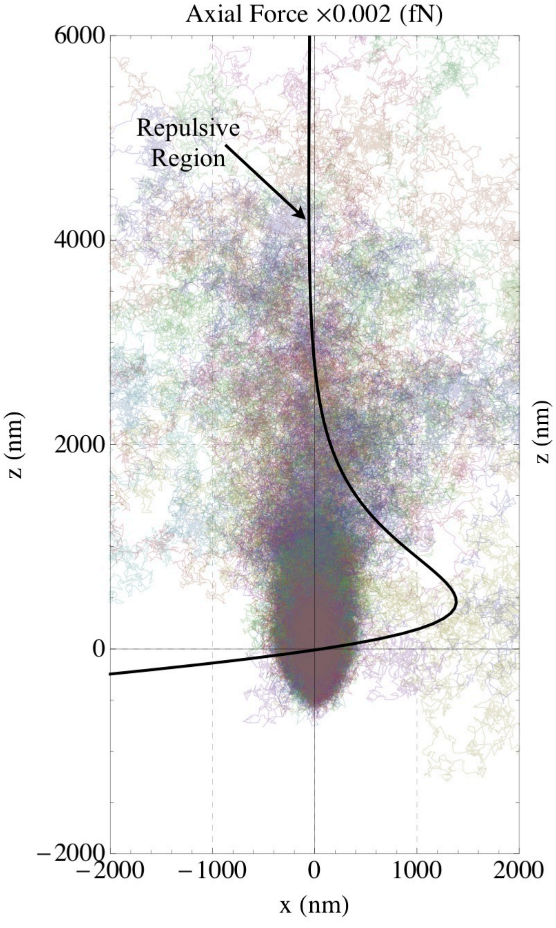
Single beam gradient optical traps use a tightly focussed optical beam to grip a nanoparticle and hold it close to the focus of the beam. As discussed in Chapter 2, the optical trapping force has two components, a conservative gradient force component that pushes the particle towards the focus of the beam and a non-conservative scattering component, arising from the radiation pressure of light, which directs the particle along the direction of propagation of light. Therefore, a stable three-dimensional optical trap is formed when the gradient forces overcome the scattering forces. The scattering force, which is proportional to the intensity of the trapping beam, is highest at the center of the beam and decreases away from the axis that defines the direction of propagation of light. Moreover, while the gradient force is strong along the transverse directions and weaker along the longitudinal directions, the scattering force by contrast acts principally along the longitudinal axis and is very small in the transverse directions. Furthermore, due to the forward radiation pressure, a microscale particle is believed to undergo a toroidal circulation within the trap rather than reach an equilibrium position [137].

Nanoparticles generally experience weak optical binding forces, in comparison with micrometer sized particles and are thereby not very strongly trapped. In order to better understand the interactions of nanoparticles with optical traps, particularly the mechanism of escape in weak traps, we have implemented the GLMT force field in our simulations [104, 67]. This force model provides a realistic description of the total optical trapping force acting on the particle and includes both the gradient and scattering force components.

Equation 5.1 shows the total trapping force calculated using the GLMT model in three-dimensional Cartesian coordinates, where n_m is the refractive index of the fluid, c is the speed of light, P is the power of the incident beam, w_0 is the size of the beam focus and $\vec{Q}(\vec{r})$ is the trapping efficiency [104]. The trapping efficiency $\vec{Q}(\vec{r})$, is independent of power and is calculated by first evaluating the scattering cross-sections ($C_{pr,x}$, $C_{pr,y}$, $C_{pr,z}$) along the three axes using the methods detailed in Ref. [67]. The GLMT force model is developed under a paraxial approximation, assuming an objective with a low numerical aperture, which does not tightly focus the beam. However, we use high numerical aperture objectives in our experiments, capable of focusing the beam to a spot approximately the size of the half the incident wavelength. Therefore, even though the GLMT is not an ideal model to approximate our experiment, it provides a realistic estimate of the trapping force and is an invaluable component in controller design, as we will see in later sections.

$$\begin{aligned} \vec{F}(\vec{r}) &= \left(\frac{n_m\,P}{c}\right) \times Q(\vec{r}) \\ \vec{Q}(\vec{r}) &= \frac{2}{\pi w_0^2}\left\{\hat{x}\,C_{pr,x}(\vec{r}),\ \hat{y}\,C_{pr,y}(\vec{r}),\ \hat{z}\,C_{pr,z}(\vec{r})\right\} \end{aligned} \tag{5.1}$$

The GLMT force field has been implemented in our simulations using the fast lookup method described in Appendix C. Therefore, we generate a table of trapping efficiencies ($\vec{Q}\,(\vec{r})$) for $350\,nm$ silica nanoparticles irradiated using a laser beam with a wavelength, $\lambda = 1064\,nm$, over a closed interval of $\pm 3\,\mu m$ along the x and y axes and $[-2, +6]\,\mu m$ along the z-axis, with a uniform grid spacing along all axes of $75\,nm$. The trapping force is then calculated within the simulation using Equation 5.1 for the grid point closest to the particle position. Since the trapping efficiency tables are independent of power, the same GLMT tables are used throughout the rest of this chapter to test the performance of static traps and controller algorithms that run at different beam powers.



(a)

Figure 5.3: 100 simulated the trajectories of $350\,nm$ glass nanoparticles in a weak GLMT force field. The trapping force along the direction of propagation of light is shown by the thick black plot. From the figure, we see that the particles escape preferentially along the positive z-axis.

Figure 5.3 shows 100 simulated trajectories of $350\,nm$ glass nanoparticles under the influence of a static GLMT force field with a beam power of $5\,mW$. The total trapping force along the Z-axis generated by the GLMT model is overlaid on top of the nanoparticle trajectories in Figure 5.3. The thick black line in the figure represents the total axial trapping force, including gradient and scattering components (directed along the positive z-axis). From the figure, we see that the trapping force is attractive on either side of $z = 0$ and the magnitude of the attractive force is much larger on the negative side of the origin. On the other hand, when $z > 0$, the trapping force reaches a maximum at approximately $500\,nm$ and then decreases, eventually becoming repulsive at approximately $3000\,nm$. The combination of weak gradient forces and axially directed scattering forces result in particles escaping preferentially along the positive z-axis as seen in the figure. Furthermore, the particles are well confined within the trap in the transverse direction, due to relatively stronger gradient forces and a negligible transverse scattering force.

$$\frac{\partial W\left(\vec{r},t\right)}{\partial t} = -\nabla_{\vec{r}}\left(\frac{1}{\gamma}\vec{F}\left(\vec{r}\right) - \frac{k_B T}{\gamma}\nabla_{\vec{r}}\right) W\left(\vec{r},t\right) \tag{5.2}$$

The escape of nanoparticles from the trap can be understood using the Fokker-Planck equation, which is shown in Equation 5.2 [131]. In this equation, the particle probability density as a function of position and time is given by $W(\vec{r},t)$, defined using Equation 5.5 in our experiments. Furthermore, $\vec{F}\left(\vec{r}\right)$ represents the trapping force field, γ is the Stoke's constant as defined previously, k_B is Boltzmann's constant and T is the temperature of the fluid.

$$\begin{aligned} &\frac{\partial W(\vec{r},t)}{\partial t} + \nabla_{\vec{r}} S = 0 \\ &S = \nabla_{\vec{r}}\left(\frac{1}{\gamma}F\left(\vec{r}\right) - \frac{k_B T}{\gamma}\nabla_{\vec{r}}\right) W\left(\vec{r},t\right) \end{aligned} \tag{5.3}$$

The Fokker-Planck equation can be re-written as a continuity equation, shown in Equation 5.3, where S is the probability current. The probability current can be described by two terms: (i) a drift term that is proportional to the product of the optical trapping force and the PDF, and (ii) a diffusion term that is proportional to the negative gradient of the PDF and driven by the thermal motion of the molecules in the fluid. For a well confined particle in a conservative force field, such as a microscale particle in a strong Gaussian trap (where the drift current can be much stronger than the diffusion current), we observe that in a quasi-steady state, the net particle flux leaving the trap is negligibly small. Therefore, the drift and the diffusion terms in Equation 5.3 balance each other and the net probability current is negligible ($S = 0$). However, in practice the particle is found not to reach an equilibrium position in the trap due to the presence of the non-conservative scattering force component. Instead microscale particles have been found to undergo a continuous toroidal circulation within the trap [137]. For smaller particles, diffusion plays a more dominant role and the magnitude of the drift current is no longer sufficient to keep the particles confined to the trap. Furthermore, the weaker restoring force along the longitudinal axis, coupled with the scattering force appears to provide an easier escape route, leading to the preferential escape of particles along the positive Z-axis that we observe in Figure 5.3.

We find that the mechanism of escape of nanoparticles from an optical trap has strong implications for controller design. As we see from Figure 5.3, particles predominantly escape the trap in the longitudinal direction. Therefore scan control in the transverse XY plane has limited impact in improving the lifetime of nanoparticles. In fact, results from simulations and in the laboratory presented in later sections show that significant improvements of lifetime are achieved without resorting to XY scan control. In the next section, we define the metrics we use to evaluate the performance of our representative controller design followed by simulation results.

5.2.2 Controller Performance Metrics

The performance of individual control algorithms is benchmarked against the performance of the static optical trap discussed in the previous section. We define two metrics to evaluate controller performance, i) the lifetime of the nanoparticle and ii) the average incident trap power, which are discussed in more detail below. Data obtained from numerical simulations, as well as data collected in the laboratory are then expressed in terms of these quantities, which allow easier relative comparisons of performance.

5.2.2.1 Trap Lifetime

We define the lifetime of the optical trap as the inverse of the rate at which nanoparticles first cross the boundary of the optical trap. Since the edge of an optical trap is not clearly defined and changes with the parameters of the system, we fix the trap boundary, for each case in the analysis. An ensemble of nanoparticles, initially located at the center of a weak optical trap, diffuses quickly to approach the boundary of the trap in a very short time, in comparison with the particle's lifetime, before proceeding to escape from the trap. Once the particles have diffused to fill the trap, the number of particles remaining in the trap as a function of time

(t), given an initial population N, is described by the expression for an exponential decay in Equation 5.4. Therefore, for time-scales larger than the time the particle takes to completely fill the trap, the exponent (λ_r) in this equation sets the rate at which nanoparticles cross the boundary of the trap and therefore the mean lifetime is given by $\frac{1}{\lambda_r}$.

$$n = N\, exp\, (-\lambda_r\, t) \tag{5.4}$$

An optical trap does not have a well defined boundary. The size of the trap depends on parameters of the system such as the beam size and the diameter of the particle. Therefore, when calculating the lifetime of a nanoparticle, we explicitly fix a three-dimensional bounding box for the trap as shown in Figure 5.4 and then discard points in the trajectory after the particle first crosses this boundary. Figure 5.4 shows the point cloud for 50 simulations of $350\, nm$ glass particles in a weak GLMT force field, with a beam power of $5\, mW$. We have seen in Section 5.2.1 that particles in a weak optical trap escape preferentially along the Z-axis. Therefore the size of the bounding box along the X and Y axis is less critical than the size in Z. Along the Z-axis, we define the top of the boundary box at a point that a majority particles pass through. Therefore, in Figure 5.4, we set the bounding box to $\pm 1.5\, \mu m$ along the X and Y axes and $-1\, \mu m$ to $+2\, \mu m$ along the Z-axis. In the figure, the positions of the particles are drawn past the bounding box along the Z-axis to highlight the predominant escape route of the particles. We calculate the lifetimes of nanoparticles in the trap in order to make relative comparisons between the controller *on* and controller *off* cases. This allows us to effectively evaluate the performance of the controller in relation to a static trap with the same average power. Therefore, the size of the bounding boxes must be identical when comparing data sets against each other.

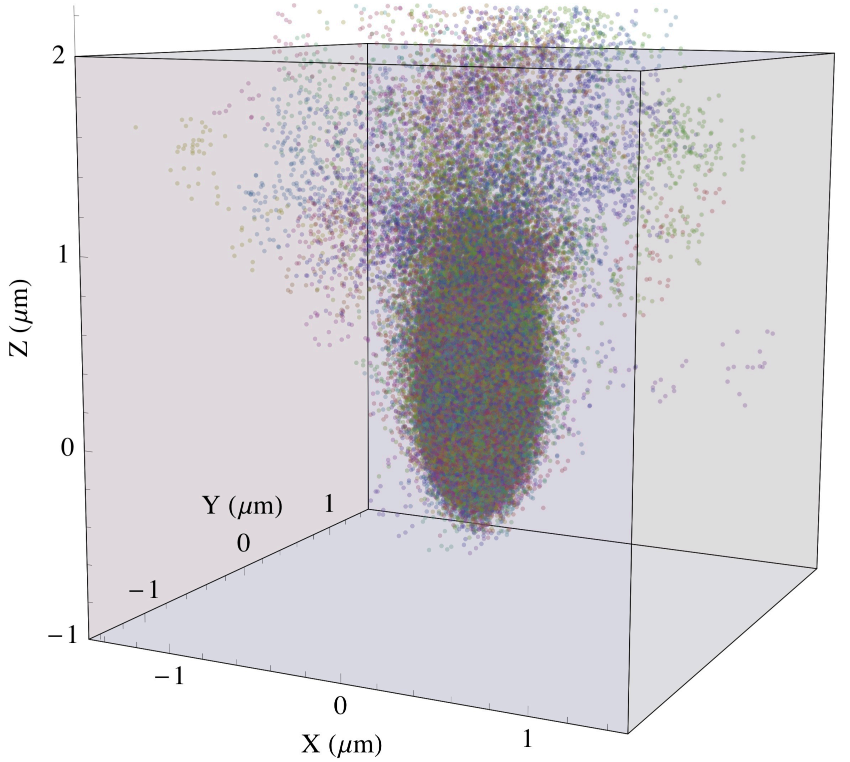


Figure 5.4: First crossing of $350\,nm$ glass nanoparticles in a weak GLMT force field. The point cloud is drawn past the positive Z boundary to highlight the predominant escape along the Z-axis.

$$\psi(x, y, z, t; \delta, \delta t) = \frac{h(x, y, z, t; \delta, \delta t)}{\delta^3 \times \delta t} \tag{5.5}$$

We use the trajectories of particles inside the bounding box to construct the probability density function of particle positions as shown in Equation 5.5, where the histogram of particle positions in X, Y, Z and time (t) is given by $h(x, y, z, t; \delta, \delta t)$. The spatial bin size, which is uniform along all three axes is defined by δ and δt defines the bin size in time. Figure 5.5 shows the conceptual overview of the process used to calculate the PDF in position and time for one spatial dimension. The sub-figure on the top left of Figure 5.5 shows the raw trajectories output by the simulation as a function of time for each particle in the ensemble. These trajectories are then truncated, shown in the sub-figure on the top right, to obtain the positions of the particle inside the optical trap (the trap boundary is shown by the two thick lines in the figure). Next, we calculate the histogram of the particle's position and time for the selected spatial and temporal bins. The PDF for the one-dimensional case is obtained by normalizing this histogram by the $\delta \times \delta t$. For our simulations, we simply extend this process to three-spatial dimensions and time and calculate the PDF using Equation 5.5.

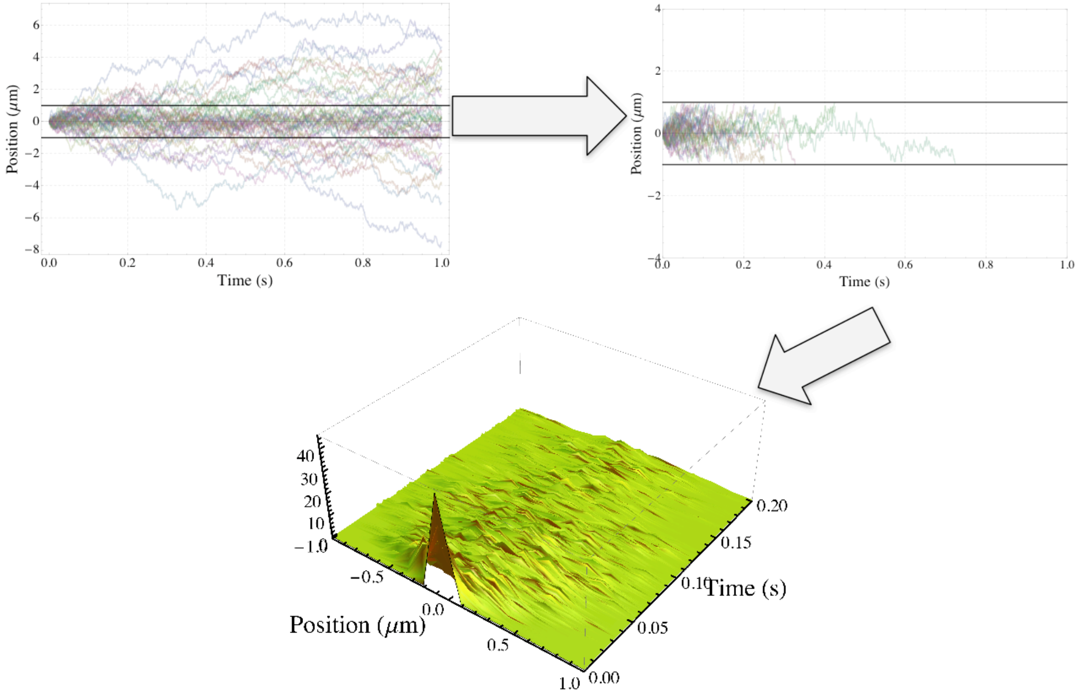


Figure 5.5: Summary of the process used to calculate a PDF from simulation trajectories in one-dimension.

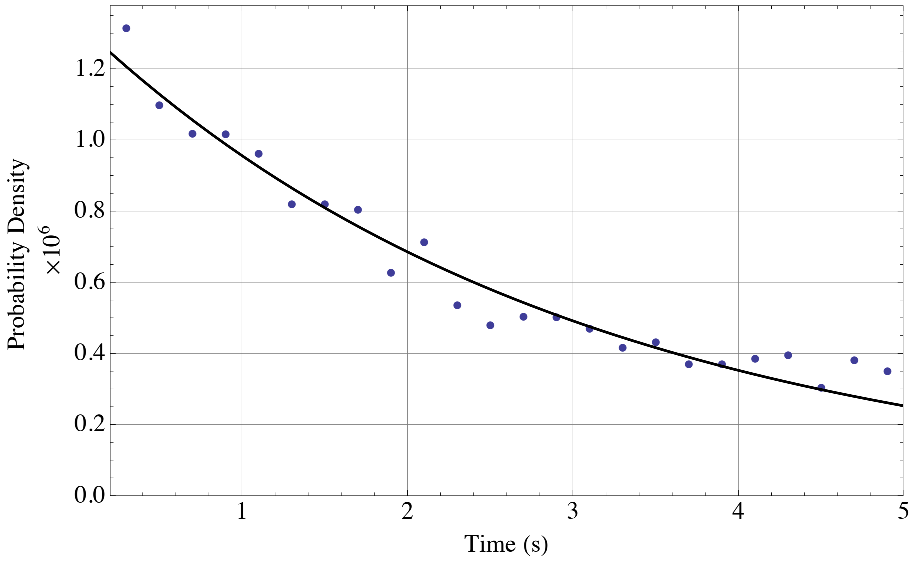


Figure 5.6: Lifetime estimate for 350 nm glass nanoparticles in a weak GLMT force field from simulations. We trace the peak of the PDF as a function of time and use Equation 5.4 to estimate the lifetime.

Once we obtain the PDF of particles within the optical trap for times longer than those required to fill the trap, we calculate the lifetime by locating the bin that contains the peak of the PDF in X, Y and Z at times longer than that required to fill the trap. Next, we trace the probability density in this bin as a function of time as shown in Figure 5.6 and estimate the lifetime of the particle by performing a least squares fit using the model in Equation 5.4. The lifetime calculated using this method for three simulation cases are shown in Table 5.2.

Lower Bounds on Lifetime for the Controller On Case: In experiments where the particle exits the trap a very small number of times over the period of observation, for example when the controller is *on*, the techniques described previously do not provide a reasonable estimate of the lifetime. In such situations, we can however estimate a lower bound for the lifetime of the particle for a pre-determined confidence level as discussed below [109].

When we observe only a small number of escape events from the trap, the expression for the exponential decay shown in Equation 5.4 can be simplified using a series expansion to obtain the decay rate (λ_r) as given in Equation 5.6 [156]. In this equation, N is the number of trials performed in an experiment, k is the number of observed escape events and t is the total observation period for each trial. The mean lifetime of the particle is then approximated by $\tau = 1/\lambda_r$.

$$\lambda_r \approx \frac{k}{Nt} \tag{5.6}$$

In some experiments, the particle never escapes the trap ($k = 0$), in which case λ_r, in Equation 5.6, equals zero. In such situations, the corresponding lifetime of the trap, estimated from the inverse of the rate in Equation 5.6 would be calculated to be infinitely large. However, such an estimate would be unphysical since optical traps have a finite lifetime, which can be observed with sufficient trials or a longer observation period. Furthermore, even when we have no observed events where the particle escapes the trap, we can estimate a lower bound on the lifetime with a confidence level α.

