Martin Gurtner

Distributed optimization for multi-object manipulation by shaping spatial force fields

CTU
CZECH TECHNICAL UNIVERSITY IN PRAGUE

2022

MARTIN GURTNER

DISTRIBUTED OPTIMIZATION FOR MULTI-OBJECT MANIPULATION BY SHAPING SPATIAL FORCE FIELDS

DOCTORAL THESIS

DISTRIBUTED OPTIMIZATION FOR MULTI-OBJECT MANIPULATION BY SHAPING SPATIAL FORCE FIELDS

MARTIN GURTNER

Prague, 2022

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Control Engineering

Ph.D. Programme:
Electrical Engineering and Information Technology

Branch of study:
Control Engineering and Robotics

Supervisor:
Zdeněk Hurák

To Saša and Filípek.

ABSTRACT

This thesis addresses three topics related to feedback manipulation through shaping force fields: position measurement, modeling of force fields, and force model inversion through distributed optimization. The first two topics focus specifically on dielectrophoresis. The third topic covers manipulation using more general force fields. The thesis presents a new sensor for measuring the 3D positions of micro-objects in real time with sufficient accuracy for feedback control. It also introduces a control-oriented model of dielectrophoretic force that can be evaluated in real time and thus is suitable for feedback control. This model is verified through experiments. The thesis proposes a distributed optimization algorithm based on ADMM for finding electric potentials that produce a desired dielectrophoretic force field, a process called model inversion. The distributed algorithm can be applied to manipulation using other physical force fields and is tested on dielectrophoresis, magnetophoresis, and acoustophoresis. Additionally, the thesis presents a compact dielectrophoretic manipulation platform that incorporates the proposed position sensor, dielectrophoretic model, and optimization-based model inversion, as well as the capability to observe manipulated objects using lensless digital holography.

KEYWORDS

distributed manipulation, distributed optimization, alternating direction method of multipliers, micro-manipulation, feedback manipulation, dielectrophoresis, digital holography

ABSTRAKT

Tato práce se zabývá třemi tématy souvisejícími se zpětnovazební manipulací prostřednictvím tvarování silových polí: měřením polohy, modelováním silového pole a inverzí modelu síly prostřednictvím distribuovaného optimalizace. První dvě témata jsou věnována specificky dielektroforetickému silovému poli. Poslední téma je věnováno obecnějším silovým polím. V této práci je představen a otestován nový senzor pro měření 3D poloh mikro objektů. Senzor měří polohy v reálném čase a s dostatečnou přesností pro zpětnovazební řízení. Dále je navržen a experimenty ověřen zjednodušený model dielektroforetické síly. Na rozdíl od jiných dostupných modelů je navržený model reprezentován výrazem, který lze vyhodnotit v reálném čase, což činí tento model vhodným pro zpětnovazební řízení. Tato práce také navrhuje distribuovaný optimalizační algoritmus založený na ADMM pro řešení problému nalezení elektrických potenciálů takových, že jsou vyvinuty požadované dielektroforetické síly v určitých bodech manipulačního prostoru. To se nazývá inverze modelu, protože dielektroforetický model vytváří síly na základě potenciálů, ne naopak. Stejný distribuovaný algoritmus lze aplikovat i při manipulaci jinými fyzikálními silovými poli. V práci je tento algoritmus testován na manipulaci pomocí dielektroforézy, magnetoforézy a akustoforézy. Nakonec je představena nová kompaktní platforma pro manipulaci pomocí dielektroforézy. Platforma kombinuje navržený senzor polohy, model dielektroforézy a inverzi modelu založenou na optimalizaci. Kromě toho platforma poskytuje také možnost pozorování manipulovaných objektů založenou na bez-objektivové digitální holografii.

KLÍČOVÁ SLOVA

distribuovaná manipulace, distribuovaná optimalizace, alternating direction method of multipliers, micro-manipulace, zpětnovazební manipulace, dielektroforéza, digitální holografie

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Zdeněk Hurák for his guidance and support throughout my learning journey. He gave me the opportunity to broaden my knowledge also beyond the rather narrow topic of this thesis, which is invaluable. I must also thank Jiří Zemánek for his invaluable ability to come up with many exciting, very enjoyable, and mostly not really useful side projects[1] that I somehow happened to work on. I would also like to thank Loi Do for his funny control-engineering-related jokes and an unlimited supply of coffee that kept me going. I must also thank Krištof Pučejdl for the SK8O project[2]. If I had to pick one project unrelated to this thesis I truly enjoyed working on, this would be it. Without Krištof, the project would not have come into existence. I cannot name all members of the AA4CC group I was part of during my studies, so to those of you who did not find your name here, thank you for the collaboration! It was fun. You all helped me move forward. I am also thankful to my family, Saša and Filípek, for their patience and understanding while I worked on my papers and thesis in the evenings. Thank you all for your help and support.

1 Interactive Christmas tree driving through Prague streets `https://youtu.be/bqqmBAz4INo`
Christmas Ballance `https://youtu.be/xpeEB2v42HQ`
Ball in hoop educational model `https://youtu.be/484GN4KBQnc`

2 SK8O robot `https://youtu.be/07z1FbjhixM`

CONTENTS

1 Introduction 1
2 Twin-beam real-time position estimation of micro-objects in 3D 11
3 Green's function-based model of DEP 21
4 ADMM for distributed manipulation 43
5 Compact DEP manipulation platform 81

1

INTRODUCTION

There is a theory which states that if ever anyone discovers exactly what the Universe is for and why it is here, it will instantly disappear and be replaced by something even more bizarre and inexplicable. There is another theory which states that this has already happened.

Douglas Adams
The Hitchhiker's Guide to the Galaxy

Manipulation by shaping force fields sounds strange and hard to grasp. But in fact, when we have a small ball on a sheet of paper, we can shape the force field in which the ball lives just by shaping the paper (see Fig. 1.1). When we make a valley next to the ball, the ball naturally goes down the value to its lowest point. When we reshape the paper so that the lowest points move from the ball, the ball follow it. In layman's terms, the topic of this thesis is how to shape the force field in time so that the ball gets from point A to point B in space.

In this thesis, we[1] deal with this topic in a rather holistic approach. We talk about modeling physical force fields, sensors, and algorithms. More specifically, we discuss more general force fields than the one acting on a ball when the sheet of paper the ball lies on is curved; we talk about force fields created by various physical phenomena, but with the main focus being on *dielectrophoretic (DEP)* force field (more on that in a moment). We do not use our eyes to see where the manipulated object (e.g., ball) is; we use *position sensors*. We do not use our hands to shape the force fields as we did in the ball on a sheet of paper example; we use *an array of actuators* to shape the force fields. Each actuator can contribute to the force field; hence the resulting force field is given by contributions from all actuators in the actuator array. Lastly, we do not use our brains to decide how to move our hands to shape the force field so that the ball moves toward the desired position; we use *distributed optimization* to decide what is the best way how

1 As the author of this thesis, I use the plural person *we* in the introduction. That has two reasons. Either I mean you and me, as I guide us together throughout what to expect from this thesis. Or, I use plular because I collaborated on all of the results of this thesis with my colleagues; Thus, I cannot say "I came with this and that result". We came up with the result.

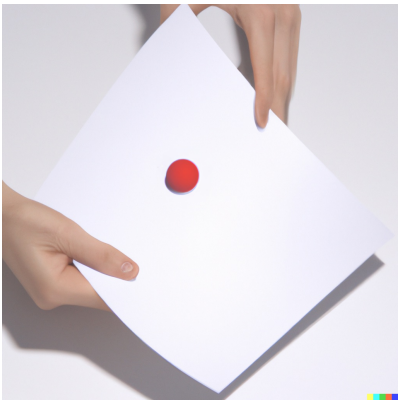


FIGURE 1.1: A ball rolling down a curved sheet of paper.

to command the actuators so that the manipulated object moves toward the desired position.

We pay special attention to micro-manipulation[2] by DEP, and more specifically, to feedback micro-manipulation by DEP. That is because the topics covered by this dissertation thesis are motivated by a research project on distributed manipulation by DEP. Nevertheless, one of the contributions —an algorithm for distributed optimization—steps out of this application area and is also applicable to manipulation by other physical force fields.

The rest of this introduction is devoted to a brief description of what can be found in the following chapters, what problems the chapters solve, and what are the contributions of the chapters. Each chapter corresponds either to an accepted, or even published journal paper, or a paper in preparation for submission.

In a nutshell, there must be a sensor when there is feedback. Thus, we discuss what kinds of sensors can be found in the literature and what limitations they have. Then, we explain why there is a need for a simple-enough model of DEP force in feedback DEP micro-manipulation, why it is necessary to compute its inversion, and how distributed optimization can help us do so. Lastly, we describe a compact manipulation platform that is a sort of culmination of all previous chapters because it integrates their results into one.

2 It is called micro-manipulation by DEP because one can typically manipulate only objects no larger than a few tens or hundreds of micro-meters by DEP.

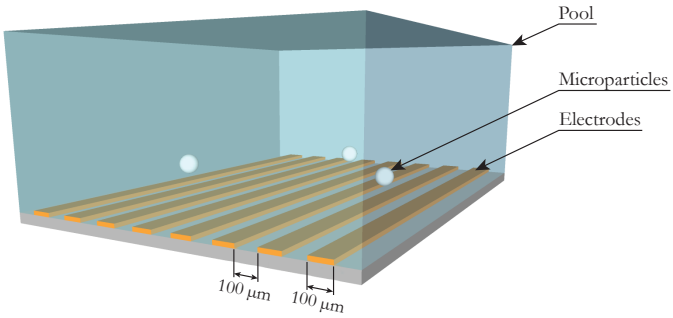


FIGURE 1.2: A sketch of micromanipulation by DEP. The microparticles are dispersed in a pool with deionized water. The electric field around the microparticles is shaped by application of varying potentials on the parallel electrodes bellow the microparticles.

DISTRIBUTED MANIPULATION BY SHAPING FORCE FIELDS

When an object is exposed to a non-zero net force, it accelerates in the direction of the force. Thus, we can move the object by shaping the force field surrounding it. When the force field is shaped by some actuators distributed in space, we talk about *distributed manipulation*.

There are many principles used in distributed manipulation. Let us mention a few. Tiny devices can be used to create mechanical force acting on manipulated objects. An array of omnidirectional wheels can be used to move objects. By shaping the electric field, one can induce a dielectrophoretic force moving the manipulated object (dielectrophoresis). Ultrasonic waves generated by ultrasonic transducers can also be used to move objects (acoustophoresis). Similarly, electromagnets can shape a magnetic force field to move objects (magnetophoresis). In this thesis, we encounter only dielectrophoresis, magnetophoresis, and acoustophoresis. However, as already mentioned, we mainly talk about DEP micro-manipulation.

DEP is a physical phenomenon enabling us to develop a force on polarizable particles by shaping the surrounding electric field. The electric field is usually created and shaped by the application of varying potentials on electrodes nearby the manipulated objects. As DEP can manipulate objects without any contact, it is especially well suited for contact-less manipulation in biology or medicine. An example of a very simple manipulation platform is sketched in Fig. 1.2.

Distributed manipulation platforms can be either sensorless or equipped with sensing capabilities. When there is no sensor, the controller does not know the position of the manipulated objects and can only "blindly" rely on some precomputed actions. When there is a sensor, and the controller reshapes the force field based on the currently sensed positions of the manipulated objects, we talk about *feedback distributed manipulation*. In this thesis, we discuss only feedback-distributed manipulation.

Without feedback, the micro-manipulation by DEP is mostly limited either to single-object manipulation or to separating objects of two or more classes. However, when we add feedback to DEP micro-manipulation, we gain the capability of parallel multi-object manipulation. Unfortunately, feedback also increases the complexity of the control system because we need a position sensor, and we also need to evaluate a DEP force model. That is because a digital control system works with some finite sampling periods, within which the work is structured into stages. The stages are depicted in Fig. 1.3. In each sampling period, the control system measures the positions of manipulated objects. Based on the measured positions and some reference positions, the control system computes forces acting on the manipulated objects such that the objects move toward the reference positions. Having the desired forces, the control system solves an optimization problem where it optimizes the actuators' commands (based on a force model) so that the desired forces are generated. That's the role of the force controller block. Then, the control system applies the optimized actuator commands and the cycle repeats in the next control period.

Let us discuss the specifics of position measurement in manipulation by DEP and how this thesis contributes to this field.

POSITION SENSING

For feedback micro-manipulation by DEP, we need to measure the position of micro-sized objects with accuracy in tens of micrometers. We need to measure the position in real time and do so frequently (at least 10 Hz) and with low latency. In addition, the manipulated objects can move in 3D. They move in the plane of the electrode array, but they also levitate above the electrode array (see Fig 1.2). When we evaluate the DEP force model, we must feed it with a full 3D position. If the levitation height was unknown, the manipulation accuracy would deteriorate. Thus, the position should be sensed in 3D. This requirement by itself eliminates conventional

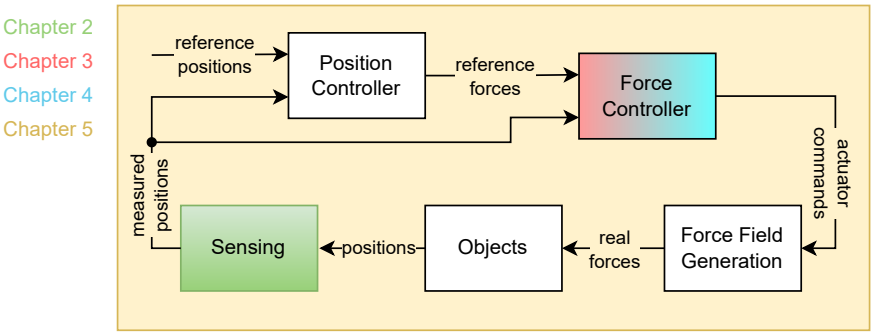


FIGURE 1.3: Block diagram of manipulation platform based on feedback force-field shaping. Individual blocks are highlighted by a specific color or a combination of colors if a chapter of this thesis is devoted to them. The whole diagram is bounded by the yellow rectangle because all blocks appear in Chapter 5.

microscopes—because these can measure the position only in 2D—and calls for some other kind of sensor.

We mention two ways to measure the positions of micro-objects in 3D: confocal microscopy and digital holography. Confocal microscopy measures the position in 3D by physically moving the optical apparatus of the microscope. The motion is typically slow, so it cannot be used to measure the position in real time for the purpose of feedback manipulation. In contrast, digital holography does not involve any physical motion. The 3D positions of objects can be extracted by digital processing of one captured image only. This is due to the fact that holograms do not capture only the 2D position (as conventional microscopes do), but they also encode the levitation height of the objects. Nevertheless, extracting information about the levitation height is computationally demanding, so these methods are also unsuitable for real-time use. Since no suitable sensor was found in the literature, a new one must be developed.

A novel method for measuring the 3D positions of micro-objects is sketched in Fig. 1.4 and described in detail in Chapter 2. This method uses twin-beam illumination. The manipulated objects are illuminated from above by two light sources that are mutually shifted. Thus, each manipulated object has two "shadows", which are also mutually shifted. The shadows are captured by an image sensor placed under the manipulated objects. The 3D positions of the manipulated objects can be extracted from

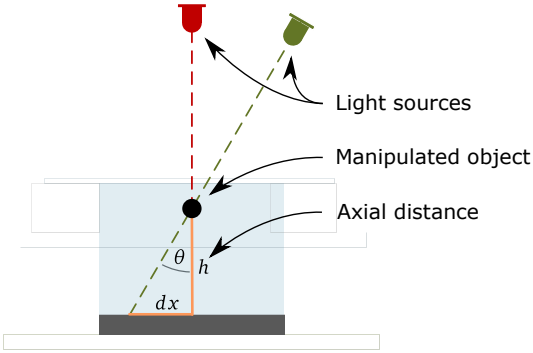


FIGURE 1.4: A principal sketch of the twin-beam position measurement method.

the locations of the captured shadows by image processing. This method runs in real time and requires only a very cost-effective and compact hardware setup. The method is further improved in Chapter 5 so that it not only measures 3D positions but also provides images that look like they were captured by a microscope.

Having a position sensor, we can move on to the force model. In fact, why exactly do we need it?

DEP MODEL

In manipulation by DEP, the position of the manipulated objects is controlled by changing the surrounding force field. The force field is shaped by electrical potentials at electrodes (or by some other actuator commands in the case of some other force fields). We cannot directly set the force field, we must do so by changing the potentials, and for that, we need a force model relating the potentials with the generated force field. As already mentioned, we, in fact, use the model in an optimization problem every control period. Therefore, the model must be simple enough so that it is cheap to evaluate and can be used in real time.

Unfortunately, a commonly used model of DEP is far from being simple and cheap to evaluate. The model is a *partial-differential equation (PDE)* with mixed boundary conditions. Such models are usually hard to solve analytically to closed-form solutions. The DEP model is no different and must be solved numerically. Thus, an exact closed-form model relating the potentials to the generated DEP force is out of our reach.

In Chapter 3, a new approximate control-oriented model of DEP force is introduced. It uses a numerical solution of some base problems and *Green's functions* to approximate the original PDE (with mixed boundary conditions) by one (with Dirichlet boundary conditions) that has a closed-form solution. The closed-form solution is simple enough to be evaluated in real time, and, at the same time, it is accurate enough to be used for control purposes.

Now we have a DEP force model that can be used in real time. How do we use it to generate the desired forces outputted by the position controller (see Fig 1.3)?

DISTRIBUTED OPTIMIZATION

The DEP force model described in the previous section outputs DEP force for given potentials. However, we need the opposite for control purposes. We need to determine the potentials for a given desired DEP force. Sadly, the model cannot be trivially inverted; there is no way to get a closed-form expression for the inverted model. Thus we pose the problem of the model inversion as an optimization problem where the objective is to minimize the difference between the generated and desired forces, and the decision variables are the potentials.

The model-inversion optimization problem, unfortunately, turns out to be non-convex and thus not easily solvable. Furthermore, the optimization problem gets harder to solve with the growing number of manipulated objects and actuators. To make the problem easier to solve and increase the scalability and robustness of the distributed manipulation platforms, we propose to decompose the optimization problem into several sub-problems that are solved in coordination.

Chapter 4 proposes a distributed optimization algorithm to solve the model-inversion problem. The main idea behind the decomposition can be described rather simply. The optimization problem is decomposed by manipulated objects. Each manipulated object is viewed as a virtual agent that optimizes only the force acting on itself and for that, it uses only actuators in its neighborhood. Since there can be two or more agents using the same actuators, they must reach a consensus on the actuator commands. We use the *Alternating Direction Method of Multipliers ADMM* for the formulation of the consensus problem. Even though we focused on DEP in the last two sections, the proposed distributed algorithm is applicable to a broader class of force fields than just to DEP. The algorithm was numerically tested on dielectrophoresis, magnetophoresis, and acoustophoresis.

COMPACT DEP MANIPULATION PLATFORM

Feedback manipulation by DEP with the proposed twin-beam position measurement method requires only relatively simple and compact hardware: an electrode array, some electronics for setting the potentials at the electrodes, a lensless image sensor, two light sources and a computer. This point is proven by a compact manipulation platform presented in Chapter 5, the last chapter of this thesis.

The manipulation platform is shown in Fig. 1.5. It measures the 3D positions of micro-objects by using the twin-beam measurement method. It uses digital holography to display the objects and Nvidia Jetson AGX for efficient image processing (both object tracking and digital holography). The manipulation are is approximately $1\,\text{mm} \times 1\,\text{mm}$. That is given by the currently used electrode array having 56 electrodes. However, the electrode array can be easily replaced by a larger one, and also the electronics for generating the electrical potentials are designed so that larger electrode arrays are supported. The manipulation platform without the computer fits in a box of size $95\,\text{mm} \times 91\,\text{mm} \times 147\,\text{mm}$. We demonstrated by experiments that the manipulation platform is capable of independent positioning of multiple objects and that it can position objects in 3D.

CONTRIBUTION

Let us summarize the main contributions of this thesis. This thesis is mainly motivated by one specific application domain: feedback manipulation through shaping dielectrophoretic force fields. Thus also, the main results of this thesis apply to this domain. Nevertheless, they are not limited to this specific domain. We developed a novel position sensor for dielectrophoretic manipulation platforms. We also developed a novel control-oriented model of dielectrophoretic force. In addition, we developed a novel distributed optimization algorithm solving the problem of finding electric potentials such that a desired dielectrophoretic force field is developed. Lastly, we developed a novel compact dielectrophoretic manipulation platform.

All contributions—except the compact manipulation platform—were published in peer-reviewed journals. A paper describing the manipulation platform is, at the time of writing this thesis, in preparation for submission to a peer-reviewed journal.

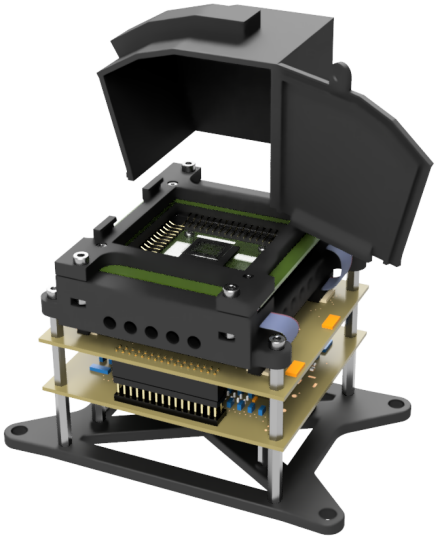


FIGURE 1.5: Compact DEP manipulation platform with the top part detached. The hardware setup without the computational unit fits within a box 95 mm × 91 mm × 147 mm.

ORGANIZATION OF THE THESIS

Each chapter in this is a reprint of a journal paper that was either already published (Chapters 2 to 4) or in preparation for submission (Chapter 5). This thesis is organized as follows. Chapter 2 describes a novel method for measuring the positions of spherical objects with a diameter in the range of tens of micrometers in real time. Chapter 3 proposes a closed-form model of dielectrophoretic force that is suitable for control purposes. Chapter 4 introduces a distributed optimization algorithm solving the force model inversion problem emerging in feedback distributed manipulation. Chapter 5 describes a DEP manipulation platform that integrates all actuators, sensing, and associated electronics into one compact setup that is also cost-effective.

2

TWIN-BEAM REAL-TIME POSITION ESTIMATION OF MICRO-OBJECTS IN 3D

Various optical methods for measuring positions of micro-objects in 3D have been reported in the literature. Nevertheless, majority of them are not suitable for real-time operation, which is needed, for example, for feedback position control. In this paper, we present a method for real-time estimation of the position of micro-objects in 3D; the method is based on twin-beam illumination and it requires only a very simple hardware setup whose essential part is a standard image sensor without any lens. Performance of the proposed method is tested during a micro-manipulation task in which the estimated position served as a feedback for the controller. The experiments show that the estimate is accurate to within $\sim$3 µm in the lateral position and $\sim$7 µm in the axial distance with the refresh rate of 10 Hz. Although the experiments are done using spherical objects, the presented method could be modified to handle non-spherical objects as well.

This chapter was published in:
Gurtner, M. & Zemánek, J. Twin-beam real-time position estimation of micro-objects in 3D. *Measurement Science and Technology* **27**, 127003 (2016)

2.1 INTRODUCTION

Estimation of positions for micro-objects in 3D is of great interest in many research domains. In microfluidics, a velocity profile of the fluid can be measured by tracking micro-objects suspended in the fluid [2]. For example, blood flow, which could indicate circulatory diseases, can be determined by measuring blood cells' trajectories [3]. Similarly, an analysis of motion of bubbles in air-water mixture can be carried out [4]. In microbiology, trajectories of sperm cells can be used to determine their motility [5].

There are many ways how to estimate 3D position of micro-objects. *Confocal microscopy* can be used for 3D position estimation [6], but it provides only a very limited time resolution since it involves hardware motion. A variety of methods are based on *digital holography* [7] where the position is estimated computationally; for a review, see [8]. These methods are either based on fitting the micro-object's hologram to a model describing the appearance of the hologram parametrized by the *axial distance* [2, 9], or on back-propagation of the hologram [10, 11]. Fitting the observed holograms to the model provides very accurate estimates of the position (up to nanometer resolution), but it is computationally very demanding and the holograms have to be captured with very high resolution—this is usually achieved by an objective lens which results in reduced observable area. Back-propagation allows us to estimate the axial distance of a micro-object by identification of the distance for which the back-propagated hologram fits to the image of the micro-objects. The back-propagation itself is not computationally demanding, but—with the exception of the method described in [12]—it has to be carried out several times. Another approach is to use multiple light sources and subsequently illuminate the micro-objects under different angles [13, 14]. Then the individual micro-object's "shadows" on the image sensor are shifted with respect to each other and this shift corresponds to the axial distance of the micro-object.

The majority of methods estimating the position of micro-objects are intended for an analysis of the motion and rely on off-line processing of the recorded data. However, when it comes to feedback position control, one needs to know the positions of the manipulated micro-objects in real-time. Hence, we were motivated to develop a novel method especially suitable for real-time processing and micro-manipulation applications. This method is based on twin-beam illumination and it needs only a very cost-effective and compact hardware setup. The setup consists of two light sources simultaneously illuminating the micro-objects and a standard image

sensor (no lens is necessary) capturing the "shadows", or more precisely interference patterns, from the micro-objects. The position of a micro-object is computationally estimated from the lateral shift of the corresponding interference patterns. The presented method is tested using a reference measurement from another camera during a micro-manipulation task.

