Czech Technical University in Prague
Faculty of Electrical Engineering

Diploma Thesis

Real-time Optimization-based Control and Estimation for Dielectrophoretic Micromanipulation

2016

Martin Gurtner

CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF ELECTRICAL ENGINEERING
DEPARTMENT OF CONTROL ENGINEERING

Real-time optimization-based control and estimation for dielectrophoretic micromanipulation

DIPLOMA THESIS

MARTIN GURTNER

Supervisor: Ing. Zdeněk Hurák, Ph.D.

Prague, January 2016

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Control Engineering

DIPLOMA THESIS ASSIGNMENT

Student: **Bc. Martin Gurtner**

Study programme: Cybernetics and Robotics
Specialisation: Systems and Control

Title of Diploma Thesis: **Real-time optimization-based control and estimation for dielectrophoretic micromanipulation**

Guidelines:

1. Design and build a simple instrumentation for a real-time sensing/estimation of the position (including the levitation height) of a microparticle on/above a planar microelectrode array.
2. Develop a computational mathematical model of the dielectrophoretic force field in the form of a Green's function.
3. Design and implement an optimization algorithm that will compute the voltages that need to be set on the microelectrodes (up to several dozens of them) so that prescribed dielectrophoretic forces are developed at one or several places over the microelectrode array. This computation will have to run periodically every few miliseconds.

Bibliography/Sources:

[1] K. E. Gustafson and D. K. M. Rao, Numerical Range. Springer New York, 1997.
[2] C. Johnson, Numerical Determination of the Field of Values of a General Complex Matrix, SIAM J. Numer. Anal., vol. 15, no. 3, pp. 595-602, June 1978.
[3] M. P. Hughes, Nanoelectromechanics in Engineering and Biology. CRC Press, 2003.
[4] T. B. Jones, Electromechanics of Particles. Cambridge University Press, 1995.
[5] H. Morgan and N. G. Green, AC electrokinetics: colloids and nanoparticles. Philadelphia, PA: Research Studies Press, 2003.
[6] D. G. Duffy, Green´s Functions with Applications, Second Edition, 2 edition. Boca Raton: Chapman and Hall/CRC, 2015.
[7] I. Stakgold, Green´s Functions and Boundary Value Problems, 3 edition. Hoboken, NJ: Wiley, 2011.

Diploma Thesis Supervisor: Ing. Zdeněk Hurák, Ph.D.

Valid until the summer semester 2016/2017

L.S.

prof. Ing. Michael Šebek, DrSc.
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, October 15, 2015

Abstract

This diploma thesis focuses on several problems related to micromanipulation based on dielectrophoresis (DEP). First, a new real-time position estimation method for microparticles was designed, implemented and tested in experiments. The method is based on twin-beams illumination and lensless digital holography. As such, it only needs a very simple, cheap and compact hardware. Second, in order to pursue a model-based control design strategy for positioning the microparticles, a simple yet accurate mathematical model relating the voltages applied to the microelectrodes with the generated DEP force is needed. In this thesis, such a model is derived within the framework of Green's functions. Third, the computational problem of determining the voltages to be applied to the microelectrodes in order to establish the required DEP force is analyzed and several algorithms solving the problem are introduced. These invoke some linear-algebraic concepts from the numerical range (or field of values) of a complex matrix and semidefinite programing.

Abstrakt

Tato diplomová práce se zabývá několika tématy souvisejícími s mikromanipulací pomocí dielektroforézy (DEP). V práci je nejprve popsána nově vyvinutá metoda pro odhadování polohy mikročástic v reálném čase. Metoda byla i implementována a otestována v laboratorních experimentech. Je založena na dvou-paprskovém osvětlení a bezčočkové digitální holografii a jako taková potřebuje ke své funkci jen velmi jednoduchý, levný a kompaktní hardware. Dále, jelikož pro účely na modelu založeného řízení polohy mikročástic je nutné mít k dispozici jednoduchý ale dostatečně přesný model svazující napětí na mikroelektrodách s vygenerovanou DEP sílou, v předložené práci je takový model odvozen pomocí aparátu Greenových funkcí. V poslední části se práce zabývá problémem určení napětí na mikroelektrodách tak, aby vytvořená DEP síla splňovala požadavky vyšších úrovní řídicího systému. V práci je zdokumentována analýza tohoto problému a několik nových metod jeho řešení. Tyto využívají výsledků pro tzv. numerický obor matice a semidefinitní programování.

Declaration

I declare that I wrote the presented thesis on my own and that I cited all the used information sources in compliance with the Methodical instructions about the ethical principles for writing an academic thesis.

Prague, January 2016
Martin Gurtner

Acknowledgements

I am immensely grateful to my supervisor Zdeněk Hurák, who gave me an opportunity to join his research group AA4CC as an intern a few years ago. I seized the opportunity and you are about to see some of the results. Without his supervision and constant help this thesis would not have been possible.

I also would like to thank to Kristian Hengster-Movric, a gentleman who do not hesitate to share his profound mathematical knowledge and who suggested exploring of Green's functions for developing the control oriented model of dielectrophoretic force.

My sincere thanks also goes to prof. Didier Henrion for his priceless help in solving the constarained inverse quadratic problem. We had only one opportunity to discuss the problem, but it was a very fruitful discussion.

Of course, I simply can't omit Jirka Zemánek in this list. I thank him that he does the utmost to keep the rest of the group well informed about all new interesting hacks, toys, gadgets and videos. I thank him for his infectious excitement to do crazy projects. Last but not least, I thank him for providing me some invaluable advice regarding physics, electronics and making things. It makes one wonder, how can anyone be so skilled in so many disciplines, all at the same time?

I also take this opportunity to express gratitude to all members of AA4CC who made my stay at the department very enjoyable.

Let me add my warmest thanks to my parents and my girlfriend who supported me throughout writing this thesis and my life in general. Parents, Saša, believe it or not, the thesis is finished and you will not hear the lamentation anymore.

Contents

Notation

Notation	*Description*
$\mathbb{R}$	The set of real numbers.
$\mathbb{C}$	The set of complex numbers.
$\mathbb{F}^{n\times m}$	A $n \times m$ matrix with entries from $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$.
$\mathscr{H}(n)$	The set of all Hermitian $n \times n$ matrices.
$\mathbf{A}^\intercal$	Transposition of a matrix $\mathbf{A}$.
$\mathbf{A}^*$	Hermitian (conjugate) transposition of a matrix $\mathbf{A}$.
$\mathrm{tr}(\mathbf{A})$	Trace of a matrix $\mathbf{A}$.
$\mathbf{H}(\mathbf{A})$	The hermitian part of a matrix $\mathbf{A}$, that is $\frac{1}{2}(\mathbf{A}+\mathbf{A}^*)$.
$\mathbf{S}(\mathbf{A})$	Skew-Hermitian part of a matrix $\mathbf{A}$, that is $\frac{1}{2}(\mathbf{A}-\mathbf{A}^*)$.
$\lambda_1(\mathbf{A}) \geq \cdots \geq \lambda_n(\mathbf{A})$	Eigenvalues of a matrix $\mathbf{A} \in \mathscr{H}(n)$.
$\mathbf{E}_k(\mathbf{A})$	The eigenspace associated with the eigenvalue $\lambda_k(\mathbf{A})$.
$\mathrm{Re}(z)$	Real part of a complex number z.
$\mathrm{Im}(z)$	Imaginary part of a complex number z.
$\mathrm{Arg}(z)$	Principal argument of a complex number z.
$\|\mathbf{x}\|_2$	Euclidean norm of a vector $\mathbf{x} \in \mathbb{R}^n$.
$\|\mathbf{x}\|_\infty$	Infinity norm of a vector $\mathbf{x} \in \mathbb{R}^n$.
∂M	Boundary of a compact set $M \subset \mathbb{R}^m$.
∂B^{m-1}	The unit hypersphere in $\mathbb{R}^m$.
$\mathrm{co}(M)$	Convex hull of a compact set $M \subset \mathbb{R}^m$.
$S(\boldsymbol{\eta}, b)$	A hyperplane defined by $S(\boldsymbol{\eta}, b) = \{\mathbf{x} \in \mathbb{R}^m \mid \boldsymbol{\eta}^\intercal \mathbf{x} = b\}$.
$s(\boldsymbol{\eta})$	The support function of a compact set M defined by $s(\boldsymbol{\eta}) = \max_{\mathbf{x}\in M} \boldsymbol{\eta}^\intercal \mathbf{x}$.

Acronyms

CMOS complementary metal–oxide–semiconductor.

DEP dielectrophoresis.

DHM digital holographic microscopy.

DOF depth of dield.

FEM finite element method.

FFT Fast Fourier transform.

JNR joint numerical range.

LED light-emitting diode.

PDE partial differential equation.

PDMS polydimethylsiloxane.

PSF point spread function.

rJNR real joint numerical range.

RMP rank minimization problem.

SDP semidefinite programming.

Chapter 1

Introduction

This thesis deals with three topics related to micromanipulation by *dielectrophoresis (DEP)*: position estimation, mathematical modeling, and control. DEP is a physical phenomenon enabling us to develop a (*dielectrophoretic*) force on polarizable particles by shaping the surrounding electric field. The electric field is usually created and shaped by application of varying potentials on electrodes nearby the microparticles. As DEP is able to manipulate microparticles without any contact, it is especially well suited for contact-less manipulation in biology or medicine. List of several applications can be, for instance, found in [1, 2], and one is also sketched in Fig. 1.1.

Topics of this thesis are motivated by real problems. The work presented here was undertaken in the research group *Advanced Algorithms for Control and Communications (AA4CC)*, Department of Control Engineering, Faculty of Electrical Engineering at Czech Technical University in Prague. Even though the author's colleagues in AA4CC already have a fully functional prototype using DEP for simultaneous micromanipulation of several microparticles, there is still a room for improvements. Specifically, the currently used position estimation method for microparticles is able to measure the position only in two dimensions, while the microparticles can move in three dimensions. The currently used model of DEP force cannot be evaluated in real-time and thus the data have to be precalculated and stored in a huge look-up table. Finally, the currently used control algorithm involves a non-convex optimization problem which is solved by a general heuristic without exploitation of the structure of the problem. We will address all these problems in this thesis.

In the following sections, we will discuss in detail the motivation for individual topics of this thesis.

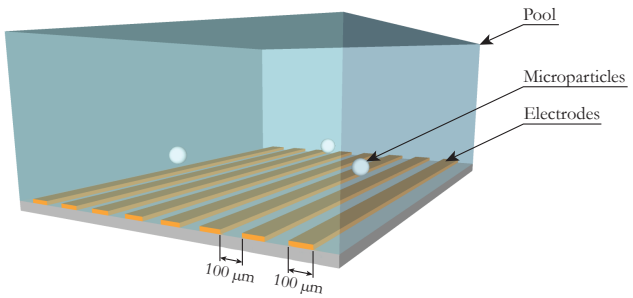


Figure 1.1: *A sketch of micromanipulation by DEP. The microparticles are dispersed in a pool with deionized water. The electric field around the microparticles is shaped by application of varying potentials on the parallel electrodes bellow the microparticles.*

1.1 Position estimation

The currently used position estimation method in the group AA4CC is based on processing of images from a microscope. The images capture the microparticles and the electrode array from the top and as such they allow us to estimate the position only in 2D; they do not allow us to estimate the levitation height of the microparticles. But the levitation height is crucial for the control, because if the levitation height is not known, it is not possible to precisely determine the DEP force acting upon the microparticles. Furthermore, if the microparticles are too low, they tend to stick to the bottom. Therefore, there is a need for a new method estimating the position in 3D.

We have basically only two requirements on the new method. The method has to provide accurate enough estimates and it has to be fast enough for real-time use. The manipulation area for the microparticles is in our case typically a cuboid with base 1500×1500 μm (given by the size of the electrode array) and height 200 μm (approximately the maximum levitation height in our hardware setup for 50 μm microparticles, which we typically use). From the manipulation area and the size of the microparticles, reasonable requirements on the sought method can be deduced. The method has to estimate the position with accuracy at least 10 μm in all three dimensions. Worse accuracy would not be of much use for the control system. Regarding the speed, to achieve a reasonable control period, the process of estimation should not take more than several tens of microseconds.

This thesis deals mostly with methods based on *digital holography* in the lensless configuration: the image sensor is used without any optical lenses. Such hardware configuration has several advantages. Due to the simplicity of the hardware setup, this solution is very cheap, simple and compact. Furthermore, compared for instance to *Confocal microscopy*, which is also used for 3D position estimation, methods based on digital holography allows us to estimate the position of microparticles moving in a rather large space. Besides, digital holography has a great potential for real-time estimation because it encodes 3D position in 2D image—that means less data to process.

Despite the immense progress in position estimation of microparticles in recent years (enabled mostly by rapid development of image sensors and processing power), majority of the methods are computationally demanding and thus not applicable for real-time use. That is not a problem if one needs to only analyze trajectories of microparticles. Nonetheless, it is a crucial issue when the estimated position is used—as it is in our case—for control. Therefore, in this thesis we will review the currently available off-line methods, pick the most appropriate one and modify it for real-time use.

1.2 Mathematical model of DEP force

In order to control the position of the microparticles, one needs a mathematical model relating the set potentials on the electrodes with the developed DEP forces acting upon the microparticles. Only then the control system can compute potentials that, when applied, generate such forces, that the microparticles move towards the desired positions. The exact model—for a dipole approximation of the microparticle—is known from the first principles, but unfortunately, it involves a boundary value problem with *Laplace equation*. Laplace equation is a *partial differential equation (PDE)* and those are usually not easy to solve analytically. Our case

is no exception and thus also the exact analytical model relating the set potentials with the DEP force remains unknown. Currently, the boundary value problem is solved numerically and the data needed for calculation of DEP force are stored in a huge look-up table. Every time one needs to determine the DEP force acting upon a microparticle, the look-up table has to be searched according to the position of the microparticle. In this thesis, we will approximate the boundary value problem in the exact model to a mathematically more tractable form that allows us to find the analytical solution by the framework of *Green's functions*.

1.3 Control algorithm

The currently used control algorithm takes the difference of the reference and desired position and based on this difference it calculates the force that has to be developed upon the microparticle in order to move it towards the desired position. Then, based on the model of the DEP force, the potentials that need to be set on the electrodes are determined. But the model relates the set potentials to the generated DEP force, not the other way around. Therefore, one actually needs an inverse of the model—relation from DEP force to the set potentials—but the inverse is not known. Thus, the problem of finding the potentials for a given DEP force is formulated as an optimization problem, which, unfortunately, turns out to be non-convex. The optimization problem is being solved by a general heuristic for non-convex problems; it is being solved by *simulated annealing* which does not utilize the structure of the problem in any way and it also does not provide any guaranties about the solution it finds. In this thesis, we will try to take a more insightful approach. We will analyze the inverse problem and propose several algorithms which make use of the structure of the problem and provide some guaranties about the solution.

1.4 Structure of the thesis

According to the topics, the thesis is divided into three parts: position estimation, modeling and control. The first part begins with Chapter 2, which is devoted to a brief introduction to the principles of digital holography. In Chapter 3, we review methods for position estimation of microparticles in 3D. In Chapter 4, the last one in the first part, we propose and evaluate our novel method for real-time position estimation of microparticles in 3D. The second part begins with Chapter 5 introducing the model of the dynamics of microparticles in fluid and the model of the DEP force. In Chapter 6, we discuss modifications of the exact model of DEP force and solution of the modified model by Green's functions. The last part is introduced by Chapter 7 designated to a description of the used control algorithm and to an analysis of the inverse problem. The inverse problem is then separately treated for 2D case in Chapter 8 and for higher dimensional case in Chapter 9. Of course, the thesis ends with a chapter summarizing the achieved results, Chapter 10.

Part I

Position estimation

Chapter 2

Digital holography

A nice and comprehensive introduction to digital holography is given in [3]. Since there is no need to duplicate their work, in this chapter, digital holography is introduced only briefly for the readers who have never heard about it. Furthermore, we will emphasize certain aspects of digital holography that will be important in the remainder of this thesis.

In its original meaning, holography is a process of recording light fields for the purpose of visual reconstruction of 3D objects. Basically, an object is illuminated by a coherent light source and the reflected light field is recorded by a film, which turns into a so called hologram. Subsequent illumination of the film (hologram) reconstructs the captured light field so that the previously illuminated object appears behind the film. It is not very usual to put links to YouTube videos in written texts, but we make an exception because that is what *Introduction to Holography* shot by *Encyclopaedia Britannica* in 1972 absolutely deserves. So here is the link: `http://y2u.be/tjWznlGst9M`. The film explains all the basic concepts of holography. The process was originally purely mechanical. Digital holography simplifies the process of recording and reconstructing the light fields, because the recording is done by an image sensor and the reconstruction is carried out computationally by a computer.

Let us have a look on the underlying physics. When the light falls on an object, several phenomenons can occur. If the object is opaque, the light can be only *reflected*, *absorbed*, or *diffracted*. If the illuminated object is small—as the microparticles in our case are—the shadow behind it does not look as we would expect based on our experience from the macro-world. Due to the diffraction of light, the shadow—or as the right terminology dictates, diffraction pattern—is not sharply bounded, it consists of bright and dark regions. If the object is translucent, the light also passes through the object. Consequently, the shadow of the object is given by superposition of the diffraction pattern and the scattered light that passed through the object. The composed shadow is called *interference pattern* and it encodes the position and the shape of the object it is formed by.

It is necessarily to note, that even though an interference pattern from an object truly encodes full information about the position and the shape of the object, the image sensor captures only part of the information. The interference pattern is nothing but a monochromatic electromagnetic wave. Provided the wave vector is known, a monochromatic electromagnetic wave is characterized at every point by its amplitude and phase. But the only measurable quantity by the image sensor is intensity. Therefore, part of the information is missing and that makes the position estimation from interference patterns more difficult.

2.1 Hardware arrangements and illumination sources

Because digital holography plays an important role in our method introduced in Chapter 4, we will spend some time commenting one particular hardware arrangement and possible illumination sources.

There are more hardware configurations used in digital holography, but in this thesis, we will present and use only the simplest one: *in-line digital holography*. This hardware setup in the lensless variant is depicted in Fig. 2.1. It consists only of a source of light illuminating the objects and an image sensor capturing the light going through and around the objects. We chose this particular setup for its simplicity; it does not need any lenses, mirrors or some other expensive optical equipment.

In digital holography, objects are illuminated by a source of coherent or partially coherent light. By coherence, we mean both temporal and spatial coherence. The coherence of light source determines the quality of the produced interference patterns (also called fringe visibility); the more coherent the light is the more visible the interference patterns are. The typical light source is a laser because it produces nearly perfectly coherent light. Another widely used light source is a *light-emitting diode (LED)*. LEDs usually produce light in a very narrow band of frequencies hence the light is partially temporally coherent. The spatial coherence can be to some extent attained by filtering the light through a small aperture (tens or hundreds of micrometers in diameter). Smaller diameter results in more spatially coherent light. An advantage of partially coherent illumination is that it removes speckle noise [4]—interference patterns caused by very small objects. A disadvantage is that very small apertures are very impractical to use because usually only a very small part of the emitted light passes through the aperture. Therefore, smaller apertures provide more coherent light, but they require high power LEDs. Nevertheless, as Mudanyali et al. [5] has shown, it is possible to use even large apertures without significant loss of spatial coherence.

LEDs as partially coherent light sources have one crucial disadvantage that has to be taken into account. As the emitted light is filtered by a small aperture, the aperture, according to Huygens's principle, becomes approximately a source of spherical waves. Spherical waves have one undesired property; if an object is illuminated by spherical waves then its shadow on the image sensor is magnified. The magnification depends on the distance of the image sensor to the object and on the distance of the light source (the aperture) to the object. The magnification is reduced when one puts the image sensor closer to the object and the light source further away from the object.

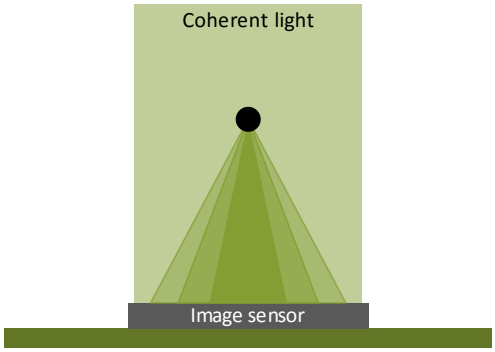


Figure 2.1: *An illustration of lensless in-line digital holography.*

2.2 Position estimation based on interference patterns

If an interference pattern encodes the position of the object it is formed by, is it possible to use it for position estimation? Is it possible to decode the position from the interference pattern? It is and we will discuss several approaches how to do it in the following chapter. But here, we would like to show you how the interference patterns look like in our case. Fig. 2.2 shows interference patterns from a 50 µm polystyrene spherical microparticle placed at different axial displacements (levitation height) from the image sensor. Clearly, if the axial distance of the microparticle is increased only by 100 µm, the interference pattern varies only slightly. That makes the position estimation—or more specifically, estimation of the axial distance—very difficult. That is despite the fact that the interference patterns were captured in almost ideal conditions; they were illuminated by a laser and captured in very high resolution.

2.3 Back-propagation

Among other things, digital holography is also used because it enables us to see micro-objects only by use of an image sensor [6, 7, 5, 8]—without a microscope and without any lenses at all, actually. The field dealing with the use of digital holography for the purpose of visualization of micro-objects is called *digital holographic microscopy (DHM)*. Nevertheless, as we already know, in digital-holography the micro-objects are not clearly visible in the images from the image sensor; the images capture interference patterns from the micro-objects. But as it was stated in the previous part of this section, the interference patterns have encoded information about the position and shape of the micro-objects in themselves. So the

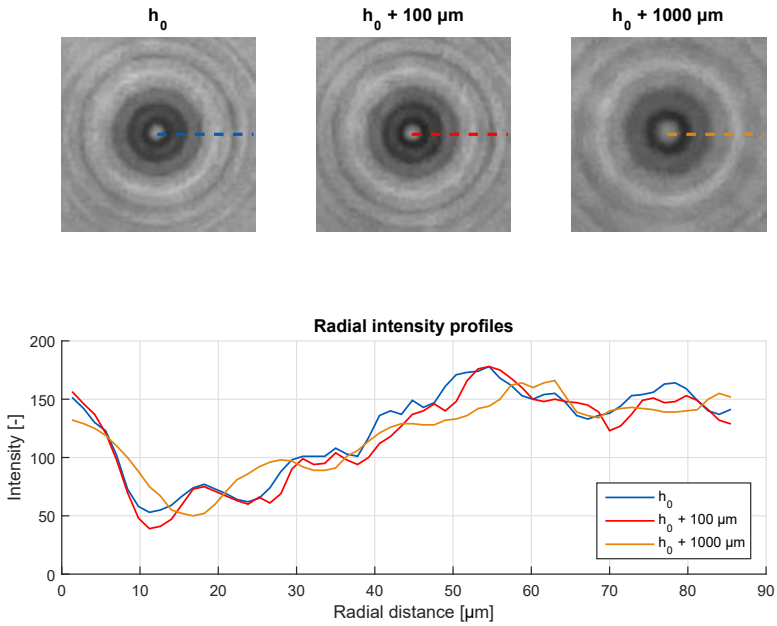


Figure 2.2: *Interference patterns and their radial intensity profiles from a spherical microparticle with diameter* 50 µm *located at different axial distances from the image sensor. The base axial distance* h_0 *is approximately* 1500 µm. *The microparticle was illuminated by a red laser with collimator and the interference patterns were captured by an image sensor with* 1.4 µm *pixels.*

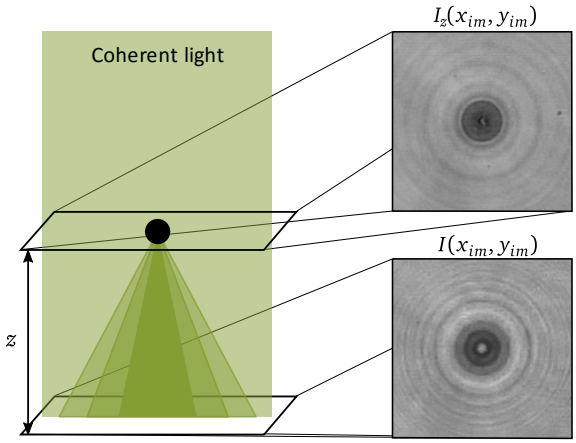


Figure 2.3: *Transformation of a captured image $I(x_{im}, y_{im})$ with an interference pattern from a microparticle to the image $I_z(x_{im}, y_{im})$ representing back-propagation of the original image to the axial distance z at which the microparticle is located.*

crucial topic in DHM is how to transform the images containing interference patterns from some objects to images displaying the objects. This transformation is called *back-propagation* because it actually simulates back propagation of the captured light field. The principle of the back-propagation is illustrated in Fig. 2.3.

An example of back-propagation is shown in Fig. 2.4. Fig. 2.4a displays an image with some interference patterns, but it is difficult to say by what objects they are formed by. Fig. 2.4b displays back-propagation of the image to the axial distance where the objects are located. Now, the situation is different. The objects are clearly visible in the back-propagated image.

The reason why it is possible to back-propagate the captured interference patterns is clarified in Appendix A.1.

2.3.1 Implementation

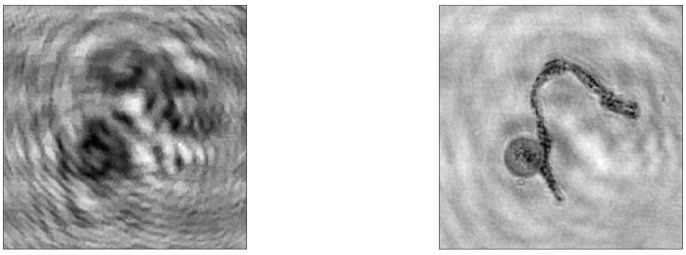
The back-propagation is carried by calculation of Rayleigh-Sommerfeld diffraction integral [9]. This is usually numerically done by the following relation

$$I_z(x_{im}, y_{im}) = \mathscr{F}^{-1}\left\{H_{-z}(f_x, f_y)\mathscr{F}\left\{I(x_{im}, y_{im})\right\}\right\}, \tag{2.1}$$

where (x_{im}, y_{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, $\mathscr{F}$ and $\mathscr{F}^{-1}$ are Fourier and inverse Fourier transformations, respectively, and

$$H_z(f_x, f_y) = \begin{cases} \exp\left(i2\pi z \frac{n}{\lambda}\sqrt{1-\left(\frac{\lambda f_x}{n}\right)^2-\left(\frac{\lambda f_y}{n}\right)^2}\right), & \sqrt{f_x^2+f_y^2} \leq \frac{n}{\lambda}, \\ 0, & \text{otherwise}, \end{cases} \tag{2.2}$$

is Fourier transform of Rayleigh-Sommerfeld propagator, where λ is the wave length of the illumination and n is refractive index of the surrounding medium.



(a) *Original* (b) *Back-propagated*

Figure 2.4: *Back-propagation in work: (a) shows an interference pattern from a microparticle and a piece of dust, (b) shows a back-propagation of the image.*

2.3.2 Issues

There are two problems with the back-propagation algorithm that was described here. As mentioned, the image sensor does not capture complete information about the light field, it captures only its intensity. That can be partially solved by estimation of the unrecorded phase [5]. We do not address the estimation here because the lack of information about the phase causes only visual artifacts in the back-propagated images, and in the proposed method, we will use back-propagation only for purposes where visual artifacts does not cause any troubles.

The second problem is that the medium between the image sensor and the microparticles is not—as you will see in Chapter 4—homogeneous while the back-propagation assumes that it is homogeneous. But that does not have to trouble us because the medium is characterized by its refractive index and refractive index only influences the speed at which the complex wave field propagates through the medium. Therefore, if the complex wave field propagates through more than just one medium and we use only one refractive index in the back-propagation algorithm then the back-propagated complex wave field to axial distance z_1 do no match to the actual complex wave field at z_1, but it matches to complex wave field at some near axial distance. With wrong refractive index, the back-propagation distances only shift with respect to the real ones. Since the back-propagation distance does not play any role in the proposed method, this imperfection is also not important.

Chapter 3

Review of methods

In this chapter we review some of the methods for position estimation of microparticles. However, we do not intend to cover all the principles because that was recently done by Yu at al. [10]. The main purpose of this chapter is to evaluate applicability of selected methods from the perspective of their possible use for real-time estimation.

The methods utilizing digital holography are here divided into three groups. The first group is based on the model of interference patterns given by Lorenz-Mie solution to Maxwell's equations. The second group exploits back-propagation of the interference patterns and the last group covers methods that estimate 3D position by making use of several light sources. In addition to methods based on digital holography, methods utilizing two-camera setup are discussed in the end of this chapter.

3.1 Lorenz-Mie solution

Lorenz-Mie solution to Maxwell's equation describes how light (electromagnetic) plane wave propagates through a homogeneous sphere [11, 12]. In other words, if we use the in-line digital holography setup, the solution describes how the interference pattern captured by an image sensor looks like when a spherical microparticle (of certain parameters like diameter, refractive index, ...) located at a certain axial distance is illuminated.

All methods localize the centers of the interference patterns at first. This way, the lateral position of the microparticles is estimated. Only the way how the axial distance is estimated differs.

Guerrero-Viramontes et al. [13] proposed one of the most simple methods based on the Lorenz-Mie solution. To estimate the axial position, the diameter of the first minimum in the radial intensity of the interference pattern is measured (see Fig. 3.1). The diameter is then searched in a look-up table that contains simulated diameters by Lorenz-Mie solution for microparticles located at various axial distances. Once the best matching entry is found, the corresponding axial distance is used as the estimate. The method is very simple and it definitely could be used in real-time. Nevertheless, as we showed in the previous chapter, the interference patterns vary only slightly in our case and thus the method would be very inaccurate.

Other methods [14, 15] estimate axial distance of the microparticles by fitting the Lorenz-Mie solution to the captured interference patterns. Since one of the parameters of the Lorenz-Mie solution is the axial distance, the solution that fits the best yields an estimate of the axial distance. The trouble is that the Lorenz-Mie solution is a rather complex model hence very

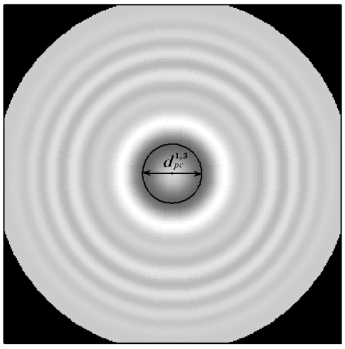


Figure 3.1: *Diameter of the first minimum in radial intensity of an interference pattern for a spherical microparticle (reprinted from [13]).*

computationally demanding to fit. And even if you manage to carry out the fitting in real-time, you still need to have the interference patterns in very high resolution to estimate the axial distance accurately. But high resolution means that you either have to magnify the interference patterns by an objective—remember, we excluded that option and want to use lensless setup—or you have to use an image sensor with very small pixels. With the pixel size, you would hit very quickly the limits of current image sensors. For instance, we use an image sensor with 3.75 µm pixels and you can get down to 1 µm with common image sensors, but no more. Furthermore, with smaller pixels you have to process more data or reduce the area where you want to estimate the position. In summary, these methods are not particulary well-suited for real-time use.

Park et al. [16] described a method where the interference patterns are at first numerically calculated for various axial distances. The captured interference pattern is then cross-correlated with the simulated interference patterns and the axial distance corresponding to the best match is taken as the estimate. This approach is very simple but it suffers from the same problem as the methods in the previous paragraph; the interference patterns have to be captured in high resolution.

To summarize, the methods based on the Lorenz-Mie solution are not suitable for our application. It is relatively easy to estimate the lateral position from the interference patterns but not so with the axial distance; it can be estimated very precisely by fitting Lorenz-Mie solution to the captured interference patterns, but that is computationally very demanding and requires high resolution interference patterns. Similarly, the axial distance can also be estimated by pattern matching or by measuring of the first minimum in radial intensity of the interference pattern, but then high resolution interference patterns have to be recorded otherwise the estimate is inaccurate.