In order to place a lower bound on the lifetime, we must first calculate an upper bound on the decay rate (λ_r), since the lifetime is the inverse of the rate of decay. The probability of observing the number of discrete events occurring over a fixed period of time can be expressed using a Poisson distribution, if the mean number of particles escaping the trap (μ, known as the parameter of the distribution) is known. However, in our experiments the parameter of the Poisson distribution is not known beforehand. Therefore, in order to calculate the upper bound on λ_r, we must first estimate an upper bound (μ_{lim}) on the Poisson parameter. We estimate μ_{lim}, such that for k observed escape events, we find a Poisson distribution with parameter μ_{lim}, which exceeds the true Poisson parameter (μ) of the distribution with probability $1 - \alpha$, for example 95%. This is shown formally in Equation 5.7, where we solve the cumulative distribution function (CDF) for the Poisson distribution at a given α to obtain the upper bound of the Poisson parameter, μ_{lim} [22]. Once we obtain an estimate of μ_{lim}, we can substitute $k = \mu_{lim}$ in Equation 5.6 to obtain an upper bound on λ_r. Consequently, the lower limit for the mean lifetime of the particle is the inverse of the estimated upper bound on λ_r.

$$\text{Solve for } \mu_{lim} : \ \text{CDF}_{Poisson}(n; \mu_{lim}) = \alpha \tag{5.7}$$

As an example, if 5 particles exit the trap ($k = 5$) over the observed time period ($t = 10\,s$), we can solve Equation 5.7 to estimate the upper limit on the parameter μ_{lim} with 95% confidence ($\alpha = 0.05$), to be 10.5. Therefore, if we have $N = 100$ total observations, we can substitute $k = 10.5$ in Equation 5.6 to estimate the upper limit of the decay rate (λ_r). The lower limit on the lifetime is then the inverse of this rate and is found to be approximately $95\,s$. This technique to estimate the lower bounds on the lifetime is used throughout this chapter in simulation and experimental cases when the controller results in very few or no escape of particles from the trap.

5.2.2.2 Average Power

The lifetime of a particle in an optical trap can be increased, with some limitations, without resorting to closed-loop control, simply by increasing the power of the incident beam. However, this can cause heating and significant temperature increase due to absorption of the incident light by both the particle, as well as the surrounding fluid. Severe heating can destroy the nanoparticle in the trap or result in the trap becoming destabilized due to strong convective forces from the heating of the fluid. Furthermore, even moderate changes in the temperature of the fluid can result in large changes to the local viscosity, thereby making it harder for a controller to keep the nanoparticle confined to the optical trap. However, in our experiments, we are only interested in comparing controller performance, relative to a static trap. Therefore, we only compare the average power input to the trap in each case. In our experiments a successful controller will allow us to trap a particle for a significant period of time at some measured average input power, when a static trap at the same power is unable to do so. In subsequent sections, we show data from simulations and laboratory experiments that clearly demonstrate this behavior.

We report the average incident power recorded in our experiments for both laboratory and simulation results, over the course of the experiment. In the laboratory, we have a continuous online measurement of the laser power on the optical table (see Appendix A for a hardware description). We record the measured power at the same rate as other signals in the experiment and are therefore able to calculate the average power over the course of the experiment. Moreover, since our simulations also record the power of the optical trap at each time-step, we are able to apply the same methods to calculate the average power to simulation and laboratory data. The average power for three simulation cases is shown in Table 5.2.

5.2.3 Intensity Control Design

We observe, from laboratory experiments and simulation results from the previous section, that nanoparticles escape the optical trap preferentially along the direction of propagation of light (positive Z-axis). Therefore scan control, in the XY-plane has a minimal impact on improving the lifetime of nanoparticles. On the other hand, intensity controllers, which change the trap power in response to particle movement, are simultaneously effective along all three axes. Therefore, in this section we discuss the design of intensity controllers, which change the power of the trapping beam as a function of the particle position.

We begin with the simplest case – an intensity controller that uses the X and Y position of the particle to determine the intensity of the trap. This type of controller is consistent with the hardware commonly found in optical trapping instruments, which include position detection in the transverse plane, but not along the longitudinal axis. In this controller, we setup a blanking region close to the trap center and increase the power to it's maximum value outside this area.

XY Blanking Control: The XY blanking controller used in the simulation results presented below is defined using the piecewise function given in Equation 5.8. In the equation, $r = \sqrt{x^2 + y^2}$ is the radial coordinate of the particle position in two-dimensional Cartesian coordinates and r_b defines the radial extent of the blanking region, within which the trap power is at it's minimum value P_{min}. Furthermore, the controller increases the trap power linearly to P_{max} over the transition region that extends between r_b and r_b' and maintains a maximum power of P_{max} when $r \geq r_b'$.

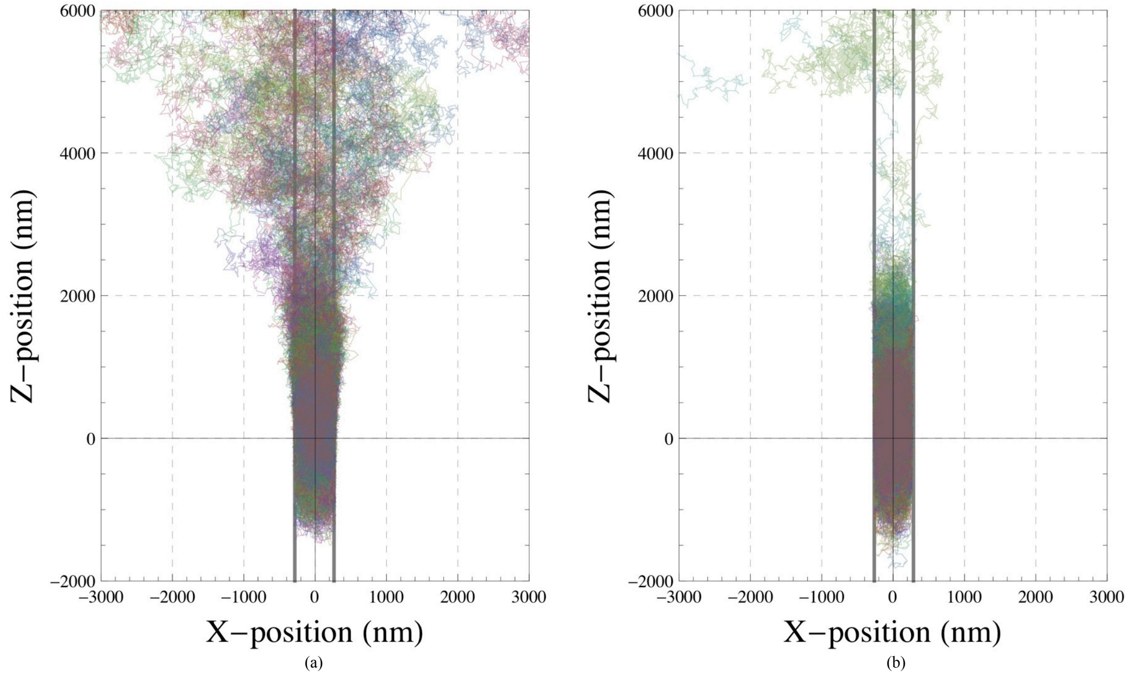
$$I_{xy}(r) = \begin{cases} P_{min} & r < r_b \\ P_{min} + \left(\frac{P_{max} - P_{min}}{r_b' - r_b} \right) \times (r - r_b) & r_b \leq r < r_b' \\ P_{max} & r \geq r_b' \end{cases} \tag{5.8}$$

Table 5.1: Intensity Control Parameters

	Static Trap	XY Blanking	XY Blanking + Z Intensity
Diameter (nm)	350		
Material	SiO_2 (glass)		
λ (nm)	1064		
w_0 (nm)	550		
n_m	1.33		
n_g	1.57		
r_b (nm)	N/A	275	275
r_b' (nm)	N/A	275	275
Detector Range (μm)	N/A	$\pm\{1.4, 1.4\}$	$\pm\{1.4, 1.4, 6.0\}$
P_{min} (mW)	N/A	0	0
P_{max} (mW)	5	25	200
κ_p (mW/μm)	N/A	N/A	$\{0, 0, 100\}$

The parameters we have selected for the XY blanking controller are shown in the second column of Table 5.1. We use a GLMT force field, generated for $350\,nm$ glass nanoparticles under a trapping beam with a wavelength of λ, with a spot size, w_0. The refractive indices of the particle (n_g) and the surrounding water (n_m) are also shown in the table, as is the gain for a proportional controller along the Z-axis (κ_p), which is used in the next section.

We set the blanking region, $r_b = 275\,nm$ to coincide with the maximum force point predicted by the GLMT model in the transverse XY-plane. Furthermore, we set the transition region to 0 ($r_b = r'_b$), so that the controller increases the power instantly to P_{max} and exerts a maximum force on the particle when it leaves the blanking region. Finally, we set the power inside the blanking region to $P_{min} = 0$, so that the trap is completely off. We also limit the maximum power available to the controller to $25\,mW$, which is five times the static trap power (see column 1 in Table 5.1) for the static trap simulations described in Section 5.2.1. Furthermore, this static trap simulation forms our baseline measurement, against which we benchmark controller performance.



(a)

(b)

Figure 5.7: Intensity control results from numerical simulations of $350nm\ SiO_2$ nanoparticles. Figure (a) shows 100 simulated trajectories using the XY blanking controller in the XZ-plane, where most particles escape the trap in a relatively short period of time. Figure (b) shows that adding proportional intensity control in Z together with XY blanking is more effective in confining the particles to the trap.

Figure 5.7(a) shows the trajectories in the XZ-plane from 100 simulation runs, each 5 s long, using the XY blanking controller parameters specified in Table 5.1. Since the trap is circularly symmetric in the transverse plane about the origin, we do not show the YZ-plane in the figure. The two solid lines in the plot represent the blanking region. When the particle is within this region, the trap is turned off and the particle undergoes free diffusion. When the particle crosses the boundary of the blanking region, the controller increases the trap power to P_{max} and the particle experiences a restoring force towards the center of the trap. As seen from the plot, the particle diffuses freely to fill up the volume of the blanking region. Since the controller does not react to the Z-position of the particle, it can freely diffuse in the Z direction, as long as it is within the blanking zone in X and Y. This results in a preferential escape route for the particle along the positive Z-axis, which causes the particle to quickly exit the trap as seen in Figure 5.3.

Table 5.2: Trapping lifetime and power comparison

	Trap Lifetime±SE (s)	Avg. Incident Power±SE (mW)
Static Trap	1.8 ± 0.3	5.0 ± 0.0
XY Blanking	1.7 ± 0.4	5.4 ± 0.5
XY Blanking + Z Intensity	> 80 ($\alpha = 0.05$)	5.8 ± 0.6

We calculate the lifetime for nanoparticles in the trap when using this controller algorithm using the technique outlined in Section 5.2.2. Table 5.2 summarizes the results of both the average power the controller uses and the calculated lifetime. From the table, for 100 simulated trajectories that are each 5 s long, we see no appreciable gain in lifetime when using this algorithm. However, the controller limits the average power to approximately 5 mW, which equals the static trap value. The blanking region near the trap center plays an important role in reducing the average power used by the controller. However limiting the controller to only the XY particle positions leaves an escape route along the Z-axis. This preferential escape route can be eliminated by including controller action along the Z-axis, in addition to the transverse plane and is discussed next.

XY Blanking Plus Z Intensity Control: In order to eliminate the escape route for particles leaving the trap along the longitudinal axis, we have implemented a control algorithm that retains the blanking region in the XY-plane, but adds a proportional intensity controller that responds to a change in the particle position along the Z-axis.

$$I_z(z) = \kappa_p \times |z| \tag{5.9}$$

$$I_T(r, z) = max\,(\ I_{xy}(r), I_z(z)\ ) \tag{5.10}$$

As in the previous example, the XY blanking controller is defined using Equation 5.8, with parameters summarized in column 3 of Table 5.1. In addition, we define a proportional controller along the Z-axis using Equation 5.9, where κ_p is the proportional gain of the controller and $|z|$ is the absolute Z-position of the nanoparticle in the trap. The final control signal is then calculated using Equation 5.10, which selects the largest intensity between the XY and Z controllers.

The controller gain, κ_p is selected to be $100\,mW/\mu m$ (see Table 5.1), so that the Z-controller saturates at $z = \pm 2\,\mu m$ with a maximum power (P_{max}) of $200\,mW$. The trajectories for 100 simulated particles in the XZ plane is shown in Figure 5.7(b). The trajectories of particles appear qualitatively different in this figure.

Only two particles from the ensemble escape the trap, while most particles are contained within ±275 nm along the X and Y axes and within ±2 μm along the Z axis. Moreover only two particles escape the trap, we are unable to calculate the lifetime of the particle using the same parameters and methods used in the previous case. However, we can estimate a lower limit on the lifetime on the particle, as described in Section 5.2.2.1. For this particular controller, for 100 simulated trajectories that are each 5 s long, we estimate the lower limit of the lifetime for two particles escape events with 95% confidence to be approximately 80 s (see Table 5.2), which is significantly greater than the static trap value. Furthermore, as seen from Table 5.1, while the controller has an upper power limit of 200 mW (40 times the static trap power), the average power varies between only 4% and 24% higher than the static trap value. The improved lifetime obtained using this intensity controller depends heavily on the addition of Z-axis control. We have successfully implemented this controller in the laboratory and tested it with different particle sizes and materials. In the next section, we present results from the implementation of this controller in the laboratory.

5.3 Laboratory Implementation: Experimental Design and Setup

In this section, we report results obtained by implementing the intensity controller, developed in the previous sections using numerical simulations, in the laboratory. We have tested the controller using two different types of particles, a) 350 nm diameter glass nanoparticles and b) 100 nm gold nanoparticles. We implement the controller algorithms digitally using an FPGA instead of relying on analog electronics that are commonly used in optical trapping experiments. In the next section, we describe our FPGA implementation, followed by controller performance results for each of the nanoparticle cases.

5.3.1 Intensity Control on Field Programmable Gate Arrays

Control systems are used extensively in optical trapping experiments, as we have already discussed in Section 2.5. However most experiments use analog PID controllers to enhance the stiffness of the optical trap, when trapping particles that are much larger than one micrometer. While analog controllers are adequate for PID control, their utility in implementing a generalized control function is more limited. Moreover, microprocessor based digital signal processors (DSP), which have long served as an alternative to analog control often require significant setup costs.

More recently, open source projects that rely on low cost FPGAs have started to gain popularity. One example is the GNU Radio project, which is a software defined radio that can be run on low cost FPGAs such as the Ettus Research Universal Software Radio Peripheral (USRP) [126, 94]. FPGAs are essentially user programmable integrated circuits that consist of several logic blocks that can be wired together using programmable interconnects to create complex functionality and in some cases even emulate microprocessors. Furthermore, most modern FPGAs include memory elements within individual logic blocks which provide increased flexibility when designing algorithms that run on these architectures. FPGAs are inherently parallel architectures where individual logic blocks are executed simultaneously and therefore have higher throughputs than comparable DSP solutions [78]. Furthermore since inputs and outputs are directly connected to the FPGA rather than sharing a common bus with other I/O devices, FPGAs typically have very low latencies. FPGAs have seen extensive use in telecommunications, integrated circuit design and software design radio. Recently the USRP, which is designed for software defined radio applications has been adapted to control a magnetic resonance force microscope [78]. The controller was implemented as a two stage cascaded filter that operates at $500\,kHz$ and used to control a cantilever with a resonant frequency of $8\,kHz$.

We have implemented a XYZ intensity controller, discussed previously in Section 5.2.3, in the laboratory using a USRP. The USRP contains an Altera Cyclone EP1C FPGA [41], the requisite I/O cards (LFRX/LFTX) that allow two analog channels per card in a frequency range of $DC - 30\,MHz$ [126, 94]. The input of each I/O card is then digitized by an Analog Devices AD9862 codec that contains two analog to digital converters (ADC). The Analog Devices AD9862 codec also contains two digital to analog converters (DAC) to convert the output of the FPGA to an analog signal [77]. The EP1C FPGA contains 12 060 logic blocks and 239 616 bits of memory. The main board of the USRP supports two receive (LFRX) and two transmit (LFTX) daughter boards, each with two channels, thereby supporting a total of four inputs and four outputs. The ADCs on the receive side sample the input signals at $64\,MHz$ with a 12-bit resolution. Furthermore, the output wires of the four ADCs in the two AD9862 codecs are connected directly to the input pins of the FPGA. On the other hand the four DACs, which run at 128 MHz with 14-bit resolution are connected to the FPGA using two sets of 14 wires. Therefore in order to use all four DACs simultaneously, the output signals must first be multiplexed (MUX) on the FPGA. Since we only have one output signal for intensity control, we are able to directly interface with the DAC without first resorting to multiplexing.

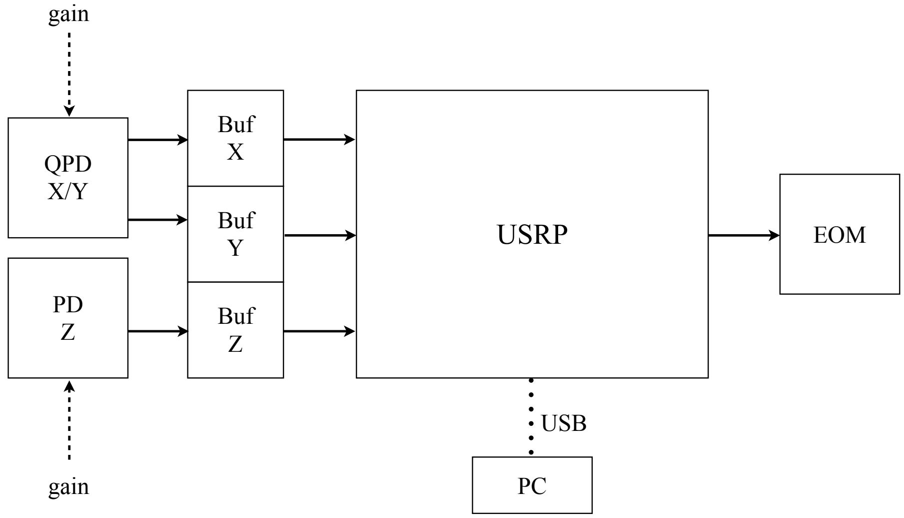


Figure 5.8: Schematic layout of the instrumentation used in implementing the FPGA based intensity controller.

Figure 5.8 shows the schematic layout of the instrumentation used to implement the intensity controller in the laboratory. The solid arrows in the figure represent analog signals between the individual components with the arrow head indicating the direction of flow, while the dashed lines represent instrumentation parameters that are user selectable. The USRP is setup and run using a computer over a universal serial bus (USB) connection shown by the dotted line in the figure. The software required to run the USRP consists of Verilog hardware description language (HDL) code that is downloaded to the USRP at startup and contains the actual control algorithms, and python code that is used to setup a range of system parameters including FPGA registers and registers that control the AD9862 codec.

The position of the nanoparticle within the optical trap is measured using back focal plane detection as described in Appendix A. The position along the X and Y axes is measured using a QPD (2901, New Focus) with a maximum bandwidth of $100\,kHz$, while the position along the Z axis is measured using a photo-diode (2032, New Focus) with a bandwidth of $150\,kHz$. The inputs to the USRP daughterboards have a unity gain and an input impedance of $50\,\Omega$ whereas, the output impedance of the detectors is $1\,k\Omega$. Therefore the output of the QPD and PD are connected to the USRP through voltage buffers to minimize loading the outputs of the detectors. The voltage buffers (SIM910, Stanford Research Systems) have user selectable gains that range from unity to 100 that are not shown in the figure. The output of the voltage buffers are then connected to the USRP, which uses the input positions to calculate an appropriate intensity control signal. This control signal is then used to drive the electro-optic modulator, which changes the intensity of the trapping beam and has a maximum bandwidth of $200\,kHz$.

The control algorithms implemented on the USRP calculate a control signal using the X,Y and Z position of the particle. Figure 5.9 shows the layout of this algorithm, which is based on blanking control in X and Y and proportional control in Z. The receive and transmit daughter cards, which are not programmable are not shown in the figure. In the figure, the thick solid lines represent analog data, while digital data that flows between the different algorithmic blocks is shown by thinner

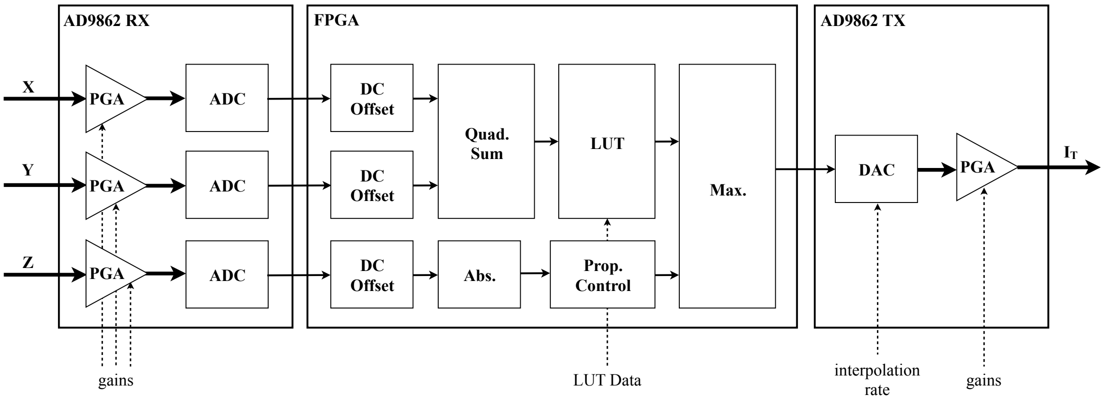


Figure 5.9: Control signal paths in the FPGA are shown. The solid lines show data flow between the controller logic blocks implemented on the FPGA, while the thick lines represent analog inputs and outputs. The dashed lines represent parameters that are provided either at runtime or during synthesis of the Verilog code.

solid lines. Parameters required at startup or during the synthesis of the Verilog HDL code are shown by dashed lines. The AD9862 codec allows input and output signals with an amplitude of $2\,V_{pp}$. For weak input signals, there is a programmable gain amplifier (PGA) in each channel that can be configured to amplify the input signal to use the full dynamic range supported by the ADC. In our experiment, we set the PGA gains for each input channel to unity. The X, Y and Z position signals are then digitized simultaneously by three ADCs at $64\,MHz$ using a signed 12-bit integer representation.