2.2 HARDWARE SETUP

Before we delve into the description of the proposed method, we describe the hardware setup (see figure 2.1). The objects to be tracked are polystyrene spherical micro-objects of diameter 50 µm and they are suspended in water contained in a 2 mm deep pool above an electrode array. The micro-objects are manipulated through the phenomenon known as *dielectrophoresis*—application of different potentials on the electrodes generates a force acting on the micro-objects [15]. Light sources are red (625 nm) and green (525 nm) LEDs which are butt-coupled to plastic optical fibers (500 µm in diameter). The tips of the fibers are placed so, that the light from the red LED illuminates the micro-objects from above and the light from the green LED falls under approximately 45°. The tips of the fibers are approximately 6 cm above the pool. Since the light is partially coherent—spatially due to the diameter of the optical fibers and temporally due to the bandwidth of the LEDs—it forms interference patterns on the image sensor (e-Con Systems, See3CAM_10CUG, 1.3 megapixels, 3.75 µm pixel size) which is approximately 1.5 mm below the micro-objects. The image sensor is cooled by a Peltier cooler to avoid heating-up the water in the pool which would cause undesired heat-driven currents. In our case, the *lateral position* of the micro-objects is restricted by the size of the electrode array to approximately 1.5 mm × 1.5 mm and the axial distance by dielectrophoresis to maximum levitation height of 200 µm above the electrode array. For the calibration of the method and validation purposes, there is also a side-view camera that allows us—in a very limited depth of field—to see the micro-objects in the pool from aside.

2.3 WORKING PRINCIPLE

The principle of the proposed method is depicted in figure 2.1. As the micro-objects are illuminated by one source from the top and by the other one from the side, there are two interference patterns on the image sensor under each micro-object. These two interference patterns are laterally shifted with

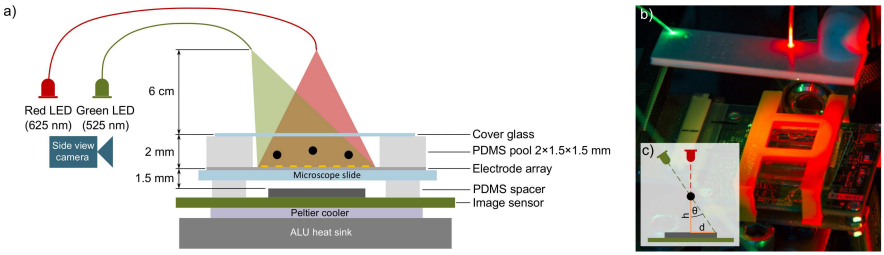


FIGURE 2.1: Diagrams (a) of the hardware setup and (b) of the working principle. PDMS is an abbreviation of polydimethylsiloxane.

respect to each other and this shift corresponds to the axial distance of the micro-object. Because the wavelengths of the LEDs were chosen so that they match the peaks in the sensitivity of the red and green channels of the image sensor, the red and green channels contains only interference patterns from the perpendicular and oblique illumination, respectively (see figure 2.2). If we assume that the micro-objects are illuminated by planar waves and neglect the refraction of light, the dependence of the axial distance of the micro-objects on the lateral shift is simply given by

$$h = d\frac{1}{\tan\theta}, \tag{2.1}$$

where h is the axial distance of a micro-object from the image sensor, d is the lateral shift of its interference patterns in the captured image and θ is the angle of the oblique incidence (see figure 2.1(c)).

Nevertheless, the tips of the optical fibers behave more like sources of spherical waves and the refraction of light clearly occurs since the light propagate through several different media on the way from the tips to the image sensor. The assumption of planar wave illumination is a good approximation if the tips of the optical fibers are sufficiently far away from the micro-objects and the lateral shift of the interference patterns is measured close enough to the micro-objects—ideally in the same medium to avoid the additional refraction of light. However, putting farther the light sources would enlarge the hardware setup and would require more energy for the same intensity of light incident on the image sensor. Putting the image sensor closer to the micro-objects is also rather difficult because that would mean making the electrode array and the supporting microscope glass thinner. To overcome this, side-marks are placed along the electrodes (see figure 2.2) which allows us to find a transformation from the image

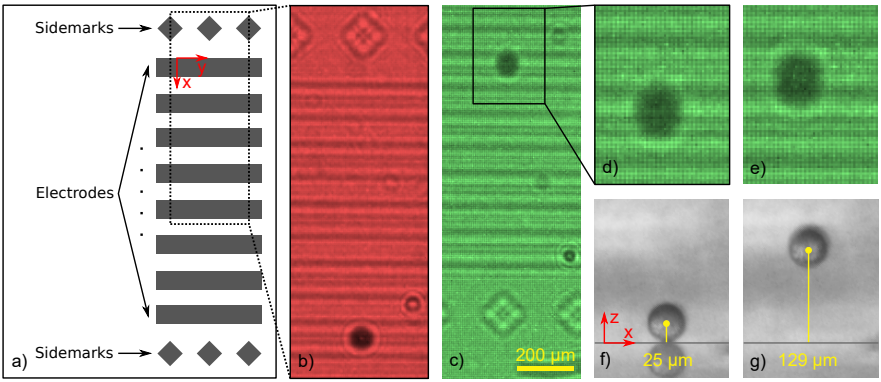


FIGURE 2.2: Demonstration of the principle of the proposed method. (a) Illustration of the electrode array with the side-marks. Images (b) and (c) are red and green channels of a cut-out of a captured image and they contain only interference patterns from perpendicular and oblique illumination, respectively. The images contain shifted interference patterns from the electrode array and one micro-object. (d-e) Blown-up regions of the green channel for a micro-object located at different levitation heights as it is shown from the side view in (f-g). The interference pattern of the micro-object (d) is shifted with respect to (e) and this shift corresponds to the difference in the axial distance of the micro-object.

coordinate system to the electrode array coordinate system. This way we can effectively measure the lateral shift at the level of the electrode array, which is very close to the micro-objects. Hence, we significantly reduced the influence of refraction of light and the influence of the non-planar illumination.

We assume that the transformations from the red and green channels (image coordinate systems) to the electrode array coordinate system can be described by a *projective transformation* [16]. That is, for the red channel, image coordinates $(x_{\mathrm{im}}, y_{\mathrm{im}})$ are transformed to electrode array coordinates $(x_{\mathrm{el}}, y_{\mathrm{el}})$ by the following relation

$$\begin{bmatrix} x_{\mathrm{el}} w \\ y_{\mathrm{el}} w \\ w \end{bmatrix} = \mathsf{H}_{\mathrm{R}} \begin{bmatrix} x_{\mathrm{im}} \\ y_{\mathrm{im}} \\ 1 \end{bmatrix}, \tag{2.2}$$

where $H_R \in \mathbb{R}^{3\times3}$. The same relation applies for the green channel, only the transformation matrix differs.

This is where the side-marks are useful; in order to determine the parameters of the projective transformation, one needs at least four pairs of corresponding points in both coordinate systems [16]. Relative positions of side-marks in the electrode array coordinate system are known and positions of several—for robustness, more than the needed four—side marks in the image coordinate systems are provided by a user. Therefore, such transformation parameters can be found and we can transform the positions of interference patterns to the electrode array coordinate system and measure their lateral shift there.

Now, we identify the precise locations of individual interference patterns and pair the patterns corresponding to the same micro-particle in the red and green channels. At the initial stage, approximate positions of the interference patterns from the perpendicular illumination (red channel) are provided by the user. Since the axial distance of the micro-objects is limited to a very narrow range, the mutual position of the interference patterns from the perpendicular and oblique illumination differs only slightly (up to 14 pixels). Thus, given the position of interference patterns from the perpendicular illumination the approximate position of the corresponding interference patterns from the oblique illumination can be calculated. To refine the approximate positions, the color channels are back-propagated to a distance where the interference patterns focus to a point. We do this because it is easier to determine a precise location of a focused point than of a larger interference pattern. The back-propagation is carried out by calculating the Rayleigh-Sommerfeld diffraction integral [17] which is numerically done by the following relation

$$I_z(x_{\mathrm{im}}, y_{\mathrm{im}}) = \mathcal{F}^{-1}\left\{H_{-z}(f_x, f_y)\mathcal{F}\left\{I(x_{\mathrm{im}}, y_{\mathrm{im}})\right\}\right\}, \tag{2.3}$$

where $(x_{\mathrm{im}}, y_{\mathrm{im}})$ are the image coordinates, (f_x, f_y) are the spatial frequencies, I is the original image, I_z is the image back-propagated to a distance z, $\mathcal{F}$ and $\mathcal{F}^{-1}$ are Fourier and inverse Fourier transformations, respectively, and

$$H_z(f_x, f_y) = \begin{cases} \exp\left(i2\pi z \frac{n}{\lambda}\sqrt{1 - \left(\frac{\lambda f_x}{n}\right)^2 - \left(\frac{\lambda f_y}{n}\right)^2}\right), & \sqrt{f_x^2 + f_y^2} \leq \frac{n}{\lambda}, \\ 0, & \text{otherwise}, \end{cases} \tag{2.4}$$

is Fourier transform of the Rayleigh-Sommerfeld propagator, where λ is the wave length of the illuminating light and n is the refractive index. Despite

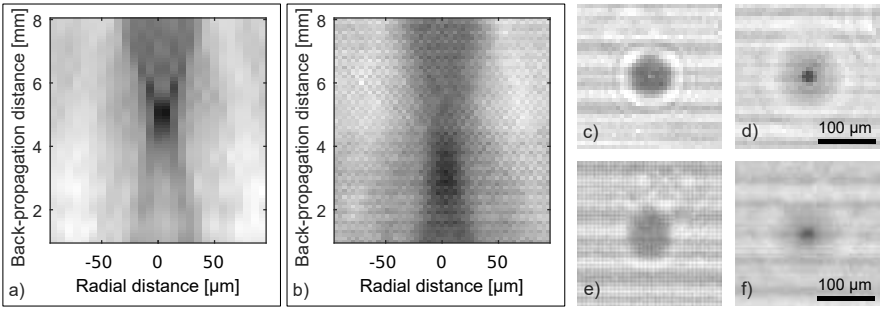


FIGURE 2.3: Back-propagation of an interference pattern from a micro-sphere: Images (a) and (b) show the dependence of radial intensity of the interference pattern from perpendicular and oblique illumination, respectively, on the back-propagation distance. Images (c) and (e) display raw interference patterns from perpendicular and oblique illumination, respectively, and (d) and (f) shows their back-propagation to a distance where they are focused to a point. Units of back-propagation distance correspond to the fixed refractive index.

the fact that the light propagates through several media on the way from the microparticle to the image sensor, we use a fixed value of the refractive index to make the back-propagation computationally faster.

The back-propagation is illustrated in figure 2.3. For each light source, we can separately fix a back-propagation distance for which all the interference patterns are focused to a point, no matter where the micro-objects are located.

The position of focused interference patterns is estimated by computing the center of mass of a small region around the approximate position (the position in the previous frame or, at the initial stage, the position given by the user). To make the estimate more accurate, the center of mass is calculated for regions of successively smaller sizes. This eliminates the influence of the surrounding specks.

What remains to be done is to identify the constant $1/\tan\theta$ in (2.1). For this purpose we have the side-view camera in the hardware setup. The side-view camera allows us to measure the levitation height (axial distance) of micro-objects within a limited depth of field directly (see figure 2.2 (f-g)). We manipulate a micro-object to several levitation heights and measure those heights and lateral shifts of the corresponding interference patterns. The constant $1/\tan\theta$ is then identified by fitting (2.1) to the set of measured

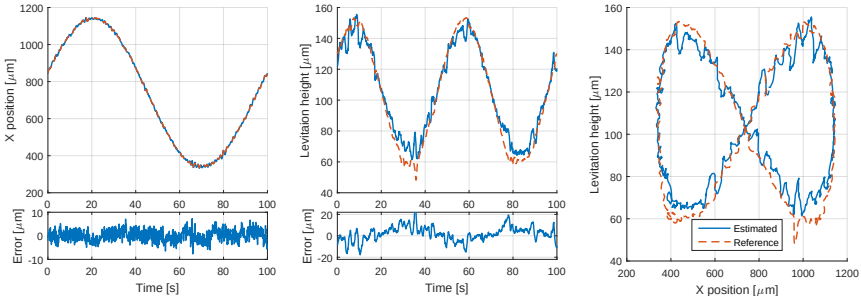


FIGURE 2.4: Comparison of the estimated positions by the proposed method with the reference measurements obtained by the side-view camera.

points. This calibration has to be done only once for the hardware setup then the side-view camera is not needed any more.

2.4 EXPERIMENTAL RESULTS

To validate the performance of the proposed method we manipulate a micro-object (polystyrene microsphere with 50 µm in diameter) along a figure eight trajectory and compare the position estimated by the proposed method with the reference measurement obtained from the side-view camera with accuracy ∼0.25 µm. The proposed method is implemented in Simulink. The estimation is carried out in real-time at 10 Hz on an ordinary PC (Intel Core i7, 8 GB RAM) and it is used in the feedback loop of the control algorithm described in [18]. The estimation algorithm itself takes only 40 ms, the remaining 60 ms is taken by the control algorithm and the execution overhead. The comparison is displayed in the form of graphs in figure 2.4 and in the form of video (containing also the side-view) in attached file Movie 1. The side-view camera enables us to measure only one coordinate of the lateral position, but from the method of estimation, the estimate in the other coordinate has necessarily the same accuracy. The standard deviation of the error in x-coordinate is 2.41 µm (0.6 px) and in the levitation height 6.64 µm. Even though the experiment is performed with only one micro-object, there is no obstacle preventing exploiting the proposed method for tracking of several micro-objects. Such an experiment is shown in attached file Movie 2.

2.5 DISCUSSION AND CONCLUSION

We have developed a simple novel method for real-time estimation of position of spherical micro-objects. The method requires only a very simple, cost-effective and compact hardware setup. We demonstrated the accuracy to be within $\sim$3 µm in the lateral position and $\sim$7 µm in the axial distance. Since the accuracy depends on precise localization of the interference patterns it can be improved by using an image sensor with smaller pixels. On the other hand, this usually reduces an observable area. Furthermore, we successfully used the method for real-time manipulation of a micro-object. Despite the fact that the method is developed for transparent spherical micro-objects, it can be potentially extended to track non-spherical and/or opaque micro-objects as well. The only thing that would have to change is the localization procedure for the interference patterns, because they might not focus to a point any more. Concerning limitations of the proposed method, if the micro-objects are in contact or located at the same lateral position (they lie along the same axial line) it might be difficult to track them with the current system.

REFERENCES

1. Gurtner, M. & Zemánek, J. Twin-beam real-time position estimation of micro-objects in 3D. *Measurement Science and Technology* **27**, 127003 (2016).
2. Cheong, F. C., Sun, B., Dreyfus, R., Amato-Grill, J., Xiao, K., Dixon, L. & Grier, D. G. Flow visualization and flow cytometry with holographic video microscopy. *Opt. Express* **17**, 13071 (2009).
3. Choi, Y.-S. & Lee, S.-J. Three-dimensional volumetric measurement of red blood cell motion using digital holographic microscopy. *Appl. Opt.* **48**, 2983 (2009).
4. Tian, L., Loomis, N., Domínguez-Caballero, J. A. & Barbastathis, G. Quantitative measurement of size and three-dimensional position of fast-moving bubbles in air-water mixture flows using digital holography. *Appl. Opt.* **49**, 1549 (2010).
5. Di Caprio, G., El Mallahi, A., Ferraro, P., Dale, R., Coppola, G., Dale, B., Coppola, G. & Dubois, F. 4D tracking of clinical seminal samples for quantitative characterization of motility parameters. *Biomed. Opt. Express* **5**, 690 (2014).

6. Van Blaaderen, A. & Wiltzius, P. Real-space structure of colloidal hard-sphere glasses. *Science* **270**, 1177 (1995).
7. Schnars, U., Falldorf, C., Watson, J. & Jüptner, W. *Digital Holography and Wavefront Sensing* (Springer, Berlin, Heidelberg, 2015).
8. Yu, X., Hong, J., Liu, C. & Kim, M. K. Review of digital holographic microscopy for three-dimensional profiling and tracking. *Opt. Eng.* **53**, 112306 (2014).
9. Guerrero-Viramontes, J. A., Moreno-Hernández, D., Mendoza-Santoyo, F. & Funes-Gallanzi, M. 3D particle positioning from CCD images using the generalized Lorenz–Mie and Huygens–Fresnel theories. *Measurement Science and Technology* **17**, 2328 (2006).
10. Lee, S. & Grier, D. G. Holographic microscopy of holographically trapped three-dimensional structures. *Opt. Express* **15**, 1505 (2007).
11. Dubois, F., Schockaert, C., Callens, N. & Yourassowsky, C. Focus plane detection criteria in digital holography microscopy by amplitude analysis. *Opt. Express* **14**, 5895 (2006).
12. Bouchal, P. & Bouchal, Z. Non-iterative holographic axial localization using complex amplitude of diffraction-free vortices. *Opt. Express* **22**, 30200 (2014).
13. Su, T.-W., Isikman, S. O., Bishara, W., Tseng, D., Erlinger, A. & Ozcan, A. Multi-angle lensless digital holography for depth resolved imaging on a chip. *Opt. Express* **18**, 9690 (2010).
14. Memmolo, P., Finizio, A., Paturzo, M., Miccio, L. & Ferraro, P. Twin-beams digital holography for 3D tracking and quantitative phase-contrast microscopy in microfluidics. *Opt. Express* **19**, 25833 (2011).
15. Morgan, H. & Green, N. G. *AC electrokinetics: colloids and nanoparticles* **2** (Research Studies Press, Philadelphia, PA, 2002).
16. Hartley, R. & Zisserman, A. *Multiple View Geometry in Computer Vision* 2 edition (Cambridge University Press, Cambridge, UK ; New York, 2004).
17. Goodman, J. W. *Introduction to Fourier optics* (Roberts and Company Publishers, Greenwood Village, 2005).
18. Zemánek, J., Michálek, T. & Hurák, Z. Feedback control for noise-aided parallel micromanipulation of several particles using dielectrophoresis. *Electrophoresis* **36**, 1451 (2015).

3

GREEN'S FUNCTION-BASED CONTROL-ORIENTED MODELING OF ELECTRIC FIELD FOR DIELECTROPHORESIS

In this paper, we propose a novel approach to obtaining a reliable and simple mathematical model of a dielectrophoretic force for model-based feedback micromanipulation. Any such model is expected to sufficiently accurately relate the voltages (electric potentials) applied to the electrodes to the resulting forces exerted on microparticles at given locations in the workspace. This model also has to be computationally simple enough to be used in real time as required by model-based feedback control. Most existing models involve solving two- or three-dimensional mixed boundary value problems. As such, they are usually analytically intractable and have to be solved numerically instead. A numerical solution is, however, infeasible in real time, hence such models are not suitable for feedback control. We present a novel approximation of the boundary value data for which a closed-form analytical solution is feasible; we solve a mixed boundary value problem numerically off-line only once, and based on this solution we approximate the mixed boundary conditions by Dirichlet boundary conditions. This way we get an approximated boundary value problem allowing the application of the analytical framework of Green's functions. Thus obtained closed-form analytical solution is amenable to real-time use and closely matches the numerical solution of the original exact problem.

This chapter is reproduced from Gurtner, M., Hengster-Movric, K. & Hurák, Z. Green's function-based control-oriented modeling of electric field for dielectrophoresis. *Journal of Applied Physics* **122**, 054903 (2017), with the permission of AIP Publishing.

3.1 INTRODUCTION

Since H. Pohl coined the term *dielectrophoresis (DEP)* and started studying this phenomenon in the 1950s and 1960s [2, 3], DEP has proved an efficient tool for transportation, separation, and characterization of microparticles such as biological cells (see Refs. [4, 5] for a recent survey and comprehensive introduction). More often than not, DEP is used to manipulate ensembles of large numbers of microparticles; it is these applications—without the need of adjusting the DEP force field in real time—where DEP reached a very mature state and is commercialized (for instance by ApoCell, Inc. [6]). However, recently some attempts were successful to use DEP in a feedback control scheme for a high accuracy noncontact manipulation of a single microparticle [7–10]; these developments can be viewed as a reopening of the topic first started in the 1990s [11]. The technology has also boosted development in this area; there are reported CMOS chips integrating both actuation and sensing and thus enabling individual and independent manipulation of thousands of cells [12]. This technology has later also been commercialized by Silicon Biosystems as a commercial product called DEPArray^{TM}. As opposed to DEP tweezers where DEP is used just to attract a microparticle and a human manipulates with it [13, 14], these non-contact tweezers are usually based on a feedback control scheme, typically invoke an automatic visual tracking and can manipulate with more microparticles simultaneously. The feedback control, in turn, requires a sufficiently accurate mathematical model of the underlying physical phenomenon of DEP. The relationship between the voltages applied to the microelectrodes and the DEP force exerted on a microparticle located at a given position needs to be evaluated periodically as the microparticle moves around the workspace. Sampling periods on a time-scale of few tens of milliseconds or even a few milliseconds are not unusual. The commonly used approaches to modeling DEP—which are based on numerical solution of the corresponding *Boundary Value Problem (BVP)*, typically using *Finite Elements Method (FEM)* or *Method of Moments (MOM)*—are not feasible in real time. It is possible to precompute and store these solutions in a computer memory (as reported in Ref. [7]) but this approach imposes stringent requirements on the volume of data stored. There are approaches described in the literature that provide analytical solutions [15–17]; however, they are only usable for simple electrode arrays and fixed harmonic voltage signals applied to the electrodes while the feedback control requires the ability to change the voltages in real time.

In this paper, we propose a modeling methodology that provides a computationally simple yet sufficiently accurate model of a DEP force for the purposes of feedback micromanipulation. We propose to combine numerical and analytical approaches to modeling of DEP. Existing models of DEP usually involve a numerical solution of an analytically intractable *mixed Boundary Value Problem (mBVP)* for the potential in the workspace. As the numerical solution is infeasible in real time and might be too large for storing in a computer memory, it is desirable to find a closed-form approximate analytical expression for the potential. To find such an expression, we solve numerically the original mBVP. Based on this numerical solution, we approximate the mBVP by a BVP for which the closed-form solution can be found by Green's functions. Using the approximate closed-form expression for the potential we obtain a model of DEP force that is computationally effective and requires almost no memory space. The numerical solution of the mBVP needs to be computed off-line and only once. Thus the high computational burden associated with the numerical solution is carried out off-line and the feedback control system uses only the approximate closed-form analytical solution in real time.

The paper is organized as follows. In Section 3.2, we briefly present the commonly used dipole model of DEP and show what prevents its direct use in feedback micromanipulation. In Section 3.3 we propose a control-oriented model derived from the dipole model by Green's functions. Experimental verification of the viability of the proposed model is provided in Section 3.4. The paper concludes with Section 3.5 where the main contributions of this paper are discussed.

3.2 FEEDBACK MANIPULATION BY DIELECTROPHORESIS

A great advantage of feedback manipulation by DEP, in contrast to the conventional use of DEP, is that it allows one to manipulate individual microparticles. Nevertheless, this comes with the cost of higher computational demands on the control system because the voltages applied to the electrodes cannot be precomputed anymore and have to be adjusted in real time as required by the feedback loop.

We can explain this with an aid of Fig. 3.1 depicting the scheme of feedback manipulation by DEP. The measured position of the microparticle is subtracted from the required position. The deviation is fed to a control system that calculates a DEP force needed to reduce this deviation—to steer the microparticle towards the required position. Therefore, the voltages

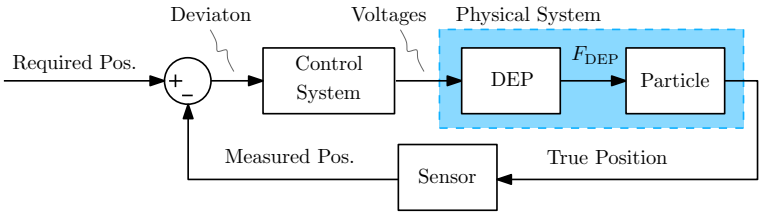


FIGURE 3.1: A block diagram of feedback manipulation by DEP.

on electrodes are required to generate such a force. Such voltages are then applied to the electrodes and the generated DEP force acts on the microparticle moving it towards the required position. The new position is then measured and the whole cycle is repeated. The crucial part of this algorithm is hidden in the control system where, in order to compute the voltages generating the desired DEP force, a model relating the voltages to the DEP force has to be used in real time [7].

Nevertheless, exact DEP models are rather complicated to use in real time. For instance, the widely used *dipole model* has the following form: time-averaged dielectrophoretic force acting on a homogeneous spherical particle in a harmonic field is [18]

$$\begin{aligned}\langle \mathbf{F}_{\mathrm{DEP}}(t)\rangle = \pi \quad & \varepsilon_{\mathrm{m}} r^3 \Big(\mathrm{Re}\{K(\omega)\}\nabla|\mathbf{E}|^2 + 2\mathrm{Im}\{K(\omega)\} \\ & \times \left(E_x^2 \nabla\varphi_x + E_y^2 \nabla\varphi_y + E_z^2 \nabla\varphi_z \right) \Big), \end{aligned} \tag{3.1}$$

where ε_{m} is the permittivity of the surrounding medium, r is the particle's radius, $K(\omega)$ is a frequency dependent constant known as Clausius–Mossotti factor, $\mathbf{E} = [E_x, E_y, E_z]$ is the amplitude of the harmonic electric field $\mathrm{Re}\{\mathbf{E}\, e^{i\omega t}\}$, the phase of the electric field is denoted by $\varphi_a, a \in \{x, y, z\}$ and finally, $\mathrm{Re}\{\cdot\}$ and $\mathrm{Im}\{\cdot\}$ denote real and imaginary parts of a complex number, respectively. For brevity, the spatial dependence is omitted in the notation.

According to (3.1), to determine the DEP force due to applied voltages to electrodes, one needs to know the electric field $\mathbf{E}$. The electric field $\mathbf{E}$ is given by $\mathbf{E} = -\nabla\phi$, where the potential ϕ is calculated from Laplace equation $\nabla^2\phi = 0$ with mixed boundary conditions. Orienting the reference frame so that the electrodes lie in the x-y plane and manipulated objects are situated above it, the domain is defined by the half-space $z > 0$. The boundary conditions are given by the voltages applied to the electrodes (Dirichlet boundary condition) and a zero-flux condition in the normal direction to the

electrode plane in the intervening space between the electrodes (Neumann boundary condition) [15]. This BVP is analytically intractable and can be solved only approximately by numerical solvers. Since the control algorithm is supposed to run in real time and the calculation of the DEP force must not take more then a few milliseconds, solving on-line this exact BVP numerically is infeasible.