3.2 Back-propagation

As it was described in Section 2.3, back-propagation is a method used in DHM to numerically reconstruct a captured image with interference patterns from some objects to an in-focus image where the objects have sharp contours and are clearly visible. There is an obvious way how to use back-propagation to estimate the axial distance of a microparticle. An image with an interference pattern from the microparticle can be successively back-propagated to increasing axial distances. The back-propagation is carried out up to the distance where the micropar-

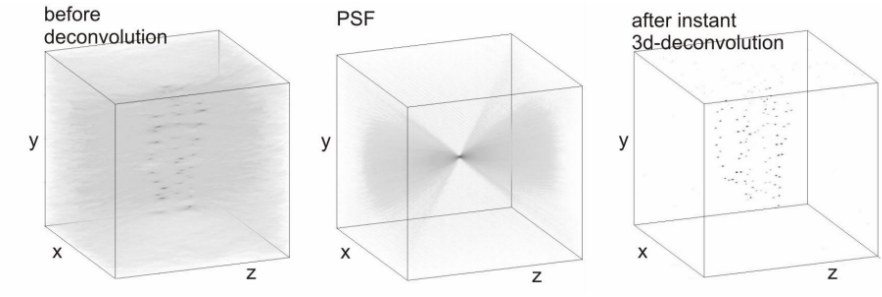
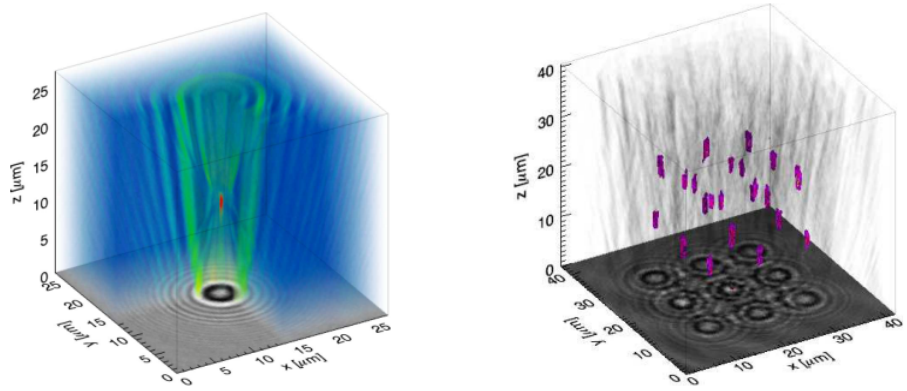


Figure 3.2: *Deconvolution of volumetric reconstruction of an image containing interference patterns from several microparticles with a PSF of the corresponding microparticle type (reprinted from [22]).*

ticle is in-focus. This distance can be used as the estimate of the axial distance [17, 18, 19].

Another way is to search the microparticles in a volumetric reconstruction. Volumetric reconstruction is a stack of images back-propagated for equidistantly divided axial distances. Dependent on the various aspects, the volumetric reconstruction is searched for microparticles by different methods. For instance, one way is to simulate a volumetric reconstruction for a single microparticle. This volumetric reconstruction is called *point spread function (PSF)*. The volumetric reconstruction for a captured image is then deconvolved with the PSF of the microparticle. The deconvolution yields bright spots at places where the microparticles are located (see Fig. 3.2). The volumetric reconstruction also can be searched for local extremes in the intensity which indicate a presence of a microparticle [20, 21]. Fig. 3.3 shows results of this method for an image with one microparticle and for an image with several microparticles.

Bouchal et al. [23] presented a method that can estimate position of microparticles in 3D based on only one back-propagation of the captured image. The method proceeds as follows. First, the captured image with interference patterns is back-propagated to a distance z_0 close to the expected axial distance of the microparticles. Then, the frequency spectrum of the back-propagated image is filtered by a filter designed so that the PSF of the microparticles changes from, roughly speaking, a cone-like PSF (see Fig. 3.3a or Fig. 3.2) to two spirals with a desired longitudinal period. In such a reconstructed image, there are two bright spots at



(a) *Single object* (b) *Multiple objects*

Figure 3.3: *Bright spots in volumetric reconstructions containing one and several microparticles (reprinted from [20]).*

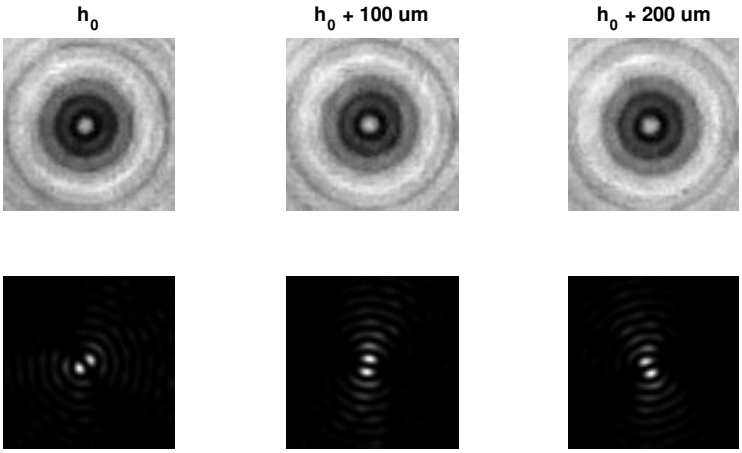


Figure 3.4: *The top row shows interference patterns from a microparticle located at different axial distances with the base axial distance* h_0=1700 µm. *The bottom row shows results of the processing of the interference patterns described in [23].*

places where the microparticles are located. The angle of the line joining these two spots with respect to the image coordinate system is proportional to the difference of the axial distance of the microparticle and z_0. Therefore, the axial distance of a microparticle is estimated based on this angle and lateral position of the microparticle is estimated as the center point between the two bright spots. Fig. 3.4 shows interference patterns from a 50 µm microparticle located at different axial distances together with their numerical reconstructions we have just described. Clearly, the two bright spots rotate as the axial distance of the microparticle changes hence their angle can be used to estimate the axial distance of the microparticle. This method is rather unique among other methods based on back-propagation because it needs to back-propagate the captured image only once and thus the method has a great potential for real-time use.

Despite the fact that methods based on back-propagation are not as computationally demanding as methods based on fitting of Lorenz-Mie solution, they are still rather complicated and they need to process a lot of data. Back-propagation of one image to one axial distance is computationally relatively cheap, but these methods need to perform the back-propagation for many distances. Furthermore, the precision of the methods is usually related to the number of performed back-propagations. To summarize, with one exception, the methods based on back-propagation are still too slow for real-time use.

3.3 Multi-angle illumination

In this section, we describe methods that use multiple light sources to estimate the axial distance of microparticles. The light sources illuminate the microparticles under different angles hence the interference patterns from one light source are shifted with reference to the interference patterns from the other light sources. The shift of the shadows from one microparticle and different light sources can be used to estimate the axial distance of the microparticle. This principle is shown in Fig. 3.5.

Su et al. [24] presented a method where microparticles are successively illuminated by individual light sources under different angles. For each light source, the image with interference patterns is recorded by an image sensor and for each interference pattern in the image an

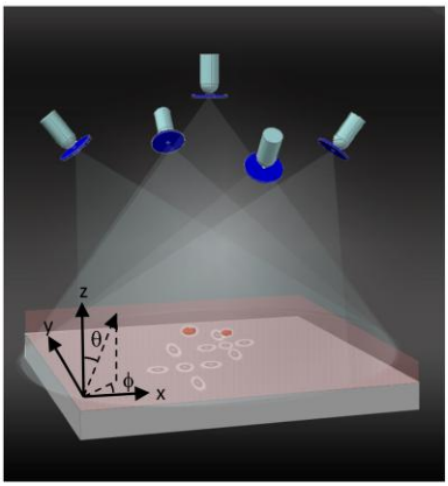


Figure 3.5: *Illustration of a multi-angle illumination hardware setup used for 3D position estimation of micro-objects (reprinted from [24]).*

imaginary ray connecting the center of the interference pattern and the light source is formed. Ideally, imaginary rays connecting interference patterns from one microparticle should intersect at one point indicating the position of the microparticle. Even though the method is very simple and might seem to be also very quick, it is not. The bottleneck is that the images are captured separately for each light source. That reduces the frame rate of the image sensor.

Su et al. further modified this method in [25]. They used only two light sources with different wavelengths. One light source illuminates the microparticles from the top, the other is tilted and illuminates the microparticles under an acute angle. The light sources illuminate the microparticles simultaneously hence there are interference patterns in the captured images from both of them and thus they have to be separated. This is done by back-propagation because objects are sharp in a back-propagated image only if the back-propagation is carried out with the right parameters (axial distance, wavelength of the light source, refractive indexes, ...). First, a captured image is back-propagated with the wavelength of the straight illumination. The distance where an object becomes sharp is taken as a rough estimate of its axial position and the position of the interference pattern in the image is taken as the estimate of its lateral position. According to the rough estimate, approximate position of its interference pattern in the oblique channel is calculated and then refined by localization of the nearest interference pattern. Since we know positions of both interference patterns from one object, we can simply calculate their lateral shift which is proportional to the axial distance. Again, in order to estimate the axial distance, several back-propagation have to be carried out and this take some time.

3.4 Two-camera setup

Very intuitive approach how to estimate position of objects in 3D is to use a two-camera (microscope) setup [26]: one microscope for a top view and one for a side view. Lateral position can be easily estimated from the top view and the axial position from the side view. This method is very simple in principle but has several drawbacks. The major drawback is the limited *depth of dield (DOF)* of the microscopes. DOF determines the range of distances

at which the microparticles are in focus. Thus, DOF also restricts the space where we can estimate the position. The top view enables us to estimate lateral position of microparticles in a very wide range, but the microparticles have to be located in a very narrows range of axial distances from the microscope and similarly for the side-view. Since we want to estimate both, lateral and axial position, the combination of the top and side view results in a very restricted space where the microparticle can be located. Furthermore, the hardware setup with two microscopes is rather complicated and expansive.

3.5 Conclusion

Despite the fact that there are plenty of methods for 3D position estimation, we did not find any method directly usable for real-time use in our case. All the methods described in the preceding sections have some aspects that make them difficult to use in real-time. Nevertheless, there is one method that seems to be modifiable for real-time applications and that is the method based on twin-beams illumination. Although it is rather computationally demanding in the original version presented in [25] it is possible to make it more efficient and in result faster. The following chapter is devoted to a description of such a modified method.

Chapter 4

Twin-beams method

This section presents a novel method for real-time position estimation of microparticles in 3D. To build at least a wisp intuition how the method works, the method is here briefly presented. The method is based on a very basic principle; it is based on *triangulation*. The principle is pictorially shown in Fig. 4.1. The microparticles are illuminated by two light sources under different angles. The first one illuminates the particles directly from the top and the second one is tilted. Because the particles are illuminated by two light sources under different angles, there are two shadows on the image sensor under each particle. These two shadows are laterally shifted with respect to each other and this lateral shift corresponds to the axial position of the microparticle. In the simplest case, if we measure the lateral shift of the shadows and assume that the light does not refract as it passes through different media, we can form a right triangle where one leg is the lateral shift and the other one is the axial distance. Provided we know the angle under which the second light source illuminates the microparticles, the axial distance can be easily calculated from the triangle.

The rest of the chapter is organized as follows. Section 4.1 describes the hardware setup. Section 4.2 briefly shows how the position estimation is carried out. In Section 4.3, we take into account propagation of light through different media in the hardware setup and derive the relation between the axial distance of a microparticle and the lateral shift of the shadows. Section 4.4 describes how the lateral shift of the shadows is measured. Section 4.7 addresses calibration of the method. In Section 4.8, we present some experimental results together with an assessment of the accuracy of the method. Finally, we discuss possible improvements of the proposed method in Section 4.9.

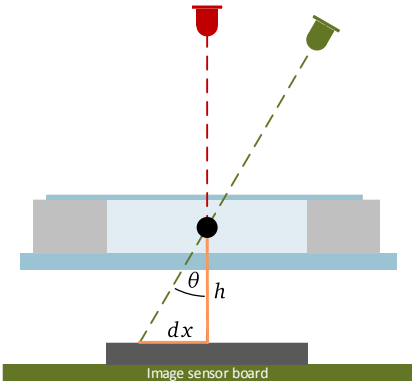


Figure 4.1: *A sketch of the basic principle of the twin-beams method for 3D position estimation of microparticles.*

4.1 Hardware description

Before we delve into a more detailed discussion of the method, we describe the hardware setup. The hardware setup developed for the proposed method is depicted in Fig. 4.2. We describe the image from the top to the bottom.

The microparticles are illuminated from the top by two plastic optical fibers with 500 µm core. The fibers are butt-coupled to red and green LEDs. The colors are chosen so that they match the sensitivity peaks of the image sensor hence each LED excites mostly one color channel of the image sensor with minimum leak to the other color channels. That allows very simple separation of interference patterns from one light source and from the other light source. The red and green LED have center wavelengths 625 nm and 525 nm, respectively. As can be seen from Fig. 4.4b, the peaks in the sensitivity of red and green channels of the image sensor are 540 nm and 610 nm, respectively. The optical fibers serve two purposes. They simplify the setup because the bulky and hot LEDs do not have to be mounted somewhere above the rest of the setup; they can be placed somewhere else where it is more convenient to place them and cool them. The second purpose is that an optical fiber functions as an optical aperture hence it filters the emitted light from LED so that the outgoing light is partially coherent and can produce interference patterns (for more on the topic see Chapter 2).

There is a pool made from *polydimethylsiloxane (PDMS)* under the optical fibers. Besides other reasons, PDMS is chosen because it is transparent. Therefore, if the pool is fabricated with some care one or more of its walls can be made transparent and hence it is possible to use a camera to see the microparticles from side. The pool is filed with deionized water where the microparticles are dispersed. The pool is covered by a cover glass because without it the surface of the water would function as a lens and thus it would distort the captured interference patterns on the image sensor.

The pool is placed on a foil with layer of *indium tin oxide* (ITO) wherein the electrode array is etched. The drawing of the electrode array is shown in Fig. 4.3. Apart form the electrodes, one can also see small rhombuses in the drawing. The rhombuses serve as marks for identification of transformation from the image coordinate system to the electrode array coordinate system. Such a transfomration is necessary for the control purposes where the electrodes are used for the manipulation of the microparticles. This topic will be discussed later on in Section 4.5. To prevent bending of the foil, the foil is placed on a microscope slide and the slide is placed above the image sensor by spacers made from PDMS.

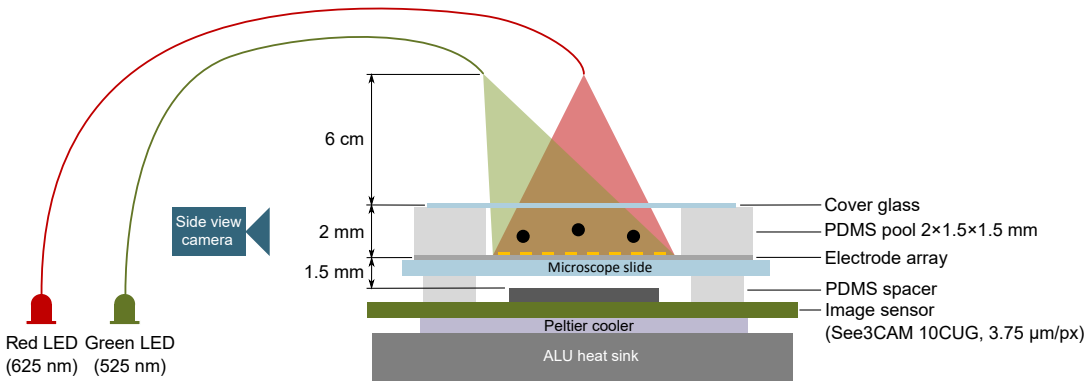


Figure 4.2: *A side view on the hardware setup used throughout this thesis.*

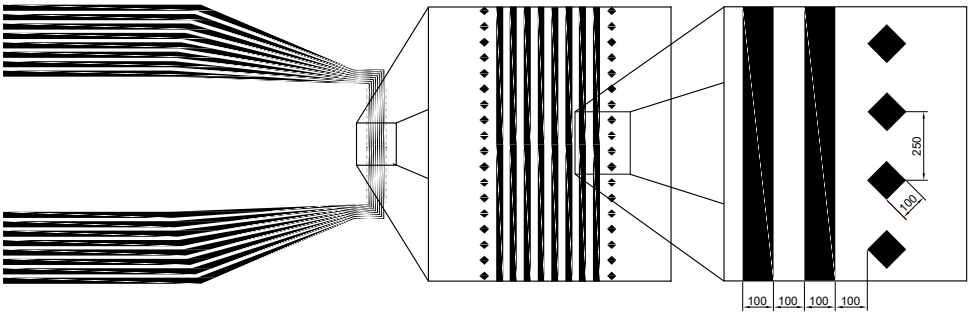
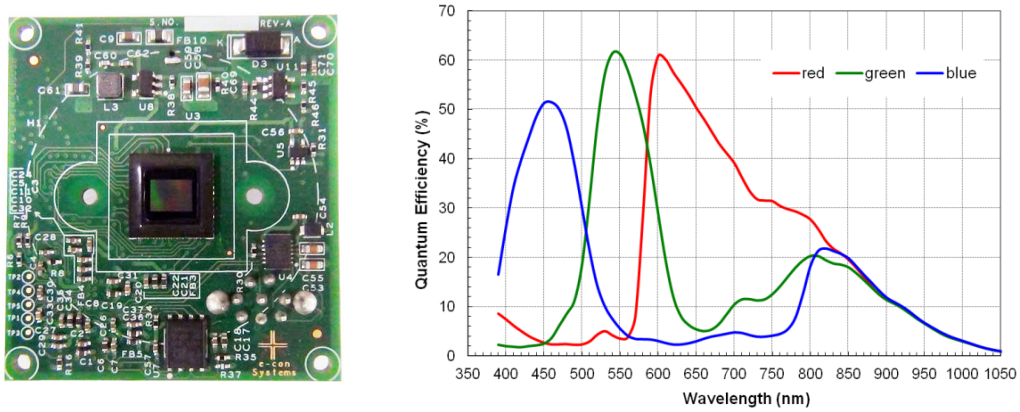


Figure 4.3: *A drawing of the electrode array used in the hardware setup.*

The used image sensor is See3CAM_10CUG camera from e-Con System (see Fig. 4.4a). The camera is based on a *complementary metal–oxide–semiconductor (CMOS)* chip with the maximum resolution 1280x960 pixels of size 3.75 µm. The camera communicates with a computer via USB 3.0 with maximum frame rate for the full resolution 45 frames per second. The CMOS chip does not capture color images directly; instead, it uses a so called Bayer filter which is a color filter array placed on the pixel array. The color filter array filters the incident light so that some pixels are sensitive mostly to red light, some mostly to green light, and some mostly to blue light. The sensitivity functions are shown in Fig. 4.4b. The color filter is organized in a grid composed of 2×2 squares with green filters at the main diagonal and red and blue filters at the remaining corners. The intensity values of the pixels sensitive to green light are used as intensity values for the green channel. Analogously, the values of pixels sensitive to red and blue light are used as values for the red and blue channels. This way, however, only a small part of the pixels in the color channels have assigned an intensity value. Values of the remaining pixels are usually obtained by an interpolation.

The image sensors heats up during operation and the heat transfers to the pool where it causes undesired fluid currents. To overcome this problem, the image sensor is cooled by a peltier cooler. The cooling side of the peltier cooler is coupled by thermal conductive paste to the image sensor board and the hot side is coupled to a custom-made aluminum heatsink.

The remaining undescribed component of the hardware setup is the side-view camera. The side-view camera provides a view on the microparticles from side hence it allows us to directly



(a) *Photo* (b) *Sensitivy characteristics*

Figure 4.4: *Image sensor See3CAM_10CUG used in the hardware setup.*

estimate the levitation height of the microparticles. As it is discussed in Section 3.4, side-view cameras are not particularly suitable for 3D position estimation in larger volumes, but here we use the camera only for the purposes of calibration and evaluation of accuracy of the proposed method; the limitied DOF does not cause us any troubles here.

4.2 Method description

Algorithm 1 briefly describes how the proposed method proceeds. The comments in the algorithm provide references to the sections or chapters where the corresponding part of the algorithm is discussed.

First of all, an image with interference patterns from both illumination sources is captured. The captured image is then separated to images with interference patterns from the straight and oblique illumination sources. Due to reasons discussed later, these images are

Algorithm 1. *Brief description of the method*

```
k := 0;                                                          // Time index
while true do
    img := captureImg();
    imgStr, imgObl := separImg(img);      // Separation of images from straight and
    oblique illumination sources (Section 4.4.1)
    /* Back-propagation of the image (Section 2.3)                          */
    imgStr := backProp(imgStr);
    imgObl := backProp(imgObl);
    imgVis := backProp(imgStr); //Back-propagation for visualization (Section 4.6)
    if initialization then
        for i ← 1 to N do
            strPos(k, i) := selectObj(imgVis);     // Manually mark ith object to track
            oblPos(k, i) := findCorrsp(strPos(k, i), imgObl);   // Find the corresponding
            interference pattern in imgObl (Section 4.4.2)
        end
    else
        for i ← 1 to N do
            strPos(k, i) := posEst(strPos(k-1, i), imgStr);    // Update position in imgStr
            (Section 4.4.3)
            roughPosEst := oblRoughPosEst(strPos(k-1, i), axDist(k-1, i));         // Rough
            estimate of position in imgObl (Section 4.4.2)
            oblPos(k, i) := posEst(roughPosEst, imgObl);   // Refine position in imgObl
            (Section 4.4.3)
        end
    end
    for i ← 1 to N do
        /* Transform positions to el.  array coordinate system (Section 4.5)  */
        strPos(k, i) := strTrans(strPos(k, i));
        oblPos(k, i) := oblTrans(oblPos(k, i));
        axDist(k, i) := axDistEst(strPos(k, i), oblPos(k, i));   // Estimate axial distance
        (Section 4.3)
    end
    k := k + 1;
end
```

back-propagated to distances where positions of interference patterns are easier to precisely localize. The image from the straight illumination is also numerically reconstructed in terms of DHM for the purpose of visualization.

In the initialization phase of the algorithm, an user marks the microparticles that will be tracked by the algorithm in the numerically reconstructed image from the straight illumination. The positions of the microparticles are further automatically refined and the resulting positions are taken as estimates of lateral positions of the microparticles. Based on these positions and the model of the lateral shift of interference patterns—dependence of the shift on the axial distance of a microparticle—approximate positions of the interference patterns in the image from oblique illumination are calculated.

In the run-time phase, the current position of a microparticle is estimated according to its last estimated position. Specifically, in the image from the straight illumination the interference patterns from the tracked microparticles are searched in small windows around the last measured position and the interference patterns in the image from the oblique illumination are searched in small windows around the position where the interference patterns should be located according to the last measured axial and lateral position.

For each microparticle, positions of its interference patterns are transformed from the image coordinate system to the electrode array coordinate system. The difference of these positions represents lateral shift of the interference patterns at the height of the electrode array and this lateral shift is used for the estimation of the axial distance of the microparticle.

4.3 Estimation of axial distance from lateral shift

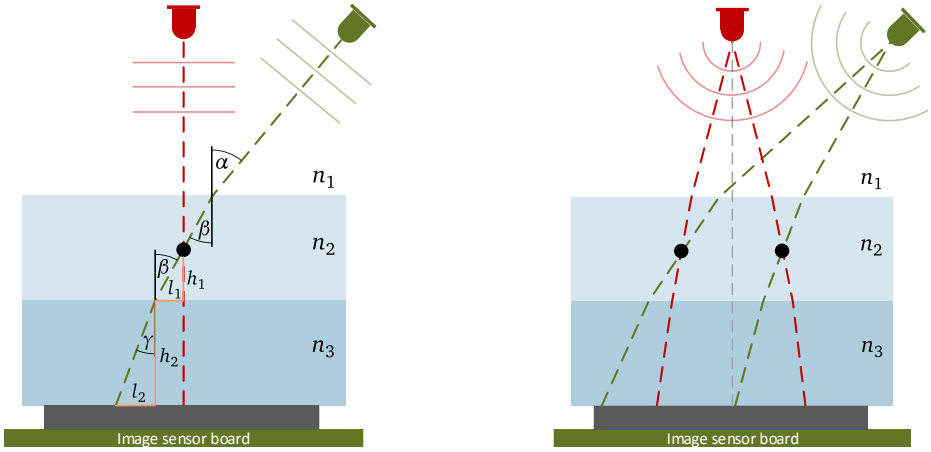
In the introduction to this chapter, the principle of the proposed method was illustrated on the simplified model where we neglected the refraction of light and assumed that the microparticles are illuminated by planar waves. In such a case, the axial distance of a microparticle together with lateral shift of its interference patterns forms a right triangle that can be used for the estimation of the axial distance (see Fig. 4.1). But the refraction of light clearly occurs in the hardware setup and the tips of the optical fibers behave more like sources of spherical waves. This section describes a model where the refraction of light is taken into account and the influence of non-planar illumination is minimized.

Refraction of light is a phenomenon occurring when light is incident upon media of different refractive indexes. The refraction of light obeys a simple physical law called Snells's law:

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{n_2}{n_1}, \tag{4.1}$$

where θ_1 is the angle of incidence of the light, θ_2 is the angle of refraction of the light and n_1, n_2 are indexes of refraction of the media.

Fig. 4.5a depicts the propagation of the light in the used hardware setup. For simplicity, only air, water in the pool and the supporting microscope slide are considered while the cover glass and the foil with the electrode array are neglected. In addition, it is assumed that the microscope slide lies directly on the image sensor. This simplifications should not introduce a significant error into the model because cover glass, electrode array and the space between the microscope slide and the image sensor are of substantially smaller thickness than the rest of the hardware setup. From the figure it is obvious that it is no longer possible to form a right



(a) (b)

Figure 4.5: *Propagation of the light from straight and oblique light sources illuminating (a) particle directly bellow the straight illumination source and (b) particles that are not directly below the straight illumination source.*

triangle with axial distance $H = h_1 + h_2$ as one leg and the measured lateral shift $L = l_1 + l_2$ in the captured image as the other leg. But we can form a smaller right triangle with legs h_1 and l_1 and use this triangle for the estimation of the axial distance. That seems reasonable because h_1 is the levitation height of the microparticle above the electrode array and that is what we actually want to measure anyway. Although lateral distance l_1 is not measured directly, it can be easily calculated from distance L by Snell's law. Or even more simply, we can use the following relation between h_1 and L

$$h_1 = \frac{L - l_2}{\tan\beta} = \frac{L}{\tan\beta} - \frac{l_2}{\tan\beta}. \tag{4.2}$$

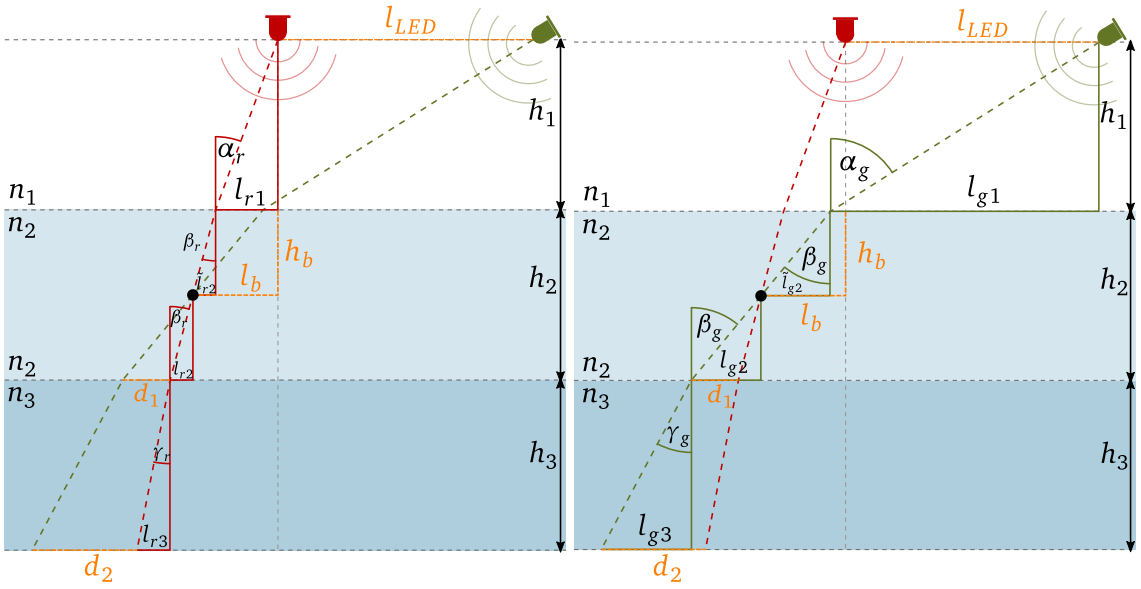
If we assume that l_2 and β are constant, we can make the substitution $k_1 = 1/\tan\beta$ and $k_2 = -l_2/\tan\beta$. Then we get

$$h_1 = k_1 L + k_2, \tag{4.3}$$

where k_1 and k_2 are constants which have to be calibrated. So we obtained a simple linear relationship between the axial distance h_1 and the measured lateral shift L.

The assumption of constant parameters l_2 and β is met if the microparticles are illuminated by planar illumination. According to the Huygen's principle, the tips of the optical fibers, however, behave more like sources of spherical waves. This is illustrated in Fig. 4.5b. The assumption still can be reasonable if the distance between the tips of the optical fibers and the microparticles is large enough. Then the light incident upon the microparticles can be approximated by planar waves. Unfortunately, it turns out that this not the case in the used hardware setup and that it is necessary to consider the light sources as sources of spherical waves.

A model of the lateral shift incorporating the refraction of light and the non-planar illumination would be too complex to use and calibrate. Nonetheless, the influence of these undesired effects depends on the distance at which the lateral shift is measured: the influence would be smaller if the microscope slide was thinner and the distance between the microscope slide and the image sensor smaller. We cannot make the microscope slide thinner nor put the



(a) *Straight illumination* (b) *Oblique illumination*

Figure 4.6: *Ray optics based model of propagation of the light illuminating a microparticle that is not placed directly below the straight illumination source. Left and right image describe propagation of the light from the straight and oblique illumination sources, respectively.*

image sensor closer, but we can transform the measured position of the interference patterns to the electrode array coordinate system and calculate the lateral shifts there. This way, we effectively measure the lateral shift at the height of the electrode array; we effectively measure directly the distance l_1 in Fig. 4.5a. Furthermore, positions of the microparticles in electrode array coordinate system need to be known for the control purposes anyway. It follows that the levitation height h_1 is simply

$$h_1 = l_1 \tan \beta. \tag{4.4}$$

At first glance, this model seems to be even simpler than model (4.3) because it has only one parameter. Nevertheless, in contrast to model (4.3), where L is directly measurable in the captured image, the calculation of l_1 in (4.4) requires identification of transformations from the measured positions of the interference patterns in the image coordinate system to the electrode array coordinate system. The transformations are described in Section 4.5.

We should analyze how accurate are models (4.3) and (4.4). Fig. 4.6 depicts how the light propagates through the simplified model of the hardware setup. Both parts of the figure captures the same situation. The only difference is that the straight illumination is annotated in the left part and the oblique illumination in the right part of the figure. Clearly, the image is based on ray optics and the light sources produce spherical waves. Notice that the microparticle is not placed directly below the straight illumination. That is because the influence of the spherical wave illumination get larger as the microparticle moves further away from the straight illumination; in other words, the influence is larger as l_b grows.

In order to compare the models, we at first find a relation between l_b and the lateral shifts d_1 and d_2—the lateral shifts of the interference patterns measured at the electrode array level and the image sensor level, respectively. Then, we will calculate the levitation height by models (4.3) and (4.4) based on d_1 and d_2 for different l_b and compare the errors.