As seen in Figure 5.9, the digitized signals from the ADC first undergo an offset removal step in the FPGA to remove a small voltage offset introduced by the LFRX daughter boards. The offset removal is implemented in two stages. First, the input signal is decimated in time slices of $262\,ms$, which corresponds to approximately 2^{24} samples for the $64\,MHz$ FPGA clock. The offset is then calculated by further smoothing the data by calculating a moving average of the last eight decimated data points and subtracting this value from the input signal. Therefore, the offset correction routine effectively updates the offset approximately every 2 s.

After the offset removal step, the X and Y positions of the nanoparticle are summed in quadrature to obtain the radial coordinate, r. The control signal for the X and Y positions is then calculated using a lookup table (LUT) that returns the trap intensity as a function of r. Utilizing a LUT where the control signal is calculated ahead of time significantly speeds up the implementation of the controller in the laboratory since we are able to rapidly test new controller designs in a very short period of time. In our experiment, the LUT simply holds the tabulated control function given by Equation 5.8 for our XY blanking controller. This control function is pre-calculated and incorporated into the controller code during the synthesis of the Verilog program. The table is implemented as a read only memory (ROM) function, with a 12-bit address field and contains 4096 points to correspond with the resolution of the input signals. The DACs in the AD9862 codec support a 14-bit signal on the transmit side. In order to used the full range of the output signal, we set the data field in the LUT to 14-bits. The LUT is implemented using an Altera ROM mega-function [41], which combines memory elements from several logic blocks to create a single large memory space. The LUT for our application uses 57 344 bits of data or approximately 24% of the total memory on the FPGA.

The Z-position of the particle is processed by a different set of blocks which implement a simple proportional controller using Equation 5.9. The output of the DC offset step, for the particle's Z-position, is first input to the *Abs.* block which returns the absolute value of the Z-coordinate ($|z|$). The control signal for the Z-intensity is calculated by simply multiplying the absolute Z-coordinate with the controller proportional gain (κ_p) within the *Prop. Control* block. The final control signal is then calculated by taking the maximum intensity value between the output of the XY blanking controller and the Z proportional controller as seen from Equation 5.10.

A principal advantage of using the FPGA for control systems is that all of the blocks in Figure 5.9 are executed in parallel, which results in a large throughput. Moreover the latency across the system is very low. We have measured a latency as low as $400\,ns$ in some of our tests, which corresponds to a bandwidth of $2\,MHz$, an order of magnitude larger than our instrumentation bandwidth. In the next section, we present laboratory results from FPGA-based intensity control.

5.4 Laboratory Results

In Section 5.2.3, we described a design for an intensity control algorithm that was tested using numerical simulations. Here, we report the successful implementation of this control design in the laboratory using FPGAs. We have tested our algorithms using two different model systems: i) $350\,nm$ diameter glass nanoparticles and ii) $100\,nm$ diameter gold nanoparticles. In both cases, we show that under controlled trapping the lifetime of the nanoparticles are significantly enhanced, highlighting that this technique is applicable to particles made from dielectrics as well as metals. Moreover, for average powers that are comparable to those required by the controller to keep a particle trapped, a corresponding static trap was unable to localize nanoparticles.

The controller performance was experimentally verified in the laboratory using the optical tweezers instrument, described in Appendix A. For each particle size, we prepared a dilute suspension of nanoparticles in de-ionized water. A sample cell, approximately $100\,\mu m$ thick, was prepared using a microscope slide and cover glass and sealed on all sides using double sided adhesive tape. Furthermore, the nanoparticle was always trapped and lifted several micrometers above (along the positive Z-axis) the bottom surface of the sample cell to reduce surface effects in the viscous drag, as discussed in Chapter 2. Next, we describe the controller results for each particle size, starting with the $350\,nm$ glass nanoparticles. For each case we first describe the controller parameters chosen, followed by performance results.

5.4.1 350nm Silica Nanoparticles

In this section, we report significantly enhanced trapping lifetime, in comparison with a static trap, when using intensity control on $350\,nm$ glass nanoparticles (Bangs Laboratories, P/N: SS02N). The parameters for the intensity controller used in this experiment are described in Table 5.3. The notation of the parameters in the table is identical to that used in Section 5.2.3 and λ is the wavelength of the trapping laser.

The intensity control algorithms implemented in the laboratory consist of two parts, a XY blanking controller and a proportional controller that depends only on the Z-position of the nanoparticle in the trap. In Table 5.3, the parameters r_b and r_b' define the blanking region and the transition region, where the controller

Table 5.3: 350 nm Glass Spheres: Control Parameters

Control Paramters	Parameter Values
Diameter (nm)	350
Material	SiO_2 (glass)
λ (nm)	1064
r_b (nm)	250
r'_b (nm)	260
P_{min} (mW)	0
P_{max} (mW)	100
κ_p (mW/μm)	70

increases the trap power from a minimum (P_{min}) to a maximum value (P_{max}). For maximum impact, we select r_b to be close to the maximum force point of the optical trap. Therefore, we first determine the maximum force points in the transverse XY-plane using the step input method described previously in Section 4.2. The optical trapping force is not perfectly symmetric along the X and Y axes due to the polarization of the incident light, as well as misalignments in the optics. However, in our experiments, we ignore these differences and simplify the controller design by selecting the location of the maximum force point along the Y-axis to define the blanking region.

$$\vec{\Phi}(\vec{v}) = \{\hat{x}\, 10\, v_x,\ \hat{y}\, 10\, v_y,\ \hat{z}\, (27.9\, v_z\ +\ 47.7\, v_z^2)\} \tag{5.11}$$

In order to estimate the trapping force acting on the nanoparticle, we first calibrate the detectors that measure the nanoparticle position using 350 nm glass spheres immobilized in gelatin as outlined in Section B.2. Furthermore, we perform the calibration independently, assuming no coupling between the axes. The resulting calibration function, which accepts the detector output in volts and returns the particle position in μm, is given in Equation 5.11. However, this calibration function is only valid over a range of ±1 μm along the X and Y axes and $-1\,\mu m$ to $+2\,\mu m$ along the Z-axis.

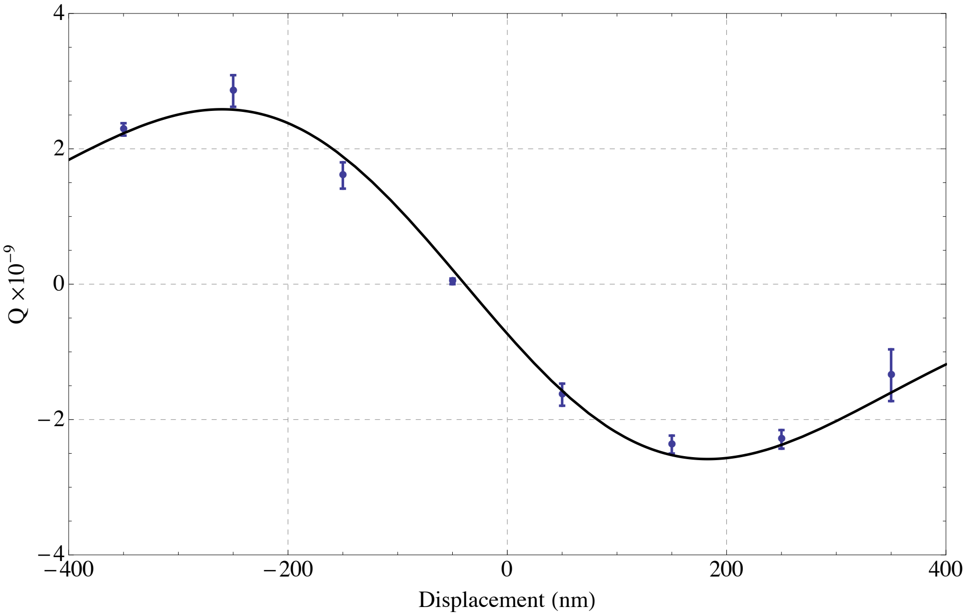


Figure 5.10: Laboratory measurement of the trapping force using the step input method for $350nm$ SiO_2 nanoparticles. The distance of the maximum force point was measured to be $236 \pm 18\,nm$ from the center and used in setting the controller parameters.

Once the calibration is complete, the trapped nanoparticle is scanned using a $1\,Hz$ square wave input, with a peak-to-peak amplitude of $350\,nm$, which represents the distance of the particle from the center of the trap. The position response of the particle to the square wave input is recorded using the QPD with a sampling frequency of $40\,kHz$. The measurement is repeated 150 times and then averaged. We then use the analysis techniques detailed in Section 4.2 to calculate the resultant trapping force profile. Furthermore, in order to eliminate variations in the trap power between measurements, we normalize the force curves for each measurement by the power recorded during the measurement. The resulting plot, which is independent of trap power is shown in Figure 5.10. The error bars in the plot are the standard error of the measurement. As we can see from the figure, we are able to recover the trapping force well beyond the maximum force point of the trap. As before (see Section 4.2), we estimate the location of the maximum force points by fitting the force data to the derivative of a Gaussian function using a least squares estimator, which gives the location of the maximum force point at $236 \pm 18\,nm$. The size of the blanking region impacts the average power used by the controller. Therefore, we tune the size of the blanking region and obtained better performance by slightly increasing the size of the blanking region (r_b) to $250\,nm$ with an additional transition region of $10\,nm$ so that $r_b' = 260\,nm$. Moreover, we setup the trap power to have a minimum value of $P_{min} = 0\,mW$ inside the blanking region and a linear transition to $P_{max} = 100\,mW$ when the particle is outside the transition region. Furthermore, we have tuned the gain for the Z-intensity controller, $\kappa_p = 70\,mW/\mu m$ to maximize the lifetime of the nanoparticle in the trap. With this gain, the controller attains it's maximum power when $Z = \pm 1.4\,\mu m$ from the center of the trap.

With the parameters listed in Table 5.3, the intensity controller significantly improves the lifetime of a nanoparticle in the trap. This is seen clearly in Figure 5.11, which shows a contour plot of the particle positions in the trap in the XZ-plane for the two cases when the controller is *on* and when the controller is *off*. The left side of the figure, which shows the controller *on* case, is calculated using the position trajectories of 10 particles. Each trajectory contains $10\,s$ of data recorded at $20\,kHz$. The contours are generated by aggregating all the trajectories and then

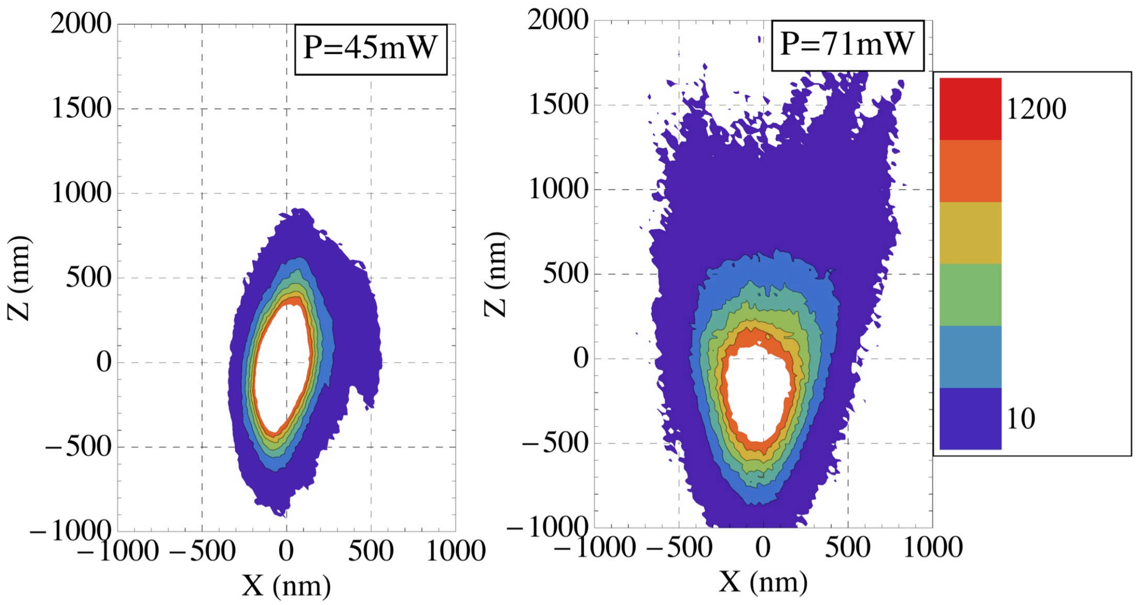


Figure 5.11: Contour plots of controlled trapping vs. a static optical trap for 350nm SiO_2 nanoparticles in the XZ-plane. The results in the YZ-plane are similar and are not shown here. The particle remains confined to the trap when the controller is *on*, but escapes for a static trap.

calculating a histogram along the X and Z axes, with a uniform bin size of 25 nm. The contour lines in the plot therefore trace regions with the same particle counts in individual bins. Due to the symmetry of the system, the corresponding plot in the YZ-plane is not shown. The contours lines represent particle counts that range from 10 counts to 1200 counts in five steps. Bins with more than 1200 particle counts are shown in white. The left side of Figure 5.11 also shows labels for the counts represented by each contour line and these values are common to both plots.

As we see from the left side of Figure 5.11, when the controller is *on* the particles stay localized close to the center of the trap. We estimate the 2σ value of the particle distribution along the X-axis to be 235 nm, where σ is the standard deviation. Therefore 95% of all particles lie inside the blanking region (r_b), which was set to 250 nm. Moreover we see no particle counts outside of $Z = \pm1000\,nm$ along the longitudinal axis. Finally, the controller is able to keep the particle localized to the trap by using an average power of $45\,mW$.

The contour plot for a static trap is shown on the right hand side of Figure 5.11. We see that for a static trap with average power of $71\,mW$, the contour plot looks qualitatively different than the static trap case. The particle density is broader along both the transverse and longitudinal axes. Moreover, as with simulation results, we see particles escaping along the positive Z-axis, as seen from the right side of Figure 5.11. When particles escape the trap, they very quickly leave the volume that defines the detectors operating range. This accounts for the relatively low density of particles above $Z = +1000\,nm$. In contrast, since particles spend more time close to the center of the trap, we see a much higher density around the origin of the plot. Furthermore, in laboratory experiments, we were unable to confine a particle to the trap with a static trap at $71\,mW$.

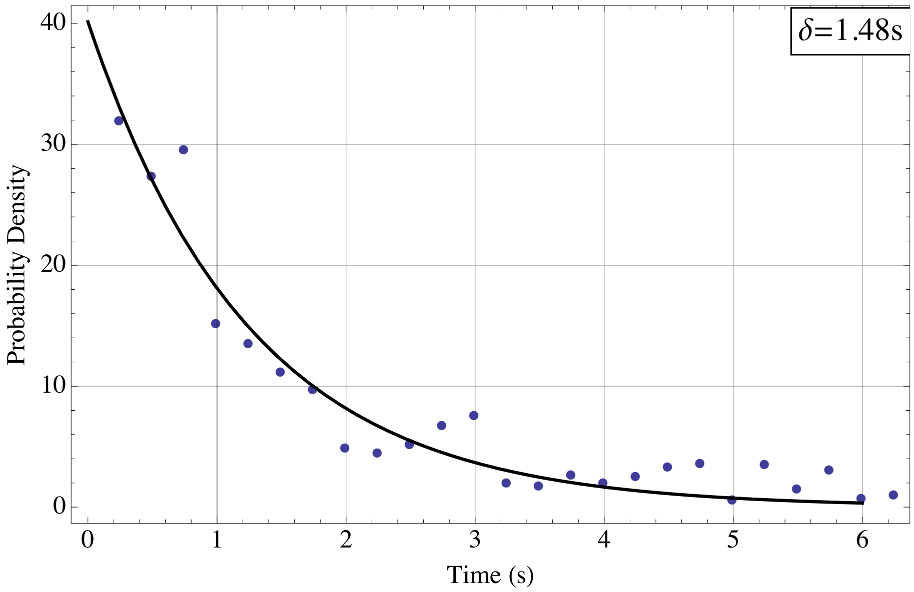


Figure 5.12: Laboratory measurement of lifetime for a $350nm$ SiO_2 nanoparticle in a static trap.

We quantify the performance of the static trap by calculating the lifetime of a 350 nm glass nanoparticle using the method described in Section 5.2.2. We have found that most particles for the static trap pass through $Z = 1000\,nm$ before they exit the trap. Therefore we calculate the first crossing times for each trajectory using a bounding box that has an edge of 1000 nm on all sides. As seen in Section 5.2.2 before, the peak of the probability density function of particles in the trap, in a quasi-steady state for times longer than the time required to fill the trap volume, is then plotted as a function of time in Figure 5.12. Finally, we estimate the lifetime of the nanoparticle to be approximately 1.5 s by fitting the data in the figure to Equation 5.4. Furthermore, this value of the calculated lifetime is consistent with our observations, where the particle would exit the trap within $1 - 2\,s$.

When using the same bounding box values as the static trap, the above technique to calculate lifetime does not work very well when applied to data recorded when the controller is *on*. As we clearly see from the left hand side of Figure 5.11, all of the particle counts are well within the region enclosed by a bounding box that is 1000 nm on all sides. However, as discussed in Section 5.2.2.1, we can place a lower bound on the lifetime. We estimate the lower bound on lifetime for 10 measured trajectories, each 10 s long where we observe *zero* escape events to be approximately 33 s with 95% confidence. Moreover this represents at least a 22 times increase over the static trap lifetime and is sufficient time to allow this technique to be useful in nanoassembly applications. Moreover, the controller is able to achieve this lifetime improvement over the static trap when using only 63% of the power. The results

Table 5.4: Lifetime and power comparison: laboratory measurements, 350 nm glass nanoparticles

	Controlled Trap	Static Trap
Number of observations	10	
Observation Period (s)	10	
Lifetime (s)	> 33 ($\alpha = 0.05$)	1.5
Lifetime Improvement	> 22	–
Avg. Power (mW)	45	71
Power Improvement	0.63	N/A

for the controlled and static trap results are summarized in Table 5.4.

5.4.2 100nm Gold Nanoparticles

Table 5.5: 100 nm Gold Spheres: Control Parameters

Control Paramters	Parameter Values
Diameter (nm)	100
Material	Au
λ (nm)	1064
r_b (nm)	100
r'_b (nm)	100
P_{min} (mW)	0
P_{max} (mW)	250
κ_p (mW/μm)	550

We have also tested the intensity controller described in Section 5.2.3 on gold particles with a diameter of 100 nm (Ted Pella, P/N: 15711-20). The controller parameters, which have the same notation as before are described in Table 5.5. For the gold nanoparticles, we set the blanking region to 100 nm and set no transition region. The controller then increases the power from 0 mW (P_{min}) inside the blanking region to 250 mW (P_{max}) when the particle is at r_b. Furthermore, we set the gain for the Z-intensity controller to 550 $mW/\mu m$ so that the Z-intensity controller saturates approximately 450 nm from the center of the trap. The average power required to confine a particle increases with particle size. Therefore the values of the maximum power and the gain of the Z proportional controller are accordingly increased from the values used for the 350 nm glass nanoparticles. However, as we see in this section, the controller is able to successfully confine a nanoparticle to the trap, when a static trap at the same power is unable to do so.

$$\vec{\Phi}(\vec{v}) = \{\hat{x}\,(2.8\,v_x - 0.8\,v_x^2),\ \hat{y}\,(2.8\,v_y - 0.8\,v_y^2),\ \hat{z}\,(4.72\,v_z\ -\ 2.72\,v_z^2)\} \tag{5.12}$$

We first calibrate the detectors using 100 nm gold nanoparticles immobilized in gelatin using the techniques described in Section B.2. The resulting calibration function, which accepts the detector output in volts and returns the particle position in micrometers, is given by Equation 5.12. Moreover the calibration function shown in the equation is only valid over a range of ±750 nm along the X and Y axes and $-1\,\mu m$ to $+2\,\mu m$ along the Z-axis.