A partial remedy to this issue is to express explicitly the dependence of $\mathbf{F}_{\mathrm{DEP}}$ on the voltages applied to the electrodes. By superposition, we express the net potential $\phi(x, y, z)$ as a weighted sum of normalized contributions from individual electrodes. That is,

$$\phi(x, y, z) = \sum_{i=1}^{n} u_i \phi_i(x, y, z), \tag{3.2}$$

where n is the number of electrodes, u_i serves as a scaling factor given by voltage on ith electrode, ϕ_i is the contribution to the net potential from the ith electrode when 1V is applied to it while the remaining electrodes are grounded. Now, to determine the net potential $\phi(x, y, z)$, we have to solve n BVPs ($\nabla^2 \phi_i = 0$, $i = 1, \ldots, n$) that are still analytically intractable, but that do not change with the voltages applied to the electrodes.

One can solve each of these BVPs numerically on a grid of points in advance, store the solution and use it as a look-up table in real time. Nevertheless, this lookup table grows unacceptably large. As an example, let the microparticles be manipulated within an area of size $1500\,\mu\mathrm{m} \times 300\,\mu\mathrm{m}$. If we grid this area equidistantly with points separated by $5\,\mu\mathrm{m}$, we obtain $300 \times 300 \times 60 = 5,400,000$ points. Naïve implementation of this approach would thus require to store $[E_x, E_y, E_z]$ and their relevant partial derivatives $[\frac{\partial E_x}{\partial x}, \frac{\partial E_x}{\partial y}, \frac{\partial E_x}{\partial z}, \frac{\partial E_y}{\partial y}, \frac{\partial E_y}{\partial z}]$ for each point in order to evaluate $\mathbf{F}_{\mathrm{DEP}}$ and all that is only for one electrode.

The volume of the data needed to be stored can be reduced by a method introduced by Kharboutly et al. [19]. They use a so-called *Boundary Element Method* and instead of storing directly the derivatives of the potential in points spread throughout all the domain, they store precomputed surface charge density in a grid on electrodes. In real time, when it is required to compute the DEP force at a point, the surface charge density is numerically integrated to calculate the electric field and its derivatives at that point and that allows the computation of the DEP force. Nevertheless, for the previous case that still means that a large portion—depending on what extent of the electrode plane is occupied by the electrodes—of $300 \times 300 = 90,000$ points have to be stored. Furthermore, the reduction in the volume of data comes

at the cost of higher computational complexity because all the stored data points are needed for evaluation of (3.1).

In this paper, we propose a different approach. We approximate the previously mentioned, analytically intractable boundary value data so that a closed-form approximated solution can be found.

3.3 GREEN'S FUNCTIONS FOR MODELING OF DIELECTROPHORESIS

The solution of the Laplace equation in a half-space domain, which is the case here, with Dirichlet conditions only can be transformed into an integration by use of the Green's theorem. The derivation can be found in Ref. [17] and the resulting formulas are

$$\phi(x,y,z) = \frac{z}{2\pi} \iint_{-\infty}^{\infty} \frac{h(x',y')}{[(x-x')^2+(y-y')^2+z^2]^{3/2}}\, dx'dy' \tag{3.3}$$

for a 3D case and

$$\phi(x,z) = \frac{z}{\pi} \int_{-\infty}^{\infty} \frac{h(x')}{(x-x')^2+z^2}\, dx' \tag{3.4}$$

for the 2D case where one axis, in our case y, is redundant, meaning the electrode array has infinitely long electrodes along y axis. The functions $h(x,y)$ and $h(x)$ are Dirichlet boundary conditions that represent values of the potential on the electrode plane, that is $\phi(x,y,0)$ and $\phi(x,0)$, respectively. Thus, to obtain a closed-form description of the potential—and subsequently also of the DEP force—one only needs to compute the integral (3.3) or (3.4). However, in order to achieve that, it is necessary to know $h(x,y)$ (or $h(x)$) and that means also the potential on the electrode plane in the intermediate space between the electrodes where the mixed boundary conditions impose a zero normal flux. Furthermore, functions $h(x,y)$ or $h(x)$ have to be such that the evaluated integral (3.3) or (3.4) is expressible as a closed-form expression containing only elementary functions; only then is the solution for the potential applicable in real-time feedback control.

To determine the values of the potential on the electrode plane in between the electrodes, various approximations of the decay of the potential away from the electrode can be found in the literature [15–17]. Nevertheless, they are all designed only for interdigitated electrode arrays. For more complex electrode array designs, they are either inapplicable or the integrals (3.3) and (3.4) are analytically intractable for the approximation of the potential

and have to be solved numerically. Then, however, formulating the solution of the BVP as an integration loses meaning since both the new and the original problem have to be solved numerically.

In this paper, instead, we numerically solve the original BVP with mixed boundary conditions. In order to obtain $h(x, y)$ (or $h(x)$), we approximate this numerical solution on the electrode plane (i.e. on the boundary of the domain) by an analytical model. As we require the integrals (3.3) and (3.4) to be expressible in closed form, we restrict the class of approximating models to piece-wise polynomial models in the 2D case and piece-wise constant models in the 3D case. Having the approximation of the potential on the electrode plane, an approximate closed-form expression for the potential in the half-space domain is obtained by evaluating the integral (3.3) (or (3.4)). Thus, by (3.1) and (3.2), we also get a model of DEP suitable for feedback control. Note, that the numerical solution is needed only to derive the control-oriented DEP model; the numerical solution is computed only once and off-line. Therefore, all the heavy computational burden is carried off-line and the control system uses the computationally more efficient model in real time.

It is worth mentioning that since the potential—as the solution of the Laplace equation—is a harmonic function, it is infinitely differentiable [20]. Furthermore, due to the *Maximum principle* [20] the error of the approximated potential diminishes as one moves further away from the electrodes. Thus, the accuracy of the model can be controlled by the accuracy of the approximation of the potential on the boundary of the domain.

In the remainder of the paper, we apply the described methodology to two electrode arrays.

3.3.1 *Example 1: Interdigitated Electrode Array*

Let us consider an electrode array with $(2n + 1)$ electrodes, the single electrode width b and center-to-center distance between the electrodes d (see Fig. 3.2(a)). We assume that the electrodes are infinitely long and thus drop the dependence on the y coordinate altogether.

We decompose the net potential by (3.2) into a weighted sum of normalized contributions ϕ_i from individual electrodes. Furthermore, we assume that the potential contribution ϕ_i is shift-invariant for a shift d along the x-axis. That means the potential contribution ϕ_i is identical for each electrode up to a shift. Mathematically,

$$\phi_i(x, z) = \phi_{i\pm1}(x \pm d, z), \quad i = -n + 1, \ldots, n - 1. \tag{3.5}$$

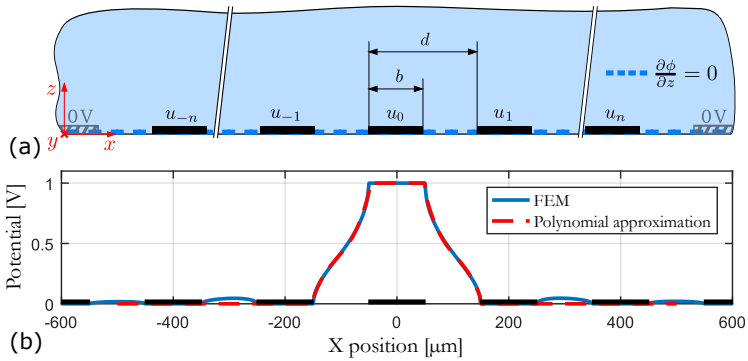


FIGURE 3.2: Interdigitated electrode array: (a) a side-view diagram and (b) an approximation of the boundary conditions. The black rectangles represent electrodes and the shaded rectangles on the left and right represent the possibly added grounding plates.

Clearly, this assumption does not hold for the electrodes close to the perimeter of the electrode array. For instance, the potential contribution $\phi_n(x,z)$ (i.e. from the electrode on the perimeter) decays more quickly towards the $(n-1)$th electrode, which is grounded, than towards the other side, where there is no grounded plate near. Nevertheless, this issue can be resolved (and the assumption (3.5) justified) by manufacturing grounding plates along the perimeter.

As a result of the assumption (3.5), we have to compute the integral (3.4) only for one $\phi_i(x,z)$, e.g. $\phi_0(x,z)$. The remaining potential contributions are determined simply by shifting, that is $\phi_i(x,z) = \phi_0(x+id,z)$ with $i \in \{-n,\dots,n\}$, and the net potential is then given by (3.2). Nevertheless, to compute (3.4) for $\phi_0(x,z)$, we need to know $h(x) := \phi_0(x,0)$ while the values of $\phi_0(x,0)$ are known only on the electrodes and unknown on the rest of the bottom boundary. To overcome this problem, we solve the original Laplace equation with mixed boundary conditions numerically by *Finite Element Method (FEM)* in COMSOL Multiphysics. Then, we take the values of the potential on the bottom boundary (i.e. $z = 0$) between the 0-th

electrode and its left adjacent electrode and fit a polynomial $p(x)$ to these values. The polynomial approximation of $h(x)$ is

$$\tilde{h}(x) = \begin{cases} p(x) & x \in \left[-d - \frac{b}{2}, -\frac{b}{2}\right), \\ 1 & x \in \left[-\frac{b}{2}, \frac{b}{2}\right], \\ p(-x) & x \in \left(\frac{b}{2}, d + \frac{b}{2}\right], \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

Specifically, for an electrode array with parameters $d = 2b = 200\,\mu\text{m}$, we fitted a third-order polynomial to the FEM solution and the fitted polynomial is

$$p(x) = 1.36 \times 10^{-6} x^3 + 4.34 \times 10^{-4} x^2 + 5.15 \times 10^{-2} x + 2.59, \tag{3.7}$$

where x is in micrometers.

The FEM solution together with the polynomial approximation is displayed in Fig. 3.2(b). Apparently, the polynomial approximation describes the FEM solution very accurately. However, one should not overlook the small humps in the gaps between the electrodes, which are completely omitted by the approximation.

With $\tilde{h}(x)$ approximating $\phi_0(x, 0)$ we can compute the integral (3.4) and obtain an approximate closed-form solution for $\phi_0(x, z)$. Then, by (3.5) and (3.2) we get the net potential $\phi(x, z)$ for any choice of electrode potentials u_i. Thus, we can compute the electric field intensity and the pertaining DEP force. We do not present the evaluated integral here, because it is rather lengthy and it would not serve any purpose. Nevertheless, it is crucial to mention that the evaluated integral is indeed expressible as a closed-form expression containing only elementary functions and thus it is easily applicable in real time.

To validate the proposed model, we compare DEP force fields computed by (3.1) for the potential obtained by numerical solution of the original BVP with mixed boundary conditions and for the potential obtained by solution of the approximated BVP. The comparison is shown in Fig. 3.3 (Multimedia view) . The comparison is carried out for an electrode array with nine electrodes ($n = 4$), with grounded perimeter (also visible in the figure) and with the single electrode width $b = 100\,\mu\text{m}$ and the distance between the electrodes $d = 200\,\mu\text{m}$. The remaining parameters are: $r = 25\,\mu\text{m}$, $\varepsilon_\text{m} = 7.08 \times 10^{-10}\,\text{F}\,\text{m}^{-1}$ and $K(\omega) = -0.4618 - 0.1454\text{i}$. As usual for the standing wave DEP, the harmonic signal applied on all electrodes has the

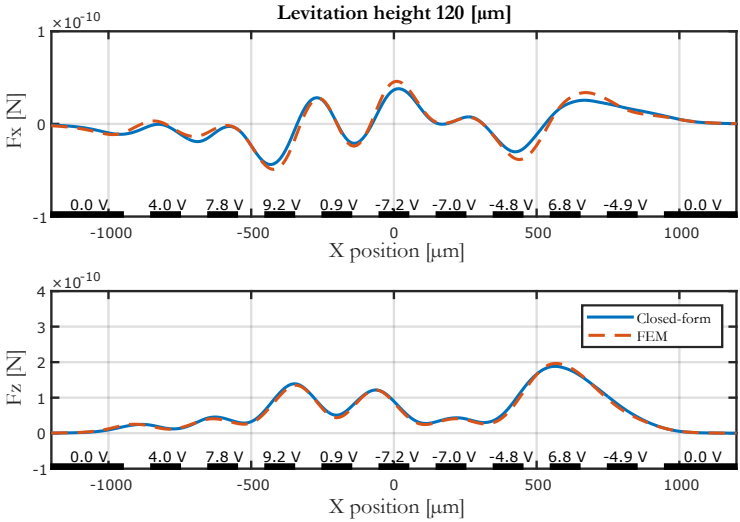


FIGURE 3.3: A component-wise comparison of DEP force fields computed for interdigitated electrode array ($d = 2b = 200\,\mu\text{m}$) numerically by FEM and analytically based on the approximate closed-form solution for the potential. The force fields are computed for the height of $120\,\mu\text{m}$ above the electrode array. The applied potentials are indicated above the electrodes represented by the black rectangles. (Multimedia view)

same frequency and phase. Since it is rather inconvenient to compare the vector fields, the comparison is done for one particular height above the electrodes (120 µm) and for varying potentials on the electrodes.

3.3.2 *Example 2: Four-Leaf Clover Electrode Array*

In the second example, we show a related approach how to approximate the values of the potential on the electrode plane for a more complex electrode array shown in Fig. 3.4(a). It consists of four quadrants and allows manipulation of microparticles in all three directions above the electrode array, as it was experimentally verified [21]. The width of the electrodes is $b = 50\,\mu\text{m}$ and the center-to-center distance between the electrodes is $d = 100\,\mu\text{m}$. We assume that the electrodes extend to infinity at one end.

Along similar lines as in Example 1, based on the superposition principle (3.2) we express the net potential as a weighted sum of normalized contributions ϕ_i from individual electrodes and make a similar assumption that the normalized contributions are identical up to a shift and/or rotation with respect to each other; for instance, for electrodes with indexes ranging from 1 to 7 (the indexes are shown in Fig. 3.4(a)), it holds that

$$\phi_{i+1}(x, y, z) = \phi_i(x - d, y + d, z), \tag{3.8}$$

where d is the center-to-center distance between the electrodes.

Again, thanks to this assumption, we need to compute the integral (3.3) only for one $\phi_i(x, y, z)$. In this case, we choose $\phi_{44}(x, y, z)$ because it is close to the center of the sector and, analogously to the previous example, the assumption (3.8) holds best for the electrodes in the center.

It remains to find the approximation of $h(x, y) := \phi_{44}(x, y, 0)$. In order to do that, we approximate the FEM solution for $\phi_{44}(x, y, 0)$ shown in Fig. 3.4(b). This time, however, the integral (3.3) is analytically intractable for $h(x, y)$ being polynomial or even linear—we are unable to express the integral in closed form for anything other than for constant boundary condition. Thus, instead of using a polynomial or linear approximation, the desired shape of the potential is constructed from blocks. Initially, we approximate the boundary condition in the roughest possible way; we assume that the potential between the electrodes drops immediately to zero as one moves away from the electrode (see Fig. 3.5(a)). Then summing the "scaled" and "shifted" versions of this boundary condition (see Fig. 3.5(b)) approximately gives the desired shape (see Fig. 3.5(c)).

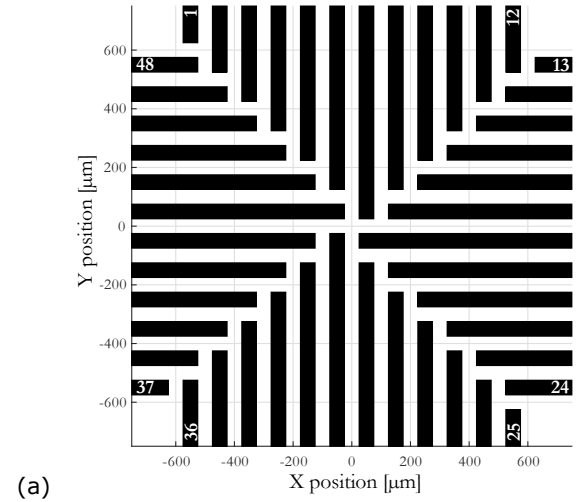


FIGURE 3.4: Four-leaf clover electrode array: (a) a top-view diagram and (b) FEM solution for the normalized potential contribution from one electrode.

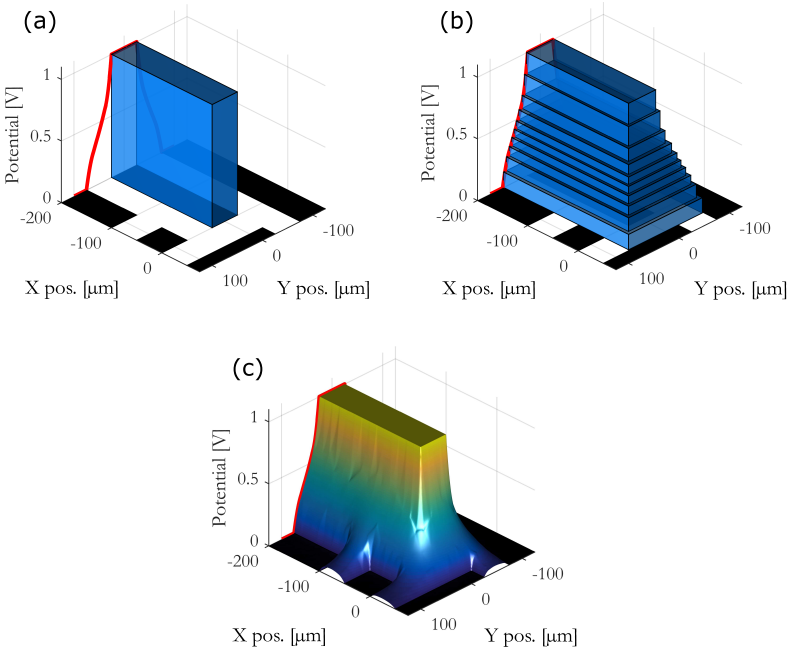


FIGURE 3.5: An approximation of the desired shape of the bottom boundary condition ($z = 0$) for the four-leaf clover electrode array: (a) one block approximation, (b) staircase approximation and (c) boundary condition obtained by FEM. Black rectangles represent electrodes.

Let us begin with the one-block approximation. We define the block boundary condition for one-side infinitely long electrode as

$$h_0(x,y) = \begin{cases} 1 & x \leq 0 \text{ and } y \in \left[-\frac{b}{2}, \frac{b}{2}\right], \\ 0 & \text{otherwise,} \end{cases} \tag{3.9}$$

where b is the width of the electrode. The staircase approximation of the desired shape is then obtained by

$$\tilde{h}(x,y) = \sum_{i=1}^{N} \alpha_i \, h_0\left(x - \frac{(\beta_i - 1)b}{2}, \frac{y}{\beta_i}\right), \tag{3.10}$$

where N is the number of blocks, α_i determines the height of the block and β_i is a scaling parameter, meaning that $\beta_i = 2$ scales the block so that it is twice as wide as the original electrode. Notice, that we assumed that the potential decays identically along the x and y axes and thus the coefficients β_i determine not only the width but simultaneously also the shift of the blocks along the x axis.

Given the FEM solution for mixed boundary conditions, the coefficients α_i and β_i in (3.10) can be determined by solving the following optimization problem

$$\begin{aligned} \min_{\alpha_i, \beta_i \in \mathbb{R}, i=1,\ldots,N} \quad & \|\tilde{h}(x_0, y) - \phi_{\text{FEM}}(x_0, y, 0)\|_2 \\ \text{subject to:} \quad & \sum_{i=1}^{N} \alpha_i = 1, \\ & \beta_i \in [1,3], \quad i = 1, \ldots, N, \end{aligned} \tag{3.11}$$

where $\phi_{\text{FEM}}(x,y,z)$ is the FEM solution. Note that the 2-norm above measures the size of a function of the real y variable, but in the numerical optimization we are only able to consider samples of y, which is not encoded in the optimization problem statement for the sake of simplicity. With the assumption that the potential decays identically along x and y axes, both α_i and β_i can be determined from a y-z cross-section of $\phi_{\text{FEM}}(x,y,0)$, we fixed x to be a negative constant value x_0. For instance, the red curve in Fig. 3.5(a) represents $\phi_{\text{FEM}}(x_0,y,0)$ for $x_0 = -200\,\mu\text{m}$. The coefficients α_i have to sum up to one because only then is the height of the piled up blocks equal to one. We assume that $d = 2b$ and thus restrict the coefficients β_i to the interval $[1,3]$ because then the blocks cannot be narrower than the electrode and they cannot interfere with other electrodes. Even though the

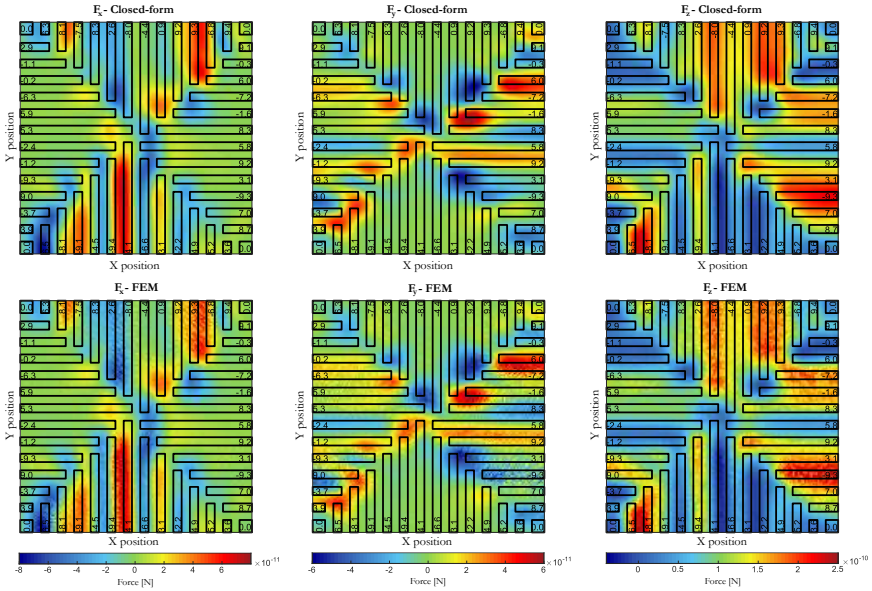


FIGURE 3.6: A comparison of individual components of DEP force fields calculated for the four-leaf clover electrode array with single electrode width $b = 50\,\mu\text{m}$ and center-to-center distance between the electrodes $d = 100\,\mu\text{m}$. The force fields are calculated numerically by FEM and analytically based on the approximate closed-form solution for the potential. The force fields are computed for the height of $120\,\mu\text{m}$ above the electrode array. The numerical values inside the electrodes represent applied potentials in volts. (Multimedia view)

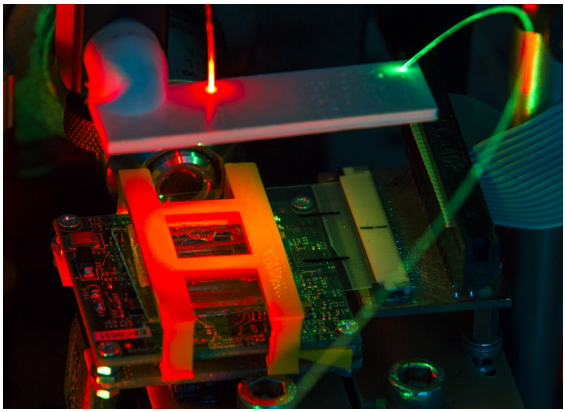


FIGURE 3.7: Photo of the apparatus used for the experimental verification of the proposed control-oriented model of DEP.

optimization task is not convex, it still provides very good results when given a good initial guess. For the initial guess, we let β_i grow linearly from 1 to 3 and set α_i to be proportional to $\frac{\partial}{\partial y}\phi_{\mathrm{FEM}}(x_0, y, 0)$. Figure 3.5(b) shows results of the optimization process for $N = 10$.

Having the approximation of the boundary condition $\tilde{h}(x, y)$, one can calculate the integral (3.3) and obtain a closed-form solution for $\phi_{44}(x, y, z)$. Instead of using the boundary condition $\tilde{h}(x, y)$ composed of several blocks directly, due to linearity of the integral, we can use the one-block boundary condition $h_0(x, y)$, calculate the integral (3.3) and compose the closed-form approximation for $\phi_{44}(x, y, z)$ in the same way as $\tilde{h}(x, y)$ is itself composed. This is exactly how we proceed. Substitution of $h_0(x, y)$ into the integral (3.3) gives

$$\phi_{i0}(x, y, z) = \frac{z}{2\pi} \int_{-\frac{b}{2}}^{\frac{b}{2}} \int_{-\infty}^{0} \frac{1}{\left((x-x')^2+(y-y')^2+z^2\right)^{3/2}} \, dx'dy'. \tag{3.12}$$

Again, we do not present the evaluated integral since one can easily compute it in Mathematica, Maple, Matlab or any another computer algebra package. Nevertheless, we emphasize that the evaluated integral is

expressible in closed form usable in real time. The approximate closed-form solution for $\phi_{44}(x, y, z)$ is then obtained as

$$\phi_i(x, y, z) = \sum_{i=1}^{N} \alpha_i \, \phi_{i0} \left(x - \frac{(\beta_i - 1)b}{2}, \frac{y}{\beta_i}, z \right). \tag{3.13}$$

Similarly, as in the 2D case, we do not compare the potentials directly because what we are interested in are the DEP force fields derived from the potentials. Since from the visualization point of view it is rather inconvenient to directly compare 3D force fields, we compare their components separately. The comparisons were carried out for the same parameters as in Example 1 and they are shown in Fig. 3.6 (Multimedia view). Fig. 3.6 shows a comparison carried out for randomly varying potentials on the electrodes. Based on this comparison, the force field computed by the proposed approximate closed-form model is seen to match that computed based on the FEM solution.