We begin with the straight illumination (Fig. 4.6a) and determine l_{r2}. Once the angle β_r is known, distance l_{r2} can be easily calculated from the right triangle with legs l_{r2} and $(h_2 - h_b)$. From the figure we see that

$$l_b = l_{r1} + \tilde{l}_{r2}. \tag{4.5}$$

If we express l_{r1} and $\tilde{l}_{r2}$ by trigonometry from the right triangles they form, we obtain

$$l_b = h_1 \tan\alpha_r + h_b \tan\beta_r. \tag{4.6}$$

Now we make use of the Snell's law and express α_r by β_r

$$l_b = h_1 \tan\alpha_r + h_b \tan\beta_r = h_1 \tan\left(\arcsin\left(\frac{n_2}{n_1}\sin\beta_r\right)\right) + h_b \tan\beta_r. \tag{4.7}$$

Equation (4.7) is a nonlinear relation between levitation height of the microparticle, lateral shift l_b of the microparticle in reference to the position of the straight illumination and angle of incidence β_r. If we fix parameters l_b and h_b, we can numerically solve the equation for β_r and calculate distance l_{r2} of the interference pattern from the straight illumination at the electrode array level

$$l_{r2} = (h_2 - h_b)\tan\beta_r. \tag{4.8}$$

The distance of the interference pattern from the straight illumination at the image sensor level is then calculated by

$$l_{r3} = l_{r2} + h_3 \tan\gamma_r, \tag{4.9}$$

where $\gamma_r = \arcsin\left(\frac{n_2}{n_3}\sin\beta_r\right)$.

Similarly, we proceed with the oblique illumination source and obtain relationship for β_g

$$l_b + l_{LED} = l_{g1} + \tilde{l}_{g2} = h_1 \tan\alpha_g + h_b \tan\beta_g = h_1 \tan\left(\arcsin\left(\frac{n_2}{n_1}\sin\beta_g\right)\right) + h_b \tan\beta_g. \tag{4.10}$$

Lateral shifts of the interference patterns are then calculated as follows

$$l_{g2} = (h_2 - h_b)\tan\beta_g, \tag{4.11}$$

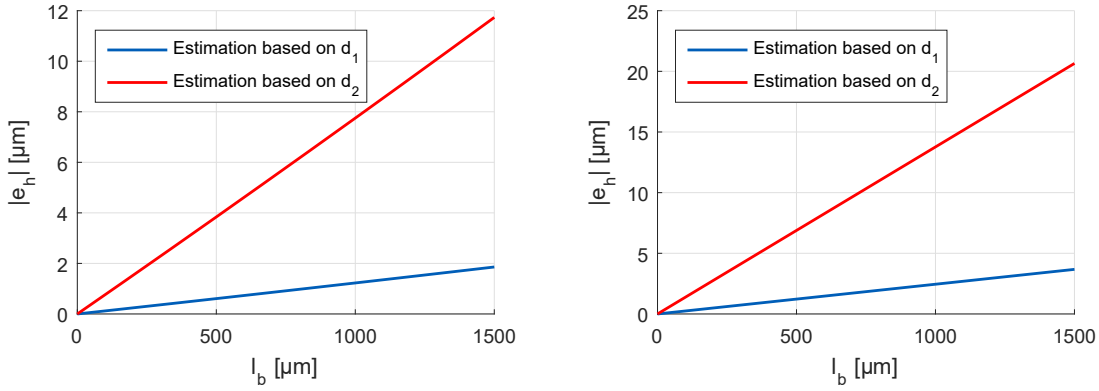
$$l_{g3} = l_{g2} + h_3 \tan\gamma_g = l_{g2} + h_3 \arcsin\left(\frac{n_2}{n_3}\sin\beta_g\right). \tag{4.12}$$

Finally, lateral shifts of the interference patterns from the oblique and straight illumination sources measured at the electrode array level and at the image sensor level are

$$d_1 = l_{g2} - l_{r2}, \tag{4.13}$$

$$d_2 = d_1 - l_{r3} + l_{g3}. \tag{4.14}$$

Having the relations for d_1 and d_2, we can finally compare the accuracy of the models. As a reminder, model (4.3) estimates the levitation height based on d_2—the lateral shift of the interference patterns measured directly in the image—while model (4.3) estimates the levitation height based on d_1—the lateral shift of the interference patterns measured in the electrode array system. Fig. 4.7a compares the accuracy of models (4.3) and (4.4). We used parameters β, k_1 and k_2 identified in the situation where the microparticle is placed directly under the straight illumination source at height $h = 200\,\mu\text{m}$ above the electrode array which



(a) *Axial distance error* (b) *Lateral position error*

Figure 4.7: *Error analyses of (a) axial distance and (b) lateral position estimation. The errors are calculated for the parameters of the used hardware setup and for a microparticle levitating at* 200 µm *above the electrode array.*

is approximately the maximum levitation height and thus the error corresponds to the worst case scenario. Fig. 4.7a shows the deviation of the estimates from the true levitation height 200 µm for varying lateral shift l_b. The lateral shift l_b varies from 0 to 1.5 mm because that is approximately the range where the microparticles can be manipulated. The remaining parameters are given by the dimensions of the hardware setup and they are: $h_1 = 60\,\mathrm{mm}$, $h_2 = 2\,\mathrm{mm}$, $h_3 = 1\,\mathrm{mm}$ and $l_\mathrm{LED} = 40\,\mathrm{mm}$. From the figure it is obvious that the model represented by (4.4) estimates the position more accurately; the error is approximately six times smaller than that of the other model. Furthermore, we can see that this simple model provide sufficient precision.

As it was stated in Section 4.2, it is assumed that the interference patterns from the straight illumination are located directly below the microparticles hence their positions are used as estimates of lateral positions of the microparticles. Nevertheless, as it is obvious from the previous discussion, due to the non-planar illumination the lateral positions of the interference patterns from the straight illumination are shifted with respect to the lateral position of the microparticles. If the position of the interference pattern is measured at the electrode array level, then this shift is equal to l_r2 (see Fig. 4.6a) and if the position is measured at the image sensor level then the shift is equal to $l_\mathrm{r2} + l_\mathrm{r3}$. Fig. 4.7b shows absolute values of the error in the estimated lateral position at the electrode array level and at the image sensor level. Again, apparently the error is significantly smaller if the position is measured at the electrode array level.

The comparison of accuracy of the models clearly shows that the model estimating axial distance from the lateral shift measured at the electrode array level achieves better accuracy than the other one. It is important to note that in the error analyses it was assumed that the positions of the interference patterns are known. But they are not known, they have to be measured and that, of course, introduces another source of error. Here, we only wanted to show that it is necessary to somehow deal with the non-planar illumination and that model (4.4) does this job sufficiently well.

4.4 Measurement of lateral shift of interference patterns

As it is apparent from the previous section, measurement of the position of the interference patterns is of crucial importance because it determines the accuracy of the proposed method. In this section, we will show how the position of an interference pattern is measured and consequently how the lateral shift of the interference patterns from one microparticle is estimated.

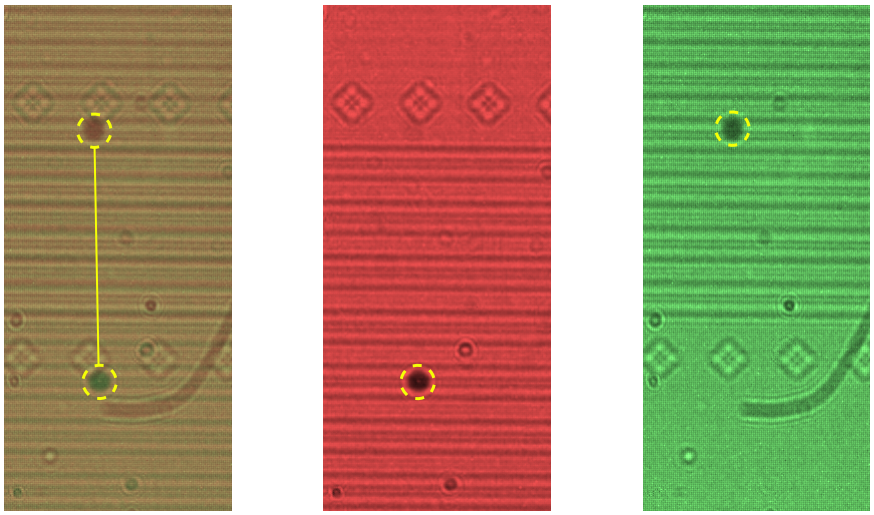
4.4.1 Separation of interference patterns

First of all, we need to separate the interference patterns from straight and oblique illumination. Thanks to the choice of the illumination sources (see Section 4.1), this is easily done: the red channel of a captured image contains only the interference patterns from the straight illumination and the green channel contains only the interference patterns from the oblique illumination. The separation of the interference patterns is shown in Fig. 4.8.

4.4.2 Identification of interference patterns corresponding to one microparticle

In order to measure the lateral shift of the interference patterns from one microparticle we have to be able to find which interference patterns from the straight illumination and oblique illumination correspond to the same microparticle. This task differs in the initialization and run-time phase of the algorithm.

In the initialization phase, an user selects the microparticles that will be tracked. The selection is done by marking the microparticles in the numerically reconstructed image from the straight illumination (see Section 4.6). As a byproduct, the user directly provides positions of the interference patterns in the red channel (straight illumination), only the corresponding interference patterns in the green channel (oblique illumination) have to be found.



(a) *Captured image* (b) *Red channel* (c) *Green channel*

Figure 4.8: *Separation of interference patterns from the straight (red color) and the oblique (green color) illumination sources. The captured image contains interference patterns from one microparticle. The fact they correspond to each other is visualized by the yellow line. The red and green channel obviously separate the interference patterns from individual illumination sources.*

To find the corresponding interference pattern in the green channel, let us have a look where it actually can be located. The angle between the corresponding interference patterns is given by the mutual position of the illumination sources and thus we assume that it is constant. The lateral shift depends on the axial distance of the microparticle. Since the microparticles can levitate only in a very limited range, the lateral shift is limited as well. Therefore, given the position of an interference pattern in the red channel, the corresponding interference pattern in the green channel can be located only along a very short line segment (see Fig. 4.9) and that is where it is searched by an algorithm described in the following section. A drawback of this approach is that no other interference pattern from other microparticles can be located along the line segment, because then the algorithm may identify a wrong interference pattern as the corresponding one. But given the length of the line segment, this does not have to bother us. The microparticles would have to nearly touch each other to cause any troubles.

In the runtime phase it is assumed that the positions of the interference patterns in the red channel have not changed much from the last measurement and thus the last measured positions are used as rough estimates of the current positions. The rough estimates are further refined by the algorithm described in the following section.

The corresponding interference patterns in the green channel are found by similar approach. Here, the position of an interference pattern does not depend only on the lateral position of the microparticle; it also depends on its axial distance. Therefore, the rough estimate is calculated according to the currently estimated position of the corresponding interference pattern in the red channel and the last estimated axial distance. Again, the rough estimates are refined by the algorithm described in the following section.

4.4.3 Precise localization of interference patterns

This section describes how the rough estimate of the position of an interference pattern in the red or green channel is refined.

The refinement of the position is not carried out directly in the recorded color channels. As it is discussed in Section 4.1, the image sensor does not record RGB image with each color

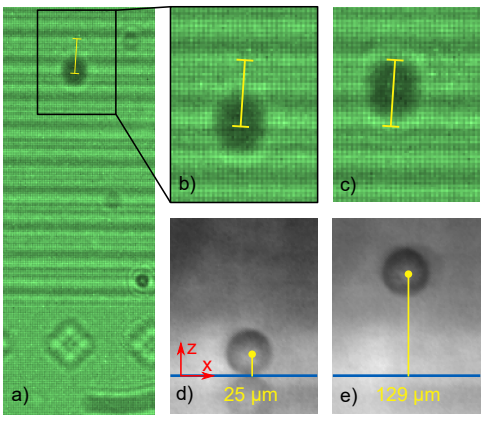


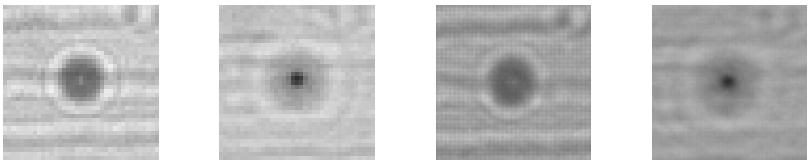
Figure 4.9: *Variation of the position of the interference pattern from a microparticle levitating at different heights while its lateral position is fixed. (a) shows a cutout of the green channel and the yellow line represents all possible positions of the interference pattern, (b-c) Blown-up regions of the green channel for a microparticle located at different levitation heights as it is shown from side view in (d-e).*

channel at full resolution. Specifically, only one quarter of the intensity values in the red channel are actually recorded by the image sensor. The remaining values are usually obtained by an interpolation. Since the interpolation cannot add any new information to the image, we can get rid of the pixels without recorded intensity value by collapsing the red channel to an image with quarter of the original resolution. This way, we have to process less data and do not lose any information. The grid of the Bayer filter does not allow us to do the same with the green channel, where every other pixel contains recorded value of intensity. Therefore, we use the green channel in the full resolution and the unrecorded values are computed by an interpolation. Fig. 4.10a shows a cutout from the reduced-resolution red channel, and Fig. 4.10c shows a cutout from the interpolated green channel.

Fig. 4.10 also shows the back-propagation of the cutouts for distances where the interference patterns focus to a point. This distance was find by back-propagation of the cutouts for successively increasing distances. The radial intensity (intensity in dependence on the distance from the center of the interference pattern) of the interference patterns is displayed in Fig. 4.11. We can see from the figure that the interference patterns are focused for a rather large range (several hundreds of micrometers) of back-propagation distances. Choosing the back-propagation distance in the middle of the range we can be sure that the interference patterns will be focused for all microparticles (independent on their levitation height) because the axial distance of the microparticles is limited to 200 μm.

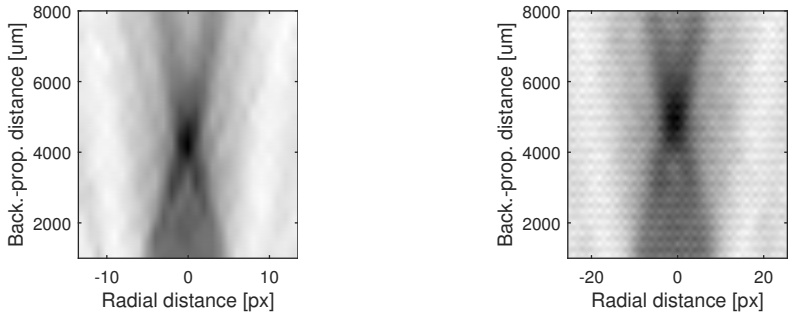
The back-propagation is very beneficial here. From the perspective of image processing, it is easier to identify the position of interference patterns in the back-propagated images than in the original images. Positions of the focused interference patterns can be simply identified by weighting pixels around the expected position—the rough estimates—by their intensity values. Whereas in the original image, one would have to use a more sophisticated algorithm. Thresholding followed by calculation of the center of mass could do the work, but it is very sensitive to the choice of the threshold. In addition, the optimal threshold can change as the ambient conditions change.

Back-propagation is not a cheap computational operation because it involves *Fast Fourier transform (FFT)* of the image to be back-propagated. FFT is more efficient for images with dimensions of a power of two. Therefore, the captured red and green channels are cropped to sizes 256 × 256 px and 512 × 512 px, respectively. The size of the cropped red channel is smaller due to the reduction of the resolution discussed in the beginning of this subsection. The green channel has resolution 3.75 μm/px which is the original resolution of the image sensor. We can easily calculate that the cropped green channel captures 1920 × 1920 μm part of the electrode array. The red channel has resolution 7.5 μm/px but it captures the same



(a) (b) (c) (d)

Figure 4.10: *Localization of the interference patterns in the red (straight illumination) channel and green (oblique illumination) channel: (a) displays the reduced-resolution red channel, (c) shows the green channel after the interpolation, (b) and (d) show back-propagations of (a) and (c) where the interference patterns focus to a point.*



(a) *Red channel* (b) *Green channel*

Figure 4.11: *Radial intensity profiles of interference patterns for (a) the red channel (straight illumination) and (b) the green channel (oblique illumination).*

area. The color channels are cropped so that each one contains the same part of the electrode array.

Position estimation of a focused interference pattern is carried out as follows. Since we have a rough estimate, we can look only at its small neighborhood and search the interference pattern there. At first, we search the interference pattern in a rather large neighborhood because the rough estimate can be very inaccurate. The position of the focused interference patterns in the neighborhood is estimated by the following equations

$$x_c = \frac{\sum_{ij} x_{ij} I_{ij}^p}{\sum_{ij} I_{ij}^p}, \tag{4.15}$$

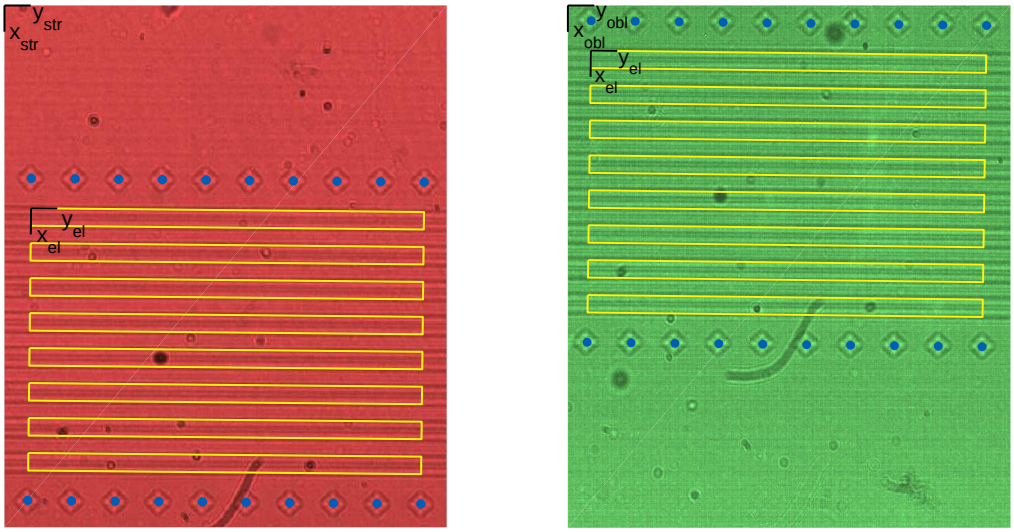
$$y_c = \frac{\sum_{ij} y_{ij} I_{ij}^p}{\sum_{ij} I_{ij}^p}, \tag{4.16}$$

where x_c and y_c are position coordinates of the interference pattern in the image coordinate system, I_{ij} is the intensity of the pixel at coordinates i and j, p is a parameter of an even value that determines how much the intensity is weighted and the summation is carried out over a specified neighborhood of the rough estimate. The position x_c and y_c is taken as a new rough estimate and the estimation is done repeatedly until a certain number of iterations is reached or until the newly estimated position is within a specified tolerance the same as the last one.

To make the estimate more accurate, the estimated position is taken as a new rough estimate and the estimation is carried out once more, but this time with a smaller neighborhood. This way, the motion of the microparticles should be covered by the larger neighborhood and the smaller neighborhood allows us to estimate the position more precisely because it suppresses influence of the surroundings of the interference pattern.

4.5 Transformation of the coordinate systems

As it was discussed in Section 4.3, lateral and axial position of a microparticle is estimated more precisely if the positions of its interference patterns measured in the image coordinate system (x_{im}, y_{im}) are transformed to the electrode array coordinate system (x_{el}, y_{el}). Since we have two color channels we need two transformations from (x_{im}, y_{im}) to (x_{el}, y_{el}). In the following paragraphs we will describe how to find and use the transformations.



(a) *Red channel* (b) *Green channel*

Figure 4.12: *Transformation of the image coordinate system to the electrode array coordinate system. Blue dots show positions of the side marks given by the user. Yellow rectangles represent transformed positions of the electrodes. (a) shows an image with interference patterns from the straight illumination and (b) shows an image with interference patterns from the oblique illumination.*

We assume that $(x_{\text{im}}, y_{\text{im}})$ and $(x_{\text{el}}, y_{\text{el}})$ are related by a planar projective transformation, that is by a linear transformation on homogeneous vectors represented by a regular matrix from $\mathbb{R}^{3x3}$ [27]. Thus, a projective transformation for the red channel from $(x_{\text{im}}, y_{\text{im}})$ to $(x_{\text{el}}, y_{\text{el}})$ is expressed by the following relation

$$\begin{bmatrix} x_{\text{el}}w \\ y_{\text{el}}w \\ w \end{bmatrix} = \mathsf{H}_{\text{R}} \begin{bmatrix} x_{\text{im}} \\ y_{\text{im}} \\ 1 \end{bmatrix}, \tag{4.17}$$

where $\mathsf{H}_{\text{R}} \in \mathbb{R}^{3x3}$.

Of course, the projective transformation for the green channel looks exactly the same, only the transformation matrix—let us denote it by H_{G}—differs.

To identify the matrices H_{R} or H_{G}, at least four pairs of corresponding points in the coordinate systems have to be known. Here comes the time for the side-marks along the electrodes (see Fig. 4.3). Since the positions of the side-marks in the electrode array coordinate system are known and the side-marks are clearly visible in the image, they are used for the identification of the projective transformations. An user manually marks positions of several consecutive side-marks in the image at the top and bottom of the electrode array. The first side-mark in the bottom line have to lie bellow the first one in the top line because then the relative positions of the marked side-marks can be simply determined. Now, for each marked position in $(x_{\text{im}}, y_{\text{im}})$, the corresponding position in $(x_{\text{el}}, y_{\text{el}})$ is easily found and the transformation H_{R} or H_{G} can be identified. The problem of determining the transformation matrix from a set of corresponding points is a standard task. Thus it is not described here and the reader is referred to [27].

4.6 Visualization of lateral position of microparticles

To provide an user a possibility to visually inspect the motion of the microparticles, the captured image with interference patterns from the straight illumination is numerically reconstructed so that it resembles a view from a microscope. The numerically reconstructed image allows the user to see directly the microparticles and their motion independent on the proposed position estimation method. Furthermore, in contrast to the position estimation algorithm, the reconstructed image shows also the untracked objects hence it can also possibly show some obstacles in the trajectories the microparticles are controlled along.

The numerical reconstruction is carried out by back-propagation (see Section 2.3) of the image from the straight illumination (the red channel of the captured image) to an axial distance where the objects become sharp. In order to make the back-propagation efficient, the same 256×256 px cut-out of the red channel as was used for the position estimation is used here. The same cut-out is chosen because now we want to back-propagate an image which was already back-propagated. Therefore, the FFT of the image, which is necessary for the back-propagation, was already computed and we do not have to compute it again. This saves some time and makes the method more efficient.

To help the user to identify positions of the microparticles in the electrode array coordinate system, the electrodes are visualized in the reconstructed image. Positions of the electrodes in the image are computed by the inverse projective transformation H_R. The numerically reconstructed cut-out of the red channel with the visualized electrodes is shown in Fig. 4.13. Apparently, the numerically reconstructed image servers its purpose pretty well because the microparticles in the figure are clearly recognizable.

4.7 Calibration

The proposed method has several parameters that are not a priori known and have to be identified. Namely, angle β in model (4.4) and projective transformations H_R and H_G repre-

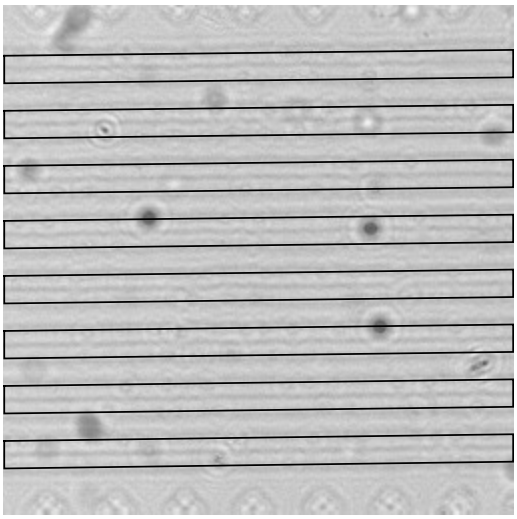


Figure 4.13: *Numerical reconstruction of an image from the straight illumination to an image resembling a view by a microscope. The black rectangles represent the electrodes.*

senting mapping from the image coordinate system to the electrode array coordinate system. Identification of H_R and H_G was discussed in Section 4.5 hence there is no need to discuss it again here. This section is devoted to identification of the angle of incidence β which turns out to be a more difficult task.

There is a dedicated device solely for the purpose of the identification of β in the hardware setup: the side-view camera. Side-view camera allows us to see the microparticles from the side hence measure their levitation height (axial distance). As a result, it provides us the necessary reference for the identification of β. We can focus the side-view camera to a microparticle, change its levitation height by DEP and for each levitation height record the image from the side-view camera and lateral shift of the interference patterns from the microparticle. The angle β is then easily found by fitting model (4.4) to the recorded set of data.

In order to be able to measure the axial distance of a microparticle in an image from the side-view camera, we need to transform the image coordinates to another coordinate system where we can measure distances in micrometers. To establish such a coordinate system, we put the mask used for the fabrication of the electrode array into the pool. Fig. 4.14 shows an image from the side-view camera with the mask in the pool. In addition, the figure shows two coordinate systems. The coordinate system (x_{sv}, y_{sv}) is the coordinate system of the image and the coordinate system (x_{msk}, y_{msk}) is a coordinate system established in reference to the electrode array in the mask. The mask is placed in the pool so that it is perpendicular to the bottom of the pool. Therefore, the axial distance of a microparticle can be measured in the coordinate system (x_{msk}, y_{msk}). Again, we relate the two coordinate systems by a projective transformation and identify it similarly as in Section 4.5: an user marks corners of the side-marks in the image, positions of the side-mark corners in (x_{msk}, y_{msk}) are known and thus we have pairs of corresponding points for the identification of the projective transformation.

Let us use the same notation as in (4.4): h_1 denotes the levitation height and l_1 the lateral shift. Now, we can conduct an experiment and measure lateral shifts l_1 for a microparticle located at different levitation heights h_1, and use this set for the identification of the angle

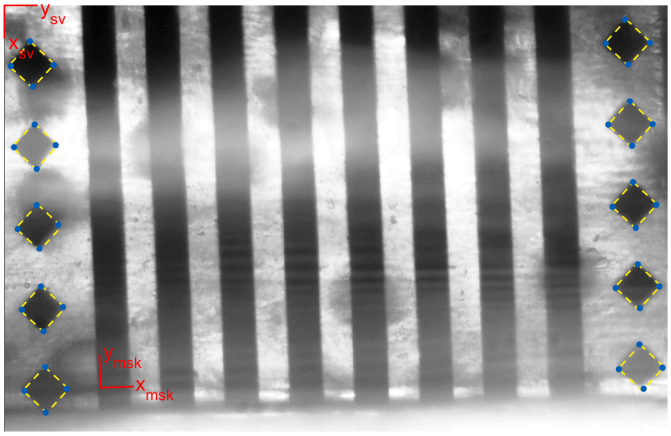
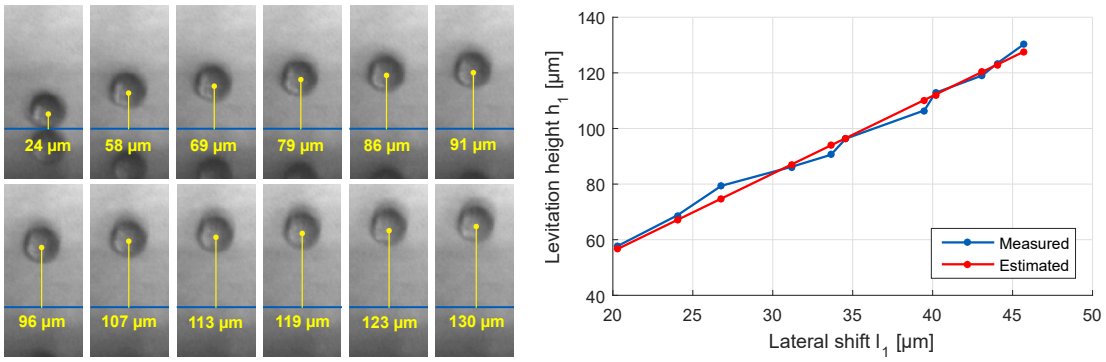


Figure 4.14: *An image from the side-view camera capturing a mask with electrodes and side-marks. The corners of the side-marks marked by an user are represented by blue dots. Yellow lines show edges of the transformed positions of the side-marks by the identified projective transformation.*



(a) (b)

Figure 4.15: *Calibration of the model of the levitation height-lateral shift dependence. (a) side views on a microparticle at different levitations heights and (b) fitting of model to the measured data.*

β. Nevertheless, what we are really after is not directly the angle β but $\tan\beta$ because that is the scalar factor between lateral shift l_1 and levitation height h_1. Let us denote $\tan\beta$ by k_β. Identification of k_β is easily done by fitting the model

$$h_1 = k_\beta l_1 \tag{4.18}$$

to the measured set of data.

Fig. 4.15a shows side-views on a microparticle at different levitation heights. The levitation heights are computed by the projective transformation as the distance in the coordinate system $(x_{\text{msk}}, y_{\text{msk}})$ from the marked center of the microparticle to the line representing the bottom of the pool. The line representing the bottom of the pool is easily identified from a side-view where a microparticle lies on the bottom of the pool.

A set of measured levitation heights h_1 and the corresponding lateral shifts l_1 is shown in Fig. 4.15b. The figure also shows the estimation of the levitation height based on the identified model represented by (4.18). Apparently, the estimated values match the measured ones.

4.8 Experimental results

To evaluate the performance of the proposed method, we manipulated a microparticle along an eight-shaped trajectory and compared the estimated position by the proposed method with the reference measurement obtained by the side-view camera. The estimation was carried out in real-time at 10 Hz on an ordinary PC (Intel Core i7, 8 GB RAM) and it was used in the feedback loop of the control algorithm proposed in Chapter 8. The comparison is displayed in Fig. 4.16. The side-view camera enables us to measure only one coordinate of the lateral position but from the principle of the estimation the estimate in the other coordinate should have the same accuracy. The standard deviation of the error in x-coordinate is 2.41 µm and of the levitation height 6.64 µm. Even though the experiment was conducted with only one microparticle, the proposed method can simultaneously estimate positions of more microparticles.

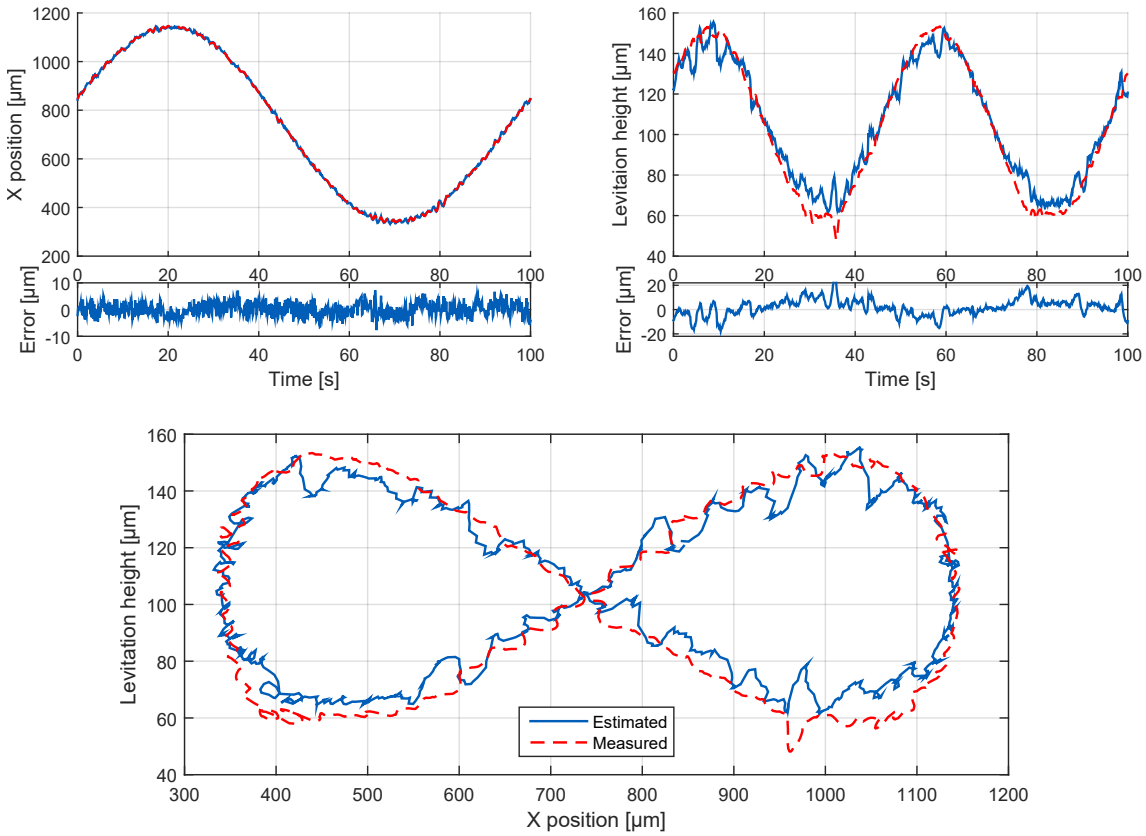


Figure 4.16: *A comparison of estimated position by the proposed method with reference measurement obtained by the side-view camera.*

4.9 Possible improvements

This section discusses two changes in the proposed method that could lead to either improved accuracy or better performance in terms of speed.