With the parameters listed in Table 5.5, the intensity controller significantly improves the lifetime of a 100 nm gold nanoparticle in the trap. This is clearly seen in Figure 5.13, which shows a contour plot of the particle positions in the trap in the XZ-plane when the two cases when the controller is *on* and when the controller is *off*. The contours in this plot are generated exactly as before with the 350 nm glass nanoparticles. The contours in each plot are generated with the position trajectories of 25 particles, each containing 5 s of data recorded at 20 kHz. The contours are generated using a histogram of the aggregated trajectories with a uniform bin size of 25 nm along each axis. The contour lines in the plots therefore trace regions with

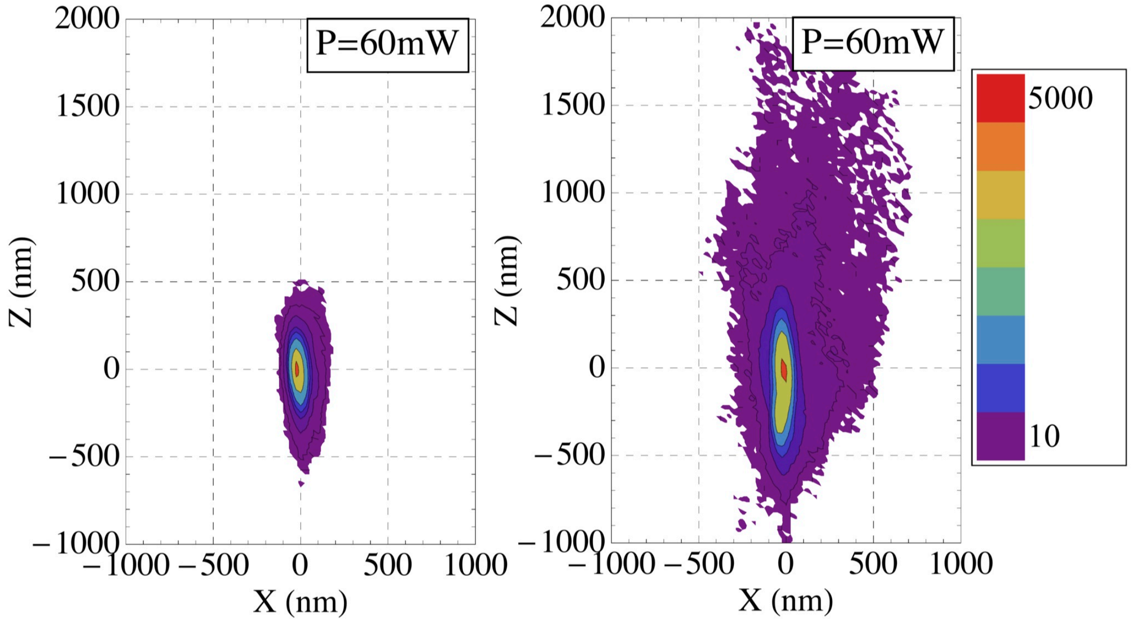


Figure 5.13: Contour plots of controlled trapping vs. a static optical trap for 100nm gold nanoparticles in the XZ-plane. The results in the YZ-plane are similar and are not shown here. The particle remains confined to the trap when the controller is *on*, but escapes for a static trap.

the same particle counts in individual bins. Six contours are shown in each plot that range in particle counts from 10 to over 5000. The right side of Figure 5.13 also shows labels for the counts represented by each contour line that are common to both plots.

As we see from the left side of Figure 5.13, when the controller is *on* the particles stay localized close to the center of the trap. We estimate the 2σ value of the particle distribution along the X-axis to be $90\,nm$, where σ is the standard deviation. Therefore 95% of all particles lie inside the blanking region (r_b), which was set to $100\,nm$. Moreover we see almost no particle counts outside of $Z = \pm 600\,nm$ along the longitudinal axis. Finally, the controller is able to keep the particle localized to the trap by using an average power of $60\,mW$.

The contour plot for a static trap is shown on the right hand side of Figure 5.13. We see that for a static trap with average power of $60\,mW$, the particle density is broader along both the transverse and longitudinal axes. Moreover, as with simulation results and laboratory results with $350\,nm$ glass nanoparticles, we see particles escaping along the positive Z-axis.

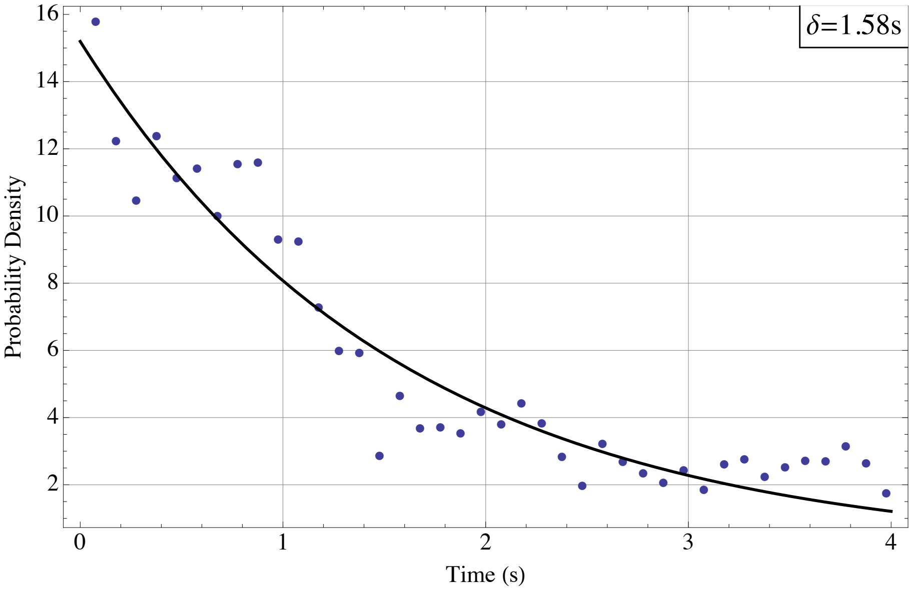


Figure 5.14: Laboratory measurement of lifetime for a $100nm$ Au nanoparticle in a static trap.

We have quantified the performance of the static trap by calculating the particle's lifetime, as we did before with the larger glass nanoparticles. The peak of the PDF as a function of time, for times longer than the time required by the particles to fill the trap, is shown in Figure 5.14. The particle's lifetime is then estimated to be approximately $1.6\,s$ by fitting the data in the plot to Equation 5.4. Furthermore, we are unable to effectively quantify the lifetime of the $100\,nm$ particle under controlled trapping, for reasons discussed previously. However, using techniques outlined in

Section 5.2.2.1, we estimate the lower bound on lifetime for 25 measured trajectories, each $5\,s$ long where we observe *zero* escape events, with 95% confidence to be approximately $42\,s$. Moreover, this represents a 26 times improvement over the static trap value for the same average power. The performance results for $100\,nm$ gold nanoparticles are summarized in Table 5.6.

Table 5.6: Lifetime and power comparison: laboratory measurements, $100\,nm$ gold nanoparticles

	Controlled Trap	Static Trap
Number of observations	25	
Observation Period (s)	5	
Lifetime (s)	> 42 $(\alpha = 0.05)$	1.6
Lifetime Improvement	> 26	–
Avg. Power (mW)	60	60

5.5 Summary

We have successfully demonstrated a framework for controlled optical trapping in the laboratory using $350\,nm$ glass nanoparticles and $100\,nm$ gold nanoparticles. We have used numerical simulations to develop a representative control algorithm that significantly improves the lifetime of the nanoparticle without a corresponding increase in the average incident power. By implementing the physically realistic GLMT force model in our simulations, we show that the scattering component of the total trapping force plays a significant role in the escape mechanism of nanoparticles. The preferential longitudinal escape of particles, in the direction of propagation of the incident light greatly impacts controller design. As a result, we show that scan control, which only acts in the transverse XY-plane has minimum impact in improving the particle's lifetime.

Using numerical simulations, we developed an intensity controller that blanks the beam, as a function of the particle's X and Y position, near the center of the trap. Moreover as the particle leaves a pre-defined blanking region, the controller quickly increases the power to it's maximum value. However, this controller was found to be ineffectual without the inclusion of the particle's Z-position. Therefore, we developed our prototype controller design, which includes blanking control along the X and Y axes and a proportional intensity controller along the Z axis. This controller was then able to significantly improve lifetime with trap power comparable to a weak static trap with a very low lifetime.

We have successfully implemented this intensity controller in the laboratory using $350\,nm$ glass nanoparticles and $100\,nm$ gold nanoparticles. Instead of designing an analog controller, we implemented the controller algorithms using an open-source and low cost FPGA. Parts of the controller are implemented using lookup tables to allow the controller to be easily modified. For the $350\,nm$ glass nanoparticles, this FPGA-based controller implementation has led to at least a 22 times improvement in lifetime (with 95% confidence) over a static trap and using approximately 40% less power. Likewise, for $100\,nm$ gold nanoparticles, we have demonstrated at least a 26 times increase in the lifetime (with 95% confidence) without a corresponding increase in power.

The successful implementation of controlled trapping in the laboratory should greatly help in advancing the use of optical tweezers in nanomanipulation and eventually nanoassembly applications. Moreover, since manipulating nanoscale objects using this technique does not result in a significant increase in the average incident power, this method is also beneficial for biophysical applications. Furthermore, we believe that including control systems in optical tweezers can result in a directed assembly tool capable of realizing fully three-dimensional nanoscale prototypes of functional devices.

Chapter 6

Conclusions

Our ability to confine nanoparticles to an optical trap for sufficient time for use in assembling practical devices is currently severely limited. One problem that prevents optical trapping from being an effective nanomanipulation tool is the weak interactions between nanoparticles and an optical trap. While the trapping time of nanoparticles can be increased simply by increasing the power of the optical trap, doing so often results in significant heating of the surrounding fluid or the particle, which can either destabilize the trap or destroy the particle. Therefore, we have developed and implemented new methods that form a basic tool set that can be used to extend optical trapping to the nanoscale. One component of this tool set is new control algorithms that significantly extend the trapping time of nanoparticles using sufficiently low average power, so that they can be effectively used in nanoassembly operations. Moreover, a static trap set the same power as the controller is unable to confine a same-sized particle.

Controlled optical trapping is an important component of the tool set, which can overcome some of the challenges facing optical trapping. The controller is able to alter two trap parameters, it's position and power of the incident beam, and control algorithms are realized by combining these parameters. We have demonstrated that controllers that modify only the intensity of the trap are effective in improving the confinement of nanoparticles. Furthermore, we have implemented controllers in the laboratory and successfully confined particles made from multiple materials, such as 100 nm gold and 350 nm glass nanoparticles, as discussed in Chapter 5.

Developing new control algorithms directly in the laboratory is challenging and often very time consuming. Therefore, physically accurate simulations, developed in Chapter 3, play a key role in the discovery of control algorithms. The simulation algorithms developed here are not limited only to the development of control algorithms, but are a more general tool for use in numerous applications of optical trapping such as developing novel assembly techniques for practical devices or operator training, that are not developed as part of this work.

The final component in the toolbox is an accurate model of the trapping force exerted by an optical trap on a nanoparticle. While a few theoretical trapping force models exist, none of them accurately model tightly focussed beams commonly used in trapping experiments or work for a range of materials. Consequently, there is very poor agreement between existing theoretical models and experimental measurements of the trapping force, even for larger particles. Another approach is to directly measure the optical trapping force in the laboratory and include this force data as input to the simulation model. However, existing force measurement techniques, developed for large particles, do not work well for weakly bound nanoparticles. Therefore, we have developed two new force measurement techniques in Chapter 4 to characterize the optical trapping force acting on nanoparticles within an optical trap. These methods are then tested using numerical simulations prior to being implemented in the laboratory.

6.1 Intellectual Contributions

Important contributions that will eventually improve the field of optical trapping as a result of the methods and algorithms developed are described below.

(i) *New techniques for nanomanipulation with optical tweezers:* The ability to confine nanoparticles to an optical trap for sufficient time for use in assembling practical nanodevices is greatly limited and is currently a central challenge in extending optical trapping to nanoscale assembly. We have developed and implemented new control systems in the laboratory that significantly improve the trapping time of nanoparticles in a trap without a corresponding increase in the average power, as described in Chapter 5. These new techniques form an important first step in achieving routine manipulation of nanoscale particles. We have developed and carefully tested a Brownian dynamics simulation framework that we have used in the discovery of new algorithms and tools. Furthermore, these new techniques will allow us to extend the flexibility of pick and place assembly using optical tweezers, already demonstrated for microscale particles, to the nanoscale.

(ii) *New characterization and manipulation tools through accurate simulations:* Accurate numerical simulations, like those discussed in Chapter 3, have played a crucial role in extending optical trapping to nanoscale manipulation. For example, in algorithms developed in Chapter 5, laboratory experiments to quantify system parameters can be difficult and time consuming. We have successfully used numerical simulations to overcome this challenge and provide an efficient route to the discovery of novel control algorithms. Furthermore, the flexibility and modular nature of our simulation framework can be widely useful to the field of optical trapping and assist in several applications, such as developing novel assembly techniques for practical devices or operator training.

(iii) *New force measurements for nanoparticles in an optical trap:* Current characterization tools for optical traps, which are developed for large microscale particles are often glaringly inadequate at the nanoscale due to the weak interactions of nanoscale particles with the trap. Force measurements are an important when quantifying the performance of optical traps. Therefore the new force measurement techniques, developed as part of this work, fill an important gap in our current understanding of the interactions of nanoparticles with an optical trap and provide a foundation for nanomanipulation. We have developed two new force measurements for weakly bound particles in Chapter 4 and tested them in the laboratory.

6.2 Benefits

The tools developed as part of this work have been used in concert to develop new methods that significantly improve the lifetime of nanoparticles in an optical trap. Moreover each of the tools developed as part of this work have several benefits, even outside the scope of the current research, as described below.

6.2.1 Physically Accurate Simulations

Physically accurate simulations provide a versatile platform for the discovery of new algorithms to extend optical trapping to the nanoscale. As part of this work, we have successfully used numerical simulations to develop new characterization tools such as force measurements that work for a wide range of particle sizes down to nanoscale particles. Furthermore, we have also used physically accurate simulations in conjunction with realistic trapping force fields to develop control algorithms that are able to significantly improve the lifetime of nanoparticles without also increasing the average input power. Since simulations allow us to quickly change the system parameters to test new designs, they have proven invaluable when developing new control algorithms as part of this work.

Numerical simulations also have other important uses, outside the scope of this work, which make them an important tool for optical trapping. Simulations can be run in a batch mode to test new assembly algorithms. Often, assembly operations must be repeated several times to collect statistical information about the process. This may not always be practical or feasible to perform in the laboratory. However, such tasks can be easily performed using simulations to obtain statistical information about the repeatability and reliability of the process. In some cases simulations may be able to quantify trapping parameters that cannot be easily measured in the laboratory, for example models for corrections to the bulk viscosity of the fluid due to local temperature gradients near the particle in the trap, which can significantly change trapping behavior. In these situations, numerical simulations combined with limited experimental data can be used to provide quantitive information that improves the performance of an optical trap. Numerical simulation algorithms, designed to run in real-time can also be included within a closed-loop controller for feed forward or model in the loop operation. Similarly, real-time numerical simulations can be used for operator training, where the simulation completely replaces the physical optical tweezers instrument.

6.2.2 Novel Force Measurement Techniques

Accurate measurements of the trapping force exerted by an optical trap are an important component of the characterization tools necessary to quantify trapping behavior and also for use in controlled trapping algorithms developed as part of this work. Existing theoretical models fall short of adequately capturing the behavior of traps created from tightly focussed beams that are commonly used in optical trapping experiments. Moreover, microscope objectives used to form optical traps can suffer from spherical aberrations, as well as diffraction effects due to the finite size of their back apertures, effects that are not addressed by existing theoretical models.

We have developed two new force measurement techniques that make no assumptions about the strength of the traps. Moreover since these are laboratory-based methods, they naturally include deviations from non-ideal behavior of the optics into the measured force data. Therefore these methods have significant value as a characterization tool for optical trapping. The measured trapping force yields insight into the performance of the optical trap. For example, the magnitude of the measured force, for a given particle size and trap power, provides information about the efficiency of the trap. Furthermore, the symmetry of the force data yields information about the alignment of the optics. Trapping force data measured in the laboratory can also be included in the simulations (described in Appendix C) and is useful in numerous applications where simulations are used to enhance optical trapping, such as the new control algorithms described next.

6.2.3 Enhanced Optical Trapping Using Control Systems

We have developed new methods that allow us to confine nanoparticles to a trap for sufficiently long periods of time, so they can be used in nanoassembly applications, when a static trap at the same power is not able to do so. Nanoparticles approximately $20\,nm$ in diameter have been confined to the trap for very short periods of time and using up to $1\,W$ of power (which corresponds to thousands of mega watts per square centimeter for diffraction limited trap), while particles below $20\,nm$ have not been trapped successfully when using even higher powers [72]. However, we believe that controlled trapping will allow us to trap the smallest nanoparticles, well below the $100\,nm$ particles demonstrated here, using significantly lower power than is currently possible. The methods developed as part of this work are therefore a first step towards achieving this goal.

As part of this work, we have also shown that the controller is able to dramatically reduce the power required to trap a nanoparticle, by as much as 40%

in some of our experiments. This power reduction has direct consequences for the power absorbed by the particle, as well as the power absorbed by the surrounding fluid. High incident powers can cause significant heating of a nanoparticle, which can result in thermal damage. Moreover the power absorbed by the fluid can significantly increase its temperature, causing destabilizing convective forces and also local viscosity changes which ultimately reduce the performance of the trap. Biophysical experiments are particularly sensitive to large temperature gradients, which can permanently damage the specimen in the trap rendering it unusable. Therefore, we believe that controlled trapping will particularly be useful to facilitate the manipulation of small biological nanoparticles using low trap powers.

6.3 Limitations and Future Work

The methods and techniques described in this research are key components in the tool set required to eventually realize nanomanipulation with optical tweezers. However, much work is required to eventually realize a flexible nanomanufacturing system that can quickly prototype nanoscale devices.

(i) *Physically accurate simulations:* Physically accurate simulations developed as part of this work have core modules required to develop and test nanomanipulation methods. However in order to continue to be useful for nanoassembly tasks, simulations must include new models that are not currently implemented as part of the framework. For example, the simulation framework must include models of electrostatic interactions between two or more individual components or between components and a substrate on to which they are assembled. The modular framework of the simulation allows for multiple models to be included and therefore the models for electrostatic interactions may be simple approximate models that define a sticking probability between components or be more complete models of the zeta-potential, depending on the specific application. Hydrodynamic interactions between components can also be developed, however these models may be less important when working in liquids with high viscosity, where the effects from fluid inertia is sufficiently small. Lastly, a scalable nanomanufacturing system must be sufficiently automated so that we can quickly prototype new devices. Simulations can play an important role in developing new automation tools. Automation algorithms can be easily accommodated in the outer loop of our simulation framework as described in Appendix C.

(ii) *Improved nanoscale characterization tools:* New force measurements developed as part of this work are an important first step towards developing a full suite of characterization tools for nanoparticles in an optical trap. The force measurements can be improved by developing algorithms that use image processing to automatically trap a nanoparticle, run a force calibration and return parameters of the trap that can be use to assess performance or obtain information about the alignment. However, the set of characterization tools is not limited to force measurement techniques, but must include routines that estimate particle size and shape to allow automation algorithms to quickly select suitable particles for assembly. Methods to estimate physical quantities such as the diffusion constant must also be developed, which can assist in particle sizing or be used to estimate the local viscosity of the fluid, if the particle size and shape are well known.

(iii) *Enhanced trapping algorithms for nanomanipulation:* Techniques to improve the lifetime of nanoparticles using control system show tremendous promise for use in a optical tweezers-based prototyping system. Prototypical control algorithms developed as part of this work have successfully demonstrated the feasibility of improving the lifetime of nanoparticles in the laboratory. However, these methods must be extended and optimized to provide the best performance in the laboratory. This will allow controlled trapping techniques to be successfully extended to particles smaller than $20\,nm$. Furthermore, controlled trapping must also be extended to work with multiple particles simultaneously and also to non-spherical particles such as spheroidal particles and even nanowires, which are essential building blocks for nanoscale devices. Another crucial area that must be developed for effective prototyping with optical tweezers is new automation algorithms. A flexible system must include automation to reduce operator load and improve the throughput of the system. Moreover, the enhanced methods required to create a flexible nanomanufacturing system will require the use of all the building blocks developed as part of this work to be tightly coupled and work together cohesively.

Appendix A

Hardware Instrumentation

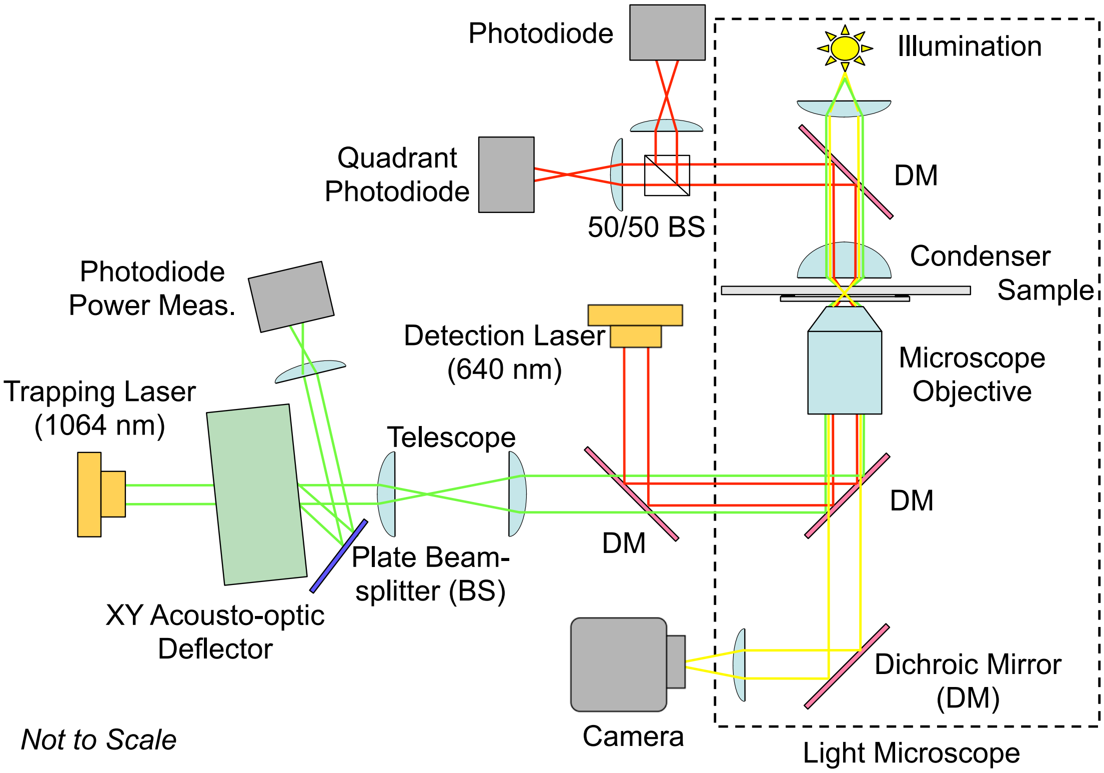


Figure A.1: Layout of an optical trapping instrument with trapping and detection beams for controlled optical trapping.