3.4 EXPERIMENTAL VERIFICATION

To verify the applicability of the proposed DEP model, we used it in an experiment where a 50 µm polystyrene microsphere was manipulated by a control system with a feedback loop. The goal of the control system is to steer the microsphere along a reference trajectory. The microsphere was suspended in water contained in a pool above an interdigitated electrode array with six electrodes and $d = 2b = 200\,\mu\text{m}$ (see Fig. 3.2(a)). A detailed description of the control and measuring system can be found in Ref. [7] and in Ref. [22], respectively. Figure 3.7 displays a photo of the hardware setup. Figure 3.8 shows reference and measured trajectories of the microsphere. In addition, the figure also displays the potentials applied to the electrodes in order to steer the microsphere along the reference trajectory. These potentials were computed in real time by the control system based on the proposed DEP model. It is noteworthy, that in Ref. [7] a numerical solution taking approximately 1 GB of memory space was used to calculate DEP force in real time, whereas here the DEP model is represented by a closed-form analytical expression that takes almost no memory space and is computationally efficient.

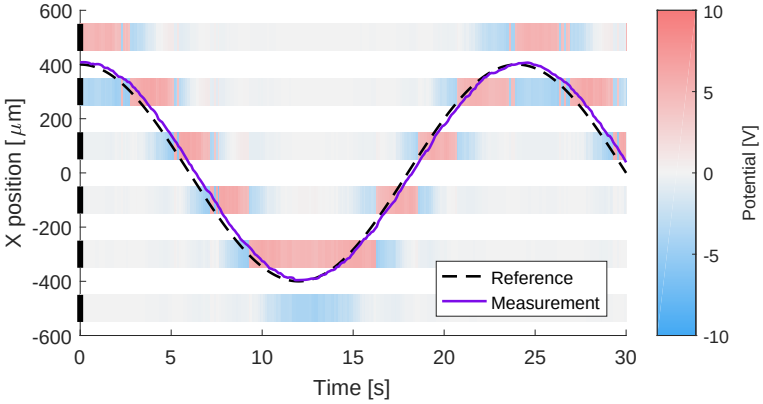


FIGURE 3.8: Experimental verification of the proposed model of DEP in an experiment where a microsphere is steered along a reference trajectory. Only the transverse coordinate (i.e. x) is shown. The colors at electrode locations show what potentials were applied to the electrodes at a given time. Notice that x axis here is time and thus any vertical cut shows what potentials were applied to the electrodes at the corresponding time.

3.5 DISCUSSION

Although the presented approach to modeling of dielectrophoresis is demonstrated on the dipole approximation of the microparticles, it can also be applied to more complex and accurate multipole approximations; no modification would be needed because multipole approximations only require higher derivatives of the potential and our proposed approximation calculates an infinitely differentiable closed-form approximation of the potential. Even though we demonstrate the approach on two concrete electrode array designs, it can also be used for other planar designs exhibiting similar symmetry; two such examples are shown in Fig. 3.9. Note that full analytical solutions for exact boundary conditions are not possible in such cases.

3.6 CONCLUSION

The major benefit of our approximate modeling methodology for a dielectrophoretic force presented here is that in comparison with the standard analytical or FEM-based (numerical) approaches it yields a mathematical

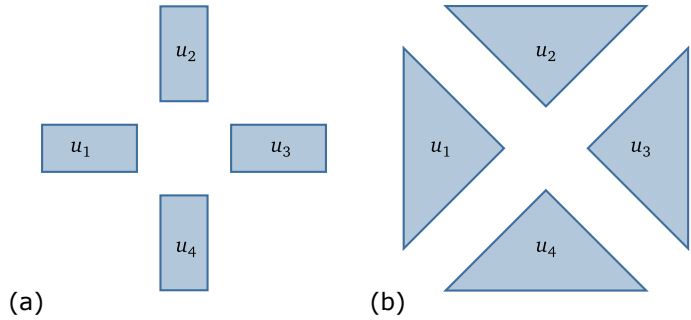


FIGURE 3.9: Some other electrode array designs in the literature [8, 19] for which the proposed modeling methodology is also applicable.

model of DEP whose application is feasible in real time (e.g. in cycles of a few milliseconds or so) on a common laboratory PC, and yet the accuracy of the model is sufficient for the purposes of feedback micromanipulation. This methodology combines numerical and analytical models so that the computational burden associated with the calculation of the numerical solution is carried out off-line and based on this numerical solution an approximated closed-form and easy-to-calculate—or briefly control-oriented—model is derived. This approach allows us to derive a control-oriented model for a broader category of electrode array designs than other approaches used in modeling of DEP.

REFERENCES

1. Gurtner, M., Hengster-Movric, K. & Hurák, Z. Green's function-based control-oriented modeling of electric field for dielectrophoresis. *Journal of Applied Physics* **122**, 054903 (2017).
2. Pohl, H. A. Some effects of nonuniform fields on dielectrics. *Journal of Applied Physics* **29**, 1182 (1958).
3. Pohl, H. A. & Hawk, I. Separation of living and dead cells by dielectrophoresis. *Science* **152**, 647 (1966).
4. Pethig, R. Review Article-Dielectrophoresis: Status of the theory, technology, and applications. *Biomicrofluidics* **4**, 022811 (2010).
5. Jones, T. B. Basic theory of dielectrophoresis and electrorotation. *Engineering in Medicine and Biology Magazine, IEEE* **22**, 33 (2003).

6. Gupta, V., Jafferji, I., Garza, M., Melnikova, V. O., Hasegawa, D. K., Pethig, R. & Davis, D. W. ApoStream™, a new dielectrophoretic device for antibody independent isolation and recovery of viable cancer cells from blood. *Biomicrofluidics* **6**, 024133 (2012).

7. Zemánek, J., Michálek, T. & Hurák, Z. Feedback control for noise-aided parallel micromanipulation of several particles using dielectrophoresis. *Electrophoresis* **36**, 1451 (2015).

8. Kharboutly, M. & Gauthier, M. *High speed closed loop control of a dielectrophoresis-based system* in *2013 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2013), 1446.

9. Kharboutly, M., Melis, A., Bolopion, A., Chaillet, N. & Gauthier, M. *2D robotic control of a planar dielectrophoresis-based system.* in *Conference on the Manipulation, Manufacturing and Measurement on the Nanoscale, 3M NANO'12.* (2012), 6.

10. Edwards, B. & Engheta, N. Electric tweezers: negative dielectrophoretic multiple particle positioning. *New Journal of Physics* **14**, 063012 (2012).

11. Kaler, K. V. & Jones, T. B. Dielectrophoretic spectra of single cells determined by feedback-controlled levitation. *Biophysical journal* **57**, 173 (1990).

12. Manaresi, N., Romani, A., Medoro, G., Altomare, L., Leonardi, A., Tartagni, M. & Guerrieri, R. A CMOS chip for individual cell manipulation and detection. *IEEE Journal of Solid-State Circuits* **38**, 2297 (2003).

13. Menachery, A., Graham, D., Messerli, S. M., Pethig, R. & Smith, P. J. Dielectrophoretic tweezer for isolating and manipulating target cells. *IET nanobiotechnology* **5**, 1 (2011).

14. Hunt, T. P. & Westervelt, R. M. Dielectrophoresis tweezers for single cell manipulation. *Biomedical microdevices* **8**, 227 (2006).

15. Morgan, H., Izquierdo, A. G., Bakewell, D., Green, N. G. & Ramos, A. The dielectrophoretic and travelling wave forces generated by interdigitated electrode arrays: analytical solution using Fourier series. *Journal of Physics D: Applied Physics* **34**, 1553 (2001).

16. Chang, D. E., Loire, S. & Mezić, I. Closed-form solutions in the electrical field analysis for dielectrophoretic and travelling wave interdigitated electrode arrays. *Journal of Physics D: Applied Physics* **36**, 3073 (2003).

17. Wang, X., Wang, X.-B., Becker, F. F. & Gascoyne, P. R. A theoretical method of electrical field analysis for dielectrophoretic electrode arrays using Green's theorem. *Journal of physics D: applied physics* **29**, 1649 (1996).

18. Hughes, M. P. *Nanoelectromechanics in engineering and biology* (CRC press, 2010).

19. Kharboutly, M., Gauthier, M. & Chaillet, N. Modeling the trajectory of a microparticle in a dielectrophoresis device. *Journal of Applied Physics* **106**, 114312 (2009).

20. Evans, L. C. *Partial Differential Equations* 2 edition (American Mathematical Society, Providence, R.I, 2010).

21. Zemánek, J., Drs, J. & Hurák, Z. *Dielectrophoretic actuation strategy for micromanipulation along complex trajectories* in *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics* (IEEE, 2014), 19.

22. Gurtner, M. & Zemánek, J. Twin-beam real-time position estimation of micro-objects in 3D. *Measurement Science and Technology* **27**, 127003 (2016).

4

ADMM-BASED DISTRIBUTED CONTROL FOR DISTRIBUTED MANIPULATION BY SHAPING PHYSICAL FORCE FIELDS

This paper proposes an algorithm for decomposing and possibly distributing an optimization problem that naturally emerges in distributed manipulation by shaping physical force fields through actuators distributed in space (arrays of actuators). One or several manipulated objects located in this field can "feel the force" and move simultaneously and independently. The control system has to produce commands for all actuators so that desired forces are developed at several prescribed places. This can be formulated as an optimization problem that has to be solved in every sampling period. Exploiting the structure of the optimization problem is crucial for platforms with many actuators and many manipulated objects, hence the goal of decomposing the huge optimization problem into several subproblems. Furthermore, if the platform is composed of interconnected actuator modules with computational capabilities, the decomposition can give guidance for the distribution of the computation to the modules. We propose an algorithm for decomposing/distributing the optimization problem using Alternating Direction Method of Multipliers (ADMM). The proposed algorithm is shown to converge to modest accuracy for various distributed platforms in a few iterations. We demonstrate our algorithm through numerical experiments corresponding to three physical experimental platforms for distributed manipulation using electric, magnetic, and pressure fields. Furthermore, we deploy and test it on real experimental platforms for distributed manipulation using an array of solenoids and ultrasonic transducers.

This chapter is reproduced from Gurtner, M., Zemánek, J. & Hurák, Z. ADMM-based distributed control for distributed manipulation by shaping physical force fields. *International Journal of Robotics Research*. As of December 2022, the paper is accepted but has not been published yet. (2022).

4.1 INTRODUCTION

Research in manipulation by actuators distributed in space, also known as distributed manipulation, was triggered by a seminal paper by [2]. Distributed manipulation can be carried either through mobile actuators [3, 4] or through actuators fixed in space where the actuators are used to shape a physical field surrounding the manipulated objects. We focus on platforms with actuators fixed in space. Various principles have been used in distributed manipulation through physical field: MEMS actuators [2], dielectrophoresis [5], acoustophoresis [6], electroosmosis [7], magnetophoresis [8], and omnidirectional wheels [9] Even though distributing not only the actuators in space but also the control algorithm is beneficial, little attention has been paid to it so far.

Early distributed manipulation platforms were sensorless. To mention a few, [10] used an array of oscillating micro-structures for actuation, arrays of microfabricated nozzles were used by [11] and [12], ultrasonic micro-actuators were reported [13], and magnetically controlled flaps were documented [14]. The absence of sensors entails that the control strategies must be open-loop and thus precomputed. Some authors refer to distributed manipulation platforms as programmable force fields [15, 16]. More on the open-loop strategies for distributed manipulation can be found in [2, 17–19]. As [20] and [21] argue, open-loop strategies tend to be slow and, in some cases, even unstable.

More recent publications introduce platforms with sensing capabilities and feedback control [9, 20, 22, 23]. When feedback control is employed, the control system solves the problem of commanding individual actuators so that forces with required directions and magnitudes are exerted at the current locations on the manipulated object(s). This problem can often be cast as a (possibly nonconvex) optimization problem where the goal is to minimize the deviation between the exerted and desired forces [5, 6]. Pseudoinverse was used to find the currents flowing through coils around the manipulation area with one or several microrobots [24, 25]. Matrix inversion [9] enabled control an array of omni-directional wheels [26]. Independent manipulation of multiple robots requires local controllability of the field, but a global signal can be used if they respond differently [27].

Distributed manipulation platforms using feedback control can be divided into those with *centralized* [9, 11–13, 20], *distributed* [22, 23] and (fully) *decentralized* control. In a *centralized control* scheme, there is one place where all the measured information is collected and processed and from which

commands to all the actuators are issued. In a *distributed control* scheme, the computation is spread (distributed) among some local units with sensing and computational capabilities (typically modules with one or several actuators). *Decentralized control* scheme could be viewed as an extreme version of the distributed one in that no interaction/communication between the units/modules is present [28]. Distributing the control logic (or computations) among the local units (modules) is advantageous because it removes the need for a central unit and makes the distributed manipulation platform modular and possibly also more scalable.

Even if ultimately the control computation is distributed among the actuator modules, it is also possible to view each manipulated object (or a group of objects) as a virtual agent, and these agents then collaborate on attaining the goal set for the control system. The agents are virtual in the sense that they only serve as a basis for decomposing the original full problem to several smaller ones to be distributed to several computational modules. This is the approach we adopt in this paper.

In particular, in this paper we adapt *Alternating Direction Method of Multipliers (ADMM)* [29, 30] for distributing the control in distributed manipulation. That is, ADMM is used to distribute the computation of the commands for the actuators. ADMM is a popular method used in convex optimization for its simplicity and applicability to a wide range of problems. Even though ADMM is a first-order method and has only a linear convergence rate [31], empirical evidence shows that it usually converges in a few tens of iterations to a modest accuracy [30]. Hence, ADMM is often used in situations where one needs a good-enough solution in a limited amount of time. This is perfectly aligned with the requirements of distributed control for distributed manipulation, where the optimization problem has to be solved repeatedly with a period as short as a few milliseconds.

Optimization problems in distributed manipulation may be nonconvex, and ADMM may diverge for nonconvex problems. Nevertheless, one can find many successful applications of ADMM to nonconvex problems in the literature, ranging from matrix completion [32–34], phase retrieval [35], polynomial optimization [36], compressive sensing [37], computer graphics [38] and image restoration [39], to name a few. These applications have driven a search for the conditions under which ADMM converges when applied to nonconvex problems [40, 41]. We show that ADMM applied to nonconvex problems emerging in distributed manipulation also converges in practice.

Contribution

Distributed manipulation through an actuator array calls for finding actuator commands for given desired forces. In contrast to the results in the literature, where this problem is usually solved as a single optimization problem, we propose a decomposition of the underlying numerical optimization, which allows for distribution of the computation to modules. To the best of our knowledge, the idea of viewing the manipulated objects as virtual agents and distributing the optimization problem related to the computation of commands for the actuators is novel. The computation can then run on (local) actuator modules, which allows for easy, and virtually unlimited, expansion of the platform. But even if centralized computation is still used, the proposed decomposition of the optimization problem is beneficial for larger platforms—its solution converges faster. Furthermore, we solved the underlying optimization subproblem of the ADMM problem only approximately so that the proposed distributed optimization algorithm runs efficiently in real time. Finally, we also show by experiments that the modified ADMM converges in practice even for a specific class of nonconvex problems.

Organization of the paper

The paper is organized as follows. The following section Control for Distributed Manipulation presents the control problem emerging in distributed manipulation by shaping a force field through a fixed actuator array. It also concisely introduces ADMM and proposes an algorithm for distributing the control problem by ADMM. In the subsequent section Experiments, we numerically and experimentally validate the proposed algorithm by applying it to three distributed manipulation platforms based on different physical principles.

Mathematical Notation

We use $[N]$ to denote the set $[N] \coloneqq \{1, \dots, N\}$, where $N \in \mathbb{N}$, and $|\mathcal{Q}|$ to denote the cardinality of a set $\mathcal{Q}$. We use $\boldsymbol{x}_{\mathcal{I}}$ to denote the vector composed of components from $\boldsymbol{x}$ with indices in $\mathcal{I} \subset \mathbb{N}$. For instance, when $\mathcal{I} = \{1, 3\}$

then $\boldsymbol{x}_{\mathcal{I}} = [x_1, x_3]^{\mathrm{T}}$. Euclidean norm of a vector $\boldsymbol{x}$ is denoted by $\|\boldsymbol{x}\|_2$. Function $I_{\mathcal{Q}}$ denotes the indicator function for a set $\mathcal{Q}$:

$$I_{\mathcal{Q}}(\boldsymbol{x}) = \begin{cases} 0, & \boldsymbol{x} \in \mathcal{Q}, \\ \infty, & \boldsymbol{x} \notin \mathcal{Q}. \end{cases} \tag{4.1}$$

Matrix $\boldsymbol{I}$ is the identity matrix of appropriate dimensions. Operator $[x]_a^b$ clips the scalar x to the interval $[a, b]$. The Jacobian of a function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ will be denoted by $\boldsymbol{D}\boldsymbol{f} \in \mathbb{R}^{m \times n}$, or simply $\boldsymbol{D}$, when it is clear from the context with respect to which function $\boldsymbol{f}$ the Jacobian is computed. Operator $\lceil x \rceil$ rounds x to the closest integer larger than or equal to x.

4.2 CONTROL FOR DISTRIBUTED MANIPULATION

Distributed manipulation deals with situations where an array of spatially distributed actuators is commanded to steer one or several objects. More specifically, here we restrict ourselves to the case where each actuator contributes to the force acting on a manipulated object (or to the physical field from which the force is derived), and the contribution decays with the distance between the actuator and the manipulated object. Thus, the net force exerted on the object is composed from the contributions of individual actuators in the object's vicinity.

The aim of the control system is to get the objects into desired locations or steer them along desired trajectories. In every control period, the control system measures positions of the manipulated objects and computes the commands for the actuators so that the objects move towards the desired locations. It is worth emphasizing that what we call actuators could be viewed as local contributors to the global force field. For instance, commanded electric currents through solenoids give rise to magnetic force field and amplitudes [8] or phase shifts [5] of voltage signals applied to microelectrode arrays give rise to dielectrophoretic force field. This position feedback control loop is structured such that first the control system computes a suitable force moving a given manipulated object towards the desired location and then it finds the commands for the actuators (contributors to the global force field) exerting this suitable force. The control system's structure is shown in Fig. 4.1. The computation of the suitable force is a task for a position controller (even a PID controller often suffices). In this paper, we focus on the problem of computing the values of actuators' commands such that the required suitable forces are developed at objects' positions. This

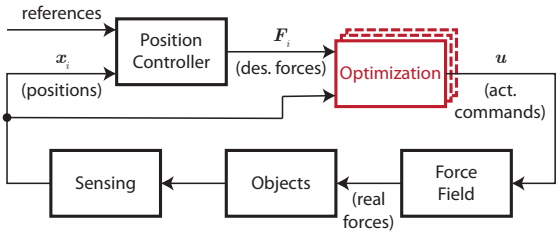


FIGURE 4.1: Block diagram of the system and the controller. The position controller sets desired forces for each object according to the reference and measured positions. Optimization block then finds actuator commands according to the desired forces and current positions.

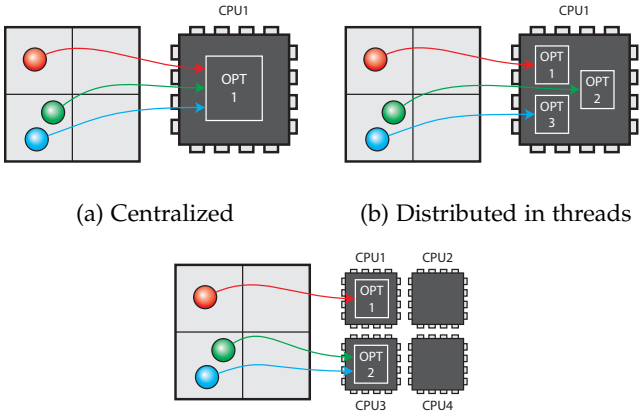
task is performed by a component labeled Optimization and marked red in the control diagram.

We assume that a mathematical model relating the actuators commands and the developed force field is available. However, the inversion—that is, a model that takes the required forces as the input and produces the actuator commands—is unavailable and must be computed numerically. For N objects to be manipulated in m dimensions (either 2D or 3D), the inversion of the force model can be formulated as the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{u}\in\mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2}\sum_{i=1}^{N} \|\boldsymbol{F}_i - \boldsymbol{f}(\boldsymbol{u}, \boldsymbol{x}_i)\|_2^2, \\ \text{subject to} \quad & \underline{u} \leq \boldsymbol{u} \leq \overline{u}, \end{aligned} \tag{4.2}$$

where $\boldsymbol{F}_i \in \mathbb{R}^m$ is the required net force acting on the ith manipulated object. Function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$ is smooth (possibly nonconvex) and represents the force model, that is, it outputs the force acting on an object at position $\boldsymbol{x}_i \in \mathbb{R}^m$ for values of the actuators' commands $\boldsymbol{u} \in \mathbb{R}^n$. Scalars $\underline{u}$ and $\overline{u}$ represent minimum and maximum limits for the values of actuators' commands, respectively. The elements of vector $\boldsymbol{u}$ are the values of the individual actuators' commands. We assume the same upper and lower limits for all actuators for notational simplicity; nevertheless, the proposed algorithms work for varying limits as well. Throughout this paper, we consider spherical objects, so we do not aim on controlling their orientation. Nevertheless, if orientation is to be controlled, the problem (4.2) can be extended to include the desired moment.

Optimization problem (4.2) can be solved either in a centralized fashion—the problem is solved as whole at one computational unit—or, as we pro-



(a) Centralized (b) Distributed in threads

(c) Spatially distributed

FIGURE 4.2: Decomposition strategies. An illustrative example shows three balls on a manipulation platform divided into four hardware modules.

pose, it can be decomposed to several smaller optimization problems. The decomposition has the advantage that one can distribute the computational load to several computational units (be it several threads at one computational core, more computational cores, or more networked computers). We propose to decompose (4.2) into smaller problems in which only those actuators are involved that can significantly contribute to the force acting on a manipulated object.

To clarify possible variants of control schemes for distributed manipulation, let us consider an illustrative example in Fig. 4.2. It shows three manipulated objects on a manipulation platform consisting of four hardware modules, each module containing possibly several actuators. In centralized control Fig. 4.2a, one optimization problem (OPT1) run by a central computer (CPU1) finds control commands leading to the desired forces exerted on all three objects. Optimization can be broken into subproblems so that each subproblem corresponds to one ball and optimization is only performed over the control commands for actuators in the vicinity. Even if the subproblems (OPT1–3) run on the same central computer (CPU1) Fig. 4.2b, for example, in separate threads, the decomposition can improve overall performance. If the modules have computational power (CPU1–4), they can solve subproblems (OPT1–2) assigned to them and the controller is (spatially) distributed Fig. 4.2c.

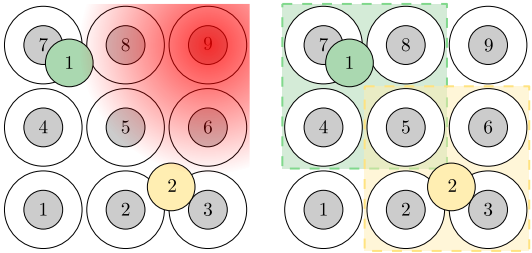


FIGURE 4.3: A motivating example of a distributed manipulation system. In the left figure, the intensity of the red color visualizes the decaying magnitude of the attractive force field generated by the top-right actuator. In the right figure, actuators that can significantly contribute to the force acting on the green and yellow manipulated objects are bounded by the green and yellow color, respectively.

Next, we show by an example why it is advantageous to decompose (4.2). Then, we briefly review ADMM and show how to use it for decomposing (4.2). Finally, we describe how the decomposed problems can be efficiently solved.

4.2.1 *Motivating example*

An example of a distributed manipulation system—inspired by the MagMan platform described later in the experimental section of the paper—is depicted in Fig. 4.3. The system consists of a 3×3 matrix of identical actuators and two manipulated objects. The actuators can generate only force fields that attract the objects to their centers. The net force acting on an object is given by the sum of forces contributed by individual actuators. The figure shows that the top-right actuator can contribute significantly to the force acting neither on the first nor the second manipulated object. In fact, one can effectively use only actuators $\{4,5,7,8\}$ to manipulate the first object and actuators $\{2,3,5,6\}$ to manipulate the second object. Thus, it makes sense to decompose (4.2) to two smaller optimization problems: first optimizes over actuators $\{4,5,7,8\}$ and is responsible for exerting the force acting on the first object, and second optimizes over actuators $\{2,3,5,6\}$ and is responsible for exerting the force acting on the second object. Since these two optimization problems share actuator number 5, they need to be solved collaboratively.

4.2.2 *Alternating Direction Method of Multipliers (ADMM)*

ADMM is a first-order method solving optimization problems of the following form:

$$\begin{aligned} \underset{\boldsymbol{u},\boldsymbol{y}\in\mathbb{R}^n}{\text{minimize}} \quad & f(\boldsymbol{u}) + g(\boldsymbol{y}), \\ \text{subject to} \quad & \boldsymbol{u} - \boldsymbol{y} = 0, \end{aligned} \tag{4.3}$$

where $f, g : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ are closed proper convex functions. ADMM introduces (scaled) dual variable $\boldsymbol{z}$ and proceeds in iterations according to the following update rules:

$$\boldsymbol{u}^{k+1} := \mathbf{prox}_{\lambda f}\left(\boldsymbol{y}^k - \boldsymbol{z}^k\right), \tag{4.4a}$$

$$\boldsymbol{y}^{k+1} := \mathbf{prox}_{\lambda g}\left(\boldsymbol{u}^{k+1} + \boldsymbol{z}^k\right), \tag{4.4b}$$

$$\boldsymbol{z}^{k+1} := \boldsymbol{z}^k + \boldsymbol{u}^{k+1} - \boldsymbol{y}^{k+1}, \tag{4.4c}$$

where k is an iteration index and the proximal operator $\mathbf{prox}_{\lambda f} : \mathbb{R}^n \to \mathbb{R}^n$ of a function f is defined as follows

$$\mathbf{prox}_{\lambda f}(\boldsymbol{v}) := \arg\min_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + (1/2\lambda)\|\boldsymbol{x} - \boldsymbol{v}\|_2^2\right), \tag{4.5}$$

with a scaling parameter $\lambda > 0$.