The accuracy of the estimation can be easily improved by use of an image sensor with smaller pixels. Actually, not only the accuracy would improve but also the numerically reconstructed image used for visual check would be more detailed. However, this comes at a cost. For the same manipulation area, smaller pixels mean more data to process and thus the method would be slower.

The most computationally demanding part of the proposed method is the image processing. Currently, the method is implemented in Simulink and all operations are performed by a CPU and CPUs are not particularly well suited for image processing. The speed of the proposed method could be improved by transferring the image processing to a GPU because GPUs are designed for this kind of operations and perform them much faster than CPUs. The major drawback of this improvement is its price. Price of a high-performance graphic card can easily climb up to several thousand of dollars. In addition, implementation of anything on a GPU is not as trivial as implementation on a CPU.

4.10 Conclusion

We developed a simple method for real-time position estimation of spherical microparticles in 3D. The method requires only a very cheap, simple and compact hardware setup. We demonstrated the accuracy of ~2–3 μm in lateral position and ~7 μm in axial position. Furthermore, we successfully used the method for real-time manipulation. Despite the fact that the method is developed for spherical microparticles, it can be potentially extended to track also non-spherical micro-objects. The only thing that would have to change is the procedure of localization of the interference patterns, because they would not have to focus to a point any more.

Part II

Model

Chapter 5

Motion of a microparticle in a DEP force field

We are able to measure the position of microparticles; next logical step on the way to the goal of controlling their position is to obtain a model of the motion and a model of the dielectrophoretic force acting upon the microparticles. The model of the dielectrophoretic force allows us to relate the set potentials on the electrodes to the generated force acting upon the microparticles. The model of the motion enables us to predict how a microparticle will react to an applied DEP force. Since it is not a goal of this thesis to develop a new mathematical model of motion of a microparticle in DEP force field and the author of this thesis did not contribute anyhow to this topic, we will provide only a very brief description of the model necessary for the remainder of this thesis. This whole chapter heavily mines from the work done by my colleagues Jiří Zemánek, Tomáš Michálek and Jakub Tomášek [28, 29, 30, 31].

5.1 Hardware setup

Before we describe the models, we extend the description of the hardware arrangement introduced in Section 4.1 to include also the electric part. We use a signal generator (METEX MXG-9810A) to generate a sinusoidal signal of frequency 300 kHz, 20 V peek to peek. The sinusoidal signal is multiplied by analog multipliers (Analog devices AD633ARZ) with analog outputs of an IO card Humusoft MF624. The resulting signal is applied on the electrodes. The IO card allows us to set the amplification of the sinusoidal signal from −1 to 1 in real-time from Matlab/Simulink environment.

5.2 Dielectrophoretic force

We briefly mentioned in the introduction that the DEP force somehow depends on the set potentials on the electrodes. Now the time comes to derive a precise mathematical relation.

The time-averaged dielectric force acting upon a spherical microparticle located at (x, y) is given by the following relation [32]:

$$\mathbf{F}^{\mathrm{dep}}(x, y) = k\nabla|\mathbf{E}(x, y)|^2, \tag{5.1}$$

where $\mathbf{E}(x, y)$ is electric field and

$$k = \frac{1}{4}v\,\mathrm{Re}\left(\frac{\varepsilon_{\mathrm{p}}(\omega) - \varepsilon_{\mathrm{m}}(\omega)}{\varepsilon_{\mathrm{p}}(\omega) + 2\varepsilon_{\mathrm{m}}(\omega)}\right), \tag{5.2}$$

where v is the volume of the microparticle, $\varepsilon_p(\omega)$ and $\varepsilon_p(\omega)$ are complex permitivities of the microparticle and the surrounding medium, respectively, and ω is the angular frequency of the sinusoidal signal generated by the signal generator. Time-averaged here means, that the force is averaged over the period of the sinusoidal signal. The only thing that ω influences are the permitivities and since we do not change ω during the manipulation with the microparticles, we can consider it to be constant. Thus, the only way how to shape $\mathbf{F}^{\text{dep}}(x,y)$ is through $\mathbf{E}(x,y)$, or more precisely, through the amplitudes of the applied potentials on the electrodes.

We should mention that relation (5.1) holds only when phase of the electric field $\mathbf{E}(x,y)$ is spatially invariant. But that is exactly our case because we are able to control only the amplitudes of the signals applied on the electrodes. Otherwise the relation would become a bit more complicated and we would have to consider also a phenomenon called *traveling wave DEP*.

The DEP force $\mathbf{F}^{\text{dep}}(x,y)$ acting upon a microparticle is proportional to the gradient of the magnitude of the electric field $\mathbf{E}(x,y)$ at the locus of the microparticle. Therefore, in order to determine $\mathbf{F}^{\text{dep}}(x,y)$, we have to determine the electric field. Since electric field is obtained as the negative gradient of the potential field $\phi(x,y)$, we start with the potential field. For simplicity, we will not explicitly state the dependence on coordinates x and y.

As linearity applies in this case, we decompose the potential ϕ to the sum of contributions from individual electrodes, that is

$$\phi = \sum_{i=1}^{8} u_i \phi_i = \mathbf{u}^\intercal \mathbf{\Phi}, \tag{5.3}$$

where u_i is a scaling factor and ϕ_i is the contribution to the net potential from ith electrode when 1 V is applied on it and the remaining electrodes are grounded. We put the scaling factors u_i and potentials ϕ_i to vectors $\mathbf{u} = [u_1, \ldots, u_8]^\intercal$ and $\mathbf{\Phi} = [\phi_1, \ldots, \phi_8]^\intercal$. The electric field is then given by

$$\mathbf{E} = -\nabla\phi = \begin{bmatrix} E_x \\ E_y \end{bmatrix} = -\begin{bmatrix} \mathbf{u}^\intercal \frac{\partial \mathbf{\Phi}}{\partial x} \\ \mathbf{u}^\intercal \frac{\partial \mathbf{\Phi}}{\partial y} \end{bmatrix}. \tag{5.4}$$

Now, we substitute $\mathbf{E}$ to the relation for $\mathbf{F}^{\text{dep}} = [F_x^{\text{dep}}, F_y^{\text{dep}}]^\intercal$:

$$F_a^{\text{dep}} = 2k\left(E_x \frac{\partial E_x}{\partial a} + E_y \frac{\partial E_y}{\partial a}\right) = 2k\left[\mathbf{u}^\intercal \left(\frac{\partial \mathbf{\Phi}}{\partial x}\frac{\partial^2 \mathbf{\Phi}^\intercal}{\partial x \partial a} + \frac{\partial \mathbf{\Phi}}{\partial y}\frac{\partial^2 \mathbf{\Phi}^\intercal}{\partial y \partial a}\right)\mathbf{u}\right], \qquad a = \{x, y\}. \tag{5.5}$$

If we introduce matrix matrices $\mathbf{A}_a$ defined by

$$\mathbf{A}_a = 2k\left(\frac{\partial \mathbf{\Phi}}{\partial x}\frac{\partial^2 \mathbf{\Phi}^\intercal}{\partial x \partial a} + \frac{\partial \mathbf{\Phi}}{\partial y}\frac{\partial^2 \mathbf{\Phi}^\intercal}{\partial y \partial a}\right), \tag{5.6}$$

we can express $\mathbf{F}^{\text{dep}}$ as

$$\mathbf{F}^{\text{dep}} = \begin{bmatrix} \mathbf{u}^\intercal \mathbf{A}_x \mathbf{u} \\ \mathbf{u}^\intercal \mathbf{A}_y \mathbf{u} \end{bmatrix}. \tag{5.7}$$

The scaling factors u_i represent the control signals; it is through them we shape the potential field. Since we will refer to u_i very frequently in the remaining part of the thesis, from now on we will refer to u_i—even though it is not terminologically correct—-as potentials u_i and say that we set potentials u_i on the electrodes. It is not correct, but it sounds more intuitive

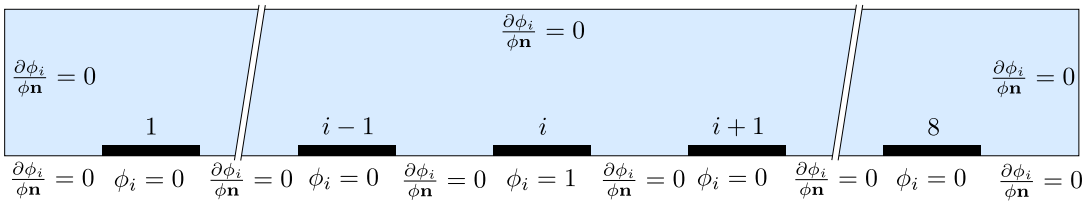


Figure 5.1: *Boundary conditions for Laplace equation* $\nabla^2\phi_i = 0$.

than saying that we change the scaling factors to shape the potential field above the electrode array.

We are still missing something to fully determine the relation between $\mathbf{u}$ and $\mathbf{F}^{\text{dep}}$: functions ϕ_i. It turns out, that this is actually the harder part. Functions ϕ_i, as potentials, have to satisfy *Laplace equation*

$$\nabla^2\phi_i = 0 \tag{5.8}$$

for given boundary conditions. Laplace equation falls into the category of PDEs which are known to be difficult to solve. To entirely specify a problem with a PDE, we have to add boundary conditions—a required behavior of the solution at the boundary of a domain where we seek a solution. The domain together with the boundary conditions for our case is shown in Fig. 5.1. Each electrode imposes a *Dirichlet boundary condition*: $\phi_i = 1$ V for the ith electrode and $\phi_i = 0$ V for the remaining ones. On the rest of the boundary, we use *Neumann boundary condition* $\frac{\partial\phi_i}{\partial\mathbf{n}}$, where $\mathbf{n}$ is the normal vector to the boundary.

We should mention that for the control purposes we will need to evaluate (5.7) in real-time hence we need to calculate functions ϕ_i—or actually its first and second derivatives (see (5.6))—in advance, store them and load the solution every time we need to determine the DEP force based on the set potentials $\mathbf{u}$.

There are basically two ways how to solve a PDE:

Numerical solution Nowadays, numerical solvers of partial differential equations are fairly developed and easy to use. One simply specifies the problem—the partial differential equation, domain and boundary conditions—and the solver gives him a solution. However, the solver provides only an approximate solution and only as numerical values at certain grid of points over the domain. As a result, we are able to determine DEP force only at certain points in the domain and the solution has to be stored as a file which grows in size as the grid gets finer. Nevertheless, a great advantage of this approach is that it enables us to solve PDEs with mixed boundary conditions (Dirichlet and Neumann) very easily. Why this is important will be obvious in the next chapter.

Analytical solution The other way is to somehow find a closed-form solution. This is a very tricky task because there is no general procedure of solving PDEs. Nevertheless, for a limited set of problems there are known solutions or approaches for obtaining them. We will present one of this approaches in the next chapter.

Tomáš Michálek and Jiří Zemánek [28, 29] successfully used numerical solution as a way of solving this problem. However, they found this approach to be rather impractical due to the file size. In particular, a numerical solution for the eight electrode array with grid spacing 1 μm takes approximately 750 MB. They also used the same approach for more complicated electrode array arrangements where the file size grew up to several tens of gigabytes. To

overcome this problem, we will use one of the approaches for obtaining an analytical solution and discuss the necessarily modifications of the original boundary value problem in the next chapter.

5.3 Dynamics of a microparticle in fluid

We assume that the inertia inertia of a 50 μm microparticle is negligible [33]. The the motion of a microparticle is governed by drag force $\mathbf{F}^{\text{drag}}$, sedimentation force $\mathbf{F}^{\text{sed}}$ and DEP force $\mathbf{F}^{\text{dep}}$.

Drag force is a frictional force acting against the motion of a microparticle in liquid. In our case, we can apply *Stoke's law* and compute the drag force as

$$\mathbf{F}^{\text{drag}} = 6\pi\mu r\mathbf{v}, \tag{5.9}$$

where μ is the dynamic viscosity of the fluid, r is the radius of the microparticle and $\mathbf{v}$ is its velocity.

Sedimentation force is a force caused by gravity and it is given by

$$F^{\text{sed}} = \frac{4}{3}\pi r^3(\rho_{\text{p}} - \rho_{\text{m}}), \tag{5.10}$$

where ρ_{p} and ρ_{m} are densities of the microparticle and the surrounding medium, respectively.

Finally, the model of motion of a microparticle is obtained by putting all the forces into balance. Specifically, to obtain a state space model, we express the components of velocity vector $\mathbf{v}$ from $\mathbf{F}^{\text{drag}}$. This way we get

$$\dot{x} = k_{\text{F}} F_{\text{x}}^{\text{dep}}, \tag{5.11a}$$
$$\dot{y} = k_{\text{F}}\left(F_{\text{y}}^{\text{dep}} - F^{\text{sed}}\right), \tag{5.11b}$$

where $k_{\text{F}} = \frac{1}{6\pi\mu r}$.

For simplicity, we denote $F_{\text{x}}^{\text{dep}}$ by F_{x} and $(F_{\text{y}}^{\text{dep}} - F^{\text{sed}})$ by F_{y}. Thus we obtain

$$\dot{x} = k_{\text{F}} F_{\text{x}}, \tag{5.12a}$$
$$\dot{y} = k_{\text{F}} F_{\text{y}}. \tag{5.12b}$$

Assuming we can arbitrarily control the forces F_{x} and F_{y}, equation (5.12) represents a very simple linear model.

5.4 Extension from 2D to 3D

We described a model of DEP force and a model of dynamics of a microparticle in 2D, but it can be analogously derived for 3D. Without going into details, here we just state that the DEP force is

$$\mathbf{F}^{\text{dep}} = \begin{bmatrix} \mathbf{u}^{\intercal}\mathbf{A}_{\text{x}}\mathbf{u} \\ \mathbf{u}^{\intercal}\mathbf{A}_{\text{y}}\mathbf{u} \\ \mathbf{u}^{\intercal}\mathbf{A}_{\text{z}}\mathbf{u} \end{bmatrix} \tag{5.13}$$

with

$$\mathbf{A}_a = 2k\left(\frac{\partial \boldsymbol{\Phi}}{\partial x}\frac{\partial^2 \boldsymbol{\Phi}^\intercal}{\partial x \partial a} + \frac{\partial \boldsymbol{\Phi}}{\partial y}\frac{\partial^2 \boldsymbol{\Phi}^\intercal}{\partial y \partial a} + \frac{\partial \boldsymbol{\Phi}}{\partial y}\frac{\partial^2 \boldsymbol{\Phi}^\intercal}{\partial z \partial a}\right), \tag{5.14}$$

and the model of the dynamics of a microparticle in fluid is

$$\dot{x} = k_\mathrm{F} F_\mathrm{x}^\mathrm{dep}, \tag{5.15a}$$
$$\dot{y} = k_\mathrm{F} F_\mathrm{y}^\mathrm{dep}, \tag{5.15b}$$
$$\dot{z} = k_\mathrm{F}\left(F_\mathrm{z}^\mathrm{dep} - F^\mathrm{sed}\right). \tag{5.15c}$$

To summary, we found a relation coupling applied potentials on individual electrodes to the generated DEP force field and we also described a model of motion of a microparticle in this force field.

Chapter 6

Control-oriented model of DEP force

In this chapter, we derive a control-oriented model of DEP force. Here, the term control-oriented is used in the usual meaning; the model can be less precise, but it has to be computationally tractable. The only problematic part of the relation for DEP force is the calculation of the potentials $\phi_i(x, y)$ from the boundary value problem (5.8). We obtain a control-oriented model by a modification of (5.8) that, at the cost of lower accuracy, enable us to solve the problem analytically by the technique of Green's functions.

At the outset, we briefly introduce the framework of Green's functions together with its limitations. Then we proceed separately for 2D and 3D case. Based on the limitations, we approximate the original boundary value problem to fit into the framework of Green's functions and solve it analytically in closed-form. Finally, we use the approximate analytical solution for calculation of DEP force and compare it with the numerical solution of the original unmodified boundary value problem.

6.1 Brief introduction to Green's functions

Green's functions bear the name after the famous mathematician George Green who in early nineteenth-century examined the solutions of *Poisson equation* $\nabla^2\phi = -f$ and Laplace equation $\nabla^2\phi = 0$ as a special case. He discovered that if he finds the solution of the Poisson equation with a point-charge source—that is, $\nabla^2 G = -4\pi\delta(\mathbf{r}-\mathbf{r}_0)$, where $\delta(\mathbf{r}-\mathbf{r}_0)$ is the Dirac delta function for the point-charge located at $\mathbf{r}_0$—and zero Dirichlet boundary conditions along the boundary S, then the solution of Laplace equation is given by the following integral [34]

$$\phi(\mathbf{r}) = \frac{1}{4\pi} \oiint_S h \nabla G \cdot \mathbf{n}\, dS, \tag{6.1}$$

where h is the value of ϕ on the boundary S imposed by Dirichlet boundary conditions and $\mathbf{n}$ is the normal vector to the boundary S. We denoted the solution G by letter G deliberately; the solution is a *Green's function*.

This result allows us to analytically find the solution of any Laplace equation with arbitrary Dirichlet boundary conditions by determining the Green's function for the problem at hand. Even though that might not seem to be a very useful result as we still have to solve a PDE, only a different one, it actually turns out that for some special geometries (shape of the domain) it is fairly simple to find the Green's function. Luckily, our boundary value problem from the previous chapter is easily modifiable to such a special geometry.

We should point out, that we merely scratched the surface of the powerful technique of Green's functions. This technique solves not only the Laplace equation but also more complex PDEs with different kind of boundary conditions.

6.2 Solving the boundary value problem in 2D

The boundary value problem (5.8) from the previous chapter cannot be directly solved by Green's functions; it does not possess any special geometry and it does not have bondary condition of just one kind. But if we slightly modify the problem and rectify these two shortcomings then Green's functions will provide us the solution. In this section we describe the necessary modifications and solve the modified problem.

6.2.1 Modifications

First of all, we notice that the domain of our problem (depicted in Fig. 5.1) can be extended to the whole positive half-plane. We move the side boundaries and the upper boundary to infinity. A half plane is a special geometry for which the Green's function is known.

What remains is to somehow get rid of the Neumann boundary conditions and specify only Dirichlet conditions along the boundary. As we go with the side boundaries and with the upper boundary to infinity, the potential has to decay to zero and thus we can replace the Neumann boundary conditions by zero Dirichlet conditions. The trouble is with the Neumann conditions on the bottom boundary between the electrodes. We assume that

$$\phi_{i\pm1}(x,y) = \phi_i(x \pm x_{\text{shift}}, y), \tag{6.2}$$

where x_{shift} is the distance between the centers of two adjacent electrodes. As a result, we have to find $\phi_i(x,y)$ for only one i, say $i=4$, and the remaining potential functions $\phi_i(x,y)$ for $i \neq 4$ can be determined simply by shifting $\phi_4(x,y)$. As a reminder, we assume that the ith electrode is set to 1 V and the remaining ones are grounded. It seems reasonable to assume that on the bottom boundary the potential decays from the ith electrode to the closest adjacent electrodes and that it does not propagate to the next gap between electrodes. Thus, we can replace all but the closest two Neumann boundary conditions on the bottom boundary by zero Dirichlet conditions. But how to get rid of the remaining two Neumann boundary conditions? We will approximate the decay of the potential between the ith electrode and its neighbors by a function and surrogate the Neumann boundary conditions by Dirichlet boundary conditions with this function.

One approach presented in [35] is to assume that the potential decays linearly. Specifically, assuming that the ith electrode is located at the origin, the Dirichlet boundary condition obtained by linear approximation is

$$h_{\text{lin}}(x) = \begin{cases} \frac{1}{100}(x+150) & x \in [-150,-50), \\ 1 & x \in [-50,50], \\ \frac{1}{100}(-x+150) & x \in (50,150], \\ 0 & \text{otherwise}, \end{cases} \tag{6.3}$$

where x is the position in microns.

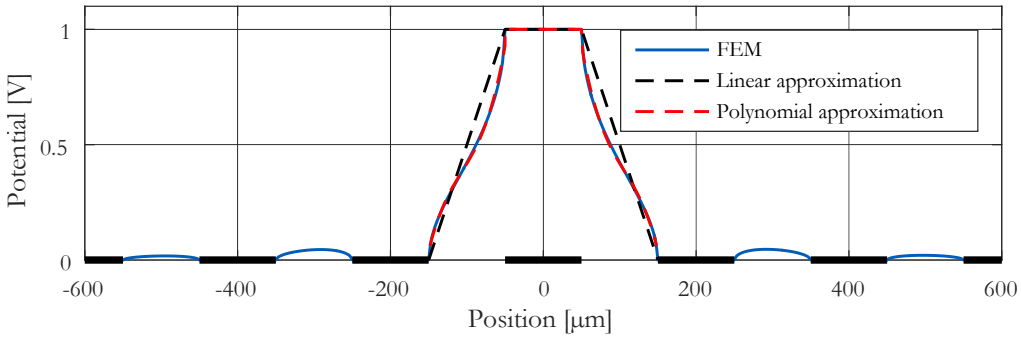


Figure 6.1: *Approximations of the contribution to the net potential from the forth electrode at the bottom boundary of the domain. The black thick segments represent electrodes.*

We use a different and more accurate approximation. At first, we solve the original boundary value problem numerically by *finite element method (FEM)* in COMSOL Multiphysics. Then, we take the values of the potential on the bottom boundary between the ith electrode and its left adjacent electrode and fit a polynomial $p(x)$ to these values. Specifically, we fitted a third-order polynomial $p(x)$ to the values because it is simple and yet accurately fits the values. The fitted polynomial is

$$p(x) = 1.3573 \cdot 10^{-6}x^3 + 4.3365 \cdot 10^{-4}x^2 + 5.1492 \cdot 10^{-2}x + 2.5882 \tag{6.4}$$

and the Dirichlet boundary condition for the polynomial approximation has the following form

$$h_{\text{poly}}(x) = \begin{cases} p(x) & x \in [-150, -50), \\ 1 & x \in [-50, 50], \\ p(-x) & x \in (50, 150], \\ 0 & \text{otherwise}, \end{cases} \tag{6.5}$$

where, again, x is the position in microns.

You can see both approximations in Fig. 6.1. Apparently, the polynomial approximation describe the FEM solution far better than the linear approximation. However, one should not overlook the humps in the gaps between the further electrodes which are completely omitted by both approximations. Furthermore, the figure shows an exact solution for $i = 4$, for the electrode which is in the middle of the electrode array, and if we chose an electrode closer to the edge, the bumps would be even larger. We have eight electrodes so let us take, for instance, $i = 1$ and $i = 2$. The FEM solutions for these cases are shown in Fig. 6.2. Obviously, the humps are getting larger and our approximate worse for the electrodes located closer to the border of the electrode array. In case of the second electrode, we still can consider the error to be acceptable but we cannot do so in the case of the first electrode. Nevertheless, we can change the design of the electrode array and add ground plates next to the border electrodes to force the potential to decay faster behind the border of the electrode array. As we have to work with the electrode array we currently have, we deliberately restrict ourselves to use only to the six electrodes in the middle and use the border electrodes as the ground plates. Based on the comparison shown in Fig. 6.1, we use $h_{\text{poly}}(x)$ as the approximation of the exact boundary conditions by a Dirichlet boundary condition.

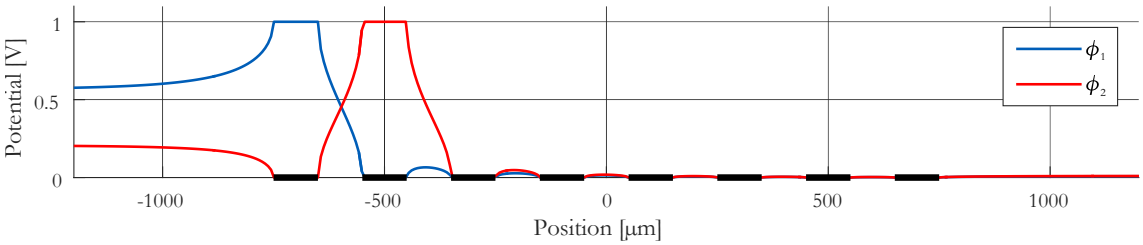


Figure 6.2: *FEM solutions for $\phi_1(x,y)$ and $\phi_2(x,y)$. The black thick segments represent electrodes.*

To summarize, the modified problem obtained by the extension of the domain to the whole upper half-plane and by the approximation of the mixed boundary conditions by Dirichlet boundary conditions is

$$\nabla^2 \phi_i(x,y) = 0, \tag{6.6}$$

with upper half-plane as the domain, zero Dirichlet boundary conditions in the infinity and bottom boundary condition given by $h_{\text{poly}}(x)$. Since the modified boundary problem has a special geometry for which the Green's function is known and the boundary conditions are only of one kind, we can find the solution by the framework of Green's functions.

6.2.2 Analytical solution

Solving the modified problem (6.6) is straightforward. The Green's function for the half-plane domain is known to be [36, p. 37]

$$G(\mathbf{r}',\mathbf{r}) = -\frac{1}{2\pi}\left(\ln|\mathbf{r}'-\mathbf{r}| - \ln|\mathbf{r}'-\mathbf{r}^*|\right), \tag{6.7}$$

where $\mathbf{r}' = (x',y')$, $\mathbf{r} = (x,y)$ and $\mathbf{r}^* = (x,-y)$. Substitution of (6.7) to (6.1) with the boundary condition $h_{\text{poly}}(x)$ gives us the solution

$$\phi_i(x,y) = \frac{y}{\pi}\int_{-\infty}^{\infty} \frac{h_{\text{poly}}(x')}{(x-x')^2+y^2}\,dx'. \tag{6.8}$$

To evaluate the integral in (6.8), we split the integral according to the cases in the definition of $h_{\text{poly}}(x)$. This way we obtain

$$\phi_i(x,y) = \phi_{i0}(x,y) + \phi_{i1}(x,y) + \phi_{i1}(-x,y), \tag{6.9}$$

where

$$\phi_{i0}(x,y) = \frac{y}{\pi}\int_{-50}^{50} \frac{1}{(x-x')^2+y^2}\,dx', \tag{6.10a}$$

$$\phi_{i1}(x,y) = \frac{y}{\pi}\int_{-150}^{-50} \frac{p(x')}{(x-x')^2+y^2}\,dx'. \tag{6.10b}$$

The integral in (6.10a) belongs to, let us say, standard integrals and is very easy to evaluate. However, things are getting a bit more complicated in the case of (6.10b) where it is not that trivial to evaluate the integral by hand. Fortunately, mathematical software like Wolfram

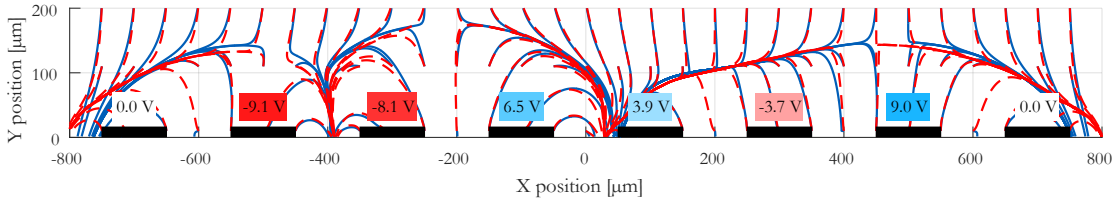


Figure 6.3: *Streamlines resulting from* $\mathbf{F}_{\mathrm{FEM}}(x, y)$ *(red dashed line) and* $\mathbf{F}_{\mathrm{anl}}(x, y)$ *(blue line).*

Mathematica, Maple or Matlab can do it for us with ease. We do not present the evaluated integrals here, because they are rather lengthy and it would not serve any purpose; everyone can calculate them in one of the mentioned software.

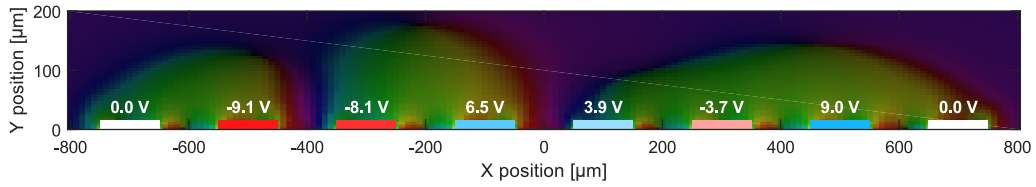
Again, we should summarize what we achieved in this section. So, we obtained the analytical closed-form solution $\phi_i(x, y)$ of the modified problem for one particular i, that is for the ith electrode. By assumption (6.2), potentials ϕ_j for $j \neq i$ are obtained simply by shifting $\phi_i(x, y)$. Since now we have the analytical solutions for potentials, we can differentiate them and obtain also the first and second derivatives necessary for calculation of the DEP force $\mathbf{F}_{\mathrm{dep}}$ in (5.7).

6.2.3 Comparison with numerical solution

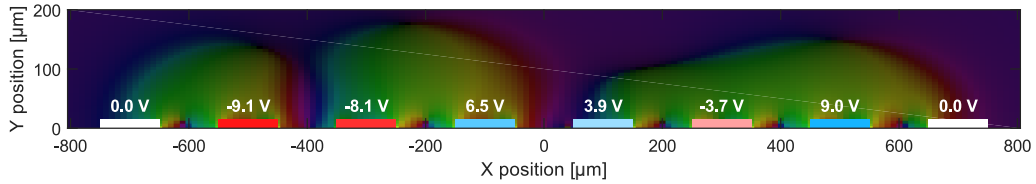
In this section, we examine how accurately the analytical solution of the modified (approximate) problem fits to a numerical solution of the original problem. As it will be obvious from the following chapter, what we are really after is the ability to determine the force acting upon a microparticle for any location above the electrode array and for any given set of potentials $\mathbf{u}$. In other words, what we are after is the ability to calculate F_{x} and F_{y} in the model (5.12). Thus, what we compare are not directly the potential fields but the derived force fields. Since $F_{\mathrm{x}} = F_{\mathrm{x}}^{\mathrm{dep}}$ and $F_{\mathrm{y}} = F_{\mathrm{y}}^{\mathrm{dep}} - F^{\mathrm{sed}}$, we determine a DEP force field based on a FEM solution of the original boundary value problem and a DEP force field based on the analytical solution of the modified boundary value problem, and subtract the sedimentation force $\mathbf{F}_{\mathrm{sed}}$ from the y component of both DEP force fields. Let us denote the two resulting force fields by $\mathbf{F}_{\mathrm{FEM}}(x, y)$ and $\mathbf{F}_{\mathrm{anl}}(x, y)$, respectively. We can evaluate how useful the approximate solution is by a comparison of these two force fields.

It is rather difficult to compare two two-dimensional vector fields where the entries in the vectors differ by orders of magnitude. Furthermore, the vector fields depend on the potentials $\mathbf{u}$. To give you at least an informative comparison, we randomly generated the vector $\mathbf{u}$ and for this particular choice we calculated the force fields $\mathbf{F}_{\mathrm{FEM}}(x, y)$ with $\mathbf{F}_{\mathrm{anl}}(x, y)$ in a grid of points ranging in x direction along the whole electrode array and in y direction from zero to 200 μm—the maximum levitation height of a microparticle. We compare these force fields in three ways. We compare streamlines, HSV images computed from the force fields and generated forces along the electrode array for a particular levitation height (constant y coordinate).