Figure A.1 shows a schematic representation of the optical trapping configuration used in our experiments, built on top of a vibration isolation table (784-33637-01, TMC). The experiment is built around a conventional inverted light microscope (TE-2000E, Nikon) in order to simplify attaching detectors such as a charge-coupled device (CCD) camera for data collection and also to allow capabilities such as confocal imaging and fluorescent microscopy to be easily added to the experiment.

Two lasers are used in the experiment, a diode pumped Nd:YAG laser with a wavelength of 1064 nm (J20I-8S-12K/BL-106C, Spectra Physics) is used for trapping and manipulating nanoparticles, while a diode laser with a lasing wavelength of 640 nm (iFLEX-2000, Point Source) is used for fast back focal plane detection [86] of the particle position. The trapping laser is fiber launched from a polarizing maintaining fiber and collimated using a single aspheric lens (Model 5726-C-H, New Focus). The beam then passes through an acousto-optic modulator (LS-110XYNIR, Isomet), which can each scan the beam independently along the X and Y axes. Two telescopes are used to bring the trapping beam to the entrance of the microscope. The first telescope is used to expand the beam to fill the back aperture of the microscope objective (Plan Apo 60X/1.40NA oil immersion, Nikon/ CFI Plan Apochromat VC Series 60X/1.20NA water immersion, Nikon), while the second telescope, which is also common to the detection beam (described below), is used to adjust the location of the trap within the sample cell. Both telescopes and the AOD are fixed so that the back entrance aperture of the microscope objective forms a conjugate plane with the AOD exit aperture. This ensures that the rotation of the beam at the AOD exit aperture results in a rotation about the entrance aperture of the microscope objective and thereby a translation in the specimen plane [52]. The AOD splits the incoming light into multiple beams. The zero order beam passes straight through the crystals of the AOD along the angle of incidence and is discarded. The AOD has two crystals to allow the beam to scan along the X and Y axis. Beam scanning along the X and Y axes is accomplished by using the first order beam that emerges from the second crystal. Furthermore, first order beam from the first crystal is deflected only along the X-axis. Therefore, we utilize this beam to continuously measure the power of the trapping laser. We reflect this beam using a plate beam-splitter and image it on to a photo-diode (PD-2032, New Focus). This online measurement provides a relative measure of the power input to the experiment and is used to evaluate the performance of controlled trapping methods outlined in Chapter 5. Furthermore, we can calibrate the detector using a power meter to obtain an absolute measure of the trapping beam power.

The detection laser is also launched from a polarizing maintaining fiber and is collimated using a single aspheric lens (Model 5724-H-B, New Focus). It is combined with the trapping beam using a dichroic mirror (Model Z640BCM, Chroma Technology Corp.) just before it enters the microscope objective. The trapping and detection beams then pass through the sample cell. When the detection laser exits the sample cell, it is collected by an aspheric lens. It then reflected by a dichroic mirror and is split into two beams by passing it through a 50/50 non-polarizing cube beam splitter. One of the split beam passes through a lens before being imaged onto a quadrant photo-diode (QPD), which senses the XY deflection of the detection beam due to the motion of the trapped nanoparticle (QPD-2901, New Focus) in the specimen plane. The second beam exiting the beam splitter is imaged by a lens onto a photo-diode and is used to measure the axial displacements of a nanoparticle at the specimen plane.

Appendix B

Detector Calibration

B.1 Method 1: Camera-based Calibration

A three step calibration procedure has been developed (Figure B.1), which maps the position of a trapped particle measured by the QPD (volts) into physical units (μm). First, the average size of the pixels in the CCD camera are measured using a ronchi ruling of known size. Next, a mapping is constructed to transform the input signal to the AOD into the position of a trapped particle (in pixels) on a camera image. A similar mapping is then constructed for the input signal to the AOD and the position of a trapped particle (in volts) on the QPD. Finally, the data from the calibration is processed to map the QPD output in volts to a distance in micrometers. The calibration procedure is described in detail below.

CCD Camera Calibration: An image of a Ronchi ruling with 100 line pairs per millimeter (Model: 38562, Edmunds) is captured using the 60X/1.40NA oil immersion objective and a CCD camera (Retiga Exi, QImaging), attached to the microscope. Using edge detection techniques, implemented in Mathematica, the width of each line pair is calculated in pixels. Since the width of each line pair in millimeters is known, the average size of each camera pixel and the standard deviation are then calculated. For the CCD camera used in these experiments, the

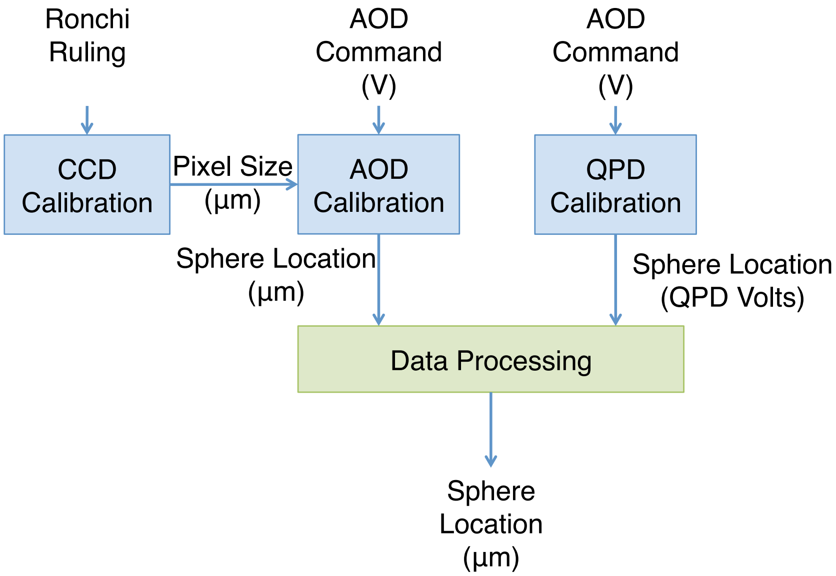
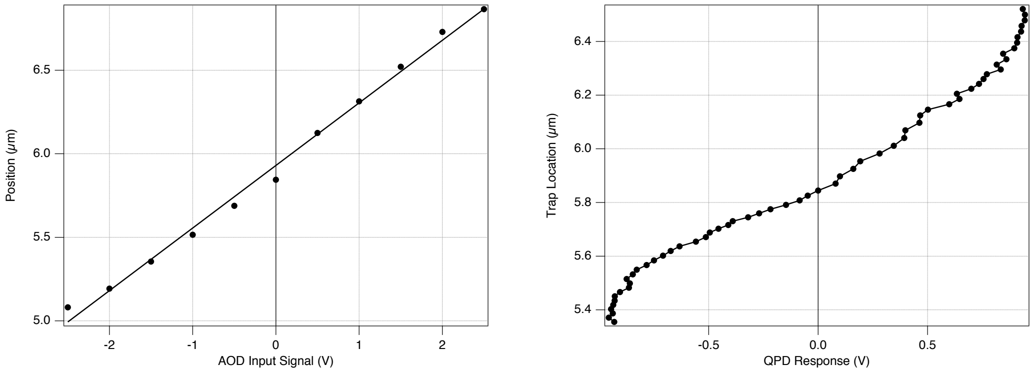


Figure B.1: Calibration procedure.

average size of the pixels was found to be $101 \pm 1\, nm$.

AOD Calibration: The AOD is calibrated next in order to construct a mapping between a command signal input to the AOD (in volts) and the deflection in position of a trapped particle measured in micrometers from a camera image. A trapped particle, held in an optical trap is moved over a finite distance by changing the input command to the AOD, V_{aod} in small increments. At each step, 100 images of the trapped particle are recorded with the CCD camera. The location of the trapped particle in the camera frame (in pixels) is then calculated for each step, using image thresholding and center of gravity algorithms. The position of the particle is then converted into micrometers using the results from the CCD calibration described previously to yield the final calibration, which is a pairing of the AOD input and particle position at each step. This calibration routine is implemented



(a) (b)

Figure B.2: (a) Calibration curve that maps the input to the acousto-optic deflectors to the trapped particle position. The curve shows a linear relationship between the input voltage and trap position. A $\pm 2.5\,V$ input voltage range approximately yields a $2\,\mu m$ scan. (b) QPD calibration obtained by deflecting the trapping beam and measuring the response of the detector. The data shown is the average of 5000 samples collected at each point.

using a Python script, to move the scanner and record images at each location, and Mathematica, for image processing functions. The AOD calibration is performed by moving the scanner over a range of $\pm 2.5\,V$ in $0.5\,V$ increments. Figure B.2(a) shows the experimental data and the curve fit for the relationship between the input signal to the AOD and the trap location. A linear function of the form $y = a\,x + b$ is fit to the AOD calibration data, where the fit parameters, with an uncertainty of one standard deviation are determined using a least squares estimator to be $a = 5.93 \pm 0.02$ and $b = 0.37 \pm 0.01$ for the data in the figure.

QPD Calibration: The QPD calibration procedure follows the AOD procedure closely. A trapped particle is moved across the detector by stepping the AOD in small increments over a desired range. At each location, 5000 samples, corresponding to the AOD input, are recorded by the quadrant photo-diode with a sampling rate of

10 kHz and then averaged. This results in a calibration function between the AOD input (volts) and the QPD response (volts). The QPD calibration is implemented in LabView and performed before the start of data acquisition. For the experimental data reported here, the QPD calibration is performed by stepping a trapped particle across the detector in increments of $0.05\,V$ over a range of $\pm 1.5V$ AOD input and measuring the particle position on the detector.

Calibration Data Processing: Data from the two preceding calibration steps is reduced to provide the final calibration function, which relates the QPD output in volts to the particle position in micrometers. This is done in two steps: i) reverse the QPD calibration map to give the input signal of the AOD (in volts) as a function of the output of the QPD (also in volts) and ii) convert the input signal to the AOD into the position of the particle in a trap (in micrometers) using the output of the AOD calibration step. The resulting calibration response curve used in the experimental results, which gives the mapping between the position of a trapped particle measured using the QPD (in volts) to its position in micrometers is shown in Figure B.2(b).

B.2 Method 2: Nanopositioner-based Calibration

We can also calibrate the three-dimensional position of a nanoparticle using a piezo nano-positioning (NP) stage. We first make a calibration sample containing a low concentration of nanoparticles immobilized in gelatin, which has a refractive index that is very close to water. We then locate a single isolated nanoparticle using the CCD camera (Retiga Exi, QImaging), attached to the microscope and perform a calibration for the XY and Z position separately as described below.

XY Calibration: First, we calibrate the XY position of the nanoparticle using XY nano-positioning stage (PI-xxx, Physik Instrumente) and the quadrant photodiode (QPD-2901, New Focus) as described in Appendix A. We start by moving the XY nano-positioning stage so that the nanoparticle is roughly centered in the detection beam. We then fine adjust the XY NP stage until the output of the QPD, which provides a XY position signal, is zero and the particle is well centered in the detection beam.

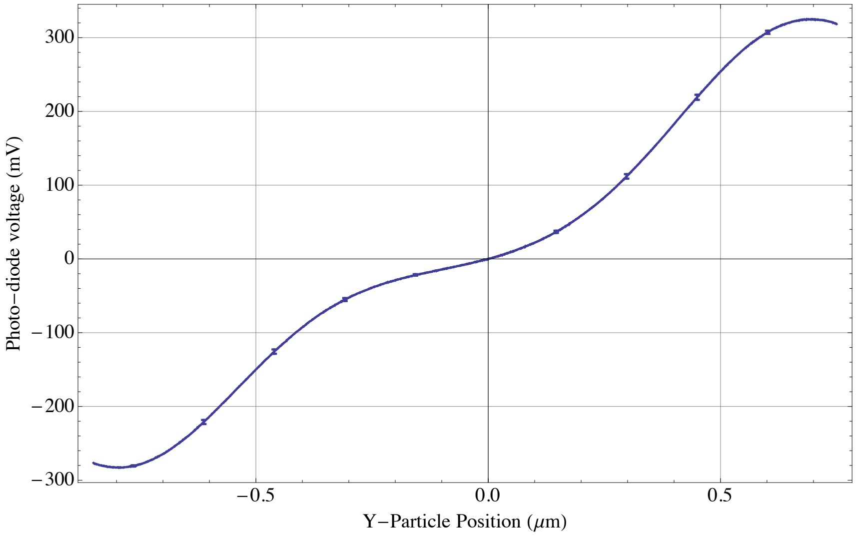


Figure B.3: Transverse detector calibration for 100 nm gold nanoparticles

To perform the calibration, we scan the NP stage using a triangle wave and record the QPD signal. Figure B.3 shows the average QPD response for a 100 nm gold nanoparticle from 100 periods of a triangle wave and the error bars in the figure are the standard error. The X-axis in the plot shows the distance traveled by the NP stage in micrometers and the Y-axis gives us the voltage recorded by the QPD in milli-volts. We are interested in a calibration function that gives us the position of the particle in micrometers, given the detector output in volts. Therefore, we must first reverse the data in Figure B.3 so that the detector output voltage is on the X-axis and the position of the particle in micrometers is on the Y-axis. Furthermore, this calibration function can be parameterized as seen in Section 5.4. As we can see from the figure, for 100 nm gold nanoparticles, the usable range of the QPD is approximately $\pm 750\,nm$. Moreover, the calibration along the X-axis is identical to the methods described above and is therefore not discussed here.

Z Calibration: Calibration of the photo-diode sensor that measures the axial displacement of the particle is similar to the transverse calibration of the QPD. As before, we first center the particle with the detection beam using the XY nano-positioning stage. When the particle is centered in the detection beam, we scan the Z-axis piezo objective stage (PI-xxx, Physik Instrumente), described Appendix A, using a triangle wave and record the output of the photo-diode (PD-2032, New Focus).

Figure B.4 shows the average photo-diode response for a 100 nm gold nanoparticle from 100 periods of a triangle wave and the error bars in the figure are the standard error. As in the transverse, we must first reverse the data in this plot so that the detector voltage is along the X-axis and the position of the particle is on the Y-axis. As we see from the figure, the response of the detector along the Z-axis is linear over most of its range, but asymmetric about the origin. Moreover,

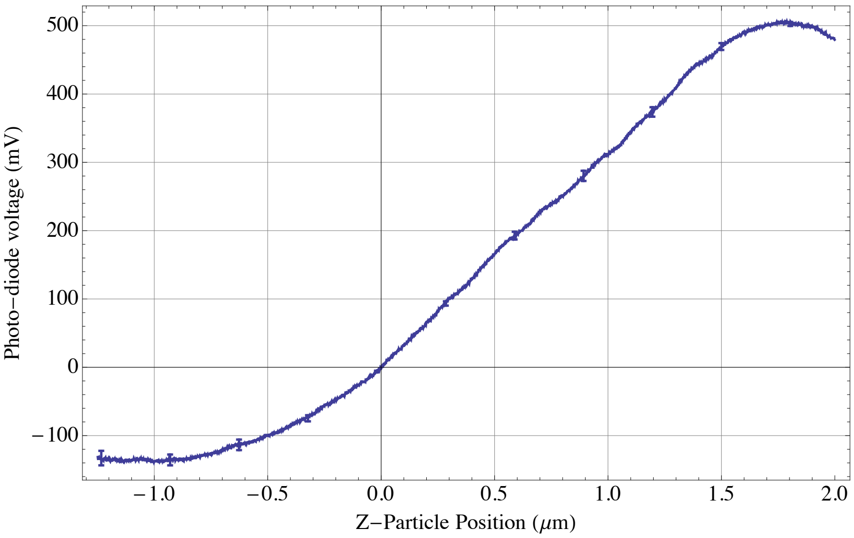


Figure B.4: Longitudinal detector calibration for 100 nm gold nanoparticles

we observe that we have a usable signal over the range of $-1\,\mu m$ to $+2\,\mu m$. Finally, the calibration function, obtained after inverting the plot in Figure B.3 can be parameterized as shown in Section 5.4.

Appendix C

Simulation Architecture

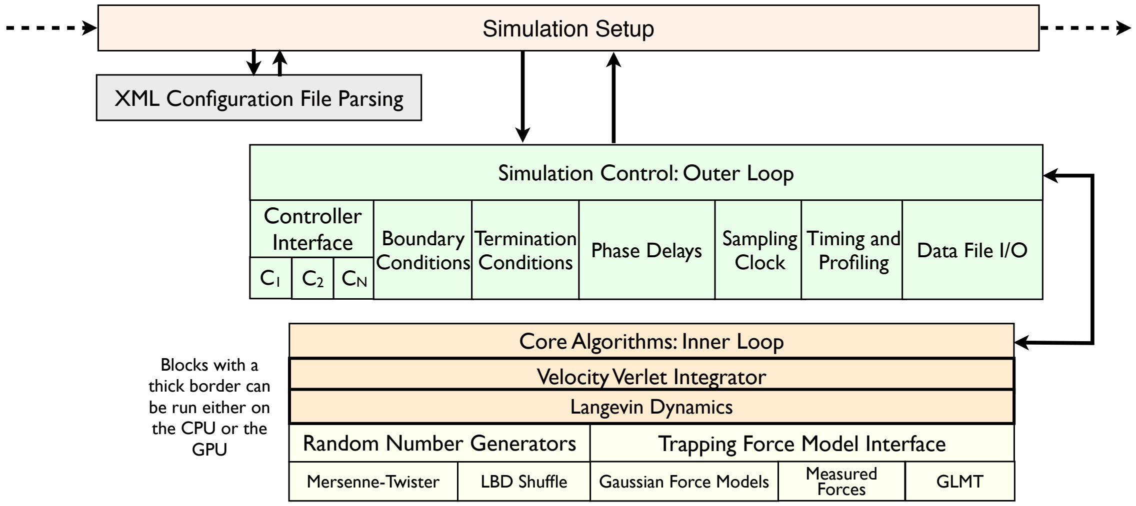


Figure C.1: Control flow of the simulation architecture, implemented in C++ for optical trapping. The program entry and exit points are at the top row, shown by dashed arrows and control flow between blocks is shown by the solid arrows. The simulation algorithms run at two time-scales: a physical time-step (Δt) for modules on the bottom row of the figure and an arbitrary sampling time-step (δt) for the outer loop modules in the middle row.

We have developed a flexible and extensible simulation framework for use in optical tweezers applications. This modular simulation framework, which is shown in Figure C.1 has been implemented in C++ and allows us to easily replace individual components, such as numerical integration modules or force field models. In the figure the control flow is from top to bottom and the simulation framework consists of three levels, a) simulation setup routines that configure the simulation parameters, b) a simulation outer loop that runs at an arbitrary sampling time-step (δt) and encapsulates functionality such as boundary conditions or feedback controllers and c) an inner loop that runs at a physical time-step (Δt) and implements core simulation functionality including Langevin dynamics and random number generation.

The *Simulation Setup* block at top of Figure C.1 is the program entry point and controls the setup and execution of a simulation instance. Parameters for individual functions in the simulation framework are fully configurable using an Extensible Markup Language (XML) configuration file. At startup, the top-level *Simulation Setup* block calls routines within the *XML Configuration File Parsing* block to read and parse the XML configuration file into a data structure. The *Simulation Setup* block then passes this data structure to the simulation outer loop modules (*Simulation Control* block). The outer simulation loop runs at a user-defined sampling frequency ($1/\delta t$) and executes functions that enforce boundary conditions, instrumentation bandwidth, as well as algorithms for controlled optical trapping. For each execution of the outer loop, the simulation inner loop is executed $\frac{\delta t}{\Delta t}$ times. Moreover the value of the sampling time-step, δt is selected so that is perfectly divisible by the physical time-step, Δt. The bottom row of Figure C.1 shows the blocks that form the core algorithms of the simulation inner loop. The simulation architecture is designed to be flexible and fully configurable at runtime. Moreover some parts of the simulation, enclosed by thick lines in Figure C.1 are designed to run on the GPU for significantly improved performance. As seen in the figure, the GPU blocks depend on CPU blocks that generate random numbers or provide the optical trapping force. This however does not reduce the performance of the simulation, since the data required by the GPU blocks can be pre-calculated ahead of time.

First, we describe the simulation setup and configuration block, which can be used to dynamically adapt the simulation architecture for multiple applications. Next, we outline the inner loop modules that form the core simulation algorithms followed by key modules that form the simulation outer loop.