The parameter λ in (4.4) can be viewed as a tuning parameter influencing the practical convergence rate. Nevertheless, ADMM is proven to converge to a minimizer for every $\lambda > 0$ under very mild assumptions [30].

4.2.3 *Decomposition*

As we mentioned, due to the decaying contribution, only those actuators in the vicinity of an object can significantly contribute to the force acting on the object. Thus, it makes sense to decompose the optimization problem (4.2) to N smaller optimization subproblems (one per manipulated object) where each of these subproblems involves only those actuators that can significantly contribute to the force acting on the manipulated object at hand. Therefore, we can view manipulated objects as virtual agents that set the commands only for the neighboring actuators so that a required force is exerted. A conflict may occur when two agents use the same actuators; hence, the agents have to cooperate. The resulting problem is often called

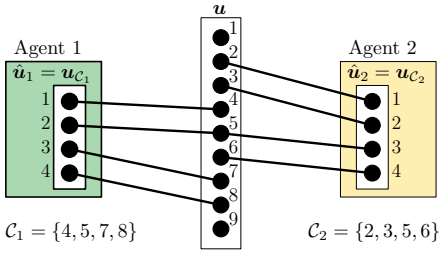


FIGURE 4.4: An example showing how the global decision vector $\boldsymbol{u}$ is distributed as local copies $\hat{\boldsymbol{u}}_i$ to the agents. The depicted case corresponds to the previously described motivation example.

a *global consensus problem*. We could as well assign each manipulated object to the closest computational module and decompose the optimization problem (4.2) so that each subproblem treats the objects assigned to the same module. For ease of exposition, we followed the former approach. In the remainder of this subsection, we adapt the approach to the global consensus problem presented by [30] and [29].

To reformulate (4.2) to a distributed optimization problem, we first decompose the optimization problem by making local copies of the decision vector $\boldsymbol{u}$ (one copy for each agent). Let us denote the local copy corresponding to the ith agent by $\hat{\boldsymbol{u}}_i$. Since only actuators in the vicinity of the manipulated object can significantly contribute to the force acting on it, $\hat{\boldsymbol{u}}_i$ contains only components of $\boldsymbol{u}$ corresponding to these actuators. Let $\mathcal{C}_i \subseteq [n]$ denote the set of indices of actuators used by the ith agent. Then, each component of $\hat{\boldsymbol{u}}_i \in \mathbb{R}^{|\mathcal{C}_i|}$ corresponds to one component in $\boldsymbol{u}_{\mathcal{C}_i}$, where $\boldsymbol{u}_{\mathcal{C}_i}$ is a vector containing elements of $\boldsymbol{u}$ with indices from $\mathcal{C}_i$. A simple example showing how the decision variable $\boldsymbol{u}$ is copied to local copies $\hat{\boldsymbol{u}}_i$ is shown in Fig. 4.4. Since we will frequently need a mapping between local and global indices in the following text, let us define a function $g = G_i(k)$ that gives the global index g for a local index k of the ith agent. In other words, $(\hat{\boldsymbol{u}}_i)_k$ corresponds to $(\boldsymbol{u})_g$ where $g = G_i(k)$. Returning to the example shown in Fig. 4.4, the third component of the local copy $\hat{\boldsymbol{u}}_2$ corresponds to the fifth component of the global decision vector $\boldsymbol{u}$ and thus $G_2(3) = 5$.

Having the local copies $\hat{\boldsymbol{u}}_i$, the centralized optimization problem (4.2) can be decomposed to

$$\begin{aligned} \underset{\hat{\boldsymbol{u}}_i \in \mathbb{R}^{|\mathcal{C}_i|},\, i=1,\dots,N}{\text{minimize}} \quad & \frac{1}{2}\sum_{i=1}^{N} \|\boldsymbol{F}_i - \boldsymbol{f}_i(\hat{\boldsymbol{u}}_i, \boldsymbol{x}_i)\|_2^2, \\ \text{subject to} \quad & (\hat{\boldsymbol{u}}_1, \dots, \hat{\boldsymbol{u}}_N) \in \mathcal{Q}, \end{aligned} \tag{4.6}$$

where function $\boldsymbol{f}_i$ is the relation between the actuators used by the ith agent and the exerted force acting on the ith object and $\mathcal{Q}$ encodes the condition on the global consensus and feasibility of local copies of the decision vector, that is

$$\begin{aligned} \mathcal{Q} \coloneqq \{(\hat{\boldsymbol{u}}_1, \dots, \hat{\boldsymbol{u}}_N) \mid \underline{u} \le \hat{\boldsymbol{u}}_i \le \overline{u} \text{ and} \\ (\hat{\boldsymbol{u}}_i)_k = (\hat{\boldsymbol{u}}_j)_l \text{ if } G_i(k) = G_j(l)\}. \end{aligned} \tag{4.7}$$

One can view (4.6) as N independent optimization problems which are coupled together by constraining the local decision vectors $\hat{\boldsymbol{u}}_i$ to lie in $\mathcal{Q}$. By defining functions

$$\varphi_i(\hat{\boldsymbol{u}}_i) = \frac{1}{2}\|\boldsymbol{F}_i - \boldsymbol{f}_i(\hat{\boldsymbol{u}}_i, \boldsymbol{x}_i)\|_2^2 \tag{4.8}$$

and using the indicator function we can convert the constrained problem (4.6) to the following unconstrained one:

$$\underset{\hat{\boldsymbol{u}}_i \in \mathbb{R}^{|\mathcal{C}_i|},\, i=1,\dots,N}{\text{minimize}} \quad \sum_{i=1}^{N} \varphi_i(\hat{\boldsymbol{u}}_i) + I_{\mathcal{Q}}(\hat{\boldsymbol{u}}_1, \dots, \hat{\boldsymbol{u}}_N) \tag{4.9}$$

which is of the same form as optimization problem (4.3)—the sum $\sum_{i=1}^{N} \varphi_i(\hat{\boldsymbol{u}}_i)$ corresponds to function f and the indicator function $I_{\mathcal{Q}}$ to function g—and thus this problem can be solved by ADMM.

4.2.4 *Inexact ADMM for distributed manipulation*

Applying update rules of ADMM (4.4) to the decomposed optimization problem (4.9), we get

$$\hat{\boldsymbol{u}}_i^{k+1} \coloneqq \mathbf{prox}_{\lambda\varphi_i}(\boldsymbol{y}_i^k - \boldsymbol{z}_i^k), \tag{4.10a}$$

$$(\boldsymbol{y}_1^{k+1}, \dots, \boldsymbol{y}_N^{k+1}) \coloneqq \mathbf{prox}_{I_{\mathcal{Q}}}(\hat{\boldsymbol{u}}_1^{k+1} + \boldsymbol{z}_1^k, \dots, \hat{\boldsymbol{u}}_N^{k+1} + \boldsymbol{z}_N^k), \tag{4.10b}$$

$$\boldsymbol{z}_i^{k+1} \coloneqq \boldsymbol{z}_i^k + \rho\,(\hat{\boldsymbol{u}}_i^{k+1} - \boldsymbol{y}_i^{k+1}). \tag{4.10c}$$

Note that we added a parameter ρ scaling the update of the dual variable $\boldsymbol{z}$. For the convex case, this is unnecessary as classical ADMM is used with $\rho = 1$. Nevertheless, in the case of ADMM applied to nonconvex problems, changing the value of ρ may significantly improve the number of iterations needed to attain a modest accuracy (as shown by [32–34]). Moreover, even in the convex case, one often observes faster convergence for $\rho > 1$. Notice that (4.10a) and (4.10c) are decoupled—to compute $\hat{\boldsymbol{u}}_i^{k+1}$ and $\boldsymbol{z}_i^{k+1}$ we need only the variables directly available to the ith agent—and can be solved by each agent independently, whereas to compute $\boldsymbol{y}_i^{k+1}$ in (4.10b), each agent needs to gather information from other agents; what information exactly, will be described in the end of this section.

Now, we describe how to efficiently compute individual update rules in (4.10). Update rule (4.10c) is trivial; hence, let us discuss (4.10a) and (4.10b). To compute $\hat{\boldsymbol{u}}_i^{k+1}$ in (4.10a), each agent has to solve a possibly nonconvex optimization problem—as we stated at the beginning, φ_i can be nonconvex—and that can be computationally prohibitive. Thus, we propose to compute $\mathbf{prox}_{\lambda\varphi_i}$ only approximately. First, we rewrite the proximal operator $\mathbf{prox}_{\lambda\varphi_i}$ to the following incremental form

$$\mathbf{prox}_{\lambda\varphi_i}(\boldsymbol{v}) = \boldsymbol{v} + \Delta\boldsymbol{v}^*, \tag{4.11}$$

where $\boldsymbol{v}$ is a shorthand for $\boldsymbol{y}_i^k - \boldsymbol{z}_i^k$ and where

$$\Delta\boldsymbol{v}^* = \underset{\Delta\boldsymbol{v}}{\arg\min} \left(\varphi_i(\boldsymbol{v} + \Delta\boldsymbol{v}) + (1/2\lambda)\|\Delta\boldsymbol{v}\|_2^2 \right). \tag{4.12}$$

Now, we focus on $\Delta\boldsymbol{v}^*$ which, after substitution for φ_i, is

$$\Delta\boldsymbol{v}^* = \underset{\Delta\boldsymbol{v}}{\arg\min} \left(\frac{1}{2}\|\boldsymbol{F}_i - \boldsymbol{f}_i(\boldsymbol{v} + \Delta\boldsymbol{v}, \boldsymbol{x}_i)\|_2^2 + \frac{1}{2\lambda}\|\Delta\boldsymbol{v}\|_2^2 \right). \tag{4.13}$$

Since (4.13) is a nonlinear least-squares problem with a regularization term, we can take some inspiration from Levenberg-Marquardt algorithm [42]—an algorithm commonly used for nonlinear least squares problems. In this algorithm, one linearizes function $\boldsymbol{f}_i$, solves the standard linear least-squares problem and repeats these two steps until convergence. We do the same, but instead of iterating until convergence, we stop after one iteration. This is motivated by the real-time iteration scheme often used in numerical optimal control [43]. The underlying assumption is that the optimization problem (4.2) does not change much in time—the positions of the objects and the required forces vary slowly—and thus if we initialize the current

optimization problem by the solution of the previous one, one iteration should suffice to obtain a reasonably good solution.

The linearized model of $\boldsymbol{f}_i$ is

$$\boldsymbol{f}_i(\boldsymbol{v} + \Delta\boldsymbol{v}, \boldsymbol{x}_i) \approx \boldsymbol{f}_i(\boldsymbol{v}, \boldsymbol{x}_i) + \boldsymbol{D}\boldsymbol{f}_i(\boldsymbol{v}, \boldsymbol{x}_i)\Delta\boldsymbol{v}. \tag{4.14}$$

Subsequently we will use $\boldsymbol{D}$ instead of $\boldsymbol{D}\boldsymbol{f}_i$ (the Jacobian of $\boldsymbol{f}_i$) to simplify the notation. With linearized $\boldsymbol{f}_i$, optimization problem (4.13) turns to

$$\Delta\tilde{\boldsymbol{v}}^* = \underset{\Delta\boldsymbol{v}}{\arg\min} \left(\frac{1}{2}\Delta\boldsymbol{v}^{\mathrm{T}} \left(\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D} + \frac{1}{\lambda}\boldsymbol{I} \right) \Delta\boldsymbol{v} + \left(\boldsymbol{f}_i - \boldsymbol{F}_i\right)^{\mathrm{T}} \boldsymbol{D}\Delta\boldsymbol{v} \right), \tag{4.15}$$

where $\Delta\tilde{\boldsymbol{v}}^*$ is the approximate solution of (4.13) and where we dropped the explicit dependence on $\boldsymbol{v}$ and $\boldsymbol{x}_i$, for simplicity. Readily, the solution of (4.15) is given by the solution of the following system of equations

$$\left(\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D} + 1/\lambda\, \boldsymbol{I} \right) \Delta\tilde{\boldsymbol{v}}^* = -\boldsymbol{D}^{\mathrm{T}} \left(\boldsymbol{f}_i - \boldsymbol{F}_i \right). \tag{4.16}$$

Having $\Delta\tilde{\boldsymbol{v}}^*$, the proximal operator $\mathbf{prox}_{\lambda\varphi_i}$ in (4.10a) can be approximated by

$$\mathbf{prox}_{\lambda\varphi_i}(\boldsymbol{v}) \approx \boldsymbol{v} + \Delta\tilde{\boldsymbol{v}}^*. \tag{4.17}$$

Now, let us turn our attention to the proximal operator $\mathbf{prox}_{\lambda I_{\mathcal{Q}}}$ in the second equation of ADMM (4.10). It is a well-known property of proximal operators of indicator functions that they simplify to Euclidean projections [29]. In the case of the set $\mathcal{Q}$, the projection is given by local averaging and clipping of local copies corresponding to the same component of the global decision vector. Formally, this is expressed as follows

$$\left(\boldsymbol{y}_i^{k+1}\right)_j = \left[\frac{1}{|\mathcal{N}_{G_i(j)}|} \sum_{G_i(j)=G_{i'}(j')} \left(\hat{\boldsymbol{u}}_{i'}^{k+1}\right)_{j'} + \left(\boldsymbol{z}_{i'}^{k}\right)_{j'} \right]_{\underline{u}}^{\overline{u}}, \tag{4.18}$$

where $\mathcal{N}_g \coloneqq \{i \in [N] \mid g \in \mathcal{C}_i\}$. In words, $\mathcal{N}_g$ is the set of agents using the actuator with the global index g.

To evaluate (4.18), each agent needs to communicate with all agents it shares some actuators with; hence, the ith agent communicates to the agents with identifiers from the set $\mathcal{M}_i \coloneqq \{j \in [N] \setminus \{i\} \mid \mathcal{C}_i \cap \mathcal{C}_j \neq \emptyset\}$.

Algorithm 1: Inexact ADMM-based algorithm for distributing the control in distributed manipulation.

Initialize $\hat{\boldsymbol{u}}_i^0, \boldsymbol{y}_i^0, \boldsymbol{z}_i^0 \in \mathbb{R}^{|\mathcal{C}_i|}$ and $\mathcal{M}_i$ for all $i \in [N]$. Set $k = 0$.
repeat
  Solve for all $i \in [N]$

$$\left(\boldsymbol{D}_i^{\mathrm{T}} \boldsymbol{D}_i + 1/\lambda\, \boldsymbol{I}\right) \Delta\tilde{\boldsymbol{v}}_i^* = -\boldsymbol{D}_i^{\mathrm{T}} \left(\boldsymbol{f}_i - \boldsymbol{F}_i\right), \tag{4.20}$$

  where $\boldsymbol{D}_i = \boldsymbol{D}\boldsymbol{f}_i$ and $\boldsymbol{f}_i$ are evaluated at $\boldsymbol{y}_i^k - \boldsymbol{z}_i^k$.
  Update

$$\hat{\boldsymbol{u}}_i^{k+1} = \boldsymbol{y}_i^k - \boldsymbol{z}_i^k - \Delta\tilde{\boldsymbol{v}}_i^*. \tag{4.21}$$

  Each agent sends MSG_i to all agents in $\mathcal{M}_i$.
  Update $\boldsymbol{y}_i^{k+1}$ by (4.18);
  Update $\boldsymbol{z}_i^{k+1}$ by (4.10c);
until *termination*;

This can be implemented in various ways, depending on the communication interface. For instance, the communication graph can be established at the beginning of the control period, when each agent can broadcast the global indices of the actuators it uses. All other agents receive this message and add the sending agent's id to $\mathcal{M}_i$ if they share an actuator. When (4.18) needs to be evaluated, each agent sends the following message to all agents in $\mathcal{M}_i$:

$$\mathrm{MSG}_i := \left(\mathcal{C}_i, \hat{\boldsymbol{u}}_i^{k+1} + \boldsymbol{z}_i^k\right). \tag{4.19}$$

The message contains vector $\hat{\boldsymbol{u}}_i^{k+1} + \boldsymbol{z}_i^k$ and the global indices $\mathcal{C}_i$ of its individual components, that is all the necessary information for computing (4.18).

Inexact ADMM for distributed manipulation is summarized in Algorithm 1. If the required forces $\boldsymbol{F}_i$ do not change abruptly in time, one can achieve faster convergence by warm-starting $\boldsymbol{y}_i^0$ and $\boldsymbol{z}_i^0$, that is by initializing these vectors by their final values from the previous control period. It is important to note that we assume synchronous and a reliable communication link, that means, the agents go through steps in Algorithm 1 synchronously and no message is lost.

4.2.5 *Efficient inexact ADMM for distributed manipulation*

Using Algorithm 1 brings the following trade-off. Each agent either uses as few actuators as it needs to exert the required force; then, the exerted force may differ from the predicted one by the model $\boldsymbol{f}_i$, as there are might be some significantly contributing and yet unconsidered actuators nearby used by other agents (this is demonstrated by the first numerical experiment in Fig. 4.7). Alternatively, each agent uses all actuators that can contribute to the exerted force acting on the manipulated object; then, the optimization problems could be large and thus more computationally demanding. To mitigate this trade-off, we propose Algorithm 2.

Algorithm 2 differs from Algorithm 1 in optimizing over a smaller set of actuators than it considers in the force model $\boldsymbol{f}_i$. Specifically, each agent considers a set of all actuators that can significantly contribute to the force acting on the manipulated object—these actuators have indices in the set $\mathcal{C}_i$—but optimizes only over a smaller set of actuators, with indices in $\mathcal{D}_i \subseteq \mathcal{C}_i$, that is sufficient to generate the required force. The assignment of the actuators to the sets depends on the actuator's distance from the object. For example, in the DEP system, $\mathcal{D}_i$ includes actuators within a circle of radius three times bigger than the actuator pitch, and $\mathcal{C}_i$ five times bigger. Thus, the accuracy of the predicted force is given by the set $\mathcal{C}_i$ and the time needed to solve (4.22) is mainly given by the size of $\mathcal{D}_i$. Also, in Algorithm 2 agents send messages to a smaller set of agents $\mathcal{M}_i'$ than in Algorithm 1. The set of agents the ith agent communicates to is now defined as $\mathcal{M}_i' := \{j \in [N] \setminus \{i\} \mid \mathcal{D}_i \cap \mathcal{C}_j \neq \emptyset\}$.

Note on convergence

Despite the observed good behavior of the proposed algorithms (as reported in the following section), the proof of convergence remains to be found. [40] proved convergence of ADMM in the form of (4.10) to a local minimum when the following conditions are met: φ_i are Lipschitz continuous, the optimization problem (4.2) is bounded, λ is large enough, and $\rho = 1/\lambda$. However, the proof of convergence by [40] relies on the proximal operators being computed exactly, whereas, for time efficiency, we compute $\mathbf{prox}_{\lambda\varphi_i}$ only approximately. Nevertheless, we can report that we did not encounter a single setup where—for properly chosen and fixed λ and ρ—the proposed algorithm diverged. Moreover, the proposed algorithm

Algorithm 2: Efficient inexact ADMM-based algorithm for distributing in control in distributed manipulation.

Initialize $\hat{\boldsymbol{u}}_i^0, \boldsymbol{y}_i^0, \boldsymbol{z}_i^0 \in \mathbb{R}^{|\mathcal{C}_i|}$ and $\mathcal{M}_i'$ for all $i \in [N]$. Set $k = 0$.
repeat
  Solve for all $i \in [N]$

$$\left(\boldsymbol{D}_{:,\mathcal{D}_i}^{\mathrm{T}} \boldsymbol{D}_{:,\mathcal{D}_i} + 1/\lambda\, \boldsymbol{I}\right) \Delta\tilde{\boldsymbol{v}}_i^* = -\boldsymbol{D}_{:,\mathcal{D}_i}^{\mathrm{T}} \left(\boldsymbol{f}_i - \boldsymbol{F}_i\right), \tag{4.22}$$

  where $\boldsymbol{D}_{:,\mathcal{D}_i}$ is the matrix composed of columns of the Jacobian $\boldsymbol{D}\boldsymbol{f}_i$ with indices from $\mathcal{D}_i$. $\boldsymbol{D}_{:,\mathcal{D}_i}$ and $\boldsymbol{f}_i$ are evaluated at $\boldsymbol{y}_i^k - \boldsymbol{z}_i^k$.
  Update

$$\left(\hat{\boldsymbol{u}}_i^{k+1}\right)_{\mathcal{D}_i} = \left(\boldsymbol{y}_i^k\right)_{\mathcal{D}_i} - \left(\boldsymbol{z}_i^k\right)_{\mathcal{D}_i} - \Delta\tilde{\boldsymbol{v}}_i^*. \tag{4.23}$$

  Each agent sends MSG_i to all agents in $\mathcal{M}_i'$.
  Update $\boldsymbol{y}_i^{k+1}$ by (4.18);
  Update $\boldsymbol{z}_i^{k+1}$ by (4.10c);
until *termination*;

commonly converges to a modest accuracy in a few iterations, as shown in section Experiments.

To measure the convergence in the numerical experiments, we define the following local convergence measure, which combines convergence of both primal and dual variables,

$$\gamma_i^k := \left\| \begin{bmatrix} \boldsymbol{y}_i^{k+1} - \boldsymbol{y}_i^k \\ \boldsymbol{z}_i^{k+1} - \boldsymbol{z}_i^k \end{bmatrix} \right\|_2^2. \tag{4.24}$$

The global convergence measure is defined as the average of the local ones, that is:

$$\gamma^k := \frac{1}{N} \sum_{i=1}^{N} \gamma_i^k. \tag{4.25}$$

Stopping criterion

The proposed algorithms are designed for real-time applications, and thus the most straightforward and yet reasonable stopping criterion is to stop

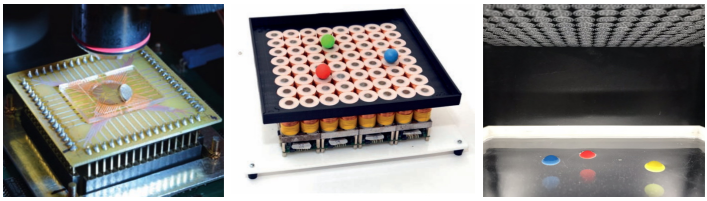


FIGURE 4.5: Photos of experimental platforms used for demonstration of the proposed algorithms.

after a fixed number of iterations. If sufficient accuracy is achieved in fewer iterations, then there is no benefit in doing the remaining iterations as in real-time applications, we care about the worst-case scenario. Nevertheless, if there is a need to terminate the algorithm earlier than after a maximum number of iterations, one can solve the consensus problem [44] also for the local convergence measures γ_i^k and terminate the algorithm when a consensus on γ^k is reached and its value is sufficiently low.

4.2.6 *Efficient Implementation*

The most computationally demanding part of the proposed algorithms is solving the system of linear equations (4.20) in Algorithm 1 (or (4.22) in Algorithm 2)[1]; hence, we focus on the efficient implementation of this part here. We discuss only the case of Algorithm 1, but the same applies to Algorithm 2.

We need to solve the following linear system efficiently

$$\left(\boldsymbol{D}_i^{\mathrm{T}}\boldsymbol{D}_i + 1/\lambda\,\boldsymbol{I}\right)\Delta\tilde{\boldsymbol{v}}_i^* = -\boldsymbol{D}_i^{\mathrm{T}}\left(\boldsymbol{f}_i - \boldsymbol{F}_i\right). \tag{4.26}$$

The matrix $(\boldsymbol{D}_i^{\mathrm{T}}\boldsymbol{D}_i + 1/\lambda\,\boldsymbol{I})$ is necessarily positive definite and thus, one might think of using Cholesky factorization for solving (4.26) which has $\mathcal{O}(|\mathcal{C}_i|^3)$ computational complexity. A better approach is to use the so-called *push-through identity*[2] [[]p. 332]Boyd2018Introduction for the equation

$$\Delta\tilde{\boldsymbol{v}}_i^* = -\left(\boldsymbol{D}_i^{\mathrm{T}}\boldsymbol{D}_i + 1/\lambda\,\boldsymbol{I}\right)^{-1}\boldsymbol{D}_i^{\mathrm{T}}\left(\boldsymbol{f}_i - \boldsymbol{F}_i\right). \tag{4.27}$$

1 In fact, the evaluation of the function $\boldsymbol{f}_i$ and the Jacobian $\boldsymbol{D}\boldsymbol{f}_i$ might be expensive as well, but as this depends solely on the force model (and not on the proposed algorithm) we do not discuss this issue here.