Fig. 6.3 shows streamlines resulting from $\mathbf{F}_{\mathrm{FEM}}(x, y)$ with $\mathbf{F}_{\mathrm{anl}}(x, y)$. A streamline shows the trajectory that a microparticle in the force field follows if we neglect its inertia and if it starts at the beginning of the streamline. Apparently, the streamlines computed from the approximate solution very accurately resemble the streamlines computed from the numerical solution.



(a) $\mathbf{F}_{\text{FEM}}(x, y)$



(b) $\mathbf{F}_{\text{anl}}(x, y)$

Figure 6.4: *A comparison of force fields* $\mathbf{F}_{\text{FEM}}(x, y)$ *and* $\mathbf{F}_{\text{anl}}(x, y)$ *by HSV images where the hue channel shows angle of the force and the visibility channel shows magnitude of the force.*

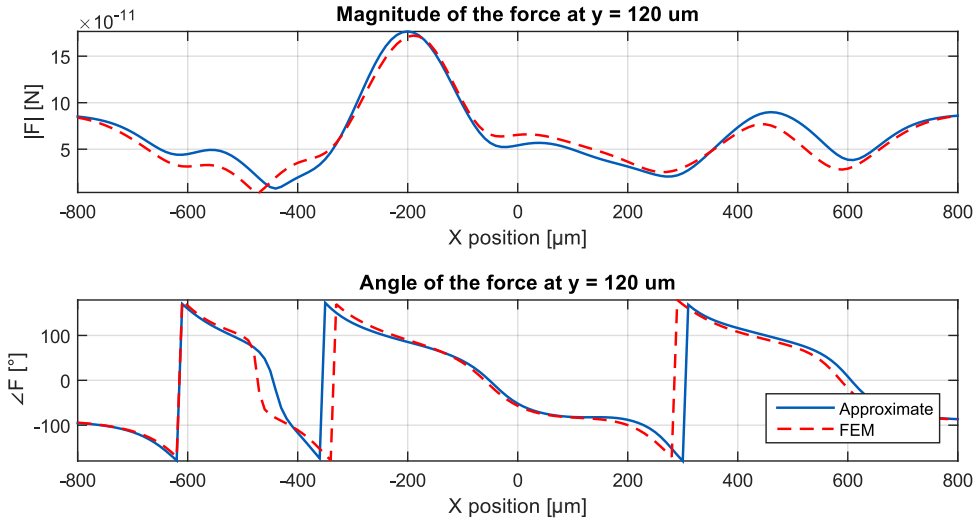


Figure 6.5: *A comparison of magnitude and angle of* $\mathbf{F}_{\text{FEM}}(x, y)$ *and* $\mathbf{F}_{\text{anl}}(x, y)$ *for* $y = 120$ μm.

Next, we generate HSV images from the force fields $\mathbf{F}_{\text{FEM}}(x, y)$ and $\mathbf{F}_{\text{anl}}(x, y)$. This comparison is inspired by *domain coloring* [37], which is a technique used for visualization of complex variables hence 2D vector fields. A HSV image has three channels: *hue*, *saturation* and *visibility*. We set angles of the vectors in the force field as the values of the hue channel, normalized logarithm of the magnitude of the vectors as the values of the visibility channel and we leave the saturation channel at a constant value. HSV images computed in this way for both force fields are shown in Fig. 6.4. Again, qualitatively the force field computed from the approximate solution is very similar to the one computed form the numerical solution.

Finally, we compare the force fields at one particular levitation height. Fig. 6.5 displays the magnitude and angle of $\mathbf{F}_{\text{FEM}}(x, y)$ and $\mathbf{F}_{\text{anl}}(x, y)$ at $y = 120$ μm. We chose this particular height because it is in the middle of the reasonable manipulation space. In the current hardware setup, it is not physically possible to manipulate a microparticle higher than 200 μm and below 40 μm the microparticles tend to fall down and stick to the bottom. Likewise in the previous two comparisons, the approximate solution closely follows the numerical solution.

6.3 Solving the boundary value problem in 3D

Let us now see how the approach from the previous section can be extended from 2D to 3D. We follow the same procedure—with some minor changes—as in the 2D case: we modify the original boundary value problem with mixed boundary conditions and without a special geometry to one with only Dirichlet boundary conditions and with a special geometry for which the Green's function is known. Then we analytically solve the modified problem and compare the result with a numerical solution of the original problem.

We did not describe in detail the 3D boundary value problem in the previous chapter because it was more convenient to derive the relation for DEP force for 2D and simply extend the result to 3D. Therefore, we very briefly describe the 3D boundary value problem here. Left part of Fig. 6.6 shows the top view on an electrode array and thus also on the domain where we seek the potential. Since the electrode array consists of four quadrants, we call it a four-quadrant electrode array. Similarly as in 2D case, the electrodes impose Dirichlet boundary conditions and the rest of the boundary imposes zero Neumann boundary conditions.

6.3.1 Modifications

Along similar lines as in the 2D case, we extend the domain in x and y directions to $\pm\infty$. That also means that we lengthen the electrodes to infinity. This way we change the domain to the positive half-space, for which the Green's function is known.

Again, we describe the net potential $\phi(x,y,z)$ as a sum of contributions $\phi_i(x,y,z)$ from individual electrodes (the indexes are shown in Fig. 6.6) and we do it so that the contributions are the same only shifted and/or rotated with respect to each other; mathematically speaking, for electrodes from the same quadrant, it holds that

$$\phi_{i+1}(x,y,z) = \phi_i(x + x_{\text{shift}}, y + y_{\text{shift}}, z), \tag{6.11}$$

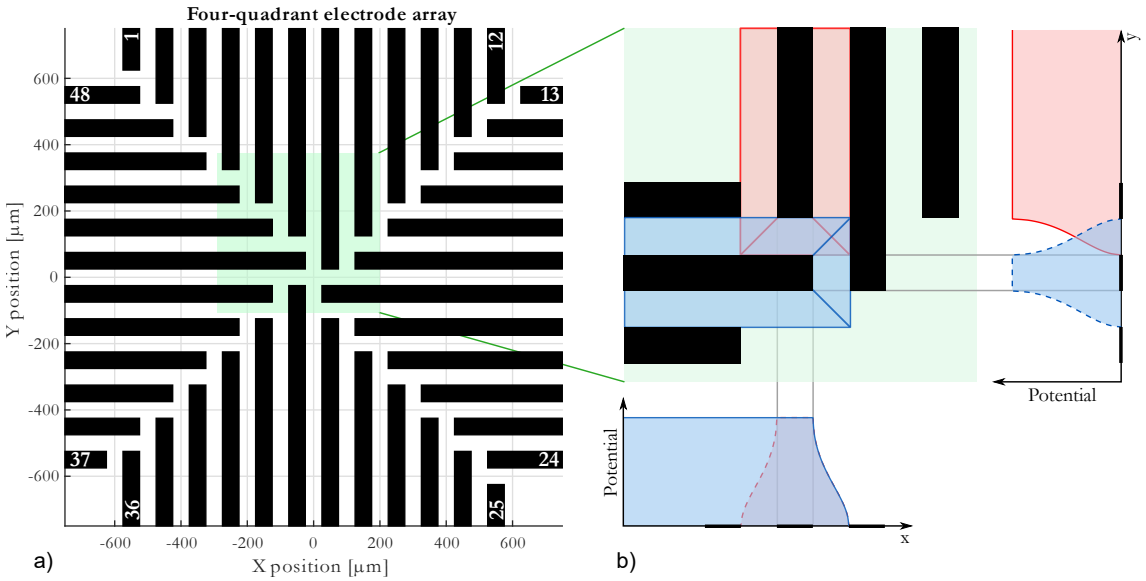


Figure 6.6: *Part (a) shows the four-quadrant electrode array and part (b) the proposed approximation of the exact boundary conditions by Dirichlet boundary conditions.*

where x_{shift} and y_{shift} are mutual distances between the electrodes in x and y direction, respectively. Then, we need to solve the boundary value problem for only one electrode, say $\phi_1(x,y,z)$. That is, all electrodes are grounded except the ith one where we set 1 V.

Next, we need to get rid of the Neumann boundary conditions. We would like to approximate the bottom boundary condition so that it imitates a FEM solution obtained by COMSOL Multiphysics. The FEM solution for $\phi_{44}(x,y,z)$ is displayed in Fig. 6.7 and the approximation we would like to use is shown in the right part of Fig. 6.6 as the red and the blue curved blocks. Nevertheless, as we will see shortly, the integral by which the analytical solution is obtained is not as simple as it was in the 2D case. Now, we are unable to evaluate the integral for anything else than for a constant boundary condition. Thus, instead of using a polynomial approximation, we build the desired shape of the potential from blocks. To be more specific, we initially approximate the boundary condition in the roughest possible way; we assume that the potential between the electrodes drops instantaneously to zero (see Fig. 6.8a). Then by "stretching", "squeezing" and "shifting" of this boundary condition (see Fig. 6.8b) we approximately build up the desired shape (see Fig. 6.8c).

We define the block boundary condition for one side infinitely long electrode as

$$h_0(x,y,d) = \begin{cases} 1 & x \le 0 \text{ and } y \in \left[-\frac{d}{2}, \frac{d}{2}\right], \\ 0 & \text{otherwise}, \end{cases} \tag{6.12}$$

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

$$h(x,y) = \sum_{i=1}^{N} a_i \, h_0\left(x - \frac{(b_i - 1)d}{2}, y, b_i \, d\right), \tag{6.13}$$

where a_i determines the height of the block and b_i is the stretching parameter in the meaning, that $b_i = 2$ stretches the block so that it is twice as wide as the electrode. Notice, that we assumed that the potential decays the same along x and y axes and thus the coefficients b_i determine not only the width but also the shift of the blocks along x axis.

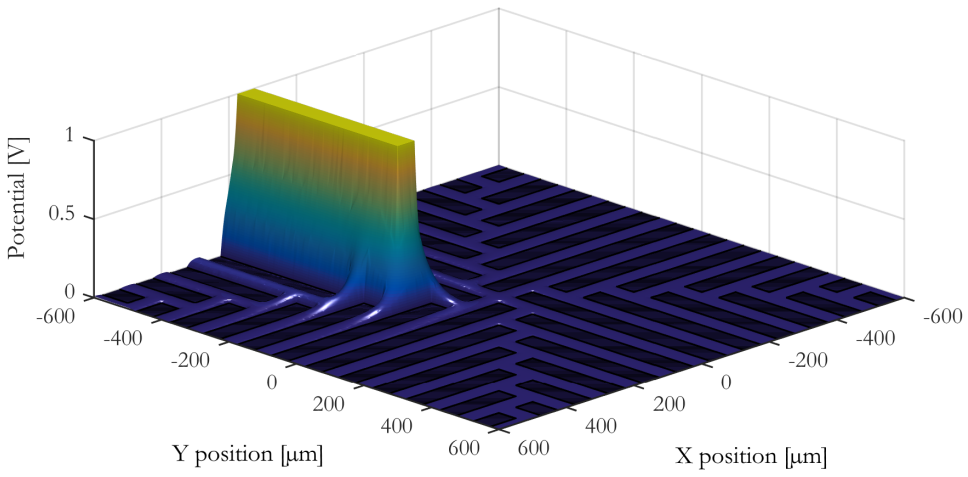
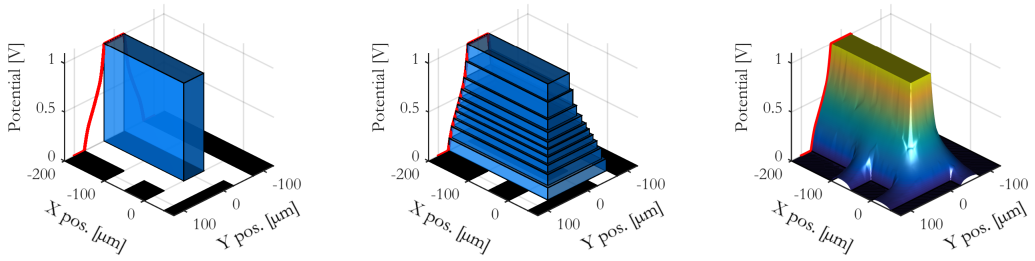


Figure 6.7: *FEM solution obtained by COMSOL Multiphysics for a* $\phi_{44}(x,y,z)$.



(a) (b) (c)

Figure 6.8: *A staircase approximation of the desired shape of the bottom boundary condition: (a) one block approximation, (b) multiple blocks approximation and (c) boundary condition obtained by FEM. Black rectangles represent electrodes.*

It remains to determine the coefficients a_i and b_i in (6.13). We formulate this task as the following optimization problem

$$\begin{aligned} &\min_{a_i,b_i} \ \|h(x_0,y)-\phi_{\mathrm{FEM}}(x_0,y,0)\|_2 \\ &\text{s.t.:} \ \sum_{i=1}^{N} a_i = 1, \\ &\qquad b_i \in [1,3], \end{aligned} \tag{6.14}$$

where $\phi_{\mathrm{FEM}}(x,y,z)$ is a FEM solution obtained by COMSOL Multiphysics. Since both a_i and b_i can be determined from a y-z cross-section of $\phi_{\mathrm{FEM}}(x,y,z)$, we fixed x to be a negative constant value x_0. The coefficients a_i have to sum up to one because only then the height of the piled up blocks will be one. We restrict the coefficients b_i to be from interval $[1,3]$ because then the blocks cannot be narrower than the electrode and they cannot interfere to other electrodes. Even though the optimization task is not convex, it still provides very good results when one provides a good initial guess of the coefficients a_i and b_i. Fig. 6.9 shows result of the optimization task (6.14). We use $N = 10$ and for the initial guess, we let b_i to grow linearly from 1 to 3 and set a_i to be proportional to the derivative of $\phi_{\mathrm{FEM}}(x_0,y,0)$.

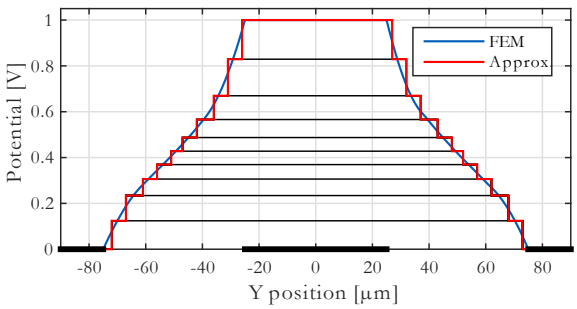


Figure 6.9: *A cross-section of the boundary potential $\phi_{\mathrm{FEM}}(x,y,0)$ and a staircase approximation $h(x,y)$.*

6.3.2 Analytical solution

Now, solving the 3D boundary value problem is easy. The Green's function for the 3D half-space domain is [36, p. 37]

$$G(\mathbf{r}',\mathbf{r}) = -\frac{1}{4\pi\|\mathbf{r}'-\mathbf{r}\|_2} + \frac{1}{4\pi\|\mathbf{r}'-\mathbf{r}^*\|_2}, \tag{6.15}$$

where $\mathbf{r}' = (x',y',z')$, $\mathbf{r} = (x,y,z)$ and $\mathbf{r}^* = (x,y,-z)$. Substitution of (6.15) to (6.1) with the boundary condition $h(x,y)$ gives us the solution

$$\phi_i(x,y,z) = \frac{z}{2\pi}\iint_{\mathbb{R}^2} \frac{h(x',y')}{((x-x')^2+(y-y')^2+z^2)^{1/2}}\,dx'dy'. \tag{6.16}$$

If we think it through, instead of using the boundary condition $h(x,y)$ composed of several blocks, we can use the one block boundary condition $h_0(x,y)$, calculate the contribution to the potential $\phi_i(x,y,z)$ and obtain $\phi_i(x,y,z)$ as a composition of the individual contributions. This is exactly how we will proceed. Substitution of $h_0(x,y)$ to the integral (6.16) gives us

$$\phi_{i0}(x,y,z,d) = \frac{z}{2\pi}\int_{-d/2}^{d/2}\int_{-\infty}^{0} \frac{1}{((x-x')^2+(y-y')^2+z^2)^{1/2}}\,dx'dy'. \tag{6.17}$$

Again, we do not present the evaluated integral here and just claim that it is possible to evaluate it in Wolfram Mathematica, Maple or Matlab.

We have a contribution of the one block approximation of width d to the net potential $\phi_i(x,y,z)$. The net potential $\phi_i(x,y,z)$ is then obtained by the same composition as we used for $h(x,y)$. Thus, the approximate analytical solution for $\phi_i(x,y,x)$ is

$$\phi_i(x,y,z) = \sum_{i=1}^{N} a_i\,\phi_{i0}\left(x - \frac{(b_i-1)d}{2}, y, z, b_i\,d\right). \tag{6.18}$$

6.3.3 Comparison with numerical solution

Similarly as in the 2D case, we do not compare directly the potentials because what we are really interested in are the force fields derived from the potentials. Bringing back the description of the force fields, we calculate a DEP force field based on the numerical solution of the original boundary value problem and a DEP force field based on the analytical solution of the modified (approximate) boundary value problem. Having the DEP force fields, we take into account also the sedimentation force and compute the force fields acting upon a microparticle.

Since it is a rather challenging task to somehow compare 3D force fields, we compare separately the components of the force fields. Fig. 6.10 shows a comparison of the force fields for a randomly generated set of potentials applied on the electrodes. It is quite surprising how well the force field computed from the approximate solution match the force field based on the FEM solution. The comparison clearly shows that all the approximations we made in the modified boundary value problem are justifiable and that the analytical solution of the modified problem is usable.

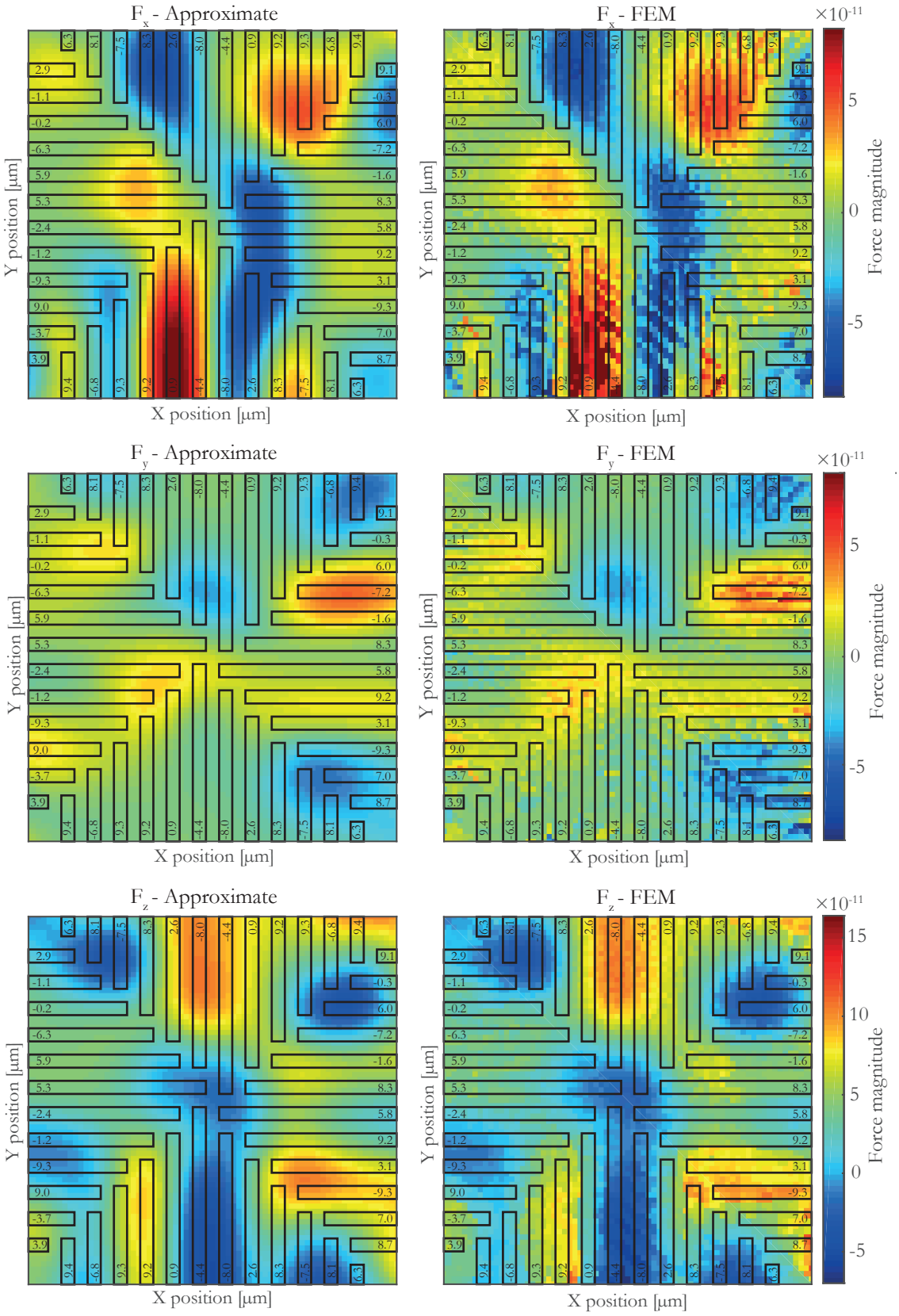


Figure 6.10: *A comparison of individual components of the force fields calculated from the approximate and FEM solution. The numerical values inside the electrodes represent set potentials in volts.*

6.4 Conclusion

In this chapter, we developed a control-oriented model of 2D and 3D DEP force fields. We took the exact description of the force field in the form of a boundary value problem and modified it so that it became solvable by the method of Green's function—a method that provides the analytical solution in the form of an integral over the boundary of the domain. The modification involved an extension of the domain and an approximation of mixed boundary conditions by Dirichlet conditions. We came up with a novel approach and approximated the boundary conditions so that they imitate values of the numerical solution of the unmodified boundary value problem. In 2D, the integral describing the solution was so simple that we could approximate the boundary conditions very accurately by polynomials. On the contrary, in 3D, we were able to evaluate the integral only for constant boundary conditions and thus we approximated the exact boundary conditions by a composition of blocks. Finally, we showed that, in both cases, the force fields obtained by the analytical solution of the modified boundary value problem closely resembles the force field obtained from a numerical solution of the original boundary value problem.

Part III

Control

Chapter 7

Control strategy

In the previous two parts, we developed a method for position estimation of microparticles and proposed a control-oriented model, so now it is time to focus our attention on the control of the position. In this chapter, we describe the control strategy for DEP that was originally introduced by Jiří Zemánek and Tomáš Michálek [28]. Then, we identify a weak spot of the control strategy, analyze it and discuss how to suppress its influence.

7.1 Currently used control scheme

The control strategy proposed by Jiří Zemánek and Tomáš Michálek [28, 29] is shown schematically in Fig. 7.1. We begin with the description of the left scheme in the figure. We assume that the microparticles are manipulated in 2D—meaning, we can control the position only in 2D—but the description is analogous for 3D. An estimate $\hat{\mathbf{x}} = [\hat{x}_1, \hat{y}_1, \ldots, \hat{x}_l, \hat{y}_l]^\intercal$ of the current position $\mathbf{x} = [x_1, y_1, \ldots, x_l, y_l]^\intercal$ of l microparticles is obtained by the algorithm described in Chapter 4. As in a classical control scheme, the measured (estimated) position is subtracted from a desired position $\mathbf{x}^{\text{des}} = [x_1^{\text{des}}, y_1^{\text{des}}, \ldots, x_l^{\text{des}}, y_l^{\text{des}}]^\intercal$ and the resulting error vector $\mathbf{e}$ is fed to a controller—in this case, a proportional one with gain K. The output of the controller is a vector of desired forces $\mathbf{F}^{\text{des}} = [F_{\text{x},1}^{\text{des}}, F_{\text{y},1}^{\text{des}}, \ldots, F_{\text{x},l}^{\text{des}}, F_{\text{y},l}^{\text{des}}]^\intercal$ acting upon the microparticles. The next block in the scheme is a crucial one and, at the same, it is the above mentioned weak spot. It takes the desired forces $\mathbf{F}^{\text{des}}$ as the input and calculates what potentials $\mathbf{u} = [u_1, \ldots, u_8]^\intercal$ have to be set on the electrodes in order to develop such forces. If this block works properly, it essentially linearize the system. This is shown in the right scheme in the figure. This fact allows us to pretend that we control the linear system represented by (5.12), where the inputs are the forces and the states are the positions of the microparticles. However, as we will see in a moment, the task of determining $\mathbf{u}$ for given forces $\mathbf{F}^{\text{des}}$ is not trivial. Far from it. The remaining part of the left scheme only shows that the potentials $\mathbf{u}$ are

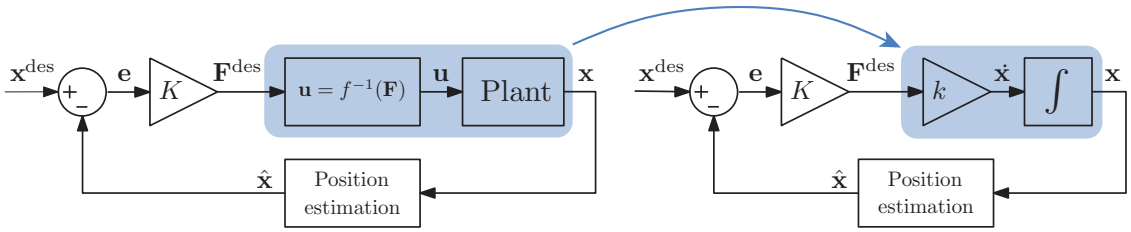


Figure 7.1: *A scheme of the used control strategy.*

fed to the plant. What happens in the block is that the potentials are set on the electrodes, DEP force field is generated and, hopefully, the microparticles move towards the desired positions.

7.2 Constrained inverse quadratic problem

As mentioned, the crucial part of the control strategy is the ability to determine such potentials $\mathbf{u}$ that the desired forces are developed. For reasons that will become obvious later on, we refer to this problem as *constrained inverse quadratic problem*. In this section, we formalize this problem and discuss some of its properties.

For simplicity, we start with only one microparticle that is manipulated in 2D. Then, the problem is to find such $\mathbf{u}$ that a desired force $\mathbf{F}^{\text{des}} = [F_{\text{x}}^{\text{des}}, F_{\text{y}}^{\text{des}}]^\intercal$ is generated. According to (5.7) and (5.11), we have

$$F_{\text{x}}^{\text{des}} = \mathbf{u}^\intercal \mathbf{A}_{\text{x}} \mathbf{u}, \tag{7.1}$$
$$F_{\text{y}}^{\text{des}} = \mathbf{u}^\intercal \mathbf{A}_{\text{y}} \mathbf{u} - F^{\text{sed}}. \tag{7.2}$$

Since the sedimentation force F^{sed} is constant we can simply add it to $F_{\text{y}}^{\text{des}}$ and obtain

$$F_a = \mathbf{u}^\intercal \mathbf{A}_a \mathbf{u}, \qquad a \in \{\text{x}, \text{y}\}, \tag{7.3}$$

where $F_{\text{x}} = F_{\text{x}}^{\text{des}}$ and $F_{\text{y}} = F_{\text{y}}^{\text{des}} + F^{\text{sed}}$. Thus for one particle, the task of determining $\mathbf{u}$ for a given $\mathbf{F}^{\text{des}}$ reduces to solving of (7.3). Analogously, for 3D manipulation and/or more microparticles, the problem also reduces to a system of equations with quadratic forms.

Vector $\mathbf{u}$ represents potentials applied on the electrodes and the potentials cannot be arbitrary; they are limited by physical capabilities of the control system. We reflect this fact by addition of a constraint on a norm of $\mathbf{u}$. The choice of the norm depends on the used control system. Since in our case the components of $\mathbf{u}$ are independent with respect to each other, the maximum norm is used.

Putting it altogether, the problem that we would like to be able to solve is

$$F_i = \mathbf{u}^\intercal \mathbf{A}_i \, \mathbf{u}, \qquad i = 1, \ldots, m, \tag{7.4}$$
$$\text{subject to } \|\mathbf{u}\|_\infty \leq 1,$$

where scalars F_i and generally indefinite matrices $\mathbf{A}_i \in \mathbb{R}^{n \times n}$ are given, and vector $\mathbf{u} \in \mathbb{R}^n$ is unknown. We call this problem *constrained inverse quadratic problem* as we seek a vector $\mathbf{u}$ that is mapped by the set of quadratic forms $\mathbf{u}^\intercal \mathbf{A}_i \, \mathbf{u}$ to a given vector $\mathbf{F} = [F_1, \ldots, F_m]^\intercal$.

7.2.1 Properties

We should spend some time commenting properties of the constrained inverse quadratic problem. First of all, matrices $\mathbf{A}_i$ are not arbitrary; they possess certain structure. Specifically, according to (5.6) and (5.14), they are given by a sum of k so-called *dyads*, where k is the dimension of the space where we can control the position of the microparticle. Therefore, matrices $\mathbf{A}_i$ are of maximum rank k. Furthermore, matrices $\mathbf{A}_i$ are not in general symmetric nor definite. But that does not matter because we can take only their symmetric part without any impact on the generated set by the quadratic forms.

As in every other mathematical problem where one seeks a solution, the question that should come to mind among the first ones is: *Does a solution exist?* It is not difficult to show that the system of equations (7.4) does not have to have a solution. For instance, if $\mathbf{u}^\intercal \mathbf{A}_i \mathbf{u}$ is a positive definite quadratic map and F_i is negative, then, obviously, ith equation of the system does not have a solution and so the whole system cannot have a solution.

Another inevitable question concerns the uniqueness of a solution. We can trivially prove that a solution does not have to be unique. Let $m = 1$, $F_1 = 1$ and $A_1 = 1$. Then the constrained inverse quadratic problem simplifies to the quadratic equation $u^2 = 1$ which obviously has two solutions $u = \pm 1$ satisfying the condition $\|u\|_\infty \leq 1$. Therefore, a solution of a constrained quadratic inverse problem does not have to exist and if it exists, it does not have to be unique.

7.2.2 Approximate solution

If there is no $\mathbf{u}$ solving (7.4), we can seek such vector $\mathbf{u}$ that the generated forces F_i are in some sense the closest to the desired ones. The closeness of the forces can be defined in various ways. Then, instead of solving (7.4), we can reformulate the problem as the following optimization task

$$\begin{aligned} &\min_{\mathbf{u}} \sum_{i=1}^{m} |F_i - \mathbf{u}^\intercal \mathbf{A}_i \, \mathbf{u}|, \\ &\text{subject to } \|\mathbf{u}\|_\infty \leq 1. \end{aligned} \tag{7.5}$$

However, this optimization task is apparently not convex hence difficult to solve.

Tomáš Michálek [29] used the method of *simulated annealing* [38] to solve this optimization problem and it turned out that this heuristic approach actually yields surprisingly good results. Despite this fact, it still only is a heuristic approach and does not provide any guaranties about the optimality of the solution it finds. Therefore, we will try to find a more insightful approach and make use of the structure of the problem.

We turn around the problem and instead of seeking a vector $\mathbf{u}$ satisfying (7.5), we seek in some sense the closest force to $\mathbf{F}$ for which the associated constrained inverse quadratic problem has a solution. Let us denote the closest force by $\tilde{\mathbf{F}}$. Thus, we optimize over the space of forces and not over the space of potentials $\mathbf{u}$. This is very beneficial because the forces lie in $\mathbb{R}^m$ which is usually much smaller than the space of potentials which lie in $\mathbb{R}^n$. However, if we find $\tilde{\mathbf{F}}$, we still have to solve the associated constrained inverse quadratic problem. Furthermore, we have to somehow be able to determine the set of all feasible forces under the constraint $\|\mathbf{u}\|_\infty \leq 1$ in order to be able to determine $\tilde{\mathbf{F}}$. As we will see, the capability of determining the set of all feasible forces is crucial and it will, as a side product, give us a hint how to find a solution of the associated constrained inverse quadratic problem.