C.1 Simulation Configuration

```xml
<?xml version=”1.0” ?>
<ConfigRoot type=’parent_node’>
  <NParticles type=’real’>15</NParticles>

  <!--Sim Time = NSamples*SampleTimestep=1s-->
  <NSamples type=’real’>100000</NSamples>

  <!--r=200nm, SiO2 microsphere-->
  <M type=’real’>8.7e-17</M>
  <T type=’real’>293</T>
  <Gamma type=’real’>3.8e-09</Gamma>

  <SimTimestep type=’real’>1e-09</SimTimestep>
  <!--100 kHz sampling-->
  <SampleTimestep type=’real’>1e-05</SampleTimestep>
  <RNG type=’text’>MT19937</RNG>

  <Trap type=’parent_node’>
    <FrcModel type=’text’>Gauss3D</FrcModel>
    <!-- depth = 1kT, width (sigma)= 500 nm along each axis-->
    <TrapParams type=’real_array’>
         4.04e-21, 5e-7, 5e-7, 1e-6
    </TrapParams>
    <TrapTraj type=’text’>traj.csv</TrapTraj>
    <TrapTrajTimestep type=’real’>10.0</TrapTrajTimestep>
  </Trap>
</ConfigRoot>
```

Listing C.1: This listing shows an example simulation XML configuration file for a 200 nm diameter glass nanoparticle in a weak Gaussian potential. The simulation is setup for 15 nanoparticles to generate 100 000 data points per particle at a sampling frequency of 100 kHz (or 1 s of data).

The simulation is configured at runtime using one or more configuration files.

Most simulation parameters can be configured with an XML input file such as the one shown in Listing C.1. The input configuration file uses the standard XML format of *<key type='<param>'>data</key>*, where *key* is the XML key name, *type* is an attribute name, *param* is the attribute value and *data* is the XML data. The parameter *type* can be one of *parent_node* for sub-sections, *real* for numbers, *real_array* for an array of numbers or *text* for text input. The XML configuration file requires the root node to be named *ConfigRoot* with an attribute *type='parent_node'*. XML parameters that affect individual program blocks are described in detail below, along with the formats for any auxiliary input files, such as comma separated value (CSV) files for initial conditions.

The configuration file, shown in Listing C.1, is used to run a three-dimensional simulation using a Gaussian force model. As seen in Listing C.1 many simulation parameters are setup directly under the top level *ConfigRoot* node, including physical parameters such as particle mass (M), temperature (T) and the Stokes constant (Gamma), as well as simulation parameters such as the physical time-step (SimTimestep), sampling time-step (SampleTimestep), the number of particles (NParticles) and type of random number generator (RNG) for use in the simulation. Moreover all physical quantities must be defined using SI units. For a simulation that includes a static optical trap, we must also include the trap parameters, which are setup using the *Trap* sub-section. The *FrcModel* tag sets up the type of force field to use in the simulation. In this example, we choose *Gauss3D* which implements the force model replicated from Section 4.3 and shown in Equation C.5. The parameters for the force field are set using the *TrapParams* tag, which accepts a list of numbers that are passed to the force calculation. For the *Gauss3D*, we pass four parameters which set the depth of the trap and the width along the x, y and z axes. In some situations, such as the force measurement described in Section 4.3, the trap must be moved according to a pre-determined trajectory. This is accomplished using an auxiliary CSV input file that contains a list of the three-dimensional coordinates of trap positions. This filename for this trajectory file is set using the *TrapTraj* tag. If the trap trajectory file is specified, the simulation outer loop updates the trap positions sequentially using the coordinates in this file. The update rate of the trap position is then controlled using the *TrapTrajTimestep* tag. Once the setup is complete, this data is passed to the simulation outer loop modules.

C.2 Simulation Inner Loop Modules

The inner loop consists of the four blocks shown in Figure C.1 that form the core of the simulation. Control flow is from top to bottom and the entry point for these functions is at the *Velocity Verlet Integrator* block. Each iteration of the inner loop blocks results in the position, velocity and acceleration of the particle for a given physical time-step (Δt). The *Langevin Dynamics* block, which is described first implements the dynamics of the simulation and generates the acceleration of the particle for each physical time-step. The *Velocity Verlet Integrator* block then performs a numerical integration to obtain the position and velocity of the particle from it's acceleration. At each step, the *Langevin Dynamics* block executes the random number generator (or reads from a list of random numbers) and requests one evaluation of the optical trapping force block (discussed in more detail at the end of this section). Some components of the simulation (shown with thick lines in Figure C.1) can be offloaded to the Graphics Processing Unit (GPU) present in most computers to gain a significant performance enhancement over the CPU implementation without a large loss in accuracy.

Langevin Dynamics Block: This block implements the dynamics of the motion of a spherical particle in a fluid as described in detail in Section 3.2. The particle dynamics are formally expressed by Langevin's equation, which is simply Newton's equation of motion coupled with a properly scaled randomly fluctuating force. Langvein's equation is shown in Equation C.1 for a particle with mass, m, radius a and velocity $V(t)$ at time t in a fluid with viscosity η, which is a function of temperature T. The characteristic time-scale of this model is given by the relaxation time, $\frac{m}{\gamma}$, where $\gamma = 6\pi\eta a$ is the drag coefficient (from Stoke's law for a spherical particle), and is the time taken by the particle to reach thermal equilibrium with the fluid, by gaining or losing velocity.

$$\frac{dV(t)}{dt} = -\frac{\gamma}{m}V(t) + \frac{\zeta}{m}\Gamma(t) \tag{C.1}$$

As described previously in Section 3.2, the scaling constant, $\zeta = \sqrt{2\gamma k_B T}$, in Equation C.1 is obtained by imposing the requirements of the fluctuation-dissipation theorem, where k_B is Boltzmann's constant. The presence of the stochastic force term, $\frac{\zeta\Gamma(t)}{m}$ prevents the direct analytical solution of this equation. Therefore, in order to calculate a single random trajectory, we express Langevin's equation in finite difference form as shown in Equation C.2, where $\chi = \sqrt{\frac{2\gamma k_B T}{\Delta t}}$ is the appropriate scaling factor for the normal distribution $N(0,1)$ and Δt is a uniform physical time-step. The external force force term (F_{ext}) in this equation allows us to include the forces from an optical trap.

$$A(t) = -\frac{\gamma}{m}V(t) + \frac{\chi}{m}N(0,1) + \frac{F_{ext}}{m} \tag{C.2}$$

The physical parameters required by the Langevin dynamics block are input using the XML configuration file as seen in Listing C.1. The specific tags that are required to properly configure this module include *M* for the particle mass, *T* for the temperature of the fluid, *Gamma* for the Stoke's constant, *SimTimestep* for the physical time-step Δt and *RNG* to select the random number generator. Currently, we have implemented two random number generators that are described in more detail later in this section.

Velocity Verlet Integration Block: As described in Section 3.2, we use a second order velocity Verlet integration algorithm in our simulations. However the modular nature of the simulation framework allows us to easily replace this method with a different numerical integration algorithm. Several numerical integration schemes, such as the Gear predictor-corrector or the Runge-Kutta algorithms, are suitable for use in Langevin dynamics simulations. However, an integration method must be selected that a) satisfies conservation laws with acceptable accuracy, b) is computationally cost effective and c) permits the use of the largest possible physical time-step [19].

$$V(t + \Delta t) = V(t) + \frac{A(t) + A(t + \Delta t)}{2}\Delta t \tag{C.3}$$

$$R(t + \Delta t) = R(t) + V(t)\Delta t + \frac{1}{2}A(t)\Delta t^2 \qquad \text{(C.4)}$$

Starting with the acceleration of the particle at the end of a physical time-step ($t+\Delta t$), given by Equation C.2, the velocity Verlet algorithm gives us the velocity of the particle at $t+\Delta t$ using Equation C.3. Next, the position of the particle is updated using the previous velocity and acceleration as shown in Equation C.4. Therefore, the velocity Verlet algorithm requires only one evaluation of the optical trapping force. The results of one pass of the integrator block are vectors of the particle's position, velocity and acceleration the end of that physical time-step, which is then returned to the inner loop function.

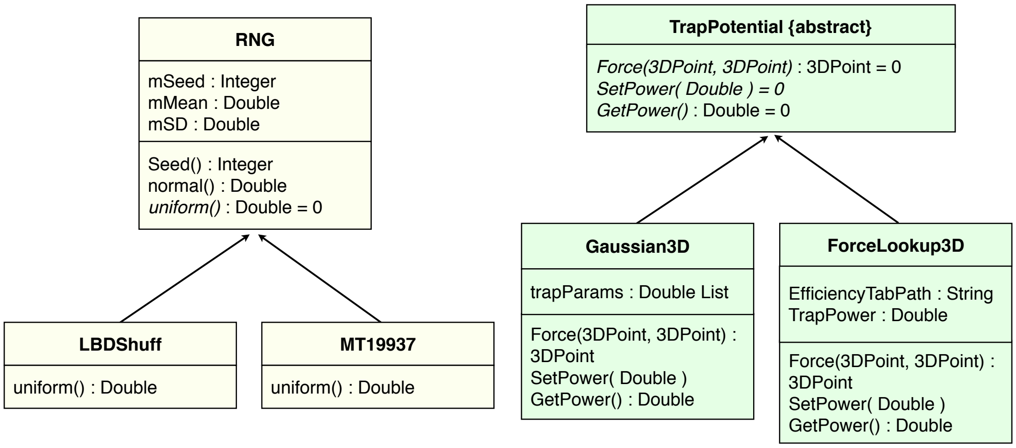


Figure C.2: Class diagram for the random number generator and optical trapping force interface

Random Number Generation: The *Langevin Dynamics* blocks call the random number generator (RNG) functions at each iteration when evaluating Equation C.2. The accuracy of the simulation results depend in part on the quality of the standard normal random variate used within the *Langvin Dynamics* block. In order to generate normal random numbers, we first generate uniform random variates and then convert them to a normal random variable using the Box-Muller transform [124]. Since the Box-Muller transform is used across all RNGs, we have implemented it in the top level *RNG* class. New RNG implementations are therefore only required to implement uniform random number generators. We have implemented two uniform random number generators: i) the L'Ecuyer generator with Bays-Durham shuffle (LBD) [124] and ii) the Mersenne Twister (MT) [101]. Both algorithms have very long periods ($> 2^{18}$ for the LBD method and $2^{19937} - 1$ for the MT algorithm) and are able to generate long uncorrelated sequences of random numbers. The random number generator for a particular simulation is selected using the *RNG* tag in the XML simulation configuration file. Currently, this tag accepts a value of either 'LBDShuff' for the LBD generator or 'MT19937' for the Mersenne Twister algorithm. Moreover, the common interface definition for the random number generator functions, shown on the left hand side of Figure C.2 allows us to add other uniform random number generators transparently.

Each implementation of a random number generator must inherit the top-level RNG class shown Figure C.2 and implement the *uniform* method to generate a uniform random number generator. When running the simulation on the GPU, the random numbers are pre-calculated and downloaded to the graphics card. This method takes no arguments but must return a single uniformly distributed random number in double precision format in the interval (0, 1]. The top level interface class, RNG, provides functions that generate seed values for the random number generators and also implements the Box-Muller transform to convert a uniform random number to a normal random number. However, these methods can be overridden to implement a different custom implementation for these functions.

Optical Trapping Forces: At each physical time-step the *Langevin Dynamics* block also evaluates the optical trapping force for use in Equation C.2. This is done using the abstract interface class *TrapPotential* shown on the right side of Figure C.2. Similar to the RNG class, having a common interface to trapping force calculations allows us to easily implement and use several trapping force models interchangeably with the simulations. Each force field model must therefore inherit the *TrapPotential* base class and implement the force function (F), which takes two 3-element vectors with the particle position ($\vec{r}$) and trap location ($\vec{\mu}$) and returns a 3-element vector containing the optical trapping force. The parameters for the force calculation are set using the *TrapParams* tag in the XML configuration as seen in Listing C.1. The force calculation must also implement two functions that can set the trapping power (*SetPower*) and read it's current value (*GetPower*) as seen in Figure C.2. These implementation of these functions can vary depending on the particular force model being used. For example, for the *Gauss3D* model described below, these power functions are used to vary the depth of the trapping potential in units of k_BT, where k_B is the Boltzmann's constant and T is the temperature of the fluid. In other implementations, such as the generalized Lorentz Mie theory (GLMT) discussed later in this section and also in Chapter 5, this parameter is the power of the trapping beam in Watts. The methods that manipulate the beam intensity are used heavily when implementing intensity control (see Chapter 5 for an example), where the power of the optical trap is changed as a function of the position of the nanoparticle within the trap.

The right hand side of Figure C.2 also outlines the class diagrams for two example force field implementations that are used in this work and inherit the *TrapPotential* class. The first method is a Gaussian force calculation (*Gaussian3D*, replicated from Section 4.5), which implements Equation C.5 within the function F to calculate the optical trapping force. In this equation, the position of the particle within the optical trap is given by the vector $\vec{r} = \{x, y, z\}$ and the trapping force is parameterized by the scalar quantity α, which characterizes the depth of the trapping potential and the trap width is given by $\vec{\sigma} = \{\sigma_x, \sigma_y, \sigma_z\}$. The position of the trap center in three-dimensions is then controlled by changing the vector $\vec{\mu} = \{\mu_x, \mu_y, \mu_z\}$. The configuration parameters for this Gaussian force field are listed in the *Trap* sub-section of the XML configuration file shown in Listing C.1. The mandatory *FrcModel* tag sets the type of force field to use and the parameters of the force field are set using the *TrapParams* tag. In Listing C.1, the trap parameters set the depth of the potential to $1\,k_BT$ and the width along the X, Y and Z axes to $500\,nm$, $500\,nm$ and $1000\,nm$ respectively. Other tags in this example are as described earlier.

$$\vec{F_T}(\vec{r}) = \left\{ \begin{array}{c} -\frac{\alpha}{\sigma_x^2}(x-\mu_x) \\ \\ -\frac{\alpha}{\sigma_y^2}(y-\mu_y) \\ \\ -\frac{\alpha}{\sigma_z^2}(z-\mu_z) \end{array} \right\} \times exp\left(-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2} - \frac{(z-\mu_z)^2}{2\sigma_z^2}\right) \tag{C.5}$$

The second force calculation is a fast lookup method (*ForceLookup3D*) to quickly find the trapping force from a pre-calculated table, which is stored in a format that is independent of trap power. The advantage of such a force field model is that it allows us to include tabulated force data from a variety of theoretical and experimental sources. The table must provide a XYZ force vector at each point along a uniform three-dimensional grid. As with the Gaussian force model discussed previously, this technique inherits the base class *TrapPotential* and implements the force function (F). The force function in this implementation, takes the particle position and simply returns the trapping force for the closest grid point without any interpolation. Therefore the accuracy of the forces returned by this technique depend directly on the spacing of the grid. Moreover the force field is truncated when the particle positions fall outside the range of the force table. We have used this force modeling method in Section **??** to implement a GLMT force calculation in our simulations to demonstrate controlled optical trapping [67, 104]. The parameters for this force field are setup under the *Trap* sub-section in the simulation configuration file and include the power of the trap and the spatial range of the forces in the table. The force table file is specified using the *TrapForceFile* tag in addition to the other trapping force parameters, described previously in Section C.1.

Physical Validation and Performance: We have performed a detailed validation of the inner loop modules of the simulation [19]. We tested the energy conservation and the diffusion constant for free spherical particles suspended in water and with a radius of $50\,nm$, $500\,nm$ and $5\,\mu m$. We found the relative error in energy was on average within $0.5\,\%$ of the theoretical value, with a maximum error of $1.7\,\%$. On the other hand, the relative errors in the diffusion constant were found to be within 1.4 % of the theoretical value on average, with a maximum error of 4.2 %. We also tested the accuracy of the simulations when run on the GPU, where the maximum relative error of the diffusion constant calculated using the GPU data was 8 %, an error that is acceptable for optical trapping applications. However, simulations run on the GPU provided an approximately 10 fold speedup (for only 1600 particles) in comparison with both single-precision and double-precision floating point calculations on the CPU. The simulation algorithms, analysis techniques and the results for both physical validation and performance are discussed in detail in Section 3.4.

C.3 Simulation Outer Loop Modules

The blocks that form the outer loop of the simulation framework are shown in the middle row of Figure C.1. The top level block in this row contains the main simulation control logic that executes once for every sampling time-step (δt). During execution, the top level block calls each of the modules below it as required to calculate a new control signal, check boundary conditions, perform data I/O or measure timing and profiling of the simulation components. We discuss two outer loop modules in detail below: methods to implement a feedback controller for enhanced trapping of nanoparticles and a module that allows us to include the phase delays from the limited bandwidth of laboratory instrumentation.

Feedback Controller Interface: The outer loop of the simulation includes an interface to incorporate a feedback controller to improve the performance of optical traps. Controller modules can alter only two trap parameters, as a function of the particles position: the trap location and the trap intensity. Therefore a generalized controller can be fully defined by the vector function $\vec{S}(\vec{r})$, which returns the location of the trap as a function of particle's position $\vec{r}$ and the scalar function $I(\vec{r})$, which returns the intensity or power of the trap as a function of the particle's position $\vec{r}$. Specific implementations of feedback controllers generally include either scan control, intensity control or both.

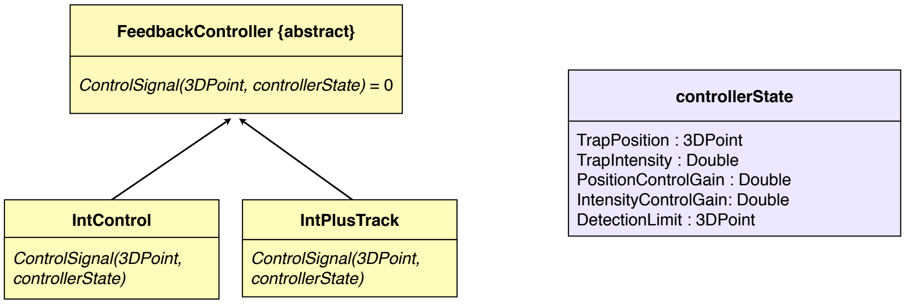


Figure C.3: Class diagram for the feedback controller interface

The class diagram for the controller interface is shown in Figure C.3. All implementations of the abstract class *FeedbackController* are required to include the function *ControlSignal*, which takes a 3-tuple containing the position of the particle and a data structure that holds the current state of the controller. The new trap position and trap power calculated by *ControlSignal* are stored within the controller state data structure. A simulation of a controller that increases the lifetime of a nanoparticle in an optical trap is described in detail in Section 5.2.3.

The right side of Figure C.3 shows the UML diagram for the *controllerState* data structure, which must be passed to the *ControlSignal* function when implementing a feedback controllers. The list of parameters shown in Figure C.3 is not comprehensive, but the data structure must include the last position of the trap, it's intensity and other parameters such as controller gains and the spatial limits over which a controller can operate.

```
<FeedbackController type='parent_node'>
  <ControllerType type='text'>CombinedController3D</ControllerType>
  <MaxTrapDepth type='real'>0.05</MaxTrapDepth>
  <MinTrapDepth type='real'>0</MinTrapDepth>
  <KpScan type='real_array'>0,0,0</KpScan>
  <KpIntensity type='real_array'>25000</KpIntensity>
  <TrapPositionSetPoint type='real_array'>0,0,0</TrapPositionSetPoint>
  <ParticleDetectionRange type='real_array'>
        1.4e-6,1.4e-6,3e-6
  </ParticleDetectionRange>
  <ControlLoopBandwidth type='real'>5e3</ControlLoopBandwidth>
 </FeedbackController>
```

Listing C.2: Sub-section of the XML configuration input file that defines the parameters for a feedback controller for use in the simulation. The controller defined in this listing is a pure proportional intensity controller that changes the trap intensity in response to the nanoparticle's motion along the z-axis.

Listing C.2 shows one example of the *FeedbackController* sub-section within the simulation XML configuration file. When this sub-section is present in the configuration file, the controller module in the outer loop is enabled. Listing C.2 describes the parameters that define a proportional intensity controller that modifies the power of the trap in response to the particles motion. The control equations for this example are given by Equations C.6−C.7, where $\vec{r}$ is the vector describing the particle position in Cartesian coordinates and x_s, y_s, z_s are the set points for the controller, defined by the *TrapPositionSetPoint* tag. In this example, scan control is effectively disabled by setting the parameter *KpScan* to 0. Furthermore, since this controller only implements intensity control along the Z-axis, the parameter *KpIntensity* only takes one input. As seen from Listing C.2, the gain for this controller is set to 25000, which implies that the controller reaches it's saturation power of $50\,mW$, $2\,\mu m$ from the center.

$$\vec{S}(\vec{r},t) = \begin{Bmatrix} KpScan[0] \times (x_s \; - \; x(t)) \\ KpScan[1] \times (y_s \; - \; y(t)) \\ KpScan[2] \times (z_s \; - \; z(t)) \end{Bmatrix} \tag{C.6}$$

$$I(z,t) = KpIntensity \times |z_s \; - \; z(t)| \tag{C.7}$$

In Listing C.2, the mandatory *ControllerType* tag is used to determine the controller algorithm to use. Each controller algorithm must register a unique text identifier that allows the user to select the controller type at run-time. In this case we set this parameter to a controller that proportionally changes the position and intensity of the trap in response to it's motion away from the trap center. The *MaxTrapDepth* tag is used by the intensity controller to define the upper limit of trap power available to the controller and conversely the *MinTrapDepth* tag sets the minimum power. In this example, we limit the maximum power available to the controller to $50\,mW$ and the minimum is set to 0 so that the trap is off when the particle is at the center of the trap. The spatial range over which the controller can operate in the laboratory is often limited by the range of the detectors used to measure the particle position in the trap. Enforcing this detection limit in the simulation is crucial to obtain realistic behavior. For example, without a detection limit a controller can be designed to track the particle position in order to always keep it within the trap. This would lead to a controller that would have give the optical trap an infinite lifetime, but would be very impractical to implement in the laboratory. Therefore for both scan and intensity control, we define the practical spatial limits over which the controller can operate and set these values using measurements in the laboratory. Finally, we can include the effect of practical limitations such as instrumentation with a finite bandwidth, which can give rise to a phase lag in the control signal. The effective bandwidth of the controller can be set using the *ControlLoopBandwidth* tag, which is set in this case to $50\,kHz$. The methods used to model these phase delays in the simulation are described next.