2 A special case of Woodbury matrix identity.

which gives us

$$\Delta\tilde{\boldsymbol{v}}_i^* = -\boldsymbol{D}_i^{\mathrm{T}} \left(\boldsymbol{D}_i \boldsymbol{D}_i^{\mathrm{T}} + 1/\lambda\, \boldsymbol{I} \right)^{-1} (\boldsymbol{f}_i - \boldsymbol{F}_i)\,. \tag{4.28}$$

Notice that now we need to compute the inversion of a much smaller $m \times m$ matrix instead of the original $|\mathcal{C}_i| \times |\mathcal{C}_i|$ matrix ($|\mathcal{C}_i|$, as the number of actuators, is significantly larger than the number m of components of the acting force). Matrix $\left(\boldsymbol{D}_i \boldsymbol{D}_i^{\mathrm{T}} + 1/\lambda\, \boldsymbol{I} \right)$ can be expressed as $\bar{\boldsymbol{A}}^{\mathrm{T}} \bar{\boldsymbol{A}}$ where

$$\bar{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{D}_i^{\mathrm{T}} \\ 1/\sqrt{\lambda} \boldsymbol{I} \end{bmatrix} \tag{4.29}$$

and thus we can express (4.28) as

$$\Delta\tilde{\boldsymbol{v}}_i^* = -\boldsymbol{D}_i^{\mathrm{T}} \left(\bar{\boldsymbol{R}}^{\mathrm{T}} \bar{\boldsymbol{R}} \right)^{-1} (\boldsymbol{f}_i - \boldsymbol{F}_i)\,, \tag{4.30}$$

where $\bar{\boldsymbol{R}}$ is the $\boldsymbol{R}$ matrix from the QR factorization of $\bar{\boldsymbol{A}}$. Computing $\bar{\boldsymbol{R}}$ by Householder QR factorization requires $2(m + |\mathcal{C}_i|)m^2$ FLOPS (ignoring non-dominant terms). Since we assume $|\mathcal{C}_i| > m$, the computational complexity of computing $\bar{\boldsymbol{R}}$ is $\mathcal{O}(|\mathcal{C}_i| m^2)$. The most computationally demanding operation in (4.30) is computing $\bar{\boldsymbol{R}}$ and thus we can infer that the computational complexity of computing $\Delta\tilde{\boldsymbol{v}}_i^*$ is $\mathcal{O}(|\mathcal{C}_i| m^2)$. That is better than $\mathcal{O}(|\mathcal{C}_i|^3)$ needed by Cholesky factorization. The complexity grows only linearly with the number of used actuators. Note that when the Jacobian $\boldsymbol{D}_i$ is independent of $\Delta\boldsymbol{v}$ (which is the case for the magnetophoresis platform described in the experimental section), matrix $\bar{\boldsymbol{R}}$ can be computed during the initialization phase of Algorithm 1.

4.3 EXPERIMENTS

We tested Algorithm 2 on distributed platforms using three different physical phenomena of actuation: manipulation through electric field (dielectrophoresis), magnetic field (magnetophoresis), and pressure field (acoustophoresis). For each platform, we experimentally found the parameters λ and ρ resulting in fast convergence, showed the practical convergence rate, and also demonstrated the proposed algorithm by numerical simulations. Presented results consider only the time needed for computation and disregard other implementation aspects, such as communication. For the platforms utilizing magnetophoresis and acoustophoresis, we also

deployed the algorithm to real hardware setups and report real experiments. Photos of the platforms are displayed in Fig. 4.5, and a video capturing the numerical simulations and real experiments is available at `https://youtu.be/P6HzgRZ4wuA`. Both the position controller and position measurement run on a central computer. Objects' positions are measured using a camera and object detection is based on color and shape. Description of the platforms and experiments follows[3].

4.3.1 *Dielectrophoresis*

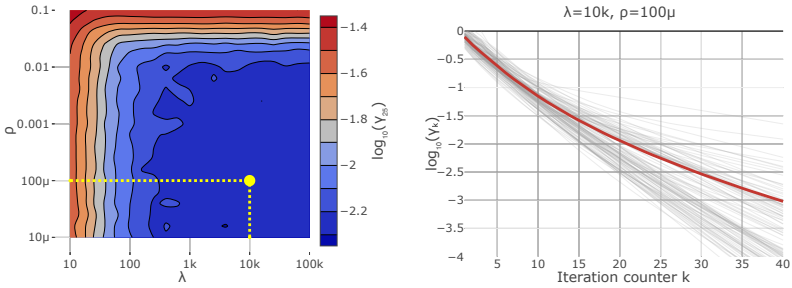
Dielectrophoresis (DEP) is a physical phenomenon in which a force is exerted on a polarizable object by shaping the surrounding electric field. The electric field is shaped by changing alternating voltages applied to electrodes in the vicinity of the manipulated objects. In case of a phase-shift control [5], the model of the DEP force acting on a spherical object is given by

$$\boldsymbol{f}_{\mathrm{DEP}}(\boldsymbol{u},\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{c}^{\mathrm{T}}\Psi_x\,\boldsymbol{c} + \boldsymbol{s}^{\mathrm{T}}\Psi_x\,\boldsymbol{s} + \boldsymbol{c}^{\mathrm{T}}\Omega_x\,\boldsymbol{s} \\ \boldsymbol{c}^{\mathrm{T}}\Psi_y\,\boldsymbol{c} + \boldsymbol{s}^{\mathrm{T}}\Psi_y\,\boldsymbol{s} + \boldsymbol{c}^{\mathrm{T}}\Omega_y\,\boldsymbol{s} \\ \boldsymbol{c}^{\mathrm{T}}\Psi_z\,\boldsymbol{c} + \boldsymbol{s}^{\mathrm{T}}\Psi_z\,\boldsymbol{s} + \boldsymbol{c}^{\mathrm{T}}\Omega_z\,\boldsymbol{s} \end{bmatrix} \tag{4.31}$$

where $\Psi_a, \Omega_a \in \mathbb{R}^{n\times n}, a \in \{x,y,z\}$; vectors $\boldsymbol{c}$ and $\boldsymbol{s}$ denote cosine and sine of the actuators' commands $\boldsymbol{u}$, that is $\boldsymbol{c} = [\cos(u_1),\ldots,\cos(u_n)]^{\mathrm{T}}$ and $\boldsymbol{s} = [\sin(u_1),\ldots,\sin(u_n)]^{\mathrm{T}}$. Even though not explicitly stated, matrices Ψ_a and Ω_a depend on the position of the object. Here, the actuators' commands $\boldsymbol{u}$ represent phase shifts of voltages applied to the electrodes. Phase shifts are, by principle, unlimited and thus $\overline{u} = \infty$ and $\underline{u} = -\infty$. The derivation of the force model can be found in the Appendix A.1 and more on feedback manipulation by DEP in [5] and [45].

The numerical experiments were carried out with several spherical objects located above a 16×16 matrix electrode array. The objects were polystyrene spheres with 50 µm in diameter, the electrodes were squares with the width of 50 µm, and the inter-electrode gap of 50 µm. Use of similar electrode arrays can be found in the literature [46, 47]. The sets $\mathcal{C}_i$ and $\mathcal{D}_i$ are given by all electrodes lying inside the circle centered at the ith object's position and with radii 550 µm ($\approx$ 5.5 electrodes) and 300 µm ($\approx$ 3 electrodes), respectively.

3 The code implementing the numerical experiments in this section is available at `https://github.com/martingurtner/DistCtrl4DistMan.jl`.



(a) Parameter tuning

(b) Convergence for the selected parameters

FIGURE 4.6: (Dielectrophoresis) The convergence of Algorithm 2 for varying values of λ and ρ. The optimization is stopped after 25 iterations.

The results are summarized in Fig. 4.6 and Fig. 4.7. First, we varied parameters λ and ρ and for each pair we solved 500 randomly generated optimization problems (4.6) with five objects by Algorithm 2. The heatmap in Fig. 4.6a visualizes the mean of γ_k (the global convergence measure) over the experiments after 25 iterations. Based on this experiment, we fixed $\lambda = 10000$ and $\rho = 0.0001$ as the values for which the algorithm converges the fastest. Convergence measure γ_k of individual experiments (in grey) and also the mean convergence (in red) for the fixed parameters are plotted in Fig. 4.6b. Figure 4.7 displays solutions of two randomly generated problems. The top figure in Fig 4.7 demonstrates the advantage of Algorithm 2 over Algorithm 1. Both algorithms optimize over the darker tone actuators, but Algorithm 2 also considers the lighter tone actuators in the force model; this helps Algorithm 2 to achieve better accuracy of the exerted force while keeping low computational complexity.

In contrast to magnetophoresis and acoustophoresis, we did not have a large enough (in terms of the number of actuators) dielectrophoretic manipulation platform at our disposal to test the proposed algorithm on real hardware.

4.3.2 *Magnetophoresis*

Magnetophoresis is a physical phenomenon enabling manipulation of feromagnetic objects by shaping the surrounding magnetic field. Here, we consider an optimization problem emerging in the modular planar ma-

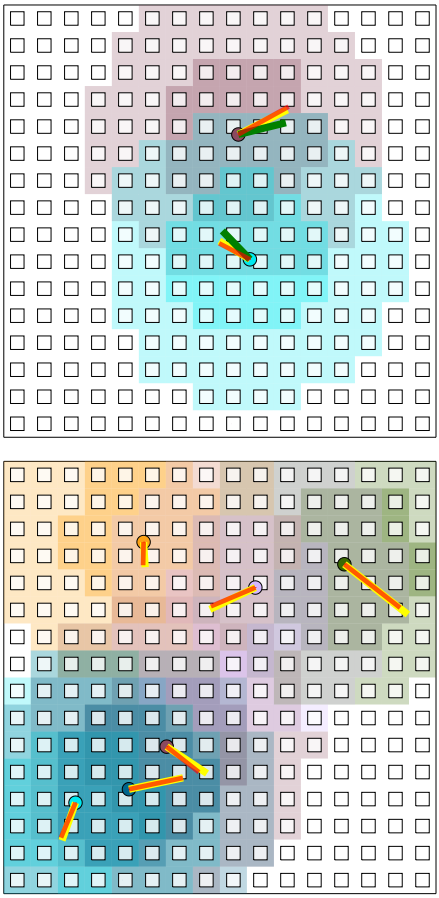


FIGURE 4.7: Solutions of two instances of the distributed optimization problem. The desired and developed forces are visualized by red and yellow lines, respectively. Furthermore, the green lines show the developed forces when Algorithm 1 (instead of Algorithm 2) is used. Electrodes used by each agent are visualized by the agent's color while they have either a darker tone if they are also optimized over or a brighter tone if they are considered in the force model but not optimized over; the distinction should be clear from the experiment in the bottom. Algorithm 2 considered only the darker tone actuators.

nipulation platform called *MagMan*. This platform manipulates steel balls by changing currents flowing through the windings of coils arranged in a regular grid under the balls. For a more detailed description, see [8] and [48].

For MagMan platform, the magnetic force acting on a steel ball can be approximately modeled by

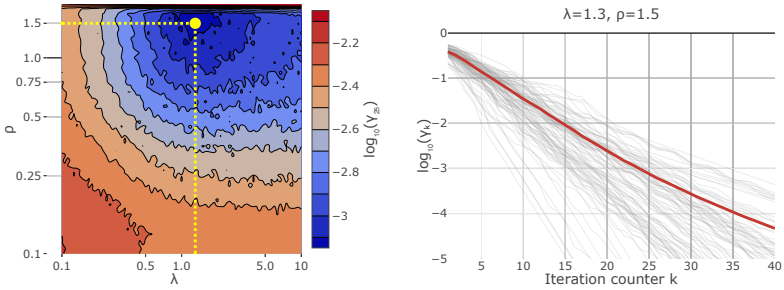
$$f_{\mathrm{MAG}}(\boldsymbol{u}, \boldsymbol{x}) = \boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}, \tag{4.32}$$

where $\boldsymbol{G}$ is a $m \times n$ matrix with n being the number of coils and $m = 2$ (since the ball's motion is restricted to a plane). Even though not explicitly stated, the matrix $\boldsymbol{G}$ depends on the position $\boldsymbol{x}$ of the ball. Actuators' commands $\boldsymbol{u}$ correspond to the currents via an invertible nonlinear map representing quadratic relation between the current and the force; thus, we optimize $\boldsymbol{u}$ and assume that one can always compute the currents by inverting the nonlinear map. We consider $\boldsymbol{u}$ normalized; hence we set $\underline{u} = 0$ and $\overline{u} = 1$. A derivation of this model can be found in [8].

For numerical experiments, we used the parameters of the force model (4.32) identified in [8] for a 20 mm ball in diameter. Here, we defined the sets $\mathcal{C}_i$ and $\mathcal{D}_i$ as the sets containing all coils within the circles centered at the ith object's position and with radii 75 mm ($\approx$ 3 coils) and 50 mm ($\approx$ 2 coils), respectively.

Similarly to the DEP platform, also here we applied Algorithm 2 to randomly generated problems to find parameters λ and ρ resulting in a fast convergence rate. For each pair of λ and ρ we generated 1000 problems (4.6) with the force model (4.32), with five steel balls, and with a 8×8 coil array. Figure 4.8a displays a heatmap visualizing the resulting mean convergence measure γ_{25} over these experiments. Based on these experiments, we fixed the parameters to $\lambda = 1.3$ and $\rho = 1.5$. Figures 4.9 show solutions of two randomly generated problems. It is worthy of note that, as opposed to the DEP platform, here, the optimization problem (4.6) is convex; thus, convergence to a global solution is ensured.

Furthermore, we deployed Algorithm 2 in the same setup to the real MagMan platform. Since the platform is currently lacking the distributed sensing capability, the algorithm was executed at a central node Raspberry Pi 3B+. Every 20 ms, the position control system computes the forces needed to move the balls towards the reference positions. Algorithm 2 was executed to solve the optimization problem of finding such actuator commands that these forces are developed. The algorithm was stopped after 25 iterations, which took approximately 1 ms. Figure 4.10 shows an experiment with six balls steered along circular trajectories.



(a) Parameter tuning

(b) Convergence for the selected parameters

FIGURE 4.8: (Magnetophoresis) The convergence of Algorithm 2 for varying values of λ and ρ. The optimization is stopped after 25 iterations.

4.3.3 *Acoustophoresis*

In acoustophoresis, one can manipulate objects by shaping the surrounding acoustic pressure field. Platforms using this principle are presented in [49] and [6]. Here, we consider manipulation due to the generation of a high-pressure region near the manipulated object to which the object reacts by moving to a lower pressure region. The acoustic pressure field is shaped by an array of ultrasonic transducers. Thus, the goal of the control system is to excite the transducers so that high-pressure points of certain values are generated at certain positions.

As with the DEP, also here we consider the phase-shift control: only the phases of the driving signals are optimized over. The model of the amplitude of the pressure at a point is given by

$$|p(\boldsymbol{u}, \boldsymbol{x})|^2 = \boldsymbol{c}^{\mathrm{T}} \boldsymbol{M}_1 \boldsymbol{c} + \boldsymbol{s}^{\mathrm{T}} \boldsymbol{M}_1 \boldsymbol{s} + \boldsymbol{c}^{\mathrm{T}} \boldsymbol{M}_2 \boldsymbol{s}, \tag{4.33}$$

where $\boldsymbol{M}_1, \boldsymbol{M}_2 \in \mathbb{C}^{n \times n}$ are matrices dependent on the position $\boldsymbol{x}$ of the pressure point, vectors $\boldsymbol{c}$ and $\boldsymbol{s}$ denote cosine and sine of the actuators' commands $\boldsymbol{u}$, that is $\boldsymbol{c} = [\cos(u_1), \ldots, \cos(u_n)]^{\mathrm{T}}$ and $\boldsymbol{s} = [\sin(u_1), \ldots, \sin(u_n)]^{\mathrm{T}}$, and $\boldsymbol{u}$ represents the phase shifts. A more detailed description of the model can be found in Appendix A.2 and in [6].

The optimization problem for this specific manipulation platform has the following form

$$\underset{\boldsymbol{u} \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^{N} \| |p(\boldsymbol{u}, \boldsymbol{x}_{i,\mathrm{p}})|^2 - P_i \|_2^2, \tag{4.34}$$

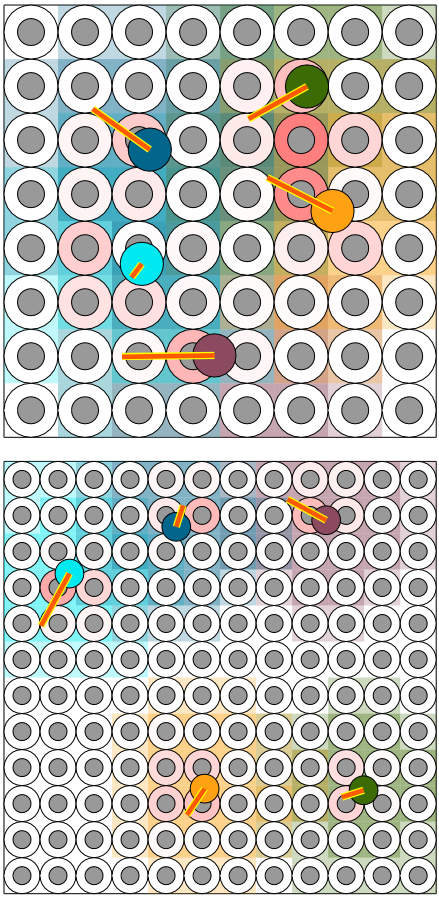
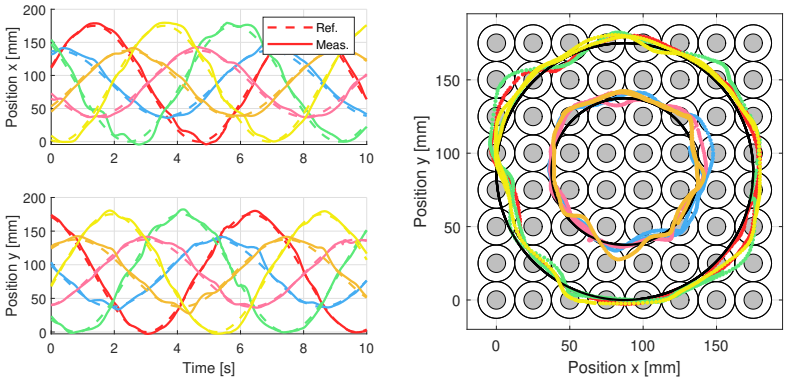


FIGURE 4.9: Solutions of two instances of the distributed optimization problem. The desired and developed force are visualized by red and yellow lines, respectively. Coils used by each agent are visualized by the agent's color while they have either a darker tone if they are also optimized over or a brighter tone if they are considered in the force model but not optimized over.



(a) Measured trajectories

(b) Long-exposure snapshot of the MagMan platform from the top view

FIGURE 4.10: Manipulation of six balls by MagMan platform along circular trajectories.

where $\boldsymbol{x}_{i,\mathrm{p}}$ is the position and P_i is the desired value of the ith high-pressure point to be generated. There is one additional step in contrast to the previous two platforms. We do not optimize the desired forces set by a position controller directly. We first need to compute the desired pressure points based on the desired forces. This step is described in detail in [6].

The numerical experiments were conducted for a 16×16 array of 10 mm ultrasonic transducers; hence $n = 256$. The sets $\mathcal{C}_i$ and $\mathcal{D}_i$ are given by all transducers lying inside the circles centered at the ith pressure point's position and with radii 65 mm (≈ 6.5 transducers) and 45 mm (≈ 4.5 transducers), respectively. All pressure points are generated at fixed distance of 65 mm from the array of transducers along the vertical axis of the array.

The results of numerical experiments are displayed in Fig. 4.12 and Fig. 4.11. Similarly to the two previous platforms, at first, we varied parameters λ and ρ and for each pair, we solved 1000 randomly generated distributed optimization problems with five pressure points by Algorithm 2. The heatmap in Fig. 4.12a visualizes the resulting mean of convergence measure γ_{25}. Based on this data, we fixed $\lambda = 10000$ and $\rho = 0.0001$. Mean convergence measure and convergence measure of individual experiments for these fixed parameters are shown in Fig. 4.12b. Figure 4.11 documents an experiment we conducted on a real acoustophoretic manipulation platform consisting of a 16×16 array of ultrasonic transducers. We manipulated three objects along predefined trajectories by generating a high-pressure point for each object. The values and positions of high-pressure points were determined by a position controller described in [6]. Phase shifts resulting in generating the desired high-pressure points were determined by Algorithm 2 with the same parameters as we used in the numerical experiments. Similarly to the magnetic platform, for the lack of the distributed sensing capability, the control system ran at a central node Raspberry Pi 4, and the control period was 40 ms.

4.3.4 *Centralized vs distributed*

One of the major advantages of the proposed distributed algorithm is increased scalability of solving the optimization problem (4.2). To prove this point, we compared the proposed distributed algorithm with a centralized version in two scenarios: computation in threads at one computational node and fully parallelized computation. We generated a set of optimization problems (4.2) with increasing complexity—the number of manipulated

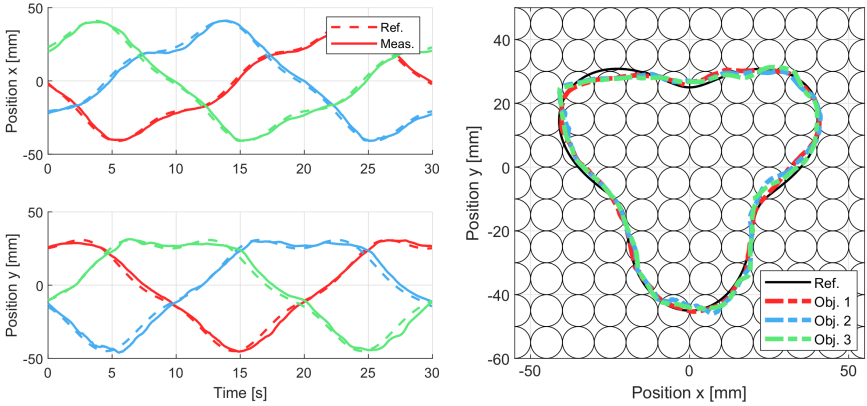
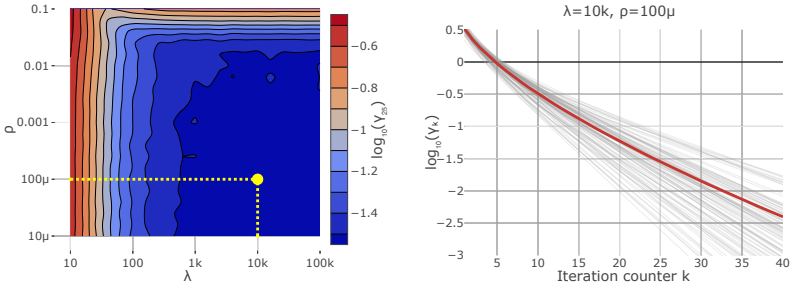


FIGURE 4.11: Measured and reference trajectories of three manipulated spherical objects are shown. These trajectories were obtained by conducting an experiment on a real acoustophoretic manipulation platform.



(a) Parameter tuning

(b) Convergence for the selected parameters

FIGURE 4.12: (Acoustophoresis) The convergence of Algorithm 2 for varying values of λ and ρ. The optimization is stopped after 25 iterations.

objects and actuators—and solved them both by the proposed distributed algorithm and by a centralized algorithm.

We used Algorithm 1 as the distributed algorithm with parameters λ and ρ specified for each platform in the preceding sections. We did not use the more efficient Algorithm 2 to make the comparison fairer as the centralized algorithm cannot make use of non-optimized actuators that are considered in the force model. To solve the problems in a centralized fashion, we used the projected Levenberg-Marquardt algorithm [50] with the same damping factor λ as was used in the distributed version. Projected Levenberg-Marquardt algorithm proceeds iteratively. In each iteration, the optimization problem is linearized, solved without constraints on the decision variables, and then the solution is projected onto the feasible set defined by the constraints on the decision variables; this makes the projected Levenberg-Marquardt algorithm probably the most similar centralized algorithm to our proposed distributed algorithm. Since the number of agents compared to the number of actuators is relatively small even for the centralized algorithm, we also used the *push-through identity* for solving the linearized problem efficiently in the projected Levenberg-Marquardt algorithm. The centralized algorithm optimized all actuators in the array, whereas the distributed one, considering all subproblems, optimized only actuators with indices from $\bigcup_{i=1}^{N} \mathcal{C}_i$ (actuators that are close enough to at least one of the manipulated objects). This potentially could make the comparison unfair as the centralized algorithm solves larger optimization problems (optimizes more actuators). Nevertheless, we designed the optimization problems in the comparisons so that the set $\bigcup_{i=1}^{N} \mathcal{C}_i$ is close to the set of all actuators. This is done through uniform distribution of positions of the manipulated objects and array size dependent on the number of manipulated objects. The size of the actuator array was set to $\lceil Nk \rceil \times \lceil Nk \rceil$, where N is the number of agents and k is a chosen number of actuators per agent. The positions of agents were drawn from the uniform probability distribution over the actuator array; hence the density of agents over the actuator array was, on average, the same. The algorithms did not have any iteration count limit and were stopped when the following condition was met:

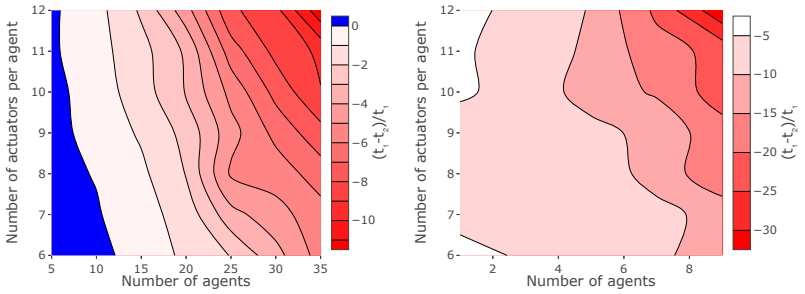
$$\frac{1}{N} \sum_{i=1\ldots,N} \|\boldsymbol{F}_i - \boldsymbol{f}_i(\hat{\boldsymbol{u}}_i, \boldsymbol{x}_i)\|_2^2 / \|\boldsymbol{F}_i\|_2^2 < \epsilon,$$

where ϵ is a chosen threshold. The experiments were run on a laptop Apple Macbook Air M1 with 8 GB of RAM.

To set the expectations, we mention that the projected Levenberg-Marquardt algorithm converges locally quadratically [50] whereas ADMM converges only linearly [31]. Thus the centralized algorithm finds a better solution in fewer iterations. On the other hand, the centralized algorithm solves larger problems (both in terms of the number of actuators and required forces to satisfy) in each iteration, and thus each iteration takes more time. Since the complexity of solving the linear system of equations in the projected Levenberg-Marquardt is cubic, one could expect that the slowly converging ADMM will outperform the centralized algorithm for large enough problems.