We should also note what it means for a feasible force to be the closest possible to a given infeasible force. We can define the closeness differently. The first choice that probably would come everyone to mind is to use Euclidean norm of the difference of the two forces. However, we use a different metric. We consider the closest feasible force to be the largest (measured by Euclidean norm) feasible force with the same direction as the given infeasible force. The reason is that for the manipulation the direction of the force is more important than the magnitude.

Chapter 8

2D constrained inverse quadratic problem

In this section, we restrict our attention to solving a 2D constrained inverse quadratic problem. In other words, we seek a solution of the following system of equations

$$F_x = \mathbf{u}^\intercal \mathbf{A}_x \mathbf{u}, \tag{8.1a}$$

$$F_y = \mathbf{u}^\intercal \mathbf{A}_y \mathbf{u}, \tag{8.1b}$$

$$\text{subject to } \|\mathbf{u}\|_\infty \leq 1, \tag{8.1c}$$

where $\mathbf{u} \in \mathbb{R}^n$ and $\mathbf{A}_x, \mathbf{A}_y$ are symmetric matrices from $\mathbb{R}^{n\times n}$. In addition, we assume that $n \geq 3$.

Before we delve into the description of a proposed algorithm solving the problem, we define a so-called *numerical range*, which will serve us as a vehicle for solving of (8.1).

8.1 Numerical range

Definition 8.1 (Numerical range)**.** The numerical range (also known as the *field of values*) of a matrix $\mathbf{A} \in \mathbb{C}^{n\times n}$ is defined as

$$\mathcal{W}(\mathbf{A}) = \{\mathbf{u}^*\mathbf{A}\mathbf{u} \,|\, \mathbf{u} \in \mathbb{C}^n,\ \mathbf{u}^*\mathbf{u} = 1\}. \tag{8.2}$$

In words, the numerical range is the image of the unit n-dimensional Euclidean sphere under the continuous map $\mathbf{u} \mapsto \mathbf{u}^*\mathbf{A}\mathbf{u}$.

The numerical range possesses a wealth of useful properties. A comprehensive list of the properties can be, for instance, found in [39, Chapter 1] or [40]. Here, we state only the properties exploited in the remaining text:

(P_1) $\mathcal{W}(\mathbf{A})$ is **compact** and **convex**.

(P_2) $\mathcal{W}(a\mathbf{A} + b\mathbf{I}_n) = a\,\mathcal{W}(\mathbf{A}) + b$, for all $\mathbf{A} \in \mathbb{C}^{n\times n}$ and $a, b \in \mathbb{C}$.

(P_3) $\mathcal{W}(\mathbf{A}) = \mathcal{W}(\mathbf{U}^*\mathbf{A}\mathbf{U})$, for any unitary matrix $\mathbf{U}$.

(P_4) If a matrix $\mathbf{A}$ is **normal** ($\mathbf{A}^*\mathbf{A} = \mathbf{A}\mathbf{A}^*$) then $\mathcal{W}(\mathbf{A})$ is equal to the convex hull of the spectrum of $\mathbf{A}$.

(P_5) $\mathcal{W}(\mathbf{H}(\mathbf{A})) = \mathrm{Re}\,\mathcal{W}(\mathbf{A})$, where Re is used to denote the real part of the numerical range.

How can the numerical range help us to solve our system of equations? At first glance, the only similarity with the original problem is that there is a quadratic form involved in the definition of the numerical range. But the quadratic form is defined above complex numbers. In addition, according to the condition $\mathbf{u}^*\mathbf{u} = 1$, the domain of the numerical range is a unit hypersphere and not the interior of a hypercube defined by $\|\mathbf{u}\|_\infty \leq 1$ in the original problem. Nevertheless, as we will see in the rest of this section, with some modifications, the numerical range is in fact particularly well suited for characterization of all feasible forces (F_x, F_y).

8.1.1 Drawing of the numerical range

From the previous discussion, it is evident that we need a way to determine the numerical range. Although two very efficient and sophisticated algorithms approximating the numerical range were introduced recently [41, 42], we will describe a simplified version of the algorithm introduced by Johnson in late seventies [43]. All mentioned algorithms approximate the numerical range $\mathscr{W}(\mathbf{A})$ by determining k of its boundary points f_k and taking convex hull of these points, but only the Johnson's algorithm also provides *generating vectors* $\mathbf{u}_k$ for all these points, that is $f_k = \mathbf{u}_k^*\mathbf{A}\,\mathbf{u}_k$. The importance of this property will become obvious in Section 8.2.

The original version of Johnson's algorithm allows us to control the precision of the approximation by running the algorithm iteratively till the desired precision is achieved. This comes at the cost of losing the ability to control the time needed for the calculation. Since this work is aimed for deployment in real-time control, we will describe a simplified version of Johnson's algorithm that does not enable us to control the precision, but it has a constant time of execution.

The algorithm is based on the following proposition:

Proposition 8.1 (Johnson [43]). *For any $\mathbf{A} \in \mathbb{C}^{n\times n}$ it holds*

$$\max_{z\in\mathscr{W}(\mathbf{A})} \mathrm{Re}(z) = \lambda_1\left(\mathbf{H}(\mathbf{A})\right), \tag{8.3}$$

$$\mathrm{Re}\left(\mathbf{u}_1^*\mathbf{A}\,\mathbf{u}_1\right) = \max_{z\in\mathscr{W}(\mathbf{A})} \mathrm{Re}(z) \tag{8.4}$$

and

$$e^{i\theta}\,\mathscr{W}(\mathbf{A}) = \mathscr{W}(e^{i\theta}\mathbf{A}), \tag{8.5}$$

where $\mathbf{u}_1$ is a unit eigenvector associated with the largest eigenvalue λ_1 of matrix $\mathbf{H}$.

Although the proof can be found in [43], we show it here since it gives some insight that we build upon later.

Proof. Observation (8.3) follows directly from properties (P_4) and (P_5). Statement (8.5) readily follows from property (P_2).

Relation (8.4) is also straightforward to prove. We begin with the very definition of eigenvalues and eigenvectors and the fact that we have chosen an eigenvector of unit length. The defining relation for eigenvalues and eigenvectors is

$$\mathbf{H}(\mathbf{A})\,\mathbf{u}_1 = \lambda_1\left(\mathbf{H}(\mathbf{A})\right)\mathbf{u}_1. \tag{8.6}$$

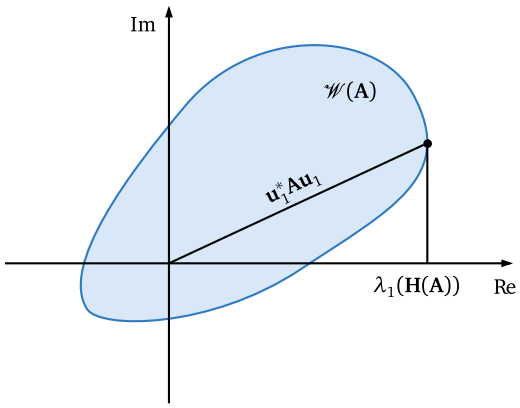


Figure 8.1: *A visualization of relations given by equations (8.3) and (8.4).*

If we multiply this relation by $\mathbf{u}_1^*$ from left, we get

$$\mathbf{u}_1^*\mathbf{H}(\mathbf{A})\mathbf{u}_1 = \lambda_1(\mathbf{H}(\mathbf{A}))\,\mathbf{u}_1^*\mathbf{u}_1 = \lambda_1(\mathbf{H}(\mathbf{A}))\,. \tag{8.7}$$

The remaining step is to show that $\mathrm{Re}\left(\mathbf{u}_1^*\mathbf{A}\mathbf{u}_1\right) = \mathbf{u}_1^*\mathbf{H}(\mathbf{A})\mathbf{u}_1$. That follows from

$$\mathrm{Re}\left(\mathbf{u}_1^*\mathbf{A}\mathbf{u}_1\right) = \frac{1}{2}\mathbf{u}_1^*\mathbf{A}\mathbf{u}_1 + \frac{1}{2}\mathbf{u}_1^*\mathbf{A}^*\mathbf{u}_1 = \mathbf{u}_1^*\left(\frac{1}{2}(\mathbf{A}+\mathbf{A}^*)\right)\mathbf{u}_1 = \mathbf{u}_1^*\mathbf{H}(\mathbf{A})\mathbf{u}_1. \tag{8.8}$$

Thus the relation (8.4) holds. □

Observations (8.3) and (8.4) have a nice geometric interpretation shown in Fig. 8.1. Eigenvalue $\lambda_1(\mathbf{A})$ is the real part of the rightmost point of $\mathscr{W}(\mathbf{A})$ and the eigenvector $\mathbf{u}_1$ is a generating vector of the rightmost point.

Now, we have a recipe how to determine the rightmost boundary point of $\mathscr{W}(\mathbf{A})$. We take the Hermitian part of a matrix $\mathbf{A}$, calculate its largest eigenvalue $\lambda_1(\mathbf{H}(\mathbf{A}))$, find an associated unit eigenvector $\mathbf{u}_1$ and map it by the quadratic form $\mathbf{u}_1^*\mathbf{A}\mathbf{u}_1$.

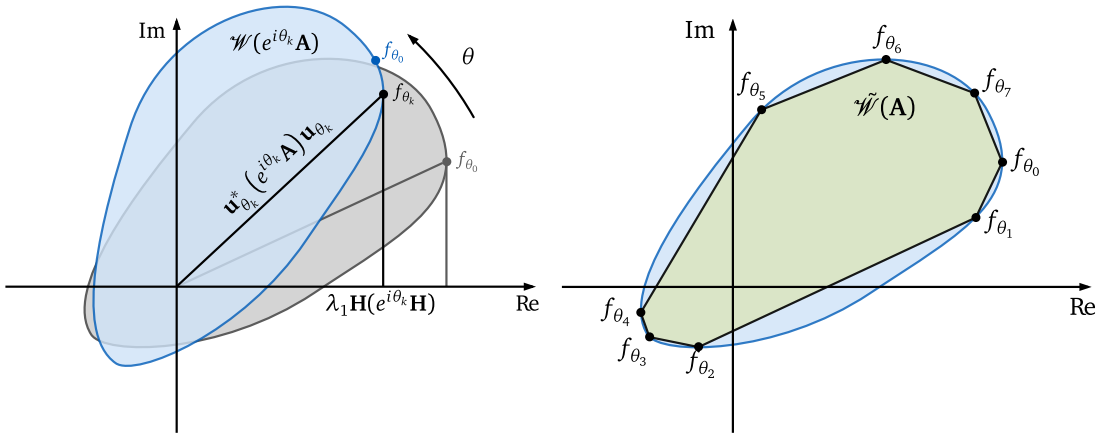
Since property (P_1) states that $\mathscr{W}(\mathbf{A})$ is convex, we can approximate $\mathscr{W}(\mathbf{A})$ by taking convex hull of several of its boundary points. So far, we know how to determine one boundary point. To find more boundary points, we rotate $\mathscr{W}(\mathbf{A})$ and use the same procedure to calculate the rightmost point of the rotated $\mathscr{W}(\mathbf{A})$. This idea is illustrated in Fig. 8.2.

The numerical range is a set in a complex plane, thus we can rotate it by multiplication with $e^{i\theta}$, where θ is the angle of rotation. As the observation (8.5) suggest, the rotation of the numerical range is carried out by multiplication of matrix $\mathbf{A}$ with $e^{i\theta}$. Therefore, to calculate boundary points of $\mathscr{W}(\mathbf{A})$, we divide interval $[0, 2\pi)$ to N equidistant angles:

$$\theta_k = k\frac{2\pi}{N}, \qquad k = 0, \ldots, N-1, \tag{8.9}$$

and for each angle θ_k, we find a unit eigenvector $\mathbf{u}_{\theta_k} \in \mathbf{E}_1\left(\mathbf{H}(e^{i\theta_k}\mathbf{A})\right)$. We know that the rightmost boundary point of the rotated numerical range $e^{i\theta}\mathscr{W}(\mathbf{A})$ is given by $\mathbf{u}_{\theta_k}^*\left(e^{i\theta_k}\mathbf{A}\right)\mathbf{u}_{\theta_k}$. In order to transform it to a boundary point of the original $\mathscr{W}(\mathbf{A})$, we have to rotate the point by angle $-\theta_k$. It follows that the boundary point of $\mathscr{W}(\mathbf{A})$ associated with angle θ_k is

$$f_{\theta_k} = e^{-i\theta_k}\left(\mathbf{u}_{\theta_k}^*\left(e^{i\theta_k}\mathbf{A}\right)\mathbf{u}_{\theta_k}\right) = \mathbf{u}_{\theta_k}^*\left(e^{-i\theta_k}e^{i\theta_k}\mathbf{A}\right)\mathbf{u}_{\theta_k} = \mathbf{u}_{\theta_k}^*\mathbf{A}\,\mathbf{u}_{\theta_k}. \tag{8.10}$$



(a) *Rotation* (b) *Inner approximation*

Figure 8.2: *Approximation of the numerical range: (a) computation of boundary points f_{θ_k} of $\mathscr{W}(\mathbf{A})$ by rotation and (b) inner approximation of $\mathscr{W}(\mathbf{A})$ by the convex hull of points f_{θ_k}.*

An inner approximation of the numerical range $\mathscr{W}(\mathbf{A})$ is then given by

$$\tilde{\mathscr{W}}(\mathbf{A}) = \mathrm{co}\big(\{f_{\theta_0}, \ldots, f_{\theta_{n-1}}\}\big). \tag{8.11}$$

Remark 1. It is important to note, that there is an analogous relation to (8.3) for the minimum eigenvalue $\lambda_n(.)$:

$$\min_{z \in \mathscr{W}(\mathbf{A})} \mathrm{Re}(z) = \lambda_n\left(\mathbf{H}(\mathbf{A})\right). \tag{8.12}$$

Therefore the algorithm can be as well based on the calculation of the leftmost boundary points. But more importantly, we can rotate the numerical range by angles from interval $[0, \pi)$ and for each angle calculate the leftmost and the rightmost boundary point. If an algorithm providing all eigenvalues at once is used then this way, we save half of the calculations of eigenvalues and thus make the approximation more computationally efficient.

Remark 2. Another way how to improve the efficiency of the algorithm is to use *Lanczos algorithm*. Lanczos algorithm is an iterative algorithm estimating a given number of the largest eigenvalues and the associated eigenvectors. The algorithm is very efficient for large and sparse matrices and that is its main area of use. Nevertheless, in our case it could be beneficial to use it even for smaller matrices. Lanczos algorithm needs a starting point, an initial guess of the dominant eigenvector, and since $\mathbf{u}_\theta$ is a continuous function of θ, there is a good chance that the eigenvector $\mathbf{u}_{\theta_k}$ lies close to $\mathbf{u}_{\theta_{k+1}}$ hence it will serves well as the starting point for estimation of $\mathbf{u}_{\theta_{k+1}}$ and make the algorithm converge faster. This idea is discussed in [44].

8.2 Set of all feasible forces

Here, we discuss how to utilize the numerical range to describe an approximation of the set of all feasible forces (F_x, F_y) of problem (8.1). As a remainder, by feasible forces we mean forces for which system of equations (8.1) has a solution. We denote the set by $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$, where the subscript emphasizes that the set represent the set of all feasible forces under the constraint $\|\mathbf{u}\|_\infty \leq 1$.

First of all, we combine real matrices $\mathbf{A}_x$ and $\mathbf{A}_y$ into a new complex matrix $\mathbf{A}_{xy} = \mathbf{A}_x + i\mathbf{A}_y$. We emphasize that $\mathbf{A}_{xy}$ obviously remains symmetric. This enables us to merge (8.1a) and (8.1b) to only one equation:

$$F_x + i\,F_y = \mathbf{u}^\intercal\,\mathbf{A}_{xy}\,\mathbf{u}. \tag{8.13}$$

Now, if we respectively identify the real and imaginary axis with F_x and F_y, the numerical range $\mathscr{W}(\mathbf{A}_{xy})$ coincides with the set of all feasible forces if we consider control signals from the set $\{\mathbf{u} \in \mathbb{C}^n \,|\, \mathbf{u}^*\mathbf{u} = 1\}$, which we do not.

Next, we deal with fact that the numerical range is defined for complex vectors $\mathbf{u}$ whereas we are restricted to real control signals $\mathbf{u}$. Of course, we can define a real numerical range as follows:

Definition 8.2 (Real numerical range)**.** For any $\mathbf{A} \in \mathbb{C}^{n\times n}$ we define the *real numerical range* as the set

$$\mathscr{R}(\mathbf{A}) = \{\mathbf{u}^\intercal\,\mathbf{A}\,\mathbf{u} \,|\, \mathbf{u} \in \mathbb{R}^n,\, \mathbf{u}^\intercal\mathbf{u} = 1\}. \tag{8.14}$$

Then the set of all feasible forces under the constraint $\mathbf{u}^\intercal\mathbf{u} = 1$ equals to $\mathscr{R}(\mathbf{A}_{xy})$. But if we do so, we lose all the mathematical results derived for the complex numerical range $\mathscr{W}(\mathbf{A})$, which is the majority. Luckily, as Brickman [45, Corollary on p. 65] showed, the real and complex numerical range, under certain condition, coincide.

Proposition 8.2 (Brickman [45])**.** *Let* $\mathbf{C} \in \mathbb{C}^{n\times n}$ *and* $n \geq 3$*. Then*

$$\mathscr{R}(\mathbf{C}) = \mathscr{W}\left(\frac{1}{2}\mathbf{C} + \frac{1}{2}\mathbf{C}^\intercal\right). \tag{8.15}$$

Since matrix $\mathbf{A}_{xy}$ is symmetric, according to the proposition, we can write $\mathscr{R}(\mathbf{A}_{xy}) = \mathscr{W}(\mathbf{A}_{xy})$ and thus the numerical ranges coincide.

Now, we deal with the different constraints on $\mathbf{u}$ in the definition of the numerical range (8.2) and in the constrained inverse quadratic problem in (8.1). We would like to bend $\mathscr{W}(\mathbf{A}_{xy})$ so that it equals $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$ or at least approximates it, but $\mathscr{W}(\mathbf{A}_{xy})$ is defined for $\mathbf{u}^*\mathbf{u} = 1$ and $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$ is defined for $\|\mathbf{u}\|_\infty \leq 1$. We can extend $\mathscr{W}(\mathbf{A}_{xy})$ to the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ containing the forces generated by $\|\mathbf{u}\|_2 \leq 1$. We observe that if we take an arbitrary vector $\mathbf{u}$ satisfying condition $\mathbf{u}^*\mathbf{u} = 1$, scale it down by $\frac{1}{2}$ and map it by a quadratic form $\mathbf{u} \mapsto \mathbf{u}^*\,\mathbf{A}\,\mathbf{u}$, we get

$$\left(\frac{1}{2}\mathbf{u}\right)^*\mathbf{A}\left(\frac{1}{2}\mathbf{u}\right) = \frac{1}{4}\mathbf{u}^*\mathbf{A}\mathbf{u}. \tag{8.16}$$

From this observation it follows that if we define a modified numerical range as

$$\mathscr{W}'(\mathbf{A}, c) = \{\mathbf{u}^*\mathbf{A}\mathbf{u} \,|\, \mathbf{u} \in \mathbb{C}^n,\, \mathbf{u}^*\mathbf{u} = c\}, \tag{8.17}$$

then we can write

$$\mathscr{W}'(\mathbf{A}, c) = c\,\mathscr{W}(\mathbf{A}). \tag{8.18}$$

In words, if we, roughly speaking, scale down the equality condition in the definition of the numerical range, we just scale down the numerical range itself. Thus the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ is easily obtained by the convex hull of $\mathscr{W}(\mathbf{A})$ and the origin of the complex plane:

$$\Omega_2(\mathbf{A}_x, \mathbf{A}_y) = \{\mathbf{u}^\intercal\mathbf{A}_{xy}\mathbf{u} \,|\, \mathbf{u} \in \mathbb{R}^n,\, \|\mathbf{u}\|_2 \leq 1\} = \mathrm{co}\big(\{\mathscr{W}(\mathbf{A}_{xy}), 0\}\big). \tag{8.19}$$

Since the numerical range can be approximated by (8.11), we can approximate $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ by

$$\tilde{\Omega}_2(\mathbf{A}_x, \mathbf{A}_y) = \mathrm{co}\big(\{f_{\theta_0}, \ldots, f_{\theta_{n-1}}, 0\}\big). \tag{8.20}$$

Now it is obvious that the notation of the boundary points f_{θ_k} was not chosen arbitrarily; we denote the boundary points by f_{θ_k} to emphasize that they lie in the same space as the forces $\mathbf{F} = [F_x, F_y]^\intercal$.

To summarize, we found a way how to describe the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ of all feasible forces by real control signals constrained by condition $\|\mathbf{u}\|_2 \leq 1$. This constraint is stricter than the original constraint $\|\mathbf{u}\|_\infty \leq 1$ hence $\tilde{\Omega}_2(\mathbf{A}_x, \mathbf{A}_y) \subset \Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$. In other words, set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ does not contain all feasible forces under the original constraint. As a result, if we classify all the forces outside $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ as infeasible, we can discard some forces that are actually physically feasible.

8.3 Existence of a solution

Here, we make an important observation and that is the fact that all the forces feasible by $\|\mathbf{u}\|_\infty \leq 1$ (that is the set $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$) lie in the cone given by the set of all feasible forces by $\|\mathbf{u}\|_2 \leq 1$ (that is the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$).

Proposition 8.3. *Let $\mathbf{A}_x, \mathbf{A}_y \in \mathbb{R}^{n\times n}$ be symmetric matrices. Then*

$$\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y) \subset \{z \in \mathbb{C} \,|\, z = \alpha f, \alpha \in \mathbb{R}, \alpha > 0, f \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)\}. \tag{8.21}$$

Proof. Let us assume that a force f from $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$ does not lie in the cone given by $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$. Let $\mathbf{u}$ denote a generating vector of f, that is $f = \mathbf{u}^\intercal \mathbf{A}_{xy} \mathbf{u}$. Now, we normalize the vector $\mathbf{u}$ hence we get $\mathbf{u}' = \mathbf{u}/\|\mathbf{u}\|_2 = c\mathbf{u}$. Then $f' = \mathbf{u}'^\intercal \mathbf{A}_{xy} \mathbf{u}' = c^2 \left(\mathbf{u}^\intercal \mathbf{A}_{xy} \mathbf{u}\right) = c^2 f$. We see that $f' \in \mathscr{W}(\mathbf{A}_{xy}) \subset \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$, because it is generated by the unit vector $\mathbf{u}'$. But since $f' = c^2 f$, f has to lie in the same cone as f'. That is a contradiction. □

This is a very important observation because it tells us what we lose if we restrict ourselves to $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ instead of the original $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$. The only thing that we lose is the magnitudes of feasible forces, but for every force from $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$ there is a force in $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ with the same direction—possibly with smaller magnitude.

We can put the same observation in a bit different way and relate it to a solution of the original constrained inverse quadratic problem (8.1). We are able to find the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$. Let us further assume that we are also able to find a generating vector $\mathbf{u}$ for every $\mathbf{F} \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$. We use the same notion as in (8.1) and simply use $\mathbf{F}$ for the desired force for which we seek a generating vector $\mathbf{u}$. Then we have three cases:

- If $\mathbf{F} \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ then trivially, (8.1) has a solution and we are able to find it.
- If $\mathbf{F} \notin \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ but there is a $\tilde{\mathbf{F}} \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ satisfying $\mathbf{F} = c\tilde{\mathbf{F}}$ for a scalar $c > 1$, then (8.1) might have a solution, but we are not able to find it.
- If $\mathbf{F} \notin \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ and there is no $\tilde{\mathbf{F}} \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ satisfying $\mathbf{F} = c\tilde{\mathbf{F}}$ for a scalar $c > 1$, then (8.1) does not have a solution.

8.4 Solving the problem

At this stage, we know how to determine whether a force $\mathbf{F} = [F_x, F_y]^\intercal$ is feasible by a generating vector satisfying $\|\mathbf{u}\|_2 \leq 1$ or not. It either lies in the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ and it is feasible or it does not lie in the set and it is not feasible. If a force is not feasible we can take the maximum feasible force in the same direction and seek a solution for this force (see Fig. 8.3). Either way, it is assured that a solution of (8.1) exists. It remains to find a way how to determine a generating vector $\mathbf{u}$ for any force from $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$.

We base our algorithm on the observation that if we find a boundary point $f_0 \in \partial\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ positively collinear (the vectors point at the same direction) with a desired force $f^{\text{des}} = F_x + i\,F_y$, we can simply scale down a generating vector $\mathbf{u}_0$ of f_0 to obtain a generating vector $\mathbf{u}$ of f^{des}, and the vector $\mathbf{u}_0$ is always known because of the way we determine boundary points (see (8.10)). This observation is proven by the following equation

$$f^{\text{des}} = c^2 f_0 = c^2 \left(\mathbf{u}_0^\intercal \mathbf{A}_{xy}\, \mathbf{u}_0\right) = (c\,\mathbf{u}_0)^\intercal \mathbf{A}_{xy}\,(c\,\mathbf{u}_0) = \mathbf{u}^\intercal \mathbf{A}_{xy}\,\mathbf{u}, \tag{8.22}$$

which also shows that the scaling factor c is $\sqrt{\|f^{\text{des}}\|_2 / \|f_0\|_2}$ and a generating vector $\mathbf{u}$ of f^{des} is $c\mathbf{u}_0$.

The requirement that the point f_0 has to be a boundary point is important because otherwise it could happen that even for $f^{\text{des}} \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ we would have to scale up $\mathbf{u}_0$ by $c > 1$ and thus the constraint $\|\mathbf{u}\|_\infty \leq 1$ (see (8.1c)) could be violated. If $f_0 \in \partial\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$, this cannot happen.

Recall that our ultimate goal is to find $\mathbf{u}$ generating f^{des} and satisfying condition $\|\mathbf{u}\|_\infty \leq 1$ hence we do not have to stop after learning that the problem is infeasible for the Euclidean norm. We can not only scale $\mathbf{u}_0$ down, we can also scale it up till $\|c\mathbf{u}_0\|_\infty \leq 1$. Therefore, in order to satisfy the original condition, we determine the scaling factor c by the following relation:

$$c = \min\left\{ \frac{1}{\|\mathbf{u}_0\|_\infty}, \sqrt{\frac{\|f^{\text{des}}\|_2}{\|\mathbf{u}_0^\intercal \mathbf{A}_{xy}\,\mathbf{u}_0\|_2}} \right\}. \tag{8.23}$$

We can divide values of c into three cases:

- If $c \leq 1$, then $f^{\text{des}} \in \Omega_2(\mathbf{A}_{xy})$ and we can scale down $\mathbf{u}_0$ to obtain a generating vector $\mathbf{u}$ of f^{des}. We found an exact solution of (8.1).

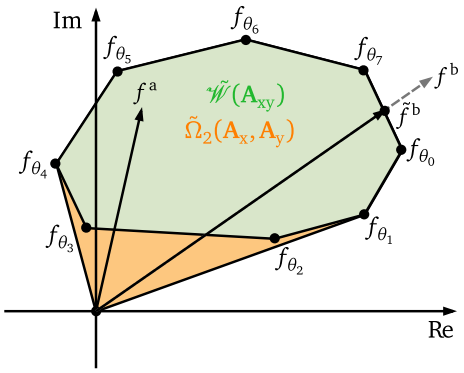


Figure 8.3: *Approximation of the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ of all feasible forces under the constraint $\|\mathbf{u}\|_2 \leq 1$ with a feasible force f^{a} and an unfeasible force f^{b}.*

- If $1 < c < 1/\|\mathbf{u}_0\|_\infty$, then $f^{\text{des}} \notin \Omega_2(\mathbf{A}_{\text{xy}})$, but we can scale up $\mathbf{u}_0$ to obtain a generating vector $\mathbf{u}$ of f^{des} without violating the constraint $\|\mathbf{u}\|_\infty \leq 1$. We found an exact solution of (8.1).

- If $c = 1/\|\mathbf{u}_0\|_\infty$, then the value of c is saturated; we cannot scale up $\mathbf{u}_0$ to obtain a generating vector $\mathbf{u}$ of f^{des} without violating the constraint $\|\mathbf{u}\|_\infty \leq 1$. We did not find an exact solution of (8.1).

Now, the problem is how to find the boundary point positively collinear with a given force f^{des}. Let us denote the point by $\tilde{f}^{\text{des}}$. Then we seek a point $\tilde{f}^{\text{des}} \in \partial\Omega_2(\mathbf{A}_{\text{x}}, \mathbf{A}_{\text{y}})$ satisfying

$$\tilde{f}^{\text{des}} = k f^{\text{des}}, \quad k \in \mathbb{R}, k > 0 \tag{8.24}$$

At first, we approximate $\Omega_2(\mathbf{A}_{\text{x}}, \mathbf{A}_{\text{y}})$ by (8.20). Thus, we obtain a set of points $f_{\theta_k} \in \partial\mathcal{W}(\mathbf{A}_{\text{xy}})$. Since we are interested only in boundary points of $\Omega_2(\mathbf{A}_{\text{x}}, \mathbf{A}_{\text{y}})$, we discard all points f_{θ_k} that lie in the interior of $\Omega_2(\mathbf{A}_{\text{x}}, \mathbf{A}_{\text{y}})$. For instance, in the situation illustrated in Fig. 8.3 we discard f_{θ_2} and f_{θ_3}.

Next, we check whether f^{des} is positively collinear—within some tolerance—with one of the points f_{θ_k}. If there is such a point, then we are done because we found $\tilde{f}^{\text{des}}$. If there is no such a point, we find a couple of boundary points f_{θ_i} and f_{θ_j} for which it holds

$$\text{Arg}(f_{\theta_i}) < \text{Arg}(f^{\text{des}}) < \text{Arg}(f_{\theta_j}). \tag{8.25}$$

For instance, in Fig. 8.3 for $f^{\text{des}} = f_{\text{a}}$, points f_{θ_5} and f_{θ_6} satisfies this condition. If the couple $f_{\theta_i}, f_{\theta_j}$ satisfying the condition does not exist, then, according to the discussion in Section 8.3, the constrained inverse quadratic problem does not have a solution.

In the rest of this section, we introduce a method seeking the point $\tilde{f}^{\text{des}}$ by compression of $\mathcal{W}(\mathbf{A}_{\text{xy}})$ with $\mathbf{A}_{\text{xy}} \in \mathbb{R}^{n\times n}$ to $\mathcal{W}(\mathbf{B})$ with $\mathbf{B} \in \mathbb{R}^{2\times 2}$. The compressed numerical range enables us to find easily real generating vectors for any $f \in \partial\mathcal{W}(\mathbf{B})$. An alternative method is presented in Appendix B.1

8.4.1 Compression of the numerical range

Our method seeking $\tilde{f}^{\text{des}}$ is based on the solution of the inverse numerical range problem introduced by Carden [46]. Carden showed that if a point $f \in \mathcal{W}(\mathbf{A})$ is a convex combination of two other points $f_1, f_2 \in \mathcal{W}(\mathbf{A})$ with known generating vectors $\mathbf{u}_1, \mathbf{u}_2$, then a generating vector $\mathbf{u}$ of f can be found by compression of the numerical range $\mathcal{W}(\mathbf{A})$ to a 2D case. In our case, we seek a point $\tilde{f}^{\text{des}}$ which lies (in angle) between points f_{θ_i} and f_{θ_j} with known generating vectors $\mathbf{u}_{\theta_i}$ and $\mathbf{u}_{\theta_j}$, but it is not necessarily given by their convex combination. Furthermore, Carden's method works only for complex generating vectors. Hence, there are some minor changes that have to be made in order to use Carden's method for our case.