Phase Delays: The simulation allows us to include practical effects arising from laboratory instrumentation such as phase delays, which can change the controller response and have a detrimental effect on performance.

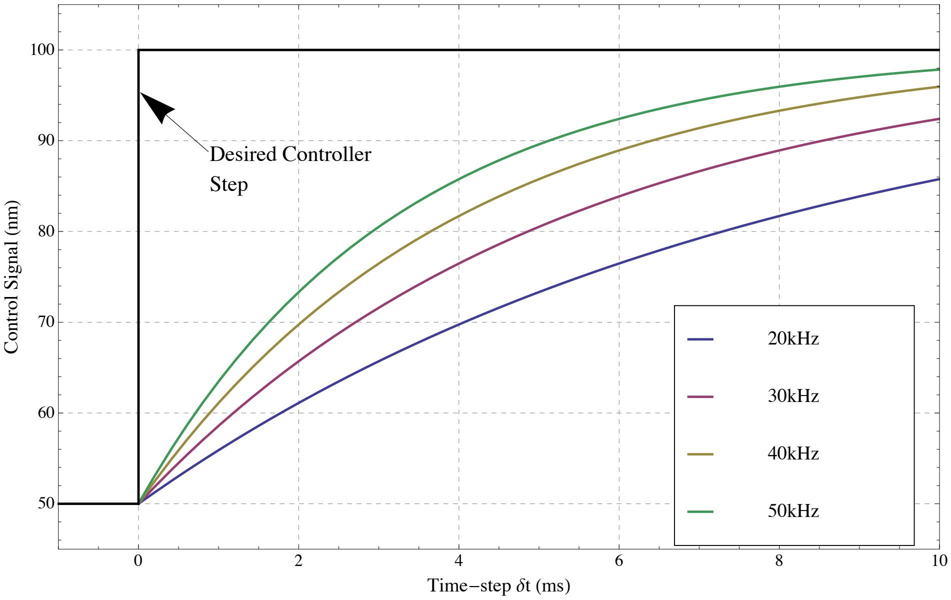


Figure C.4: A phase lag model has been implemented in the simulation outer loop to model the effects of instrumentation bandwidth. The figure shows an ideal controller step (in black) and the response of controllers with bandwidth ranging from $20\,kHz$ to $50\,kHz$ across a $10\,\mu s$ time-step.

$$x(t) = a + e^{-\omega t}(x_0 - a) \tag{C.8}$$

We model phase delays in the controller as the first order response to a step input (a), starting at an initial controller location, x_0. The solution for such a first order system is shown in Equation C.8, where x is the position of the controller and $\omega = \frac{2\pi}{T}$ parameterizes the response time T of the system. In the *Phase Delays* block shown in Figure C.1, Equation C.8 is implemented by setting x_0 to the last controller position and setting a to the new desired controller position. The response of the controller for a system with bandwidth $f = \frac{1}{T}$ at the end of the sampling time-step δt is then $x(\delta t)$. Figure C.4 show the response of a controller to a $50\,nm$ step input across one $10\,\mu s$ sampling time-step, for bandwidths ranging from $20\,kHz$ to $50\,kHz$. As expected for a system with a phase lag, the controller falls short of the desired controller position at the end of a time-step and the difference between their positions is larger for systems with lower bandwidths. In order to turn on phase delays in the simulations, we set the instrumentation bandwidth in *Hz* using the *ControllerBandwidth* tag within the *FeedbackController* section of the XML configuration file shown in Listing C.2.

Appendix D

Optical Trapping Force Calculated Using a Dipole Approximation

The optical trapping forces acting on nanoparticles, that are significantly smaller than the wavelength of the incident light (λ) are calculated by approximating the particles as simple dipole oscillators. In this, so called Rayleigh approximation, the incident light is assumed to have a uniform amplitude across the particle and the magnitude of the force is proportional to the polarizability of the material (α_p). The gradient force acting on such a particles is then given by Equation D.1, where I is the intensity of the light, incident on particle and n_m is the refractive index of the surrounding medium [44, 148].

$$F_g = \frac{|\alpha_p| n_m}{2c} \nabla I \tag{D.1}$$

Equation D.2 describes a beam, incident on the small particle with a Gaussian intensity distribution, which depends on the power, P, the half-width of the beam waist, w_0 and the radial coordinate, r.

$$I = \frac{P}{\pi w_0}\, exp\left(-\frac{r^2}{2w_0^2}\right) \tag{D.2}$$

The polarizability of the particle under the influence of the trapping beam is then given by Equation D.3, where $V^{'}$ is the effective volume of the particle, ϵ is the dielectric constant of the particle and $\epsilon_m = n_m^2$ is the dielectric constant of the surrounding fluid medium.

$$\alpha_p = 3V^{'} \frac{\epsilon - \epsilon_m}{\epsilon + 2\epsilon_m} \tag{D.3}$$

The effective volume, V' for dielectric materials such as glass is the same as the volume of the particle. However, for conducting materials such as gold, the incident electromagnetic field generates an electric current in the particle, which decays exponentially as a function of depth, away from the particle's surface. The size of the effective conductive region, close to the surface of the material, is called the skin depth of the material and in the case of optical trapping it affects the calculation of the trapping force. In order to account for the skin depth, the effective volume of the particle must be calculated using the relationship given in Equation D.4, where a is the radius of the particle and δ is the skin depth.

$$V' = 4\pi \int_0^a \left(r^2 \; exp\left(\frac{r-a}{\delta} \right) \right) dr \tag{D.4}$$

The skin depth, δ, shown in Equation D.5, is a function of the wavelength of the incident field (λ) and a function of the particle's dielectric constant k, given in Equation D.6.

$$\delta = \frac{\lambda}{2\pi k} \tag{D.5}$$

$$k = \left(-\frac{Re[\epsilon]}{2} + \frac{\sqrt{Re[\epsilon]^2 + Im[\epsilon]^2}}{2} \right)^{1/2} \tag{D.6}$$

We can then substitute Equation D.2 and Equation D.3 into Equation D.1 to obtain the explicit expression for the gradient force shown in Equation D.7. We see that the gradient force depends linearly on the power of the incident beam. We then integrate Equation D.7 to obtain an expression for the optical trapping potential (Equation D.8), where U_0 is the constant of integration, which can be set arbitrarily.

$$F_g = -\frac{P|\alpha_p| n_m r}{2\pi c w_0^4} \; exp\left(-\frac{r^2}{2w_0^2} \right) \tag{D.7}$$

$$U = -\int F_g \, dr = -\frac{P|\alpha_p| n_m}{2\pi c w_0^2} \; exp\left(-\frac{r^2}{2w_0^2}\right) \; + \; U_0 \tag{D.8}$$

The scattering force acting on the particle is given by Equation D.9, where $< S >$ is the time-averaged Poynting vector, C_s is the scattering cross-section of the particle, given by Equation D.10 and $k_m = 2\pi n_m/\lambda$ is the wave number in the medium.

$$F_s = \frac{n_m < S > C_s}{c} \tag{D.9}$$

$$C_s = \frac{k_m^4 |\alpha_p|^2}{4\pi} \tag{D.10}$$

Similarly, the force due to absorption is given by Equation D.11, where C_a is the absorption cross-section, given in Equation D.12.

$$F_a = \frac{n_m < S > C_a}{c} \tag{D.11}$$

$$C_a = k_m \, Im(\alpha_p) \tag{D.12}$$

Bibliography

[1] Technical brief: Nvidia GeForce 8800 GPU architecture overview. 2006.

[2] H Acuna-Campa and M Medina-Noyola. Collective diffusion in a two-dimensional brownian fluid. *J Chem Phys*, 113(2):869–875, 2000.

[3] R Agarwal, K Ladavac, Y Roichman, GH Yu, Cm Lieber, and Dg Grier. Manipulation and assembly of nanowires with holographic optical traps. *OPTICS EXPRESS*, 13(22):8906–8912, 2005.

[4] Erik Alerstam, Tomas Svensson, and Stefan Andersson-Engels. Parallel computing with graphics processing units for high-speed monte carlo simulation of photon migration. *Journal of Biomedical Optics*, 13(6):060504, 2008.

[5] AP Alivisatos, WW Gu, and C Larabell. Quantum dots as cellular probes. *Annu Rev Biomed Eng*, 7:55–76, 2005.

[6] Erik C Allen and Gregory C Rutledge. A novel algorithm for creating coarse-grained, density dependent implicit solvent models. *J Chem Phys*, 128(15):154115, 2008.

[7] M P. Allen and D J. Tildesley. *Computer Simulation of Liquids*. Oxford University Press, 1989.

[8] M Allersma, F Gittes, M deCastro, and R Stewart. Two-dimensional tracking of ncd motility by back focal plane interferometry. *Biophysical Journal*, 1998.

[9] JA Anderson, CD Lorenz, and A Travesset. General purpose molecular dynamics simulations fully implemented on graphics processing units. *Journal of Computational Physics*, 227(10):5342–5359, 2008.

[10] Katsuhiko Ariga, Jonathan P Hill, Michael V Lee, Ajayan Vinu, Richard Charvet, and Somobrata Acharya. Challenges and breakthroughs in recent research on self-assembly. *Sci. Technol. Adv. Mater.*, 9(1):014109, 2008.

[11] A. Ashkin. Acceleration and trapping of particles by radiation pressure. *Physical Review Letters*, 24(4):156–159, 1970.

[12] A Ashkin. Forces of a single-beam gradient laser trap on a dielectric sphere in the ray optics regime. *Biophysical Journal*, pages 569–582, 1992.

[13] A ASHKIN. History of optical trapping and manipulation of small-neutral particle, atoms, and molecules. *IEEE Journal of Selected Topics in Quantum Electronics*, 6(6):841–856, 2000.

[14] A Ashkin and JM Dziedzic. Feedback stabilization of optically levitated particles. *Applied Physics Letters*, 30(4):202–204, Jan 1977.

[15] P Auffinger, L Bielecki, and E Westhof. Symmetric K+ and Mg2+ ion-binding sites in the 5 S rRNA loop E inferred from molecular dynamics simulations. *Journal of Molecular Biology*, 335(2):555–571, 2004.

[16] Anna C Balazs, Todd Emrick, and Thomas P Russell. Nanoparticle polymer composites: Where two small worlds meet. *Science*, 314(5802):1107–1110, 2006.

[17] A. Balijepalli, T Lebrun, J Gorman, and S Gupta. Methods to directly measure the trapping potential in optical tweezers. In *Proceedings of SPIE*, volume 70380V, 2008.

[18] A Balijepalli, T Lebrun, J Gorman, and S Gupta. Evaluation of a trapping potential measurement technique for optical tweezers using simulations and experiments. *Proceedings of the ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2009.

[19] A Balijepalli, T Lebrun, and S Gupta. Stochastic simulations with graphics hardware: Characterization of accuracy and performance. *Journal of Computing and Information Science in Engineering*, Jan 2010.

[20] A. Balijepalli, T. W. Lebrun, J. J. Gorman, and S. K. Gupta. Enhanced force measurement techniques to extend optical trapping towards nanoscale manipulation. *Proceedings of the 9th IEEE Nanotechnology Conference: IEEE Nano*, 2009.

[21] A Balijepalli, TW LeBrun, and S K Gupta. A flexible system framework for a nanoassembly cell using optical tweezers. In *Proceedings of IDETC/CIE 2006 ASME 2006*, volume 2006, pages 333–342, 2006.

[22] RM Barnett, CD Carone, DE Groom, TG Trippe, CG Wohl, B Armstrong, PS Gee, GS Wagman, F James, M Mangano, K Monig, L Montanet, JL Feng, H Murayama, JJ Hernandez, A Manohar, M AguilarBenitez, C Caso, RL Crawford, M Roos, NA Tornqvist, KG Hayes, K Hagiwara, K Nakamura, M Tanabashi, K Olive, K Honscheid, PR Burchat, RE Shrock, S Eidelman, RH Schindler, A Gurtu, K Hikasa, G Conforto, RL Workman, C Grab, and C Amsler. Particles and field .1. review of particle physics. *Phys Rev D*, 54(1), 1996.

[23] J Barton and D Alexander. Fifth–order corrected electromagnetic field components for a fundamental gaussian beam. *Journal of Applied Physics*, 1989.

[24] J Baumgartl and C Bechinger. On the limits of digital video microscopy. *Europhysics Letters*, 71:487–493, 2005.

[25] D Bemporad, C Luttmann, and JW Essex. Computer simulation of small molecule permeation across a lipid bilayer: Dependence on bilayer properties and solute volume, size, and cross-sectional area. *Biophysical Journal*, 87(1):1–13, 2004.

[26] D Bemporad, C Luttmann, and JW Essex. Behaviour of small solutes and large drugs in a lipid bilayer from computer simulations. *BBA-Biomembranes*, 1718(1-2):1–21, 2005.

[27] D. C Benito, D. M Carberry, S. H Simpson, G. M Gibson, M. J Padgett, J. G Rarity, M. J Miles, and S Hanna. Constructing 3D crystal templates for photonic band gap materials using holographic optical tweezers. *Optics Express*, 16(17):13005–13015, 2008.

[28] H J C Berendsen and W F Van Gunsteren. Practical algorithms for dynamic simulations - molecular dynamics simulation of statistical mechanical systems. *Proceedings of the Enricfo Fermi Summer School*, pages 43–64, 1985.

[29] K Berg-sorensen and H Flyvbjerg. Power spectrum analysis for optical tweezers. *Review of Scientific Instruments*, 75(3):594–612, 2004.

[30] K Berg-sorensen, EJG Peterman, T Weber, Cf Schmidt, and H Flyvbjerg. Power spectrum analysis for optical tweezers. ii: Laser wavelength dependence of parasitic filtering, and how to achieve high bandwidth. *Review of Scientific Instruments*, 77(6):063106, 2006.

[31] S Berneche and B Roux. Molecular dynamics of the KcsA K+ channel in a bilayer membrane. *Biophysical Journal*, 78(6):2900–2917, 2000.

[32] S Berneche and B Roux. Energetics of ion conduction through the K+ channel. *Nature*, 414(6859):73–77, 2001.

[33] SM Block, CL Asbury, JW Shaevitz, and MJ Lang. Probing the kinesin reaction cycle with a 2D optical force clamp. *P Natl Acad Sci USA*, 100(5):2351–2356, 2003.

[34] SM Block, LSB Goldstein, and BJ Schnapp. Bead movement by single kinesin molecules studied with optical tweezers. *Nature*, 348(6299):348–352, 1990.

[35] A Brańka and D Heyes. Algorithms for brownian dynamics simulation. *Physical Review E*, 1998.

[36] H Brenner. The slow motion of a sphere through a viscous fluid towards a plane surface. *Chem Eng Sci*, 16(3-4):242–251, 1961.

[37] A Buosciolo, G Pesce, and A Sasso. New calibration method for position detector for simultaneous measurements of force constants and local viscosity in optical tweezers. *Optics Communications*, 230(4-6):357–368, 2004.

[38] M Capitanio, G Romano, R Ballerini, M Giuntini, FS Pavone, D Dunlap, and L Finzi. Calibration of optical tweezers with differential interference contrast signals. *Review of Scientific Instruments*, 73(4):1687–1696, 2002.

[39] S Chandrasekhar. Stochastic problems in physics and astronomy. *Reviews of Modern Physics*, 15(1):0001–0089, 1943.

[40] J. C. Chen and A. S. Kim. Brownian dynamics, molecular dynamics, and monte carlo modeling of colloidal systems. *Advances in colloid and interface science*, 112(1-3):159–173, 2004.

[41] Altera Corporation. Cyclone FPGA family data sheet, san jose, ca, URL: http://www.altera.com/literature/lit-cyc.jsp.

[42] J Crocker and D Grier. Methods of digital video microscopy for colloidal studies. *Journal of Colloid And Interface Science*, 179:298–310, 1996.

[43] L. C DaSilva, L Candido, L. da F Costa, and Osvaldo N Oliveira. Formation energy and interaction of point defects in two-dimensional colloidal crystals. *Physical Review B*, 76(3):035441, 2007.

[44] T. J Davis. Brownian diffusion of nano-particles in optical traps. *Optics Express*, 15(5):2702–2712, 2007.

[45] W Denk and WW Webb. Optical measurement of picometer displacements of transparent microscopic objects. *Applied Optics*, 29(16):2382–2391, 1990.

[46] Kishan Dholakia, Woei Ming Lee, Lynn Paterson, Michael P MacDonald, Richard McDonald, Igor Andreev, Patience Mthunzi, C. Tom A Brown, Robert F Marchington, and Andrew C Riches. Optical separation of cells on potential energy landscapes: Enhancement with dielectric tagging. *IEEE Journal of Selected Topics in Quantum Electronics*, 13(6):1646–1654, 2007.

[47] Kishan Dholakia, Michael P MacDonald, Pavel Zemanek, and Tomas Cizmar. Cellular and colloidal separation using optical forces. *Method Cell Biol*, 82:467–495, 2007.

[48] Göddeke Dominik, Strzodka Robert, and Turek Stefan. Accelerating double precision FEM simulations with GPUs. *Proceedings of the 18th Symposium on Simulation Technique (ASIM 2005)*, pages 139–144, 2005.

[49] ER Dufresne, TM Squires, MP Brenner, and DG Grier. Hydrodynamic coupling of two brownian spheres to a planar surface. *Physical Review Letters*, 85(15):3317–3320, 2000.

[50] A Einstein. On the movement of small particles suspended in stationary liquids required by the molecular-kinetic theory of heat. *Annalen der Physik*, 17:549–560, 1905.

[51] DL Ermak and JA McCammon. Brownian dynamics with hydrodynamic interactions. *The Journal of Chemical Physics*, 69:1352, 1978.

[52] E Fallman and O Axner. Design for fully steerable dual-trap optical tweezers. *Applied Optics*, 36(10):2107–2113, 1997.

[53] MX Fernandes and JG de la Torre. Brownian dynamics simulation of rigid particles of arbitrary shape in external fields. *Biophysical Journal*, 83(6):3039–3048, 2002.

[54] JT Finer, RM Simmons, and JA Spudich. Single myosin molecule mechanics: piconewton forces and nanometre steps. *Nature*, 368(6467):113–119, 1994.

[55] EL Florin, A Pralle, EHK Stelzer, and JKH Horber. Photonic force microscope calibration by thermal noise analysis. *Appl Phys A-Mater*, 66:S75–S78, 1998.

[56] Daan Frenkel and Berend Smit. *Understanding molecular simulation: From algorithms to applications*. Academic Press, second edition, 2002.

[57] Mark S Friedrichs, Peter Eastman, Vishal Vaidyanathan, Mike Houston, Scott Legrand, Adam L Beberg, Daniel L Ensign, Christopher M Bruns, and Vijay S Pande. Accelerating molecular dynamic simulation on graphics processing units. *Journal of Computational Chemistry*, 30(6):864–872, 2009.

[58] Mej Friese, H RubinszteinDunlop, Nr Heckenberg, and EW Dearden. Determination of the force constant of a single-beam gradient trap by measurement of backscattered light. *Applied Optics*, 35(36):7112–7116, 1996.

[59] T Fujiwara, K Ritchie, H Murakoshi, K Jacobson, and A Kusumi. Phospholipids undergo hop diffusion in compartmentalized cell membrane. *J Cell Biol*, 157(6):1071–1081, 2002.

[60] J Gelles, B Schnapp, and M Sheetz. Tracking kinesin-driven movements with nanometre-scale precision. *Nature*, 331:450–453, 1988.

[61] L Ghislain, N Switz, and W Webb. Measurement of small forces using an optical trap. *Review of Scientific Instruments*, pages 2762–2768, 1994.

[62] DT Gillespie. Fluctuation and dissipation in brownian-motion. *American Journal of Physics*, 61(12):1077–1083, Jan 1993.

[63] F Gittes and Cf Schmidt. Interference model for back-focal-plane displacement detection in optical tweezers. *Optics Letters*, 23(1):7–9, 1998.

[64] D Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1), 1991.

[65] Zan Gong, Hongtao Chen, Shenghua Xu, Yinmei Li, and Liren Lou. Montecarlo simulation of optical trap stiffness measurement. *Optics Communications*, 263(2):229–234, 2006.

[66] J J Gorman, TW LeBrun, and Arvind Balijepalli. Control of optically trapped particles for brownian motion suppression. *Accepted to be published in the IEEE Transactions on Control Systems Technology*, 2009.

[67] G Gouesbet, B Maheu, and G Grehan. Light–scattering from a sphere arbitrarily located in a gaussian beam, using a bromwich formulation. *Journal of the Optical Society of Americal A–Optics Image Science and Vision*, 5(9):1427–1443, 1988.