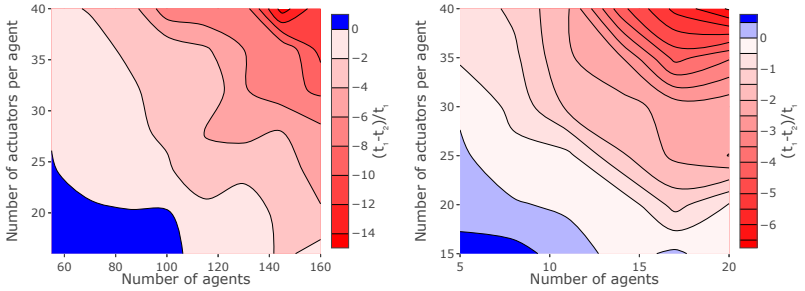
Experiment where the subproblems of the distributed algorithm are solved at one computational unit but in separate threads (see Fig. 4.2b) is summarized in Fig. 4.13 (a,c,e). The time needed for solving each problem was measured as *wall-clock time*. That is, the sub-problems ran in parallel, but the concurrency is limited by the number of threads the processor can process at one time. One can see that the proposed distributed algorithm scales better and outperforms the centralized algorithm for large enough optimization problems. Nevertheless, the number of agents the proposed distributed algorithm needs to outperform the centralized one is relatively large. Similarly, the number of actuators per agent must be relatively large so that the distributed algorithm outperforms the centralized one. The latter can be explained by the fact that more actuators per agent results in the object being farther apart. Thus, the actuators needed to generate the required force are not shared, it is easier to reach consensus, and the algorithm converges faster to a local optimum. In contrast, the centralized algorithm cannot take any advantage of the objects being farther apart; it simply optimizes more actuators; the optimization problem is larger and thus converges slower to a local optimum.

The simulation of a fully distributed experiment is summarized in Fig. 4.13 (b,d,f). Here, the subproblems of the distributed algorithm are also solved in separate threads. Nevertheless, the time needed to solve each subproblem is measured separately, and the maximum time is compared to the time needed to solve the centralized variant. This way, we simulate a fully distributed deployment of the algorithm where each subproblem would be solved by a separate computational unit (see Fig. 4.2c). Apparently, the proposed distributed algorithm outperforms the centralized one even for a very low number of agents and a very low number of actuators per agent.



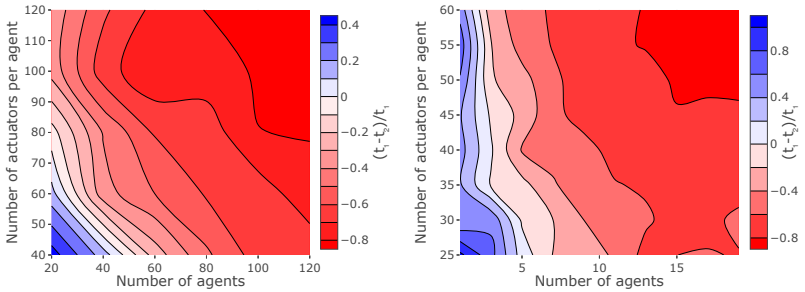
(a) Magnetophoresis: one computational node

(b) Magnetophoresis: fully parallel computation



(c) Dielectrophoresis: one computational node

(d) Dielectrophoresis: fully parallel computation



(e) Acoustophoresis: one computational node

(f) Acoustophoresis: fully parallel computation

FIGURE 4.13: Scalability of the proposed distributed algorithm in comparison to a centralized algorithm. The contour maps visualize a normalized difference in solution time t_1 of the proposed distributed algorithm with the solution time t_2 of the centralized algorithm. The red-tone colors mark problem instances for which the proposed algorithm was faster, the blue-tone colors mark problem instances for which the centralized algorithm performed better, and the white color marks the edge cases for which both algorithms performed the same.

4.4 CONCLUSION

We introduced algorithms distributing an optimization problem emerging in distributed manipulation: computing commands for actuators such that some desired forces are exerted at given positions. The algorithms decompose this optimization problem into several smaller optimization problems; each is responsible for exerting a desired force acting on one manipulated object or for exerting desired forces acting on several objects. These smaller optimization problems are then solved collaboratively so that the resulting actuators' commands are not in conflict. The proposed algorithms enable making the distributed manipulation platforms modular and scalable as the need for a central coordinating node is eliminated; the actuators' commands can be computed distributively on individual actuator modules the distributed manipulation platform consists of. We used the proposed algorithms for computing actuator commands for distributed manipulation platforms utilizing three different physical phenomena of actuation (dielectrophoresis, magnetophoresis, and acoustophoresis), and we believe it can be adapted to other platforms as well. The future research will focus on improving the convergence rate by using accelerated variants of ADMM and on finding proof of convergence.

FUNDING

This work was supported by the Czech Science Foundation under Grant 19-18424S.

APPENDIX

A.1 *DEP force model*

Components of the DEP force consist of two contributions:

$$F_{\mathrm{DEP},a} = F_{\mathrm{cDEP},a} + F_{\mathrm{twDEP},a}, \qquad a \in \{x, y, z\}, \tag{4.35}$$

where $F_{\mathrm{cDEP},a}$ and $F_{\mathrm{twDEP},a}$ are forces exerted due to so-called conventional and traveling wave DEP, respectively. These forces can be modeled by the following relations

$$F_{\mathrm{cDEP},a} = 2\,k_{\mathrm{r}}\,\boldsymbol{u}^{\mathrm{H}}\boldsymbol{\Lambda}_a\boldsymbol{\Gamma}^{\mathrm{T}}\boldsymbol{u}, \tag{4.36a}$$

$$F_{\mathrm{twDEP},a} = \imath\,k_{\mathrm{i}}\,\boldsymbol{u}^{\mathrm{H}}\left(\boldsymbol{\Lambda}_a\boldsymbol{\Gamma}^{\mathrm{T}} - \boldsymbol{\Gamma}\boldsymbol{\Lambda}_a^{\mathrm{T}}\right)\boldsymbol{u}, \tag{4.36b}$$

where $\imath = \sqrt{-1}$, k_r and k_i are scalars, $\boldsymbol{\Lambda}_a, \boldsymbol{\Gamma} \in \mathbb{R}^{n\times 3}$ are matrices depending on the object's position, $\boldsymbol{u}$ represents the voltage signals applied to the electrodes (more on that later) and $\boldsymbol{u}^\mathrm{H}$ is the Hermitian transpose of $\boldsymbol{u}$. Detailed derivation of (4.36) can be found in [8].

Defining a matrix $\boldsymbol{A}_{\mathrm{DEP},a}$ as

$$\boldsymbol{A}_{\mathrm{DEP},a} = 2\,k_\mathrm{r}\,\boldsymbol{\Lambda}_a\boldsymbol{\Gamma}^\mathrm{T} + \imath\,k_\mathrm{i}\left(\boldsymbol{\Lambda}_a\boldsymbol{\Gamma}^\mathrm{T} - \boldsymbol{\Gamma}\boldsymbol{\Lambda}_a^\mathrm{T}\right), \tag{4.37}$$

we get a quadratic relation for the DEP force

$$F_{\mathrm{DEP},a} = \boldsymbol{u}^\mathrm{H}\,\boldsymbol{A}_{\mathrm{DEP},a}\,\boldsymbol{u}, \qquad a \in \{x, y, z\}. \tag{4.38}$$

The input variable can be the amplitude, phase-shift control or combination of both. When phase-shift control is used, only the phases of the voltage signals applied to the electrodes change, that is $\mathbf{u}(\boldsymbol{\theta}) = U[e^{\imath\theta_1}, \ldots, e^{\imath\theta_n}]^\mathrm{T}$, U is the amplitude of the voltage signals and $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_n]^\mathrm{T}$ is the vector of phase shifts.

To get rid of the complex numbers in (4.38), we can rewrite each quadratic form to

$$\mathbf{u}^\mathrm{H}\,\mathbf{A}_{\mathrm{DEP},a}\,\mathbf{u} = \mathbf{c}^\mathrm{T}\,\Psi_a\,\mathbf{c} + \mathbf{s}^\mathrm{T}\,\Psi_a\,\mathbf{s} + \mathbf{c}^\mathrm{T}\,\Omega_a\,\mathbf{s}, \tag{4.39}$$

where $\mathbf{c} = [\cos\theta_1, \ldots, \cos\theta_n]^\mathrm{T}$ and $\mathbf{s} = [\sin\theta_1, \ldots, \sin\theta_n]^\mathrm{T}$ and

$$\Psi_a = 2\,U^2\,k_\mathrm{r}\,\boldsymbol{\Lambda}_a\boldsymbol{\Gamma}^\mathrm{T}, \tag{4.40}$$

$$\Omega_a = 2\,U^2\,k_\mathrm{i}\left(\boldsymbol{\Gamma}\boldsymbol{\Lambda}_a^\mathrm{T} - \boldsymbol{\Lambda}_a\boldsymbol{\Gamma}^\mathrm{T}\right). \tag{4.41}$$

Therefore, the DEP model we used in this paper is

$$f_\mathrm{DEP}(\boldsymbol{\theta}) = \begin{bmatrix} \mathbf{c}^\mathrm{T}\,\Psi_x\,\mathbf{c} + \mathbf{s}^\mathrm{T}\,\Psi_x\,\mathbf{s} + \mathbf{c}^\mathrm{T}\,\Omega_x\,\mathbf{s} \\ \mathbf{c}^\mathrm{T}\,\Psi_y\,\mathbf{c} + \mathbf{s}^\mathrm{T}\,\Psi_y\,\mathbf{s} + \mathbf{c}^\mathrm{T}\,\Omega_y\,\mathbf{s} \\ \mathbf{c}^\mathrm{T}\,\Psi_z\,\mathbf{c} + \mathbf{s}^\mathrm{T}\,\Psi_z\,\mathbf{s} + \mathbf{c}^\mathrm{T}\,\Omega_z\,\mathbf{s} \end{bmatrix}. \tag{4.42}$$

A.2 *Acoustic Pressure model*

The model of the magnitude of the acoustic pressure field is given by a sum of contributions from individual ultrasonic transducers. In case of phase-shift control, the sum is

$$|p(\boldsymbol{\theta}, \boldsymbol{x})| = \left|\sum_{i=1}^{n} a_i(\boldsymbol{x})\mathrm{e}^{j\theta_i}\right|, \tag{4.43}$$

where $|\cdot|$ denotes the magnitude of a complex number, $\boldsymbol{x}$ is the position of the pressure point, $\grave{} = [\theta_1, \ldots, \theta_n]^{\mathrm{T}}$ are the phase shifts of signals exciting individual transducers and constants $a_i(\boldsymbol{x}) \in \mathbb{C}$ are given by the distance of the pressure point $\boldsymbol{x}$ in space from the transducer, by the directivity of the transducer and by nominal power of the transducer (for details, see [51]). For brevity, we will omit the explicit dependence on $\boldsymbol{x}$ and $\boldsymbol{\theta}$ where the dependence is obvious.

Let us define a vector $\mathbf{a} = [a_1, \ldots, a_n]^{\mathrm{T}}$ and vector $\mathbf{u} = [\mathrm{e}^{i\theta_1}, \ldots, \mathrm{e}^{i\theta_n}]^{\mathrm{T}}$. Then (4.43) can be rewritten to

$$|p|^2 = \boldsymbol{u}^{\mathrm{H}}\, \bar{\boldsymbol{a}}\boldsymbol{a}^{\mathrm{T}}\, \boldsymbol{u}, \tag{4.44}$$

where $\bar{\boldsymbol{a}}$ is a vector of complex conjugated components of $\boldsymbol{a}$. To get rid of the complex numbers, we define vectors $\boldsymbol{c}$ and $\boldsymbol{s}$ as cosines and sines of the components of $\boldsymbol{\theta}$, that is $\boldsymbol{c} = [\cos(\theta_1), \ldots, \cos(\theta_n)]^{\mathrm{T}}$ and $\boldsymbol{s} = [\sin(\theta_1), \ldots, \sin(\theta_n)]^{\mathrm{T}}$. Then, one can easily show, that (4.44) can be rewritten to

$$|p|^2 = \boldsymbol{c}^{\mathrm{T}}\, \boldsymbol{M}_1\, \boldsymbol{c} + \boldsymbol{s}^{\mathrm{T}}\, \boldsymbol{M}_1\, \boldsymbol{s} + \boldsymbol{c}^{\mathrm{T}}\, \boldsymbol{M}_2\, \boldsymbol{s}, \tag{4.45}$$

with matrices $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ defined as follows

$$\boldsymbol{M}_1 = \begin{bmatrix} \boldsymbol{a}_{\mathrm{r}} & \boldsymbol{a}_{\mathrm{i}} \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_{\mathrm{r}}^{\mathrm{T}} \\ \boldsymbol{a}_{\mathrm{i}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{4.46}$$

$$\boldsymbol{M}_2 = \begin{bmatrix} \boldsymbol{a}_{\mathrm{r}} & \boldsymbol{a}_{\mathrm{i}} \end{bmatrix} \begin{bmatrix} 0 & -2 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_{\mathrm{r}}^{\mathrm{T}} \\ \boldsymbol{a}_{\mathrm{i}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{4.47}$$

where $\boldsymbol{a}_{\mathrm{r}}$ and $\boldsymbol{a}_{\mathrm{i}}$ are the vectors of real and imaginary parts of components of $\boldsymbol{a}$, respectively.

REFERENCES

1. Gurtner, M., Zemánek, J. & Hurák, Z. ADMM-based distributed control for distributed manipulation by shaping physical force fields. *International Journal of Robotics Research*. As of December 2022, the paper is accepted but has not been published yet. (2022).
2. Bohringer, K.-F., Donald, B. R., Mihailovich, R. & MacDonald, N. C. *A theory of manipulation and control for microfabricated actuator arrays* in *Proceedings IEEE Micro Electro Mechanical Systems An Investigation of Micro Structures, Sensors, Actuators, Machines and Robotic Systems* (1994), 102.

3. Shorinwa, O. & Schwager, M. *Scalable collaborative manipulation with distributed trajectory planning* in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020), 9108.

4. Alonso-Mora, J., Baker, S. & Rus, D. Multi-robot formation control and object transport in dynamic environments via constrained optimization. *The International Journal of Robotics Research* **36**, 1000 (2017).

5. Zemánek, J., Michálek, T. & Hurák, Z. Phase-shift feedback control for dielectrophoretic micromanipulation. *Lab on a Chip* **18**, 1793 (2018).

6. Matouš, J., Kollarčík, A., Gurtner, M., Michálek, T. & Hurák, Z. Optimization-based Feedback Manipulation Through an Array of Ultrasonic Transducers. *IFAC-PapersOnLine* **52**, 483 (2019).

7. Chaudhary, S. & Shapiro, B. Arbitrary steering of multiple particles independently in an electro-osmotically driven microfluidic system. *IEEE Transactions on Control Systems Technology* **14**, 669 (2006).

8. Zemánek, J. *Distributed manipulation by controlling force fields through arrays of actuators* Doctoral thesis (Czech Technical University in Prague, Czech Republic, 2018).

9. Luntz, J. E., Messner, W. & Choset, H. Distributed manipulation using discrete actuator arrays. *The International Journal of Robotics Research* **20**, 553 (2001).

10. Bohringer, K.-F., Donald, B. R., Mihailovich, R. & MacDonald, N. C. *Sensorless manipulation using massively parallel microfabricated actuator arrays* in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation* (IEEE, 1994), 826.

11. Pister, K. S. J., Fearing, R. S. & Howe, R. T. *A planar air levitated electrostatic actuator system* in *IEEE Proceedings on Micro Electro Mechanical Systems, An Investigation of Micro Structures, Sensors, Actuators, Machines and Robots.* (1990), 67.

12. Konishi, S. & Fujita, H. A conveyance system using air flow based on the concept of distributed micro motion systems. *Journal of Microelectromechanical Systems* **3**, 54 (1994).

13. Furuhata, T., Hirano, T. & Fujita, H. *Array-driven ultrasonic microactuators* in *TRANSDUCERS'91: 1991 International Conference on Solid-State Sensors and Actuators. Digest of Technical Papers* (1991), 1056.

14. Liu, C., Tsao, T., Tai, Y.-C., Liu, W., Will, P. & Ho, C.-M. *A micromachined permalloy magnetic actuator array for micro robotics assembly systems* in *Proceedings of the International Solid-State Sensors and Actuators Conference-TRANSDUCERS'95* **1** (1995), 328.

15. Bohringer, K. F., Donald, B. R., MacDonald, N. C., Kovacs, G. T. & Suh, J. W. Computational methods for design and control of MEMS micromanipulator arrays. *IEEE Computational Science and Engineering* **4**, 17 (1997).

16. Bohringer, K.-F., Donald, B. R. & MacDonald, N. C. Programmable Force Fields for Distributed Manipulation, with Applications to MEMS Actuator Arrays and Vibratory Parts Feeders. *The International Journal of Robotics Research* **18**, 168 (1999).

17. Erdmann, M. A. & Mason, M. T. An exploration of sensorless manipulation. *IEEE Journal on Robotics and Automation* **4**, 369 (1988).

18. Kavraki, L. E. *Part orientation with programmable vector fields: two stable equilibria for most parts* in *Proceedings of International Conference on Robotics and Automation* **3** (1997), 2446.

19. Goldberg, K. Y. Orienting polygonal parts without sensors. *Algorithmica* **10**, 201 (1993).

20. Fromherz, M. P. J. & Jackson, W. B. Force allocation in a large-scale distributed active surface. *IEEE Transactions on Control Systems Technology* **11**, 641 (2003).

21. Murphey, T. D. & Burdick, J. W. Feedback control methods for distributed manipulation systems that involve mechanical contacts. *The International Journal of Robotics Research* **23**, 763 (2004).

22. Boutoustous, K., Laurent, G. J., Dedu, E., Matignon, L., Bourgeois, J. & Le Fort-Piat, N. *Distributed control architecture for smart surfaces* in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2010), 2018.

23. El Baz, D., Boyer, V., Bourgeois, J., Dedu, E. & Boutoustous, K. Distributed part differentiation in a smart surface. *Mechatronics. Special Issue on Distributed Intelligent MEMS: from hardware to software* **22**, 522 (2012).

24. Diller, E., Giltinan, J. & Sitti, M. Independent control of multiple magnetic microrobots in three dimensions. *The International Journal of Robotics Research* **32**, 614 (2013).

25. Kummer, M. P., Abbott, J. J., Kratochvil, B. E., Borer, R., Sengul, A. & Nelson, B. J. OctoMag: An Electromagnetic System for 5-DOF Wireless Micromanipulation. *IEEE Transactions on Robotics* **26**, 1006 (2010).

26. Bedillion, M. & Messner, W. Control for Actuator Arrays. *The International Journal of Robotics Research* **28**, 868 (2009).

27. Donald, B. R., Levey, C. G., Paprotny, I. & Rus, D. Planning and control for microassembly of structures composed of stress-engineered MEMS microrobots. *The International Journal of Robotics Research* **32**, 218 (2013).

28. Lunze, J. *Networked Control of Multi-Agent Systems* (Bookmundo Direct, S.l., 2019).

29. Parikh, N. & Boyd, S. Proximal algorithms. *Foundations and Trends® in Optimization* **1**, 127 (2014).

30. Boyd, S., Parikh, N., Chu, E., Peleato, B. & Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* **3**, 1 (2011).

31. He, B. & Yuan, X. On the $O(1/n)$ Convergence Rate of the Douglas–Rachford Alternating Direction Method. *SIAM Journal on Numerical Analysis* **50**, 700 (2012).

32. Yang, L., Pong, T. & Chen, X. Alternating Direction Method of Multipliers for a Class of Nonconvex and Nonsmooth Problems with Applications to Background/Foreground Extraction. *SIAM Journal on Imaging Sciences* **10**, 74 (2017).

33. Shen, Y., Wen, Z. & Zhang, Y. Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software* **29**, 239 (2014).

34. Xu, Y., Yin, W., Wen, Z. & Zhang, Y. An alternating direction algorithm for matrix completion with nonnegative factors. *Frontiers of Mathematics in China* **7**, 365 (2012).

35. Wen, Z., Yang, C., Liu, X. & Marchesini, S. Alternating direction methods for classical and ptychographic phase retrieval. *Inverse Problems* **28**, 115010 (2012).

36. Jiang, B., Ma, S. & Zhang, S. Alternating direction method of multipliers for real and complex polynomial optimization models. *Optimization* **63**, 883 (2014).

37. Chartrand, R. & Wohlberg, B. *A nonconvex ADMM algorithm for group sparsity with sparse groups* in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (2013), 6009.

38. Overby, M., Brown, G. E., Li, J. & Narain, R. ADMM $\supseteq$ Projective Dynamics: Fast Simulation of Hyperelastic Models with Dynamic Constraints. *IEEE Transactions on visualization and computer graphics* **23**, 2222 (2017).

39. Zha, Z., Zhang, X., Wu, Y., Wang, Q., Liu, X., Tang, L. & Yuan, X. Non-convex weighted l_p nuclear norm based ADMM framework for image restoration. *Neurocomputing* **311**, 209 (2018).

40. Hong, M., Luo, Z.-Q. & Razaviyayn, M. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization* **26**, 337 (2016).

41. Wang, Y., Yin, W. & Zeng, J. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing* **78**, 29 (2019).

42. Nocedal, J. & Wright, S. *Numerical Optimization* 2nd edition (Springer, New York, 2006).

43. Diehl, M., Bock, H. G. & Schlöder, J. P. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on control and optimization* **43**, 1714 (2005).

44. Olfati-Saber, R., Fax, J. A. & Murray, R. M. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE* **95**, 215 (2007).

45. Kharboutly, M. & Gauthier, M. *High speed closed loop control of a dielectrophoresis-based system* in *2013 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2013), 1446.

46. Gascoyne, P. R., Vykoukal, J. V., Schwartz, J. A., Anderson, T. J., Vykoukal, D. M., Current, K. W., McConaghy, C., Becker, F. F. & Andrews, C. Dielectrophoresis-based programmable fluidic processors. *Lab on a Chip* **4**, 299 (2004).

47. Hunt, T. P., Lee, H. & Westervelt, R. M. Addressable micropost array for the dielectrophoretic manipulation of particles in fluid. *Applied Physics Letters* **85**, 6421 (2004).

48. Zemánek, J., Čelikovsky, S. & Hurák, Z. Time-optimal control for bilinear nonnegative-in-control systems: Application to magnetic manipulation. *IFAC-PapersOnLine* **50**, 16032 (2017).

49. Marzo, A., Seah, S. A., Drinkwater, B. W., Sahoo, D. R., Long, B. & Subramanian, S. Holographic acoustic elements for manipulation of levitated objects. *Nature communications* **6**, 8661 (2015).

50. Kanzow, C., Yamashita, N. & Fukushima, M. Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints. *Journal of Computational and Applied Mathematics* **172**, 375 (2004).

51. Kinsler, L. E., Frey, A. R., Coppens, A. B. & Sanders, J. V. *Fundamentals of Acoustics* 4th edition (Wiley, New York, 1999).

5

COMPACT DIELECTROPHORETIC FEEDBACK MANIPULATION PLATFORM

Despite the popularity of the concept of a *lab on a chip*, many research solutions published in this domain seem to rely on the concept of a *chip in a lab* instead – they depend on bulky and expensive versatile laboratory instrumentation. While for some laboratory applications this is not an issue, it does impede further development of truly portable lab-on-a-chip applications. This particularly holds for some microfluidic and electrokinetic feedback (micro)manipulation applications, wherein the measurements of position of the manipulated particles are periodically obtained from images acquired using bulky and expensive microscopes and cameras. In this paper we demonstrate a novel contactless (micro)manipulation device capable of controlled motion of micrometer-size objects in 3D that does not need a microscope, even though a visual feedback control loop is closed. Although it does not constitute a solution fully encapsulated in a single chip, it does offer better portability than some lab-tied solutions. The device utilizes the phenomenon of dielectrophoresis as the actuation mechanism. In particular, dielectrophoretic force field above a planar microelectrode array is shaped by changing the phase shift of voltages applied on to the individual electrodes. The inline digital holography with partially coherent light sources as the mechanism for displaying the manipulated objects. Furthermore, the twin-beam method is used to measure the position of the manipulated objects in 3D (and the full 3D is needed for dielectrophoresis since particles levitate above the electrode array). Thanks to digital holography, the device has a relatively large field of view (compared to conventional miscroscopes) and needs neither lenses nor lasers (no need for bulky and expensive optical components). An experimental demonstration of manipulation of up to eight particles is documented in the paper.

This chapter is based on Gurtner, M. *et al.* Compact Dielectrophoretic Feedback Manipulation Platform. *In prepraration.* (2022)

5.1 INTRODUCTION

Non-contact manipulation of micro-sized objects is essential in many applications since the ability to position and orient micro-objects is naturally required for many tasks, from analyzing biological samples to assembling artificially-made components into functional units. Various approaches based on optical, electrical, magnetic and acoustic forces have been successfully used [2].

In this paper, we focus on manipulation by *dielectrophoresis (DEP)*. DEP is a physical phenomenon where a force acts on a polarizable object surronded by a spatially varying electrical field. By shaping the electrical field both in space and time, the position of a manipulated object can be controlled. DEP has the advantage of being relatively simple from an instrumental point of view since it needs only relatively simple hardware. Only some electrodes and some circuitry setting the electrical potentials on the electrodes are needed. See [3, 4] for a survey of applications of DEP.

Traditionally, DEP is used in an open-loop setup; positions of the manipulated objects are not measured and the electrical potentials are precomputed so that a desired goal is satisfied. Open-loop DEP platforms can, for instance, be used to separate objects with different electrical properties. When feedback is added, one gains the possibility to control the position of individual manipulated objects, but that is at the price of complicating the hardware setup because a position sensor must be added. Typically, the position of manipulated objects is measured by bulky microscopes.