Following Carden's algorithm, we compress the numerical range to a 2D case. We construct an orthonormal matrix $\mathbf{U} \in \mathbb{C}^{n\times 2}$ such that $\mathbf{u}_{\theta_i}, \mathbf{u}_{\theta_j} \in \text{Range}(\mathbf{U})$. Let $\mathbf{B} = \mathbf{U}^{\mathsf{T}}\mathbf{A}_{\text{xy}}\mathbf{U}$. Then the numerical range $\mathcal{W}(\mathbf{B})$ is a compression of $\mathcal{W}(\mathbf{A}_{\text{xy}})$ to 2D. It is noteworthy, that $\mathbf{B}$ is symmetric, because $\mathbf{A}_{\text{xy}}$ is symmetric and the unitary transformation does not destroy symmetry. Furthermore, since vectors $\mathbf{u}_{\theta_i}$ and $\mathbf{u}_{\theta_j}$ are eigenvectors of real symmetric matrices $\mathbf{H}(e^{i\theta_i}\mathbf{A}_{\text{xy}})$ and $\mathbf{H}(e^{i\theta_j}\mathbf{A}_{\text{xy}})$, respectively, they are real and thus also $\mathbf{U}$ is real, that is $\mathbf{U} \in \mathbb{R}^{n\times 2}$.

Numerical range $\mathcal{W}(\mathbf{B})$ and also $\mathcal{R}(\mathbf{B})$, by construction, contains f_{θ_i} and f_{θ_j}. According to Proposition 8.2, $\mathcal{W}(\mathbf{B}) = \mathcal{R}(\mathbf{B})$ only for $n \geq 3$ and here we have $n = 2$. Thus, $\mathcal{W}(\mathbf{B})$ does not coincide with $\mathcal{R}(\mathbf{B})$ in general. Since we seek only real generating vectors and $\mathcal{W}(\mathbf{B}) \neq \mathcal{R}(\mathbf{B})$, we leave $\mathcal{W}(\mathbf{B})$ and focus solely on $\mathcal{R}(\mathbf{B})$.

Real numerical range $\mathcal{R}(\mathbf{B})$ necessarily contains a point positively collinear with f^{des}, because it contains f_{θ_i} and f_{θ_j}, and we can get from f_{θ_i} to f_{θ_j} continuously by a unit real generating vector. Let us denote the positively collinear point by $\hat{f}^{\text{des}}$. Point $\hat{f}^{\text{des}}$, however, is not the sought point $\tilde{f}^{\text{des}}$ because it does not necessarily have to lie on the boundary of $\mathcal{W}(\mathbf{A}_{\text{xy}})$, but, as we will see, it will serve as a very good approximation.

We seek a point $\hat{f}^{\text{des}}$ as an intersection of $\mathcal{R}(\mathbf{B})$ with the line in the direction of f^{des} (see Fig. 8.4). In order to simplify the calculation, we modify the matrix $\mathbf{B}$ so that it has zero trace:

$$\mathbf{B}' = \mathbf{B} - \frac{1}{2}\operatorname{tr}(\mathbf{B})\,\mathbf{I}_2. \tag{8.26}$$

Because matrix $\mathbf{B}'$ is symmetric and has zero trace, it has the following form

$$\mathbf{B}' = \begin{bmatrix} a & b \\ b & -a \end{bmatrix}, \tag{8.27}$$

where $a, b \in \mathbb{C}$. Carden further simplified $\mathbf{B}'$ by unitary matrix transformations to a real matrix with zeros on the diagonal (see [39, p. 19]). But the unitary matrices would necessary contain complex numbers which, as it will be obvious in a moment, would make the following procedure inapplicable. Thus, we cannot further simplify $\mathbf{B}'$ and leave it as it is.

Removing the trace from $\mathbf{B}$ changes $\mathcal{R}(\mathbf{B})$. According to property (P_2), which also applies on real numerical range, we have

$$\mathcal{R}(\mathbf{B}') = \mathcal{R}\left(\mathbf{B} - \frac{1}{2}\operatorname{tr}(\mathbf{B})\,\mathbf{I}_2\right) = \mathcal{R}(\mathbf{B}) - \frac{1}{2}\operatorname{tr}(\mathbf{B}). \tag{8.28}$$

In words, $\mathcal{R}(\mathbf{B}')$ is $\mathcal{R}(\mathbf{B})$ shifted by $-\frac{1}{2}\operatorname{tr}(\mathbf{B})$. Since we seek an intersection of $\mathcal{R}(\mathbf{B})$ with the line in the direction of f^{des} and now we shifted $\mathcal{R}(\mathbf{B})$, we also will have to shift the line by $-\frac{1}{2}\operatorname{tr}(\mathbf{B})$.

Let us now see how $\mathcal{R}(\mathbf{B}')$ looks like. We can describe all 2D real unit vectors as $\mathbf{v}(\alpha) = [\cos\alpha, \sin\alpha]^\intercal$. Making use of the form of the matrix $\mathbf{B}'$, we can describe all values of the real numerical range as

$$\mathcal{R}(\mathbf{B}') = \mathbf{v}(\alpha)^\intercal\,\mathbf{B}'\,\mathbf{v}(\alpha) = a\,\cos(2\alpha) + b\,\sin(2\alpha). \tag{8.29}$$

Notice, that $\mathcal{R}(\mathbf{B}')$ is a description of an ellipse. In fact, $\mathcal{W}(\mathbf{B}')$ is an ellipse with its interior (see [39, Lemma 1.3.3]) and $\mathcal{R}(\mathbf{B}')$ is its boundary, that is $\mathcal{R}(\mathbf{B}') = \partial\mathcal{W}(\mathbf{B}')$. Since $\mathcal{R}(\mathbf{B}')$ is shifted $\mathcal{R}(\mathbf{B})$, $\mathcal{R}(\mathbf{B})$ has to be an ellipse as well. Fig. 8.4 illustrates the relation between $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$, $\tilde{\Omega}_2(\mathbf{A}_x, \mathbf{A}_y)$ and $\mathcal{R}(\mathbf{B})$. Obviously, since $\mathcal{R}(\mathbf{B})$ is an ellipse having two boundary points of $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ on its boundary, it serves as a better local approximation of $\partial\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ than $\tilde{\Omega}_2(\mathbf{A}_x, \mathbf{A}_y)$.

Having a convenient description of $\mathcal{R}(\mathbf{B}')$, we proceed with the description of the shifted

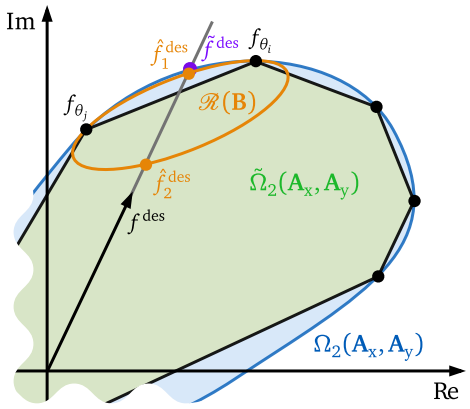


Figure 8.4: *Visualization of a compression of the numerical range $\mathscr{W}(\mathbf{A}_{xy})$ to a 2D case $\mathscr{R}(\mathbf{B})$.*

line given by f^{des}. One can easily verify that the shifted line is described by

$$f_x = t, \tag{8.30a}$$
$$f_y = k_1\, t + k_2, \tag{8.30b}$$

where $t \in \mathbb{R}$ is the parameter of the line, $k_1 = \frac{\text{Re}(f^{\text{des}})}{\text{Im}(f^{\text{des}})}$ and $k_2 = k_1 \,\text{Re}\left(\frac{1}{2}\,\text{tr}\,(\mathbf{B})\right) - \text{Im}\left(\frac{1}{2}\,\text{tr}\,(\mathbf{B})\right)$.

Next, we decompose complex numbers a and b in (8.28) to real and imaginary parts: $a = a_1 + i\,a_2$ and $b = b_1 + i\,b_2$. Then intersection points of $\mathscr{R}(\mathbf{B}')$ with the shifted line (8.30) are found as solutions of the following system of equations

$$a_1\,\cos(2\alpha) + b_1\,\sin(2\alpha) = t, \tag{8.31a}$$
$$a_2\,\cos(2\alpha) + b_2\,\sin(2\alpha) = k_1\,t + k_2, \tag{8.31b}$$

which has in general two solutions for α. The solutions are

$$\alpha_{1,2} = \arctan\left(\frac{b_2 - b_1\,k_1 \pm \sqrt{a_1^2\,k_1^2 - 2\,a_1\,a_2\,k_1 + a_2^2 + b_1^2\,k_1^2 - 2\,b_1\,b_2\,k_1 + b_2^2 - k_2^2}}{a_2 + k_2 - a_1\,k_1}\right). \tag{8.32}$$

Therefore, we have two real generating vectors for two intersection points. But these intersection points are shifted by $-\frac{1}{2}\text{tr}(\mathbf{B})$ from the original setup hence we have to shift them back. Thus, the sought intersection points are

$$\hat{f}_i^{\text{des}} = \mathbf{v}^{\mathsf{T}}(\alpha_i)\,\mathbf{B}'\,\mathbf{v}(\alpha_i) + \frac{1}{2}\text{tr}(\mathbf{B}) \tag{8.33}$$
$$= \mathbf{v}^{\mathsf{T}}(\alpha_i)\left(\mathbf{B} - \frac{1}{2}\text{tr}(\mathbf{B})\right)\mathbf{v}(\alpha_i) + \frac{1}{2}\text{tr}(\mathbf{B}) \tag{8.34}$$
$$= \mathbf{v}^{\mathsf{T}}(\alpha_i)\,\mathbf{B}\,\mathbf{v}(\alpha_i) - \frac{1}{2}\text{tr}(\mathbf{B})\mathbf{v}^{\mathsf{T}}\mathbf{v} + \frac{1}{2}\text{tr}(\mathbf{B}), \quad i = \{1,2\}, \tag{8.35}$$

but since $\mathbf{v}^{\mathsf{T}}\mathbf{v} = 1$, we get

$$\hat{f}_i^{\text{des}} = \mathbf{v}^{\mathsf{T}}(\alpha_i)\,\mathbf{B}\,\mathbf{v}(\alpha_i), \quad i = \{1,2\}. \tag{8.36}$$

The intersection points are visualized in Fig. 8.4. One can see that the point $\hat{f}_1^{\text{des}}$ is very close to the boundary hence it is a very good approximation of the sought point $\tilde{f}^{\text{des}}$.

In order to transform the 2D vectors $\mathbf{v}(\alpha_1)$ and $\mathbf{v}(\alpha_2)$ to the original n-dimensional space, we multiply them by the unitary matrix $\mathbf{U}$:

$$\hat{\mathbf{u}}_1^{\text{des}} = \mathbf{U}\,\mathbf{v}(\alpha_1), \tag{8.37}$$

$$\hat{\mathbf{u}}_2^{\text{des}} = \mathbf{U}\,\mathbf{v}(\alpha_2). \tag{8.38}$$

Obviously, matrix $\mathbf{U}$ has to be real because otherwise it would transform a real vector $\mathbf{v}(\alpha_i)$ to a complex vector $\hat{\mathbf{u}}_i^{\text{des}}$. But we seek only real generating vectors. That is why we could not further simplify the matrix $\mathbf{B}'$.

So we obtained two real generating vectors for points $\hat{f}_1^{\text{des}}$ and $\hat{f}_2^{\text{des}}$ that are positively collinear with the desired force f^{des}. The remaining step is to scale the generating vector corresponding to the intersection point closer to the boundary of $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$. Since it is not as easy as it seems to determine which of the points $\hat{f}_1^{\text{des}}$ and $\hat{f}_2^{\text{des}}$ is closer to the boundary, we scale both generating vectors by (8.23) and take the one which generates a force closer to the f^{des}.

8.5 Comparison of the set of all physically and computationally feasible forces

We described a way how to solve the constrained inverse quadratic problem (8.1) for all forces that have a generating vector $\mathbf{u}$ satisfying $\|\mathbf{u}\|_2 \leq 1$. We denoted this set by $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$. We also showed that the set of all forces for which we are able to find a solution of (8.1) can be extended from $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ to a larger set. Let us denote the extended set by $\Omega_{\text{ext}}(\mathbf{A}_x, \mathbf{A}_y)$. The set $\Omega_{\text{ext}}(\mathbf{A}_x, \mathbf{A}_y)$ is obtained as follows. Let f be an arbitrary boundary point of $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ and $\mathbf{u}$ is a generating vector of f. We normalize the generating vector $\mathbf{u}$ to $\mathbf{u}' = \mathbf{u}/\|\mathbf{u}\|_\infty$. Then the point $\mathbf{u}'^{\mathsf{T}}\mathbf{A}_{xy}\,\mathbf{u}'$ is a boundary point of $\Omega_{\text{ext}}(\mathbf{A}_x, \mathbf{A}_y)$.

Fig. 8.5 shows comparisons of the sets of all physically feasible forces $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$ with the sets $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ and $\Omega_{\text{ext}}(\mathbf{A}_x, \mathbf{A}_y)$. Since we are not able to calculate the boundary of $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$, we approximated the set by sampling the set of all admissible control signals $\mathbf{u}$. We randomly sampled the interior of the hypercube defined by the constraint $\|\mathbf{u}\|_\infty \leq 1$ and use these samples as generating vectors. The black dots in the figure represent forces obtained in this way. To make the approximation a bit more informative, we also sample all the edges of the hypercube and the forces generated by these samples are in the figure visualized by the gray dots. We compared the sets in four cases. We placed an imaginary microparticle at four places above the electrode array and for these places we calculated matrices $\mathbf{A}_x$ and $\mathbf{A}_y$. The figure clearly shows that, at least in these cases, the set $\Omega_{\text{ext}}(\mathbf{A}_x, \mathbf{A}_y)$ captures only a small part of $\Omega_\infty(\mathbf{A}_x, \mathbf{A}_y)$. In words, the proposed method clearly fails to find a solution in a rather large set of cases. We can also infer from the figure, that the extension of the set $\Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ to $\Omega_{\text{ext}}(\mathbf{A}_x, \mathbf{A}_y)$ enlarges significantly the set of forces for which the proposed algorithm finds a solution.

8.6 Possible improvements

As Carden has shown in [46], there are n linearly independent generating vectors for each point from the interior of $\mathcal{W}(\mathbf{A})$, where $\mathbf{A} \in \mathbb{R}^{n\times n}$. We solved the constrained inverse quadratic

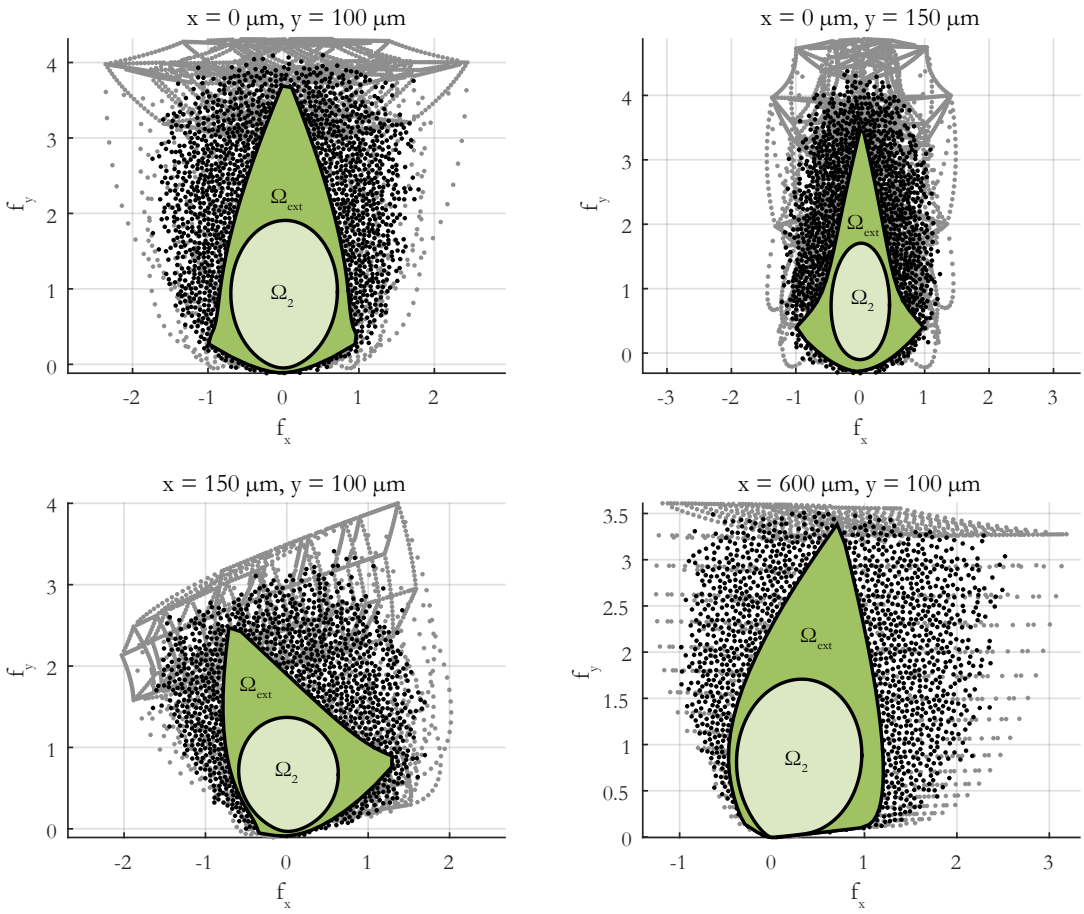


Figure 8.5: *Comparisons of all physically feasible forces (gray and black dots) with forces feasible under the condition* $\|\mathbf{u}\|_2 \leq 1$ (Ω_2) *and with forces for which the proposed algorithm is able to solve the constrained inverse quadratic problem* (Ω_{ext}).

problem by finding a point $\hat{f}^{\text{des}} \in \Omega_2(\mathbf{A}_x, \mathbf{A}_y)$ that is positively collinear with the desired force f^{des}. This point is then lengthened/shortened to match the magnitude of f^{des}. However, we cannot lengthen it more than by the factor equal to $1/\|\mathbf{u}\|_\infty$, where $\mathbf{u}$ is a generating vector of $\hat{f}^{\text{des}}$. According to Carden's result, $\hat{f}^{\text{des}}$ has n linearly independent generating vectors. Thus, if a generating vector $\mathbf{u}$ does not allows us to extend the $\hat{f}^{\text{des}}$ to the desired length, it could happen that $\hat{f}^{\text{des}}$ has another generating vector that would allow us to do it. Currently, we are able to find only of its generating vectors and we do not know how to find the others.

8.7 Experimental results

To verify the applicability of the proposed algorithm, we implemented it and used it as a part of the control strategy described in Chapter 7. Execution of the algorithm itself takes approximately 1.5 ms but due to the position estimation method, the control period is 100 ms.

Fig. 8.6 shows the measured trajectory of a microparticle steered along a desired trajectory. Apparently, the microparticle follows the desired trajectory and thus we can claim that viability of the control strategy and the proposed algorithm is proven. In order to get some insight how the generated potentials look like, we also add Fig. 8.7 showing the potentials along the trajectory.

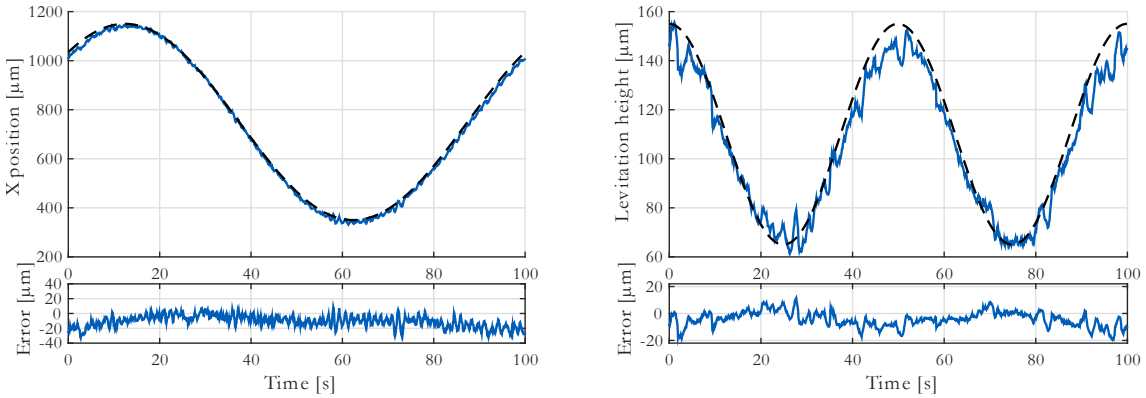


Figure 8.6: *An experiment showing manipulation of a microparticle by the described control strategy. The dashed black line shows the reference trajectory and the blue line represents the measured trajectory.*

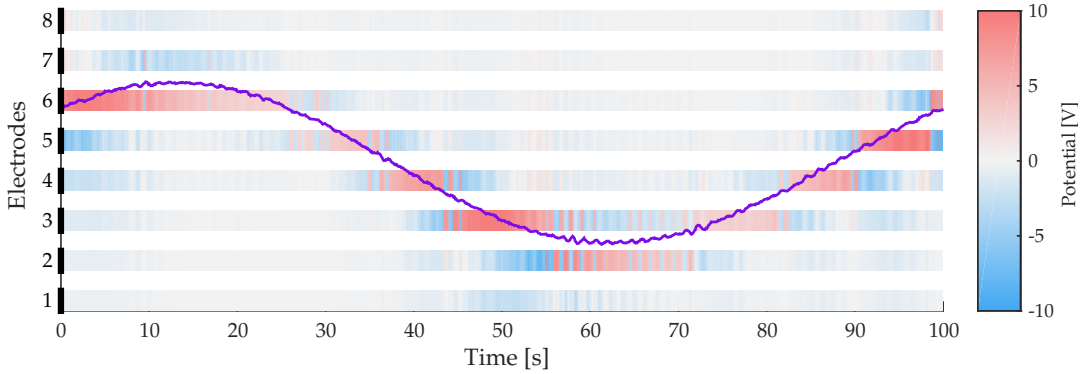


Figure 8.7: *An illustration showing the potentials set on the electrodes during the manipulation with a microparticle. The purple line represents the x component of the measured trajectory.*

Sure, the tracking error could be better and it could possibly be lowered by use of a better tunned proportional regulator or by use of a more complex regulator. But that was not the main goal of this thesis. We, to some extent, solved the 2D constrained inverse quadratic problem and here we showed that the overall control strategy works.

8.8 Conclusion

In this chapter, we proposed an algorithm partially solving the 2D constrained inverse quadratic problem. By partially, we mean that we had to replace the original constraint $\|\mathbf{u}\|_\infty \leq 1$ by the stricter constraint $\|\mathbf{u}\|_2 \leq 1$. This reduces the set of forces for which we are able to find a generating vector $\mathbf{u}$. Nevertheless, the situation is not that bad because we managed to extend the set even for some forces with $\|\mathbf{u}\|_2 > 1$. In addition, we proved that if the direction of the desired force is physically feasible, the proposed algorithm always finds a vector $\mathbf{u}$ generating a force in the desired direction, but possibly with a smaller magnitude.

Chapter 9

nD constrained inverse quadratic problem

We continue in the discussion of solving the constrained inverse quadratic problem (7.4). We will try to generalize the proposed algorithm solving the 2D case to a m-dimensional case. For convenience, here we restate the constrained inverse quadratic problem:

$$\begin{aligned} F_i &= \mathbf{u}^\mathsf{T}\mathbf{A}_i\,\mathbf{u}, \qquad i = 1,\dots,m, \\ \text{subject to } &\|\mathbf{u}\|_\infty \leq 1, \end{aligned} \tag{9.1}$$

where vector $\mathbf{u} \in \mathbb{R}^n$, $\mathbf{A}_i \in \mathbb{R}^{n\times n}$ and $\mathbf{A}_i = \mathbf{A}_i^\mathsf{T}$.

In the same spirit as in the 2D case, we use a similar set as the numerical range as a tool for solving the system of equation. This time, we use a so called *joint numerical range*.

9.1 Joint numerical range

The numerical range generalized to higher dimensions is called *joint numerical range*. Unfortunately, similarly as with the numerical range defined in Section 8.1, also the joint numerical range has several names. Some authors refer to is as *generalized numerical range* [47], some *multiform numerical range* [48] and some authors also as *k-dimensional field of values* [39]. We will stick with the name joint numerical range.

Definition 9.1 (Joint numerical range)**.** Let $(\mathbf{A}_1,\dots,\mathbf{A}_m)$ be an m-tuple of matrices $\mathbf{A}_i \in \mathscr{H}(n)$. Then the *joint numerical range (JNR)* of $(\mathbf{A}_1,\dots,\mathbf{A}_m)$ is defined as follows

$$\mathscr{W}_m(\mathbf{A}_1,\dots,\mathbf{A}_m) = \{(\mathbf{u}^*\mathbf{A}_1\mathbf{u},\dots,\mathbf{u}^*\mathbf{A}_m\mathbf{u}) \,|\, \mathbf{u}\in\mathbb{C}^n,\, \mathbf{u}^*\mathbf{u} = 1\}. \tag{9.2}$$

In contrast to the numerical range, the joint numerical range is not always convex. When $m = 2$, there is one-to-one correspondence between the numerical range and the joint numerical range hence $\mathscr{W}_2(\mathbf{A}_1,\mathbf{A}_2)$ is always convex [49, 50]. When $m = 3$ and $n > 2$, the joint numerical range $\mathscr{W}_3(\mathbf{A}_1,\mathbf{A}_2,\mathbf{A}_3)$ remains convex [50]. However, convexity may fail if $m > 3$ [49].

We also define the real joint numerical range:

Definition 9.2 (Real joint numerical range)**.** For any m-tuple $(\mathbf{A}_1,\dots,\mathbf{A}_m)$ of matrices $\mathbf{A}_i \in \mathscr{H}(n)$ we define the *real joint numerical range (rJNR)* as the set

$$\mathscr{R}_m(\mathbf{A}_1,\dots,\mathbf{A}_m) = \{(\mathbf{u}^\mathsf{T}\mathbf{A}_1\mathbf{u},\dots,\mathbf{u}^\mathsf{T}\mathbf{A}_m\mathbf{u}) \,|\, \mathbf{u}\in\mathbb{R}^n,\, \mathbf{u}^\mathsf{T}\mathbf{u} = 1\}. \tag{9.3}$$

For simplicity of notation, we denote a m-tuple $(\mathbf{A}_1,\dots,\mathbf{A}_m)$ by $\mathbf{A}$ and we will write simply $\mathscr{W}_m(\mathbf{A})$ instead of $\mathscr{W}_m(\mathbf{A}_1,\dots,\mathbf{A}_m)$, or analogously $\mathscr{R}_m(\mathbf{A})$, when no confusion can arise. Furthermore, let $\boldsymbol{\eta}\in\mathbb{R}^m$, then we write $\boldsymbol{\eta}^\intercal\mathbf{A}$ instead of $\sum_{i=1}^m \eta_i\mathbf{A}_i$.

Similarly as in the 2D case, we would like to be able to determine all the boundary points of $\mathscr{R}_m(\mathbf{A})$ or $\mathscr{W}_m(\mathbf{A})$ if they coincide. For that purpose, we base our discussion on the following proposition.

Proposition 9.1 (Gutkin [51])**.** *Let $\mathbf{A}=(\mathbf{A}_1,\dots,\mathbf{A}_m)$, where $\mathbf{A}_i\in\mathscr{H}(n)$, and let $s(\boldsymbol{\eta})$ be the support function of $\mathscr{W}(\mathbf{A})$. Then*

$$s(\boldsymbol{\eta})=\lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A}). \tag{9.4}$$

Let $\boldsymbol{\eta}\in\partial B^{m-1}$. Then

$$\mathscr{W}_m(\mathbf{A})\cap S(\boldsymbol{\eta},\lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A}))=\{\mathbf{u}^*(\boldsymbol{\eta}^\intercal\mathbf{A})\mathbf{u}\,|\,\mathbf{u}\in\mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A}),\mathbf{u}^*\mathbf{u}=1\}. \tag{9.5}$$

In words, the proposition states that all points from $\partial\mathscr{W}(\mathbf{A})$ that also lies in $\partial\,\mathrm{co}(\mathscr{W}(\mathbf{A}))$ have a generating vector (not necessarily real) from the eigenspace $\mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A})$ for a vector $\boldsymbol{\eta}$.

Although not of immediate use, based on the simulations, we make the following conjecture:

Conjecture 9.1. *Let $\mathbf{A}=(\mathbf{A}_1,\dots,\mathbf{A}_m)$ be an m-tuple of symmetric matrices $\mathbf{A}_i\in\mathbb{R}^{n\times n}$. Then for every point from $\partial\,\mathrm{co}(\mathscr{R}(\mathbf{A}))$ it holds that it is given by a convex combination of at most two points from $\partial\mathscr{R}(\mathbf{A})$.*

We feel that the key to a proof of the conjecture is hidden in Proposition 9.1, which, besides other things, tells us that every 2D projection of rJNR is convex.

9.2 Extending the proposed algorithm from 2D to mD

Now, we try to generalize the algorithm solving the 2D constrained inverse quadratic problem (see Section 8.4) to higher dimensions.

To begin with, we remind that the algorithm relied on the ability to determine all the boundary points of $\mathscr{R}(\mathbf{A}_{\mathrm{xy}})$ together with their generating vectors. In 2D we were able to determine all the boundary points of $\mathscr{R}(\mathbf{A}_{\mathrm{xy}})$ because we found a way how to determine boundary points of $\mathscr{W}(\mathbf{A}_{\mathrm{xy}})$ and $\mathscr{W}(\mathbf{A}_{\mathrm{xy}})$ coincided with $\mathscr{R}(\mathbf{A}_{\mathrm{xy}})$. We examine whether those two aspects are fulfilled in a general m-dimensional case and thus whether we can generalize the algorithm from 2D to higher dimensions.

The first part of Proposition 9.1 shows us how boundary points of $\mathrm{co}(\mathscr{W}_m(\mathbf{A}))$ can be determined: by variation of $\boldsymbol{\eta}$ we find all supporting hyperplanes of $\mathscr{W}_m(\mathbf{A})$ and thus, according to the *supporting hyperplane theorem* [52], we determine $\partial\,\mathrm{co}(\mathscr{W}_m(\mathbf{A}))$. Unfortunately, the second part of the proposition reveals only how to get generating vectors for points lying in $\partial\mathscr{W}_m(\mathbf{A})$ and at the same time in a supporting hyperplane of $\mathscr{W}_m(\mathbf{A})$. Therefore, we failed to satisfy the first aspect—by application of Proposition 9.1 we can determine generating vectors only for points from $\partial\mathscr{W}_m(\mathbf{A})\cap\partial\,\mathrm{co}(\mathscr{W}_m(\mathbf{A}))$, and since $\mathscr{W}_m(\mathbf{A})$ is not always convex there can be boundary points of $\mathscr{W}_m(\mathbf{A})$ that does not lie in a supporting hyperplane. This is troubling because it seems that the approach from the 2D case is in more dimensions applicable only for convex $\mathscr{W}_m(\mathbf{A})$—provided that the second aspect is met.

Let us now see if the second aspect is fulfilled for convex $\mathcal{W}_m(\mathbf{A})$, that is if $\mathcal{R}_m(\mathbf{A}) = \mathcal{W}_m(\mathbf{A})$ for convex $\mathcal{W}_m(\mathbf{A})$. We show by a counterexample that it does not have to be fulfilled. Let $\mathbf{A} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$. Since $\mathcal{W}_3(\mathbf{A})$ is always convex, it follows from Proposition 9.1 that for all $\mathbf{F} \in \partial\mathcal{W}_3(\mathbf{A})$ there is $\boldsymbol{\eta} \in \mathbb{R}^3$ such that $\mathbf{u} \in \mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A})$ is a generating vector of $\mathbf{F}$. But is there always a **real** generating vector for any $\mathbf{F} \in \partial\mathcal{W}_3(\mathbf{A})$? If it is, then $\partial\mathcal{R}_3(\mathbf{A}) = \partial\mathcal{W}_3(\mathbf{A})$. Let us further assume that for a vector $\boldsymbol{\eta}$ the eigenvalue $\lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A})$ is of multiplicity two and thus $\mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A}) = \operatorname{span}\{\mathbf{v}, \mathbf{w}\}$, where $\mathbf{v}, \mathbf{w}$ are unit real vectors (keep in mind that matrix $\boldsymbol{\eta}^\intercal\mathbf{A}$ will be always real and symmetric in our case). Then all real unit vectors $\mathbf{u}$ from $\mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A})$—that is $\mathbf{u} = \mathbf{v}\cos\alpha + \mathbf{w}\sin\alpha$, where $\alpha \in [0, 2\pi)$—generate a one-dimensional manifold in $\mathbb{R}^3$, but the intersection $\mathcal{W}_3(\mathbf{A}) \cap S(\boldsymbol{\eta}, \lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A}))$ is in general a 2D manifold. Hence some points from $\mathcal{W}_3(\mathbf{A}) \cap S(\boldsymbol{\eta}, \lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A}))$ necessarily have only complex generating vectors and thus $\partial\mathcal{R}_3(\mathbf{A}) \neq \partial\mathcal{W}_3(\mathbf{A})$. This result can be analogously generalized for $m > 3$. Based on this discussion, we can, however, infer that if $\mathcal{W}_m(\mathbf{A})$ is strictly convex then $\partial\mathcal{R}_m(\mathbf{A}) = \partial\mathcal{W}_m(\mathbf{A})$.