[68] P Grassia. Dissipation, fluctuations, and conservation laws. *American Journal of Physics*, 69(2):113–119.

[69] William J Greenleaf, Kirsten L Frieda, Daniel A. N Foster, Michael T Woodside, and Steven M Block. Direct observation of hierarchical folding in single riboswitch aptamers. *Science*, 319(5863):630–633, 2008.

[70] P Habdas and Er Weeks. Video microscopy of colloidal suspensions and colloidal crystals. *Current Opinion in Colloid & Interface Science*, 7(3-4):196–203, 2002.

[71] Ramin Haghgooie and Patrick S Doyle. Directed self-assembly of field-responsive fluids in confined geometries. *Soft Matter*, 5(6):1192–1197, 2009.

[72] P. M. Hansen, V. K. Bhatia, N. Harrit, and L. B. Oddershede. Expanding the optical trapping range of gold nanoparticles. *Nano Letters*, 5(10):1937–1942, 2005.

[73] M. G. Harisinghani, J. Barentsz, P. F. Hahn, W. M. Deserno, S. Tabatabaei, C. H. van de Kaa, J. de la Rosette, and R. Weissleder. Noninvasive detection of clinically occult lymph-node metastases in prostate cancer. *New Engl J Med*, 348(25):2491–U5, 2003.

[74] M Harris. Many-core GPU computing with NVIDIA CUDA. In *Proceedings of the 22nd Annual International Conference on Supercomputing*, Island of Kos, Greece, 2008. ACM.

[75] M Hecht, J Harting, T Ihle, and HJ Herrmann. Simulation of claylike colloids. *Physical Review E*, 72(1):011408, 2005.

[76] K Hillesland and a Lastra. GPU floating-point paranoia. *ACM Workshop on General Purpose Computing on Graphics Processors, page C*, 8.

[77] Analog Devices Inc. AD9860/AD9862 mixed-signal front-end MxFE, processor for broadband communications.

[78] Jonathan P Jacky, Joseph L Garbini, Matthew Ettus, and John A Sidles. Digital control of force microscope cantilevers using a field programmable gate array. *Review of Scientific Instruments*, 79(12):123705, 2008.

[79] R Khardekar, G Burton, and S McMains. Finding feasible mold parting directions using graphics hardware. *Computer-Aided Design*, 38(4):327–341, 2006.

[80] Emmett Kilgariff and Randima Fernando. The GeForce 6 series GPU architecture. 2005.

[81] P Kipfer, M Segal, and R Westermann. Uberflow: A GPU-based particle engine. *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 115–122, 2004.

[82] Andreas Kolb and Nicolas Cuntz. Dynamic particle coupling for GPU-based fluid simulation. *Proceedings of the 18th Symposium on Simulation Technique*, pages 722–727, 2005.

[83] PT Korda and DG Grier. Annealing thin colloidal crystals with optical gradient forces. *J Chem Phys*, 114(17):7570–7573, 2001.

[84] R Kubo. The fluctuation-dissipation theorem. *Reports on Progress in Physics*, 29(1):255–284, 1966.

[85] SC Kuo and MP Sheetz. Force of single kinesin molecules measured with optical tweezers. *Science*, 260(5105):232–234, 1993.

[86] MJ Lang, CL Asbury, JW Shaevitz, and SM Block. An automated two-dimensional optical force clamp for single molecule studies. *Biophysical Journal*, 83(1):491–501, 2002.

[87] Paul Langevin. On the theory of Brownian motion ("sur la théorie du mouvement Brownien,"). *C. R. Acad. Sci. (Paris)*, 146:530–533, 1908.

[88] Eunjeong Lee. An integrated system of microcantilever arrays with carbon nanotube tips for imaging, sensing, and 3d nanomanipulation: Design and control. *Sensor Actuat A-Phys*, 134(1):286–295, 2007.

[89] M. Lemkin and B. E. Boser. A three-axis micromachined accelerometer with a cmos position-sense interface and digital offset-trim electronics. *Ieee J Solid-St Circ*, 34(4):456–468, 1999.

[90] Terence S Leung and Samuel Powell. Fast monte carlo simulations of ultrasound-modulated light using a graphics processing unit. *JOURNAL OF BIOMEDICAL OPTICS*, 15(5):055007, 2010.

[91] L Li, RG Larson, and T Sridhar. Brownian dynamics simulations of dilute polystyrene solutions. *Journal of Rheology*, 44(2):291–322, 2000.

[92] Mo Li, W. H. P Pernice, C Xiong, T Baehr-Jones, M Hochberg, and H. X Tang. Harnessing optical forces in integrated photonic circuits. *Nature*, 456(7221):480–U28, 2008.

[93] Shaoliang Lin, Naoko Numasawa, Takuhei Nose, and Jiaping Lin. Brownian molecular dynamics simulation on self-assembly behavior of rod-coil diblock copolymers. *Macromolecules*, 40(5):1684–1692, 2007.

[94] Ettus Research LLC. Universal software radio peripheral USRP, URL: http://www.ettus.com/, 2006.

[95] B Lukić, S Jeney, Ž Sviben, A Kulik, and E Florin. Motion of a colloidal particle in an optical trap. *Physical Review E*, 2007.

[96] B Lukić, S Jeney, C Tischer, A Kulik, and L Forró. Direct observation of nondiffusive motion of a brownian particle. *Physical Review Letters*, 2005.

[97] MP MacDonald, S Neale, L Paterson, A Richies, K Dholakia, and GC Spalding. Cell cytometry with a light touch: Sorting microscopic matter with an optical lattice, 2004.

[98] N Malagnino, G Pesce, A Sasso, and E Arimondo. Measurements of trapping efficiency and stiffness in optical tweezers. *Optics Communications*, 214(1-6):15–24, 2002.

[99] AA Mamedov, NA Kotov, M Prato, DM Guldi, JP Wicksted, and A Hirsch. Molecular design of strong single-wall carbon nanotube/polyelectrolyte multilayer composites. *Nature Materials*, 1(3):190–194, Jan 2002.

[100] SJ Marrink and HJC Berendsen. Simulation of water transport through a lipid–membrane. *J Phys Chem-Us*, 98(15):4155–4168, 1994.

[101] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulations*, 8(1):3–30, 1998.

[102] Jeremy S Meredith, Gonzalo Alvarez, Thomas A Maier, Thomas C Schulthess, and Jeffrey S Vetter. Accuracy and performance of graphics processors: A quantum monte carlo application case study. *Parallel Comput*, 35(3):151–163, 2009.

[103] JP Mills, L Qie, M Dao, CT Lim, and S Suresh. Nonlinear elastic and viscoelastic deformation of the human red blood cell with optical tweezers. *MCB-Tech Science Press*, 1:169–180, 2004.

[104] YK Nahmias and DJ Odde. Analysis of radiation forces in laser trapping and laser-guided direct writing applications. *IEEE J Quantum Elect*, 38(2):131–141, 2002.

[105] P Nakroshis, M Amoroso, J Legere, and C Smith. Measuring Boltzmann's constant using video microscopy of Brownian motion. *American Journal of Physics*, 2003.

[106] Alexander V Nemukhin, Ilya A Kaliman, and Alexander A Moskovsky. Modeling negative ion defect migration through the gramicidin a channel. *J Mol Model*, 15(8):1009–1012, 2009.

[107] K Neuman and S Block. Optical trapping. *Review of Scientific Instruments*, 75(9):2787—2809, 2004.

[108] KC Neuman, EA Abbondanzieri, R Landick, J Gelles, and Sm Block. Ubiquitous transcriptional pausing is independent of RNA polymerase backtracking. *Cell*, 115(4):437–447, 2003.

[109] H Nishino, S Clark, K Abe, Y Hayato, T Iida, M Ikeda, J Kameda, K Kobayashi, Y Koshio, M Miura, S Moriyama, M Nakahata, S Nakayama, Y Obayashi, H Ogawa, H Sekiya, M Shiozawa, Y Suzuki, A Takeda, Y Takenaga, Y Takeuchi, K Ueno, K Ueshima, H Watanabe, S Yamada, S Hazama, I Higuchi, C Ishihara, T Kajita, K Kaneyuki, G Mitsuka, K Okumura, N Tanimoto, M. R Vagins, F Dufour, E Kearns, M Litos, J. L Raaf, J. L Stone, L. R Sulak, W Wang, M Goldhaber, S Dazeley, R Svoboda, K Bays, D Casper, J. P Cravens, W. R Kropp, S Mine, C Regis, M. B Smy, H. W Sobel, K. S Ganezer, J Hill, W. E Keig, J. S Jang, J. Y Kim, I. T Lim, M Fechner, K Scholberg, C. W Walter, R Wendell, S Tasaka, J. G Learned, S Matsuno, Y Watanabe, T Hasegawa, T Ishida, T Ishii, T Kobayashi, T Nakadaira, K Nakamura, K Nishikawa, Y Oyama, K Sakashita, T Sekiguchi, T Tsukamoto, A. T Suzuki, A Minamino, T Nakaya, M Yokoyama, Y Fukuda, Y Itow, T Tanaka, C. K Jung, G Lopez, C McGrew, R Terri, C Yanagisawa, N Tamura, Y Idehara, M Sakuda, Y Kuno, M Yoshida, S. B Kim, B. S Yang, T Ishizuka, H Okazawa, Y Choi, H. K Seo, Y Furuse, K Nishijima, Y Yokosawa, M Koshiba, Y Totsuka, S Chen, Y Heng, Z Yang, H Zhang, D Kielczewska, E Thrane, and R. J Wilkes. Search for proton decay via $p \to e^+\pi^0$ and $p \to \mu^+\pi^0$ in a large water cherenkov detector. *Phys Rev Lett*, 102(14):141801, 2009.

[110] L Oddershede, H Flyvbjerg, and K Berg-sorensen. Single-molecule experiment with optical tweezers: Improved analysis of the diffusion of the lambda-receptor in e-coli's outer membrane. *Journal of Physics–Condensed Matter*, 15(18):S1737–S1746, 2003.

[111] B Onoa, S Dumont, J Liphardt, Sb Smith, I Tinoco, and C Bustamante. Identifying kinetic barriers to mechanical unfolding of the T-thermophila ribozyme. *Science*, 299(5614):1892–1895, 2003.

[112] Mario Orsi, Wendy E Sanderson, and Jonathan W Essex. Permeability of small molecules through a lipid bilayer: A multiscale simulation study. *Journal of Physical Chemistry B*, 113(35):12019–12029, 2009.

[113] R Pastor, B Brooks, and A Szabo. An analysis of the accuracy of langevin and molecular dynamics algorithms. *Molecular Physics*, 1988.

[114] PJ Pauzauskie, A Radenovic, E Trepagnier, H Shroff, PD Yang, and J Liphardt. Optical trapping and integration of semiconductor nanowire assemblies in water. *Nature Materials*, 5(2):97–101, 2006.

[115] T Peng, a Balijepalli, S Gupta, and T Lebrun. Algorithms for on-line monitoring of micro spheres in an optical tweezers-based assembly cell. *Journal of Computing and Information Science in Engineering*, 2007.

[116] Tao Peng, Arvind Balijepalli, and Satyandra K Gupta. Algorithms for extraction of nanowires attributes from optical section microscopy images. In *Proceedings of the ASME Design Engineering Technical Conferences*, 2007.

[117] Tao Peng, Arvind Balijepalli, Satyandra K Gupta, and Thomas W LeBrun. Algorithms for extraction of nanowire lengths and positions from optical section microscopy image sequence. *Journal of Computing and Information Science in Engineering*, 9(4):041007, 2009.

[118] T. T. Perkins, S. R. QUAKE, D. E. Smith, and S. Chu. Relaxation of a single DNA molecule observed by optical microscopy. *Science*, 264(5160):822–826, 1994.

[119] E Perozo, DM Cortes, P Sompornpisut, A Kloda, and B Martinac. Open channel structure of MscL and the gating mechanism of mechanosensitive channels. *Nature*, 418(6901):942–948, 2002.

[120] A Pertsinidis and XS Ling. Diffusion of point defects in two-dimensional colloidal crystals. *Nature*, 413(6852):147–150, 2001.

[121] Kathleen F Pirollo and Esther H Chang. Targeted delivery of small interfering rna: Approaching effective cancer therapies. *Cancer Res*, 68(5):1247–1250, 2008.

[122] A Pralle, M Prummer, EL Florin, EHK Stelzer, and JKH Horber. Three-dimensional high-resolution particle tracking for optical tweezers by forward scattered light. *Microsc Res Techniq*, 44(5):378–386, 1999.

[123] T Preis, P Virnau, W Paul, and J Schneider. GPU accelerated monte carlo simulation of the 2D and 3D ising model. *Journal of Computational Physics*, Jul 2009.

[124] W Press, S Teukolsky, W Vetterling, and B Flannery. *Numerical Recipes in C++: The Art of Scientific Computing.* Cambridge University Press, second edition, 2005.

[125] AK Priyadrashi and SK Gupta. Finding mold-piece regions using computer graphics hardware. *Geometric Modeling and Processing Conference*, 2006.

[126] GNU Radio Project. USRP documentation, URL: http://gnuradio.org/redmine/wiki/gnuradio.

[127] TR Ramachandran, C Baur, A Bugacov, A Madhukar, BE Koel, A Requicha, and C Gazen. Direct and controlled manipulation of nanometer-sized particles using the non-contact atomic force microscope. *Nanotechnology*, 9(3):237–245, 1998.

[128] A Ranaweera and B Bamieh. Modelling, identification, and control of a spherical particle trapped in an optical tweezer. *International Journal of Robust and Nonlinear Control*, 15(16), 2005.

[129] AAG Requicha, S Meltzer, R Resch, D Lewis, BE Koel, and ME Thompson. Layered nanoassembly of three-dimensional structures. *IEEE International Conference on Robotics and Automation*, 4:3408–3411, 2001.

[130] A Revyakin, RH Ebright, and TR Strick. Promoter unwinding and promoter clearance by RNA polymerase: Detection by single-molecule DNA nanomanipulation. *P Natl Acad Sci USA*, 101(14):4776–4780, 2004.

[131] Hannes Risken. *The Fokker-Planck Equation: Methods of Solution and Applications.* Springer, 1996.

[132] A Rohrbach. Stiffness of optical traps: Quantitative agreement between experiment and electromagnetic theory. *Physical Review Letters*, 95(16):168102, 2005.

[133] A Rohrbach, EL Florin, and EHK Stelzer. Photonic force microscopy: Simulation of principles and applications. In *Proceedings of the SPIE*, volume 4431, pages 75–86, 2001.

[134] A Rohrbach and EHK Stelzer. Optical trapping of dielectric particles in arbitrary fields. *Journal of the Optical Society of Americal A–Optics Image Science and Vision*, 18(4):839–853, 2001.

[135] A Rohrbach and EHK Stelzer. Three-dimensional position detection of optically trapped dielectric particles. *Journal of Applied Physics*, 91:5474, 2002.

[136] A Rohrbach and EHK Stelzer. Trapping forces, force constants, and potential depths for dielectric spheres in the presence of spherical aberrations. *Applied Optics*, 41(13):2494–2507, 2002.

[137] Yohai Roichman, Bo Sun, Allan Stolarski, and David G Grier. Influence of nonconservative optical forces on the dynamics of optically trapped colloidal spheres: The fountain of probability. *Physical Review Letters*, 101(12):128301, 2008.

[138] Jimaan Sane, Johan T Padding, and Ard A Louis. Hydrodynamics of confined colloidal fluids in two dimensions. *Physical Review E*, 79(5):051402, Jan 2009.

[139] S Sato, M Ohyumi, H Shibata, H Inaba, and Y Ogawa. Optical trapping of small particles using a 1.3-mu m compact ingaasp diode laser. *Opt. Lett.*, 16(5):282–284, 1991.

[140] EA Schellenberger, D Sosnovik, R Weissleder, and L Josephson. Magneto/optical annexin v, a multimodal protein. *Bioconjugate Chem*, 15(5):1062–1067, 2004.

[141] Yeonee Seol, Amanda E Carpenter, and Thomas T Perkins. Gold nanoparticles: enhanced optical trapping and sensitivity coupled with significant heating. *Optics Letters*, 31(16):2429–2431, 2006.

[142] RM Simmons, JT Finer, S Chu, and JA Spudich. Quantitative measurements of force and displacement using an optical trap. *Biophysical Journal*, 70(4):1813–1822, 1996.

[143] D E Smith, SJ Tans, S B Smith, S Grimes, D L Anderson, and C Bustamante. The bacteriophage phi 29 portal motor can package DNA against a large internal force. *NATURE*, 413(6857):748–752, 2001.

[144] S. M. Spearing. Materials issues in microelectromechanical systems (MEMS). *Acta Mater*, 48(1):179–196, 2000.

[145] John E Stone, David J Hardy, Ivan S Ufimtsev, and Klaus Schulten. GPU-accelerated molecular modeling coming of age. *J Mol Graph Model*, 29(2):116–125, 2010.

[146] T Sumanaweera and D Liu. *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*, chapter Medical Image Reconstruction With the FFT, pages 765–768. Addison-Wesley Professional, 2005.

[147] K Svoboda and S Block. Biological applications of optical forces. *Annual Reviews in Biophysics and Biomolecular Structure*, 1994.

[148] K Svoboda and SM Block. Optical trapping of metallic Rayleigh particles. *Optics Letters*, 19(13):930–932, 1994.

[149] K Svoboda, CF Schmidt, BJ Schnapp, and SM Block. Direct observation of kinesin stepping by optical trapping interferometry. *Nature*, 365(6448):721–727, 1993.

[150] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *J Chem Phys*, 76(1):637–649, 1982.

[151] R Thurn and W Kiefer. Structural resonances observed in the Raman spectra of optically levitated liquid droplets. *Applied Optics*, 24(10):1515–1519, 1985.

[152] Simon F Tolic-Norrelykke, Erik Schaeffer, Jonathon Howard, Francesco S Pavone, Frank Juelicher, and Henrik Flyvbjerg. Calibration of optical tweezers with positional detection in the back focal plane. *Review of Scientific Instruments*, 77(10):103101, 2006.

[153] J. A van Meel, A Arnold, D Frenkel, S. F. Portegies Zwart, and R. G Belleman. Harvesting graphics power for MD simulations. *Molecular Simulation*, 34(3):259–266, 2008.

[154] L. Verlet. Computer experiments on classical fluids. I. thermodynamical properties of lennard-jones molecules. *Phys. Rev*, 1967.

[155] L. Verlet. Computer experiments on classical fluids. II. equilibrium correlation functions. *Phys. Rev*, 1968.

[156] B. Viren. Confidence limit calculation for setting proton decay lifetime limits (Super-Kaniokande note 98-03) URL: http://nngroup.physics.sunysb.edu/nngroup/publication_theses/ brett_sknote.pdf, 1998.

[157] RK Visaria, RJ Griffin, BW Williams, ES Ebbini, GF Paciotti, CW Song, and JC Bischof. Enhancement of tumor thermal therapy using gold nanoparticle-assisted tumor necrosis factor-alpha delivery. *Mol Cancer Ther*, 5(4):1014–1020, 2006.

[158] K Visscher and Sm Block. Versatile optical traps with feedback control. *Method Enzymol*, 298:460–489, 1998.

[159] K Visscher, Sp Gross, and Sm Block. Construction of multiple-beam optical traps with nanometer-resolution position sensing. *IEEE Journal of Selected Topics in Quantum Electronics*, 2(4):1066–1076, 1996.

[160] M. von Smoluchowski. Zur kinetischen theorie der brownschen molekularbewegung und der suspensionen. *Annalen der Physik*, 326(14):756–780, 1906.

[161] Anders E Wallin, Heikki Ojala, Edward Haeggstrom, and Roman Tuma. Stiffer optical tweezers through real-time feedback control. *Applied Physics Letters*, 92(22):224104, Jan 2008.

[162] Dong Wang and Gregory A Voth. Proton transport pathway in the CIC Cl-/H+ antiporter. *Biophysical Journal*, 97(1):121–131, 2009.

[163] MD Wang, MJ Schnitzer, H Yin, R Landick, J Gelles, and Sm Block. Force and velocity measured for single molecules of rna polymerase. *Science*, 282(5390):902–907, 1998.

[164] MD Wang, H Yin, R Landick, J Gelles, and Sm Block. Stretching DNA with optical tweezers. *Biophysical Journal*, 72(3):1335–1346, 1997.

[165] GM Whitesides and M Boncheva. Beyond molecules: Self-assembly of mesoscopic and macroscopic components. *P Natl Acad Sci Usa*, 99(8):4769–4774, 2002.

[166] WH Wright, GJ Sonek, and MW Berns. Parametric study of the forces on microspheres held by optical tweezers. *Applied Optics*, 33(9):1735–1748, 1994.

[167] Kurt D Wulff, Daniel G Cole, and Robert L Clark. Servo control of an optical trap. *Applied Optics*, 46(22):4923–4931, Jan 2007.

[168] Kurt D Wulff, Daniel G Cole, and Robert L Clark. Adaptive disturbance rejection in an optical trap. *Applied Optics*, 47(20):3585–3589, 2008.

[169] Tian-Xiang Xiang and Bradley D Anderson. Liposomal drug transport: A molecular perspective from molecular dynamics simulations in lipid bilayers. *Adv Drug Deliver Rev*, 58(12-13):1357–1378, 2006.

[170] SH Xu, YM Li, LR Lou, and ZW Sun. Computer simulation of the collision frequency of two particles in optical tweezers. *Chinese Phys*, 14(2):382–385, 2005.