We demonstrate a platform which is compact and allow parallel cell positioning in a large manipulation area in 3D. This is achieved by combining dielectrophoresis with digital inline holographic microscopy for imaging the manipulated objects in a large manipulation area and with twin-beam method for measuring position of the manipulated objects in 3D. Digital holographic microscopy eliminates the need for the bulky and costly optics by processing thr captured holograms so that they look as if they were captured by a misroscope after the processing. Twin-beam [5] method works so that the manipulated objects are illuminated from different angles by two light sources. There are two shadows under each manipulated object. One shadows is used to determine the position of the manipulated object in 2D and the mutual distance of the shadows decodes the position of the object in the remaining third dimension. The setup is compact and cost-effective.

5.2 METHODS

Experimental setup

The render of the hardware setup is shown in Fig. 5.1. All the black parts are 3D printed. The top black part is detachable and houses two illumination sources: LEDs with 50 µm to 100 µm apertures filtering the emitted light. One LED is Würth Elektronik 150060GS75000 (green, 515 nm) and it illuminates the manipulated objects straight from the top. The second LED is Würth Elektronik 150060RS75000 (red, 630 nm) and it illuminates the manipulated objects under the angle of 30°. This top part attaches to the rest of the setup via 3D printed locks. Straight under the green illumination source, there is an electrode array etched in indium-tin-oxide coated glass substrate. A 3D printed pool 10 mm × 10 mm × 3 mm is glued to the top of the electrode array. The pool is filled with deionized water, contains the manipulated objects and is covered by a cover glass from the top to prevent any optical distortion due to uneven water surface. An image sensor without any lenses (Leopard Imaging LI-IMX477-MIPI-M12 camera module, 1.55 µm pixel size, 4056 pixel × 3040 pixel resolution) is placed right below the electrode array. The wavelengths of the green and red LEDs were chosen so that they are close to the peak sensitivities of the green and red color channels of the image sensor. Thus the leaks to other color channels are min-

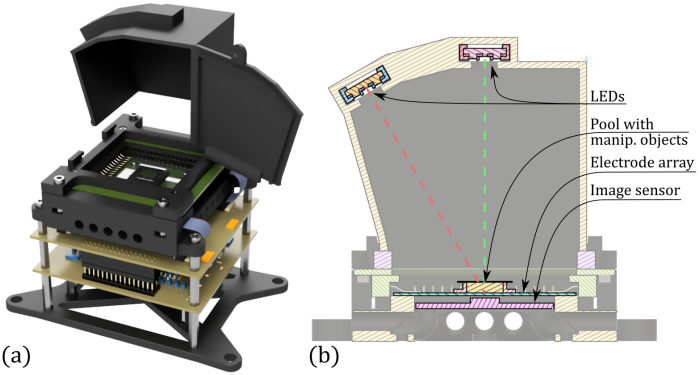


FIGURE 5.1: Hardware setup without the NVidia Jetson AGX computer and a power supply: (a) render of the hardware setup with the top part detached, (b) side cross-sectional view of the upper part. The hardware setup fits within a box 95 mm × 91 mm × 147 mm.

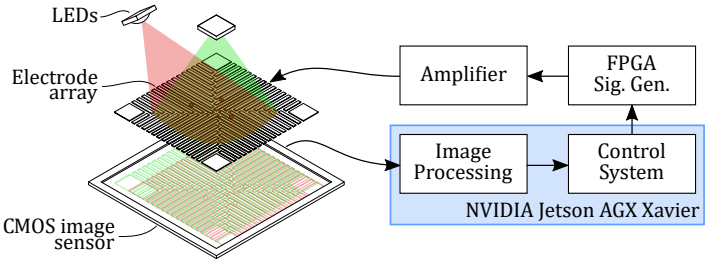


FIGURE 5.2: Principal schematics of the hardware setup.

imal and the green and red color channels of the recorded image contains mainly interference patterns from the green and red LEDs, respectively. To reduce heating of the manipulated objects by the image sensor in operation, we air cool the image sensor by a fan placed right below it. The electrode array is connected by spring-loaded connectors to a driving circuitry which consist of an open-design FPGA-based generator[1] outputting 56 square waves of frequency 300 kHz with real-time adjustable phase shifts (with resolution of 1°) and an amplifier board amplifying the generated signals from 3.3 V to 16 V which are then connected to the electrode array. The amplifier board needs external 16 V power supply. Both the camera and the generator are connected to NVIDIA Jetson AGX Xavier: a computer suitable for demanding image processing applications.

Since no objective lenses (commonly used in conventional microscopes) are used in the hardware setup, the usual limit on the size of the *Field-of-View (FOV)* is eliminated an the size of the FOV is given by the size of the image sensor. In our setup, the FOV can be as large as 6.29 mm × 4.7 mm.

A principal sketch of the whole hardware setup is displayed in Fig. 5.2. The camera is connected via the MIPI CSI-2 camera interface to the Jetson Xavier computer. The manipulated objects are tracked in the captured images with the frame rate of 30 frames per second. The positions of the manipulated objects are sent to the control system which computes the phase shifts of the voltage signal applied to the electrodes in such a way that the manipulated objects move towards some desired positions. The phase shifts are then sent to the signal generator via serial interface with the baud rate of 115200 baudrate.

1 https://github.com/aa4cc/fpga-generator

Image reconstruction and object tracking

Images captured by the camera module serve two purposes: displaying the objects and measuring their position. To properly describe the image processing algorithms used for either of the two purposes, it is first necessary to introduce some basic concepts.

The hardware setup utilises incoherent lensfree in-line digital holography [6]. Due to the partial-coherent light source, the captured image contains interference patterns from the manipulated objects in the pool (see Fig. 5.3(a)). These patterns can be numerically reconstructed to images of the objects as they would appear in a conventional microscope (see Fig. 5.3(b, d) for comparison). To reconstruct the image, we back-propagate the captured light wave by computing the Rayleigh-Sommerfeld diffraction integral [7]. This computation is numerically done as follows

$$I_z(x_{\text{im}}, y_{\text{im}}) = \mathcal{F}^{-1}\left\{H_z(f_x, f_y)\mathcal{F}\left\{I(x_{\text{im}}, y_{\text{im}})\right\}\right\}, \tag{5.1}$$

where $(x_{\text{im}}, y_{\text{im}})$ are the image coordinates, (f_x, f_y) are the spatial frequencies, I is the captured image, I_z is the image back-propagated to a distance z, $\mathcal{F}$ and $\mathcal{F}^{-1}$ are Fourier and inverse Fourier transformations, respectively, and

$$H_z(f_x,f_y)=\begin{cases}\exp\left(i2\pi z\frac{n}{\lambda}\sqrt{1-\left(\frac{\lambda f_x}{n}\right)^2-\left(\frac{\lambda f_y}{n}\right)^2}\right), & \sqrt{f_x^2+f_y^2}\leq\frac{n}{\lambda},\\ 0, & \text{otherwise},\end{cases} \tag{5.2}$$

is Fourier transform of the Rayleigh-Sommerfeld propagator, where λ is the wave length of the incident light wave and n is the refractive index[2].

For the purpose of displaying, the captured images should be processed in such a way, so that the observed objects are easily discernible from their surroundings and so that the resulting image is, if possible, not affected by any artifacts. However, since the image sensor captures only the amplitude of the incident light wave, the phase has to be either estimated [6, 8, 9] or considered constant throughout the image, in which case the presence of twin-image artifacts deteriorates the quality of the reconstructed images. A useful property of the Rayleigh-Sommerfeld diffraction is that it is easily reversible. After calculating the back-propagation of the captured image, the result can then be easily propagated back to its original axial plane

2 Even though the light wave travels through several media between the objects and the image sensor, we consider the refractive index constant to simplify the computation.

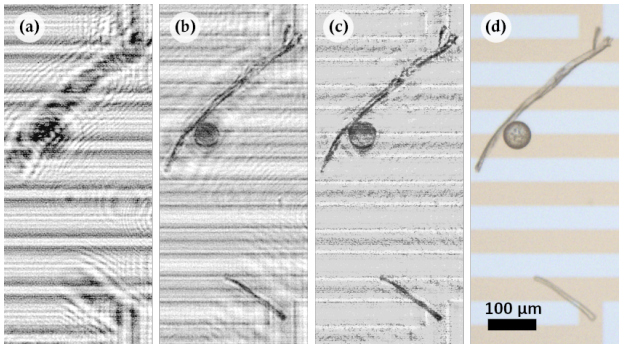


FIGURE 5.3: Image reconstruction of the green color (straight illumination) channel (a) and (b) display cutouts of the captured and back-propagated images, respectively. (c) then shows the same area of the image with phase reconstructed. Finally, (d) shows the area of the electrode array captured by a microscope (Olympus BX-FM, 10× magnification lens).

by calculating (5.1) of the result with a complex conjugate of the used propagator. To reduce the impact of twin-images and, therefore, provide clearer reconstructions for displaying we utilize an iterative phase reconstruction method [10] which allows us to iteratively estimate the phase of the holograms. The method in question approaches phase reconstruction as an inverse problem regularized by sparsity and positivity constraints. The optimization strategy is then facilitated by an accelerated proximal gradient method [11]. Each iteration of used method starts by back-propagating the captured image to the axial plane in which the reconstruction is desired, where the constraints are imposed, and then propagates the constrained reconstruction back to the plane of the hologram. Since the reconstruction algorithm computes *Fast Fourier Transform*—which are most efficient for image resolutions that are powers of two—we run the phase reconstruction algorithm only on a 2048×2048 pixel cutout equivalent to FOV of $3.17\,\text{mm} \times 3.17\,\text{mm}$ of the captured image which is further down scaled to 1024×1024 pixels. The down scaling is used to reduce the computational time required to reconstruct the phase of one captured image. The algorithm provides satisfactory results even for a low number of iterations. The results after only 3 iterations of the used algorithms are comparable to the images captured by a conventional microscope (see Fig. 5.3(c-d) for comparison). For one down scaled color channel, 3 iterations of the phase reconstruction algorithm take $\sim$30 ms. The reconstruction algorithm is implemented in

CUDA. The size of FOV can be increased if longer computational times are not an issue.

To measure the position of the objects in 3D, we utilize a twin-beam method [5]. For the control purpose, we are specifically interested in measuring the positions in an electrode array coordinate frame. The position is measured in two steps: first, the position in the electrode array plane is measured; then, the position along the axis perpendicular to the electrode array is measured.

The position of an object in the electrode array plane is related by homography to the pixel position of the object in the image from the straight illumination. The homography is estimated by relating known positions of the corners of the electrodes in both pixel and electrode array coordinates.

The position of an object along the axis perpendicular to the electrode array (*i.e.* the levitation height) is computed based on the mutual distance of positions of the object in both color channels (for details, see [5]). The levitation height is then given by

$$h = d\frac{1}{\tan\theta}, \tag{5.3}$$

where θ is the angle of the oblique and straight illumination sources (in our case, 30°).

For both steps, we first need to find the pixel positions of the object in both color channels. Raw captured images are not suitable for tracking the objects as the objects' holograms interfere with the electrodes' holograms and thus they change depending on where they are located with respect to the electrode array. Therefore, before finding the pixel positions, the objects in both color channels are reconstructed by using a simple back-propagation. The manipulated objects in the reconstructed images can be found and tracked by many standard image processing techniques. We tested two: (1) finding a regional minimum closest to the previous object's position in a blurred image and (2) tracking with Siamese networks [12]. The major advantage of the former is that it is highly parallelizable and thus the number of tracked objects is virtually unlimited. The disadvantage is that is applicable only to objects that are regular in shape and appear darker in the reconstructed image. The Siamese networks works well for arbitrary objects but the cost is that one can track a limited number of objects due to the large computational burden. We used the regional-minimum algorithm in the experiment section of this paper.

Object manipulation

The objects are manipulated by phase-shift feedback control, an approach introduced by Zemánek et al. [13]. The phase-shifts of sinusoidal voltage signals applied on the electrodes are computed through numerical optimization such that some desired dielectrophoretic forces are exerted on the manipulated objects. To optimize the voltage signals one needs a model of the dielectrophoretic force. The model relates the voltage signals and object positions to the dielectrophoretic force acting on the objects. This is where knowing the levitation height of the objects is useful; it not only adds one degree of freedom when controlling the object positions but it also allows us to more precisely optimize the voltage signals because if the levitation height was unknown it would have to be somehow estimated (as, for instance, in Zemánek et al. [13]). The desired dielectrophoretic forces are determined by a positional controller so that the manipulated objects move towards some desired positions.

In addition to Zemánek et al. [13], we also add an artificial repulsive force to avoid collisions of the particles and to avoid particles leaving the manipulation area. The net repulsive force for one particle is given by a sum of repulsive forces from all other particles and from the boundaries. The repulsive force between particles decays quadratically with the distance between the particles.

In contrast to [13], the signal generator used in the hardware setup allows finer resolution in the phase shifts, and thus we can consider the phase shifts continuous and use the Levenberg-Marquardt method to optimize the phase shifts instead of the simulated annealing used in [13], which is suitable to discrete optimization values. The control system runs at 30 Hz and is capable of manipulating up to 10 objects.

5.3 RESULTS AND DISCUSSION

To demonstrate the capabilities of the proposed device, we manipulated polystyrene particles of diameter 50 µm immersed in deionized water in the pool (see Fig. 5.1). We used the same four-sector electrode array as in [13], only with more electrodes. Each sector contained 14 parallel electrodes which were 50 µm wide and 50 µm apart from each other. In total, the electrode array has 56 electrodes. We chose this particular electrode array layout because, as the authors in [13] argue, it allows object manipulation in arbitrary direction in the area where the sectors meet; nevertheless,

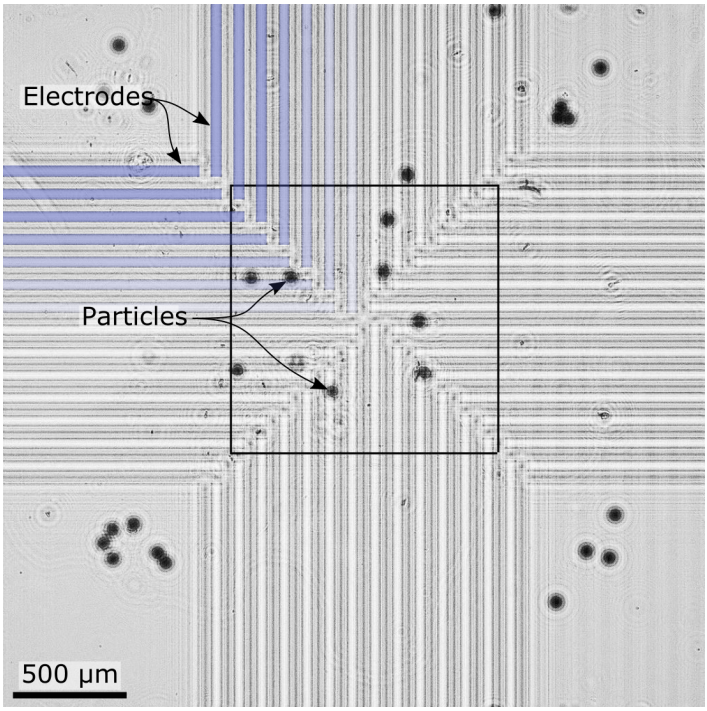


FIGURE 5.4: Snapshot of the manipulation area. Few electrodes are highlighted by blue color. The black square bounds an area where a net force (dielectrophoretic + sedimentation force) aiming in any direction can be exerted on the particles.

other layouts can be used in the proposed device as well. A reconstructed captured image with highlighted electrodes and manipulated objects is shown in Fig.5.4.

Even though we demonstrate the manipulation capabilities only with polystyrene particles, other researchers used the same manipulation principle (DEP) to manipulate living cells [14, 15].

Multi-object independent manipulation

An example of manipulating particles from some random initial positions to some predefined constant positions is displayed in Fig. 5.5. It took 20 s to manipulate the particles from the initial position to the final position. Even

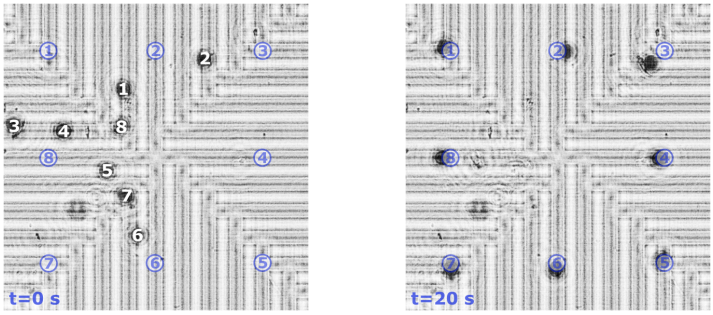
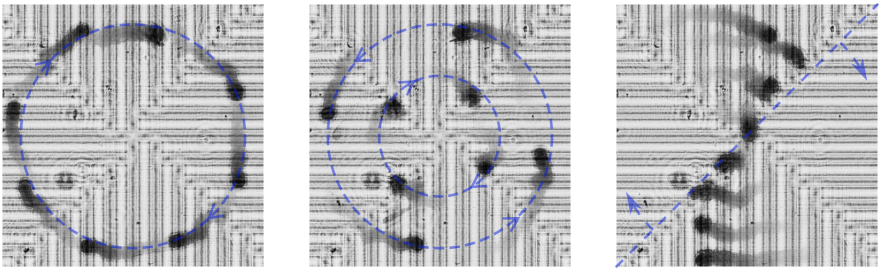


FIGURE 5.5: Manipulating the particles from some random initial positions to predefined required constant positions (marked by blue circles).



(a) Rotating circle (b) Counter-rotating circles (c) Rotating line

FIGURE 5.6: Long-exposure snapshots of experiments where the particles were steered along three trajectories.

though some particles had colliding straight paths from the initial position to the final position, the control system could handle this situation.

Figure 5.6 shows experiments where the particles were manipulated so that they tracked some moving reference positions: in Fig. 5.6a, they all followed one large circle; in Fig. 5.6b, half of the particles followed a large clockwise rotating circle and the other half followed a smaller counter-clockwise rotating circle; and in Fig. 5.6c, the particles were positioned along a line which rotated in a clockwise direction. The maximum velocity of the moving reference position was $50\,\mu m\,s^{-1}$ and the control system was able to track this reference. To visualize the motion of the particles, all snapshots in Fig. 5.6 were modified in post processing so that they look like long-exposure snapshots.

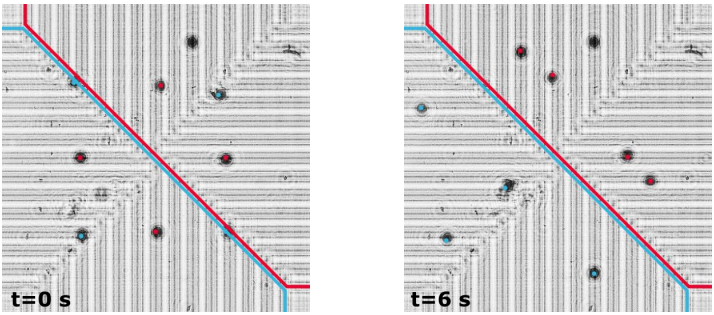


FIGURE 5.7: Separation of objects of two categories. Objects of the same type were labeled as red or blue. The control system was able to move the red-labeled and blue-labeled objects to the top-right and bottom-left corners, respectively, in 6 s.

The capability of independent objects manipulation allows separation of the objects. This is demonstrated by an experiment shown in Fig 5.7. We artificially divide particles into two categories: *blue* and *red*. The particles were initially spread over the manipulation area. The task was to move the blue particles to the lower-left half-plane and the red particles to the upper-right half-plane. This was accomplished in 6 s. The particle crossing the center of the electrode array moved with the average speed of 130 µm s^{-1}.

Single object height control

The capability of the platform to control the levitation height of the objects is presented by the experiment in Fig. 5.8. Single particle was steered along a circular trajectory. Initially, the levitation height was set to 150 µm above the electrode array. Then, the reference levitation height was changed to 200 µm. You can see, that the particle followed the reference trajectory in all three dimensions for both reference levitation heights.

CONCLUSION

In this paper, a novel contactless (micro)manipulation device is demonstrated that is capable of controlled motion of micrometer-size objects in 3D without the need for a microscope. The device utilizes the phenomenon of dielectrophoresis as the actuation mechanism, and inline digital holography with partially coherent light sources as the mechanism for displaying

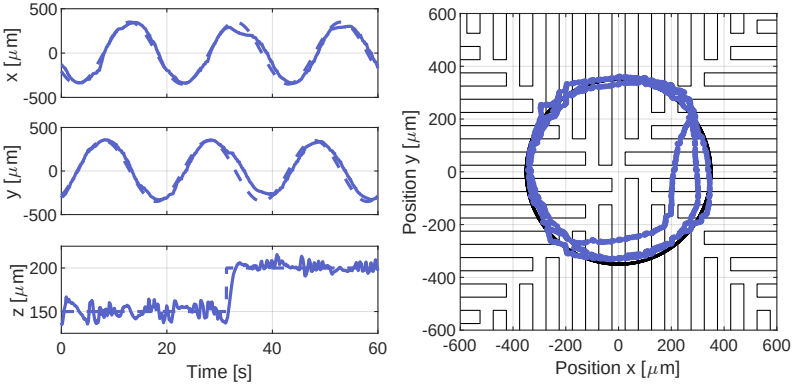


FIGURE 5.8: Controlling the height experiment. The measured position is represented by blue lines and the reference position is visualized by dashed black lines.

the manipulated objects. The twin-beam method is used to measure the position of the manipulated objects in 3D. Thanks to digital holography, the device has a relatively large field of view and needs neither lenses nor lasers, making it more portable than some lab-tied solutions. The capabilities of the proposed manipulation platform were demonstrated through parallel multi-object manipulation, object separation, and height control experiments.

REFERENCES

1. Gurtner, M., Koropecký, V.-A., Zemánek, J. & Hurák, Z. Compact Dielectrophoretic Feedback Manipulation Platform. *In prepraration.* (2022).
2. Fassi, I. & Shipley, D. Micro-manufacturing technologies and their applications. *Springer Tracts in Mechanical Engineering* **10**, 978 (2017).
3. Pethig, R. R. *Dielectrophoresis: Theory, methodology and biological applications* (John Wiley & Sons, 2017).
4. Jones, T. B. Basic theory of dielectrophoresis and electrorotation. *Engineering in Medicine and Biology Magazine, IEEE* **22**, 33 (2003).
5. Gurtner, M. & Zemánek, J. Twin-beam real-time position estimation of micro-objects in 3D. *Measurement Science and Technology* **27**, 127003 (2016).

6. Mudanyali, O., Tseng, D., Oh, C., Isikman, S. O., Sencan, I., Bishara, W., Oztoprak, C., Seo, S., Khademhosseini, B. & Ozcan, A. Compact, light-weight and cost-effective microscope based on lensless incoherent holography for telemedicine applications. *Lab on a Chip* **10**, 1417 (2010).

7. Goodman, J. W. *Introduction to Fourier optics* (Roberts and Company Publishers, Greenwood Village, 2005).

8. Fienup, J. R. Phase retrieval algorithms: a comparison. *Applied optics* **21**, 2758 (1982).

9. Koren, G., Polack, F. & Joyeux, D. Iterative algorithms for twin-image elimination in in-line holography using finite-support constraints. *JOSA A* **10**, 423 (1993).

10. Momey, F., Denis, L., Olivier, T. & Fournier, C. From Fienup's phase retrieval techniques to regularized inversion for in-line holography: tutorial. *Journal of the Optical Society of America A* **36**, D62 (2019).

11. Beck, A. & Teboulle, M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences* **2**, 183 (2009).

12. Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A. & Torr, P. H. *Fully-convolutional siamese networks for object tracking* in *European conference on computer vision* (2016), 850.

13. Zemánek, J., Michálek, T. & Hurák, Z. Phase-shift feedback control for dielectrophoretic micromanipulation. *Lab on a Chip* **18**, 1793 (2018).

14. Huang, K.-W., Su, T.-W., Ozcan, A. & Chiou, P.-Y. Optoelectronic tweezers integrated with lensfree holographic microscopy for wide-field interactive cell and particle manipulation on a chip. *Lab on a chip* **13**, 2278 (2013).

15. Voldman, J. Electrical forces for microscale cell manipulation. *Annu. Rev. Biomed. Eng.* **8**, 425 (2006).

PUBLICATIONS RELATED TO THE THESIS

Articles in peer-reviewed journals:

1. Gurtner, M. & Zemánek, J. Twin-beam real-time position estimation of micro-objects in 3D. *Measurement Science and Technology* **27**, 127003 (2016).
2. Gurtner, M., Hengster-Movric, K. & Hurák, Z. Green's function-based control-oriented modeling of electric field for dielectrophoresis. *Journal of Applied Physics* **122**, 054903 (2017).
3. Gurtner, M., Zemánek, J. & Hurák, Z. ADMM-based distributed control for distributed manipulation by shaping physical force fields. *International Journal of Robotics Research*. As of December 2022, the paper is accepted but has not been published yet. (2022).

Conference contributions:

4. Matouš, J., Kollarčík, A., Gurtner, M., Michálek, T. & Hurák, Z. *Optimization-based Feedback Manipulation Through an Array of Ultrasonic Transducers* in. **52**. 8th IFAC Symposium on Mechatronic Systems 2019 (2019), 483.

Articles in preparation:

1. Gurtner, M., Koropecký, V.-A., Zemánek, J. & Hurák, Z. Compact Dielectrophoretic Feedback Manipulation Platform. *In prepraration.* (2022).

PUBLICATIONS NOT RELATED TO THE THESIS

Articles in peer-reviewed journals:

1. Saikin, D. A., Baca, T., Gurtner, M. & Saska, M. Wildfire Fighting by Unmanned Aerial System Exploiting Its Time-Varying Mass. *IEEE Robotics and Automation Letters* **5**, 2674 (2020).

Conference contributions:

2. Gurtner, M. & Zemánek, J. *Ball in double hoop: demonstration model for numerical optimal control* in. **50**. 20th IFAC World Congress (2017), 2379.