Proposition 9.2. *Let $\mathbf{A} = (\mathbf{A}_1, \ldots, \mathbf{A}_m)$ be a m-tuple of symmetric matrices $\mathbf{A}_i \in \mathbb{R}^{n\times n}$. If $\lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A})$ is a simple eigenvalue for all $\boldsymbol{\eta} \in \partial B^{m-1}$, then $\mathcal{W}_m(\mathbf{A})$ is strictly convex and*

$$\partial\mathcal{R}_m(\mathbf{A}) = \partial\mathcal{W}_m(\mathbf{A}) \tag{9.6}$$

Proof. If $\lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A})$ is simple for every $\boldsymbol{\eta} \in \partial B^{m-1}$ then $\mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A}) = \operatorname{span}\{\mathbf{v}\}$, where $\mathbf{v}$ is a real unit eigenvector—it is real because all matrices $\boldsymbol{\eta}^\intercal\mathbf{A}$ are real symmetric. It follows from Proposition 9.1 that $\mathcal{W}_m(\mathbf{A}) \cap S(\boldsymbol{\eta}, \lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A})) = \mathbf{v}^\intercal(\boldsymbol{\eta}^\intercal\mathbf{A})\mathbf{v}$. In words, intersection of $\mathcal{W}_m(\mathbf{A})$ with any of its supporting hyperplane is a point. It follows that $\mathcal{W}_m(\mathbf{A})$ is strictly convex. Furthermore, all points $\partial\mathcal{W}_m(\mathbf{A})$ are generated by real vectors and thus $\partial R_m(\mathbf{A}) = \partial\mathcal{W}_m(\mathbf{A})$. □

To summarize, the convexity of $\mathcal{W}_m(\mathbf{A})$ does not allow us to extend the algorithm solving the 2D constrained inverse quadratic problem to higher dimensions, only the strict convexity of $\mathcal{W}_m(\mathbf{A})$ enables us to do so. An obvious question arises, does the special form of matrices $\mathbf{A}_i$—derived in Chapter 5 (see (5.14))—ensure that $\lambda_1(\boldsymbol{\eta}^\intercal\mathbf{A})$ is simple for all $\boldsymbol{\eta} \in \partial B^{m-1}$? Unfortunately, by simulation we found out that the answer is negative. Therefore, the joint numerical range $\mathcal{W}_m(\mathbf{A})$ for the special form of matrices $\mathbf{A}_i$ is not in general strictly convex and thus we cannot use the approach we used for solving the 2D case for higher dimensions.

9.3 Reformulation to semidefinite programming

Since we are unable to extend the algorithm solving the 2D case, we will try to tackle the higher dimensional case by a different approach. We reformulate the original problem (9.1). We make use of the fact that the quadratic form can be also written as

$$\mathbf{u}^\intercal\mathbf{A}\mathbf{u} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}u_iu_j = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}p_{ij} = \operatorname{tr}(\mathbf{P}\mathbf{A}), \tag{9.7}$$

where elements of matrix $\mathbf{P} = [p_{ij}]$ are given by $p_{ij} = u_iu_j$ and thus $\mathbf{P} = \mathbf{u}\mathbf{u}^\intercal$. Therefore, the quadratic form equations in the original problem (9.1) can by written as

$$F_i = \operatorname{tr}(\mathbf{P}\mathbf{A}_i), \tag{9.8}$$

where $\mathbf{P} = \mathbf{u}\mathbf{u}^\intercal$ and $\|\mathbf{u}\|_\infty \leq 1$. That did not help us much, so we proceed further and completely get rid of the dependence on $\mathbf{u}$. Notice, that matrix $\mathbf{P}$ has to be positive semidefinite, because only if it is positive semidefinite, it can be, according to the *spectral theorem* (see, for instance, [53]), decomposed to

$$\mathbf{P} = \sum_{i=1}^{n} \lambda_i(\mathbf{P}) \mathbf{v}_i \mathbf{v}_i^\intercal, \tag{9.9}$$

where $\mathbf{v}_i$ is an orthonormal eigenvector associated with $\lambda_i(\mathbf{P})$. Thus for any positive semidefinite matrix $\mathbf{P}$ we can write

$$\operatorname{tr}(\mathbf{P}\mathbf{A}_i) = \sum_{j=1}^{\operatorname{rank}\mathbf{P}} \lambda_j(\mathbf{P}) \operatorname{tr}\left(\mathbf{v}_j \mathbf{v}_j^\intercal \mathbf{A}_i\right) = \sum_{j=1}^{\operatorname{rank}\mathbf{P}} \left(\sqrt{\lambda_j(\mathbf{P})}\mathbf{v}_j\right) \mathbf{A}_i \left(\sqrt{\lambda_j(\mathbf{P})}\mathbf{v}_j\right)^\intercal = \sum_{j=1}^{\operatorname{rank}\mathbf{P}} \tilde{\mathbf{v}}_j \mathbf{A}_i \tilde{\mathbf{v}}_j^\intercal. \tag{9.10}$$

Apparently, another requirement on matrix $\mathbf{P}$ is that it has to be of rank one. In order to completely reformulate the original problem (9.1) from $\mathbf{u}$ to $\mathbf{P}$, it remains to transform the condition $\|\mathbf{u}\|_\infty \leq 1$. Clearly, since $\mathbf{P} = \mathbf{u}\mathbf{u}^\intercal$, the diagonal elements of $\mathbf{P}$ are equal to u_i^2 and thus we can reformulate the condition $\|\mathbf{u}\|_\infty \leq 1$ to $p_{ii} \leq 1$. Putting it all together, the problem (9.1) can be reformulated to

$$\begin{aligned} F_i &= \operatorname{tr}(\mathbf{P}\mathbf{A}_i), \\ \mathbf{P} &\succeq 0 \\ \operatorname{rank}\mathbf{P} &= 1 \\ p_{ii} &\leq 1, \\ i &= 1, \ldots, m. \end{aligned} \tag{9.11}$$

Even though we transformed the quadratic equations to linear ones, due to the introduced rank constraint, the problem remains to be a tough nut to break. Therefore, we relax the rank constraint and further modify the problem to a so-called *rank minimization problem (RMP)*:

$$\begin{aligned} \mathbf{P}^* = \arg\min_{\mathbf{P}} \ & \operatorname{rank}(\mathbf{P}) \\ \text{s.t.:} \ & F_i = \operatorname{tr}(\mathbf{P}\mathbf{A}_i), \\ & \mathbf{P} \succeq 0, \\ & i = 1, \ldots, m, \end{aligned} \tag{9.12}$$

which is known to be NP-hard [54], but at least one can choose from several heuristics dealing with this problem. Notice, that we removed the condition $p_{ii} \leq 1$. That is because it lost its original meaning—$\mathbf{P}^*$ can be of higher rank than one, and then $p_{ii} \neq u_i^2$.

We should discuss how to interpret the minimizer $\mathbf{P}^*$. If $\operatorname{rank}\mathbf{P}^* = 1$ and $p_{ii} \leq 1$ for every i, then a solution of (9.1) is easily obtained by the decomposition of $\mathbf{P}$ to $\mathbf{u}\mathbf{u}^\intercal$. However, if $\operatorname{rank}\mathbf{P}^* > 1$ or $p_{ii} > 1$ for any i, then a solution of (9.1) does not exist.

As we already said, optimization problem (9.12) is hard to solve. But because it emerges in many areas like signal processing or control systems design, it gained quite an extensive attention from many researchers and several heuristics [55, 56] were developed. Here, we use a so-called *trace heuristic* which converts the non-convex problem (9.12) to the following

semidefinite programming (SDP) problem

$$\begin{aligned} \mathbf{Q}^* = \arg\min_{\mathbf{Q}} \ & \operatorname{tr}(\mathbf{Q}) \\ \text{s.t.:} \ & F_i = \operatorname{tr}(\mathbf{Q}\mathbf{A}_i), \\ & \mathbf{Q} \succeq 0, \\ & i = 1, \ldots, m. \end{aligned} \tag{9.13}$$

This optimization problem is, in contrast to (9.12), convex.

Finally, we ended up with a problem that is easy to solve. But we should not be mistaken and bear in mind that this is only a heuristic and thus it does not necessarily give us a minimal rank solution. There are special cases of rank minimization problems where the trace heuristic yields the minimum rank solution [57, 58]; unfortunately, our problem does not fall into this category. Another hitch is that even though the reformulated problem is easy to solve, the optimization runs over a symmetric matrix $\mathbf{P}$ and thus the number of optimization variables grew up from the original n to $n/2(n+1)$.

Again, we should discuss how to interpret the minimizer $\mathbf{Q}^*$. If $\operatorname{rank}\mathbf{Q}^* = 1$, then the situation is the same as with the original RMP and the solution of (9.1) is obtained via decomposition of $\mathbf{Q}^*$ to $\mathbf{u}\mathbf{u}^\intercal$. However, if $\operatorname{rank}\mathbf{Q}^* > 1$ then the situation differs; we cannot state anymore that a solution of (9.1) does not exist because we cannot be sure that there is no matrix $\mathbf{Q}$ satisfying the conditions with lower rank.

Now, we describe how to tackle the case when $\operatorname{rank}\mathbf{Q}^* > 1$. We use a so-called *log-det heuristic* [59, 56]—a more sophisticated heuristic for RMP than the trace heuristic—to find $\mathbf{Q}$ with lower rank. Furthermore, we show that by *chattering control* we can use even $\mathbf{Q}^*$ of higher rank than one to generate the desired force.

9.3.1 Log-det heuristic

The log-det heuristic [59, 56] is another heuristic approximately solving RMP. Without any derivation or explanation, here we just state that the heuristic iteratively solves the following SDP

$$\begin{aligned} \mathbf{Q}_{k+1} = \arg\min_{\mathbf{Q}} \ & \operatorname{tr}\left((\mathbf{Q}_k + \delta\mathbf{I})^{-1}\mathbf{Q}\right) \\ \text{s.t.:} \ & F_i = \operatorname{tr}(\mathbf{Q}\mathbf{A}_i), \\ & \mathbf{Q} \succeq 0, \\ & q_{ii} \leq 1, \\ & i = 1, \ldots, m, \end{aligned} \tag{9.14}$$

and that the initial value $\mathbf{Q}_0$ is the output of the trace heuristic, that is $\mathbf{Q}_0 = \mathbf{Q}^*$. Apparently, the log-det heuristic can be viewed as an iterative refinement of the result obtained by the trace heuristic. The obvious disadvantage of this approach is that we have to solve the SDP several times. That could not be possible due to the time constraints given by the fact that the algorithm is supposed to be used in a real-time application. In addition, this heuristic also does not necessarily converge to a true minimum rank solution P^*.

9.3.2 Chattering control

In the chattering control, the control action is composed of several values and the control system switches between those values so that the desired output is achieved. Probably the most famous example is *pulse-width modulation (PWM)*, where, for instance, we have 0 V and 5 V, and in order to achieve the desired output voltage, say 2.5 V, the control system spend half of the control period with 0 V and the other half with 5 V.

We can use the concept of chattering control so, that we decompose the output of the trace heuristic $\mathbf{Q}^*$, according to (9.9), to $\sum_{i=1}^{l=\operatorname{rank}\mathbf{Q}^*}\lambda_i(\mathbf{Q}^*)\mathbf{u}_i\mathbf{u}_i^\intercal$. Then in order to generate the desired force, the control system switches between $\mathbf{u}_i$ with switching times proportional to $\lambda_i(\mathbf{Q}^*)$. This can be mathematically expressed as

$$\mathbf{u}(t)=\begin{cases}\mathbf{u}_1 & t\in[0,k_1T),\\ \vdots & \\ \mathbf{u}_{l-1} & t\in[k_{l-2}T,k_{l-1}T),\\ \mathbf{u}_l & t\in[k_{l-1}T,T),\end{cases} \tag{9.15}$$

where T is the control period, time t is from the interval $[0,T)$ and

$$k_i=\frac{\sum_{j=1}^{i}\lambda_j(\mathbf{Q}^*)}{\sum_{j=1}^{l}\lambda_j(\mathbf{Q}^*)}. \tag{9.16}$$

In theory, as T converges to zero, the force generated by $\mathbf{u}(t)$ converges to the desired force. One should immediately argue, that this approach assumes that we are able to switch between $\mathbf{u}_i$ during the control period T. That would be a good objection, because we are actually not able to switch between $\mathbf{u}_i$ during the control period; currently, the control system runs in Simulink/Real-Time Windows Target environment which due to the access to the image sensor does not allow us to run a faster control loop side-by-side with the main control loop.

9.3.3 A comparison of SDP solvers

To implement the algorithm solving the general m-dimensional constrained inverse quadratic problem we need to select an appropriate—the fastest—solver for the SDP. We chose solvers that are written in Matlab or that has available interface to Matlab. We used the basic variant of the proposed algorithm—the trace heuristic—as a benchmark. Table 9.1 summarizes the run times of different solvers for different parameters n and m (for details see (9.1)). Based on the table, CSDP solver is the winner and thus it is also the one we used in the experiments. Also notice the bad scalability—the average time grows very fast in dependence on n.

9.4 Relation of the SDP formulation to the rJNR

In this section, we examine the relation between the SDP formulation and rJNR. We will illustrate the relation on a 3D case. Specifically, we randomly generated matrices $\mathbf{A}_1$,$\mathbf{A}_2$ and $\mathbf{A}_3$ from $\mathbb{R}^{10\times10}$.

Fig. 9.1a shows boundary points of $\mathscr{R}_3(\mathbf{A}_1,\mathbf{A}_2,\mathbf{A}_3)$ obtained by Proposition 9.1. That means, we varied $\boldsymbol{\eta}$ and generated the boundary points by real unit vectors $\mathbf{u}\in\mathbf{E}_1(\boldsymbol{\eta}^\intercal\mathbf{A})$.

Problem size		Average time [ms]				
n	m	SeDuMi	SDPT3	CSDP	DSDP	Mosek
8	2	11.3	28.4	2.4	2.1	3.0
8	4	12.0	32.6	2.5	2.5	3.1
48	2	43.1	46.8	13.3	18.1	24.4
48	4	46.9	48.1	17.7	23.0	27.4
48	8	55.2	50.1	25.7	32.3	35.1

Table 9.1: *A comparison of SDP solvers.*

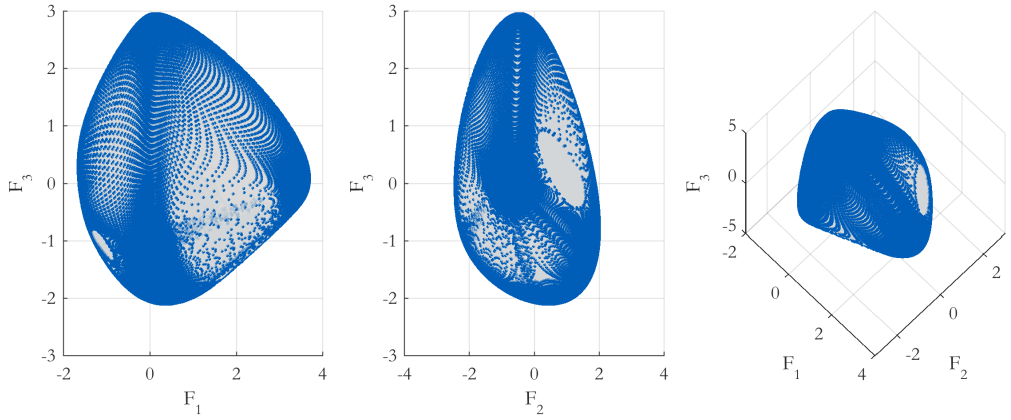
In fact, Proposition 9.1 actually allows us to find only the boundary points from the strictly convex part of the boundary. This is clearly visible in the figure, where one can see some circular blank spots—the non-strictly convex part of the boundary.

Now the situation becomes more interesting because a comparison of Fig. 9.1a with Fig. 9.1b shows that there is a deep connection between the SDP formulation and rJNR. But first things first, we should comment how Fig. 9.1b was obtained. We uniformly sampled the space of all unit forces $\mathbf{F}$. For each sample, we solved the constrained inverse quadratic problem by the trace heuristic (9.13) and, if the rank of $\mathbf{Q}^*$ was higher than one, we also solved the log-det heuristic (9.14) and obtained $\mathbf{Q}_N$. We decomposed $\mathbf{Q}^*$ to $\sum_{i=1}^{n} \lambda_i(\mathbf{Q}^*)\mathbf{v}_i\mathbf{v}_i^\intercal$ and possibly also $\mathbf{Q}_N$ to $\sum_{i=1}^{n} \lambda_i(\mathbf{Q}_N)\mathbf{w}_i\mathbf{w}_i^\intercal$. From the decompositions, we took only the eigenvectors associated with the largest eigenvalue, that is $\mathbf{v}_1$ and $\mathbf{w}_1$, and use them as generating vectors. Forces generated in this way by $\mathbf{v}_1$ and $\mathbf{w}_1$ are in the figure represented by green and red dots, respectively. Interesting, the SDP formulation actually do the same as the algorithm proposed in the previous chapter; it seeks a boundary point of $\mathscr{R}_3(\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3)$ in the direction of the desired force and scale its generating vector so that it generates the desired force. Apparently, it also fails in the same situations because without the log-det heuristic the figure would have the same blank spots. Luckily, the log-det heuristic manages to fill those blank spots.

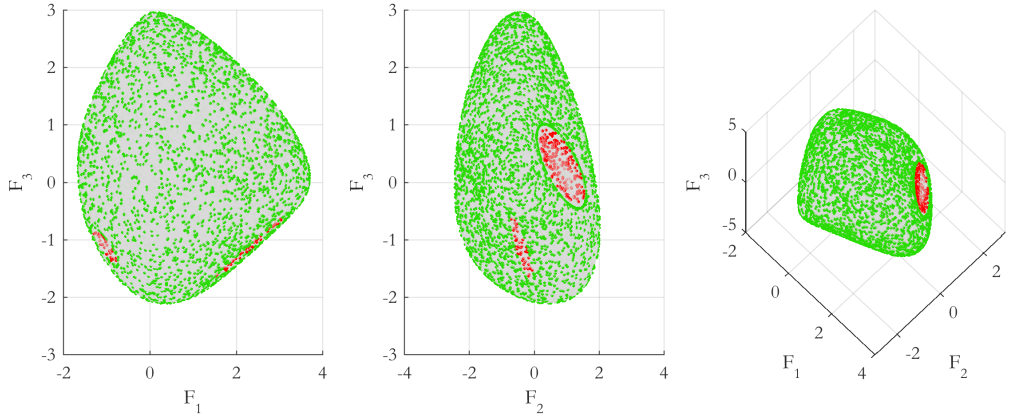
Let us figure out why the SDP formulation behaves like that. At first, we will try to get an intuition how the trace heuristic works. In the following discussion, we will frequently refer to the rank of $\mathbf{Q}$, so let us denote it by r. For simplicity, we will write λ_j instead of the full $\lambda_j(\mathbf{Q})$.

First of all, we make use of the fact that $\operatorname{tr}(\mathbf{Q}) = \sum_{i=1}^{r} \lambda_i$ and change the objective function accordingly. Then, we express the constraint $F_i = \operatorname{tr}(\mathbf{Q}\mathbf{A}_i)$ by the spectral theorem and (9.10) as $F_i = \sum_{j=1}^{r} \lambda_j \mathbf{u}_j^\intercal \mathbf{A}_i \mathbf{u}_j$. Thus, we get

$$\begin{aligned} \mathbf{Q}^* = \arg\min_{\mathbf{Q}} \ & \sum_{i=1}^{r} \lambda_i \\ \text{s.t.:} \ & F_i = \sum_{j=1}^{r} \lambda_j \mathbf{u}_j^\intercal \mathbf{A}_i \mathbf{u}_j, \\ & \mathbf{Q} \succeq 0, \\ & i = 1, \ldots, m. \end{aligned} \tag{9.17}$$



(a) *Real joint numerical range (rJNR)*



(b) *Semidifenite programming (SDP)*

Figure 9.1: *Relation of the SDP formulation to the rJNR.*

Now, we will tinker a bit more with the constraints involving F_i. We can rewrite the constraints to the vector form

$$\mathbf{F} = \begin{bmatrix} F_1 \\ \vdots \\ F_m \end{bmatrix} = \sum_{j=1}^{r} \lambda_j \underbrace{\begin{bmatrix} \mathbf{u}_j^\intercal \mathbf{A}_1 \mathbf{u}_j \\ \vdots \\ \mathbf{u}_j^\intercal \mathbf{A}_m \mathbf{u}_j \end{bmatrix}}_{\tilde{\mathbf{F}}_j} = \sum_{j=1}^{r} \lambda_j \tilde{\mathbf{F}}_j. \tag{9.18}$$

Keep in mind, that vectors $\mathbf{u}_j$ form an orthonormal basis of $\mathbf{Q}$ hence they are of unit length and vectors $\tilde{\mathbf{F}}_j$ are necessarily from $\mathscr{R}(\mathbf{A})$. We are getting closer to a nice interpretation; we already see that the trace heuristic optimization problem seeks the optimal solution as a weighted sum of points from $\mathscr{R}(\mathbf{A})$ while it minimizes the sum of the weights.

To see, why the optimal solution $\mathbf{Q}^*$ is of rank one only if $\mathbf{F}$ points at the strictly convex part of the boundary and higher rank otherwise, we further modify the constraint with $\mathbf{F}$. We divide the weighting coefficients λ_j by $\tilde{\lambda} = \sum_{j=1}^{r} \lambda_j$. Thus, we get

$$\mathbf{F} = \tilde{\lambda} \sum_{j=1}^{r} \lambda'_j \tilde{\mathbf{F}}_j = \tilde{\lambda} \tilde{\mathbf{F}}, \tag{9.19}$$

where $\tilde{\mathbf{F}}$ is given as the convex combination of points $\tilde{\mathbf{F}}_j$ with coefficients λ'_j. Since $\tilde{\mathbf{F}}$ is a convex combination of points from $\mathscr{R}(\mathbf{A})$, it necessarily lies in $\mathrm{co}(\mathscr{R}(\mathbf{A}))$. Finally, if we replace the condition involving F_i in (9.17) by the condition (9.19) and $\sum_{j=1}^{r}\lambda_j$ by $\tilde{\lambda}$, we get

$$\begin{aligned} \min_{\tilde{\mathbf{F}}\in\mathrm{co}(\mathscr{R}(\mathbf{A}))} \quad & \tilde{\lambda} \\ \text{s.t.:} \quad & \mathbf{F} = \tilde{\lambda}\tilde{\mathbf{F}}, \\ & \tilde{\lambda} \geq 0. \end{aligned} \tag{9.20}$$

Event though, this optimization problem does not tell us how to choose $\mathbf{u}$ in order to generate the desired force $\mathbf{F}$, it still is in principle the same as the original trace heuristic optimization problem (9.13). It shows us, that an optimization solver actually seeks the point $\tilde{\mathbf{F}}$ from $\mathrm{co}(\mathscr{R}(\mathbf{A}))$ that has the largest magnitude and that is positively collinear with $\mathbf{F}$—in other words, it seeks the positively collinear boundary point of $\mathrm{co}(\mathscr{R}(\mathbf{A}))$. Only then is the coefficient $\tilde{\lambda}$ minimized. We finally found the connection. If the boundary point $\tilde{\mathbf{F}}$ is on the strictly convex part of the boundary, then it lies in $\mathscr{R}(\mathbf{A})$, it has a real generating vector and $\mathrm{rank}\,\mathbf{Q}^* = 1$. Otherwise, $\tilde{\mathbf{F}}$ is given by a convex combination of two or more boundary points of $\mathscr{R}(\mathbf{A})$, it does not have a real generating vector and the trace heuristic optimization problem ends up with $\mathbf{Q}^*$ of rank higher than one.

It is here, where Conjecture 9.1 becomes useful. To remind, it states, that every point from $\partial\,\mathrm{co}(\mathscr{R}(\mathbf{A}))$ is given by a convex combination of at most two points from $\partial\mathscr{R}(\mathbf{A})$. Since the minimizing $\tilde{\mathbf{F}}$ must be from $\partial\,\mathrm{co}(\mathscr{R}(\mathbf{A}))$, it is given by a convex combination of at most two points from $\partial\mathscr{R}(\mathbf{A})$. Thus, if the conjecture is right, we can conclude that the trace heuristic optimization problem never ends up with $\mathbf{Q}^*$ of higher rank than two. As a result, in chattering control, one never needs to switch between more than two sets of potentials $\mathbf{u}$.

9.5 Possible improvements and modifications

The first modification we mention here deals with the situation where the trace heuristic ends up with matrix $\mathbf{Q}^*$ of rank higher than one. Then, instead of using the log-det heuristic or the chattering control, one can also decompose $\mathbf{Q}^*$ by the spectral theorem, take the dominant $\mathbf{u}$—a unit eigenvector associated with the largest eigenvalue—and use it as the initial value for one of many local search algorithms and the optimization problem minimizing $\sum_{i=1}^{m}|F_i - \mathbf{u}^\intercal\mathbf{A}_i\,\mathbf{u}|$ (for details, see (7.5)).

Second, the trace heuristic is based on an optimization method that needs to be initialized with a starting point. Currently, we do not provide to the used SDP solver a starting point and it generates it by itself. However, we can be more clever and use the $\mathbf{Q}^*$ from the previous control period as the starting point for the current control period.

Finally, instead of a generic SDP solver, we could develop our own problem specific solver. Solver designed for a problem with a special structure might be faster than a generic solver [60, Chapter 1].

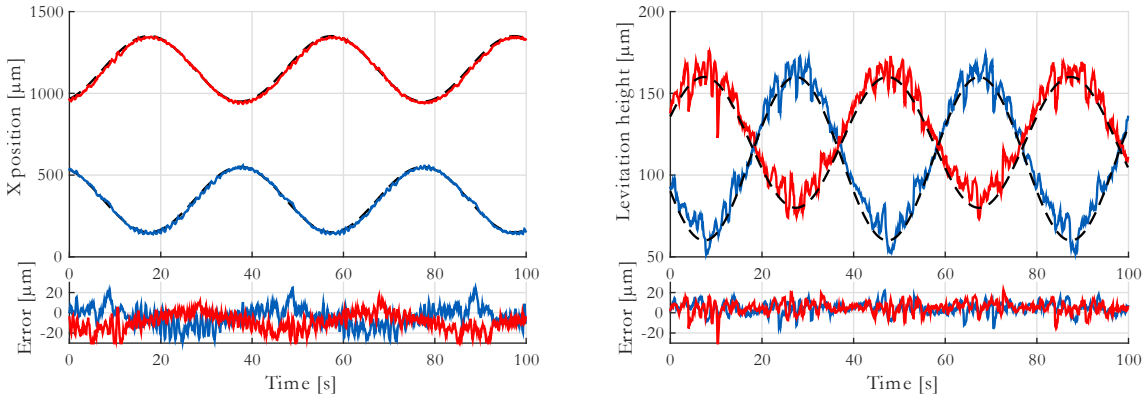


Figure 9.2: *A demonstration of manipulation of microparticles. The dashed black lines show the desired trajectories and the blue and red lines represent the measured trajectories.*

9.6 Experimental results

Similarly as in the previous chapter, also here we show that the proposed algorithm truly works and is useful. As we mentioned earlier, currently, we are unable to implement the chattering control, thus we show experimental result only for the log-det heuristic.

We used the proposed algorithm for control of position of two microparticles in the same hardware setup as in the previous chapter. That means, we have eight electrodes ($n = 8$) and we need to solve a constrained inverse quadratic problem with four equations ($m = 4$) at every control period. We use CSDP solver hence according to Table 9.1, the execution of the trace heuristic itself takes 2.5 ms. Since one iteration of the log-det heuristic takes approximately 2.5 ms as well, we limited the number of iterations of the log-det heuristic to five. Therefore, the execution of the algorithm with the log-det heuristic should not take more than 15 ms which still is acceptable for real-time use.

Fig. 9.2 shows measured and desired trajectories of the microparticles. The microparticles obviously follow the desired trajectories and the proposed algorithm with the log-det heuristic proved to be be applicable. Similarly as in the 2D case, we also add Fig. 9.3 showing the set potentials on the electrodes along the trajectories.

It is difficult to argue why the microparticles actually follow so nicely the desired trajectories despite the fact that the log-det heuristic might fail to find a rank-one solution. Of course, it could be the case that this situation just did not happen during the experiment but that is very unlikely. More plausible explanation is that even though the log-det heuristic fails from time to time it usually does not fail in more control periods in a row. Because if it fails and the generated force slightly differs from the desired one, the microparticles move to different positions where the setup of the constrained inverse quadratic problem changes and where the log-det heuristic might succeed to find a rank-one solution.

9.7 Conclusion

In this chapter, we discussed solution of the general m-dimensional constrained inverse quadratic problem. Since we did not find an algorithm fully solving the 2D version, we did not aspire to find one for higher dimensions. At the beginning, we tried to extend the algorithm partially solving the 2D version to higher dimensions. Similarly as in 2D, where we based

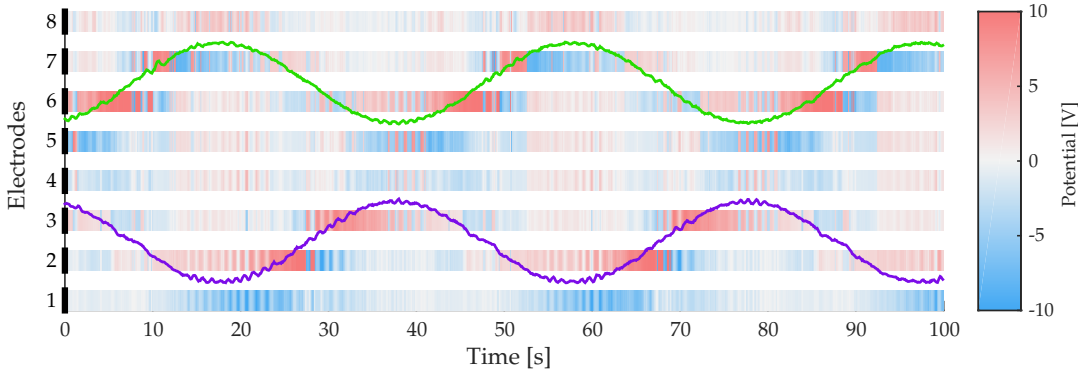


Figure 9.3: *An illustration showing the potentials set on the electrodes during the manipulation with the microparticles. The green and purple lines show the x component of the trajectories of the microparticles.*

the algorithm on the numerical range, here, we based the algorithm on generalization of the numerical range to higher dimensions: joint numerical range. However, in contrast to the numerical range, joint numerical range is not always convex and that is one of the aspects preventing us to extend the algorithm partially solving the 2D case to higher dimensions.

Thus we tried a different approach and reformulate the constrained inverse quadratic problem to a semidefinite program. Specifically, we reformulate it to a rank minimization problem which is also hard to solve but at least one can use one of many available heuristics. We used the trace heuristic followed by the log-det heuristic. In the end of the chapter, we showed on an experiment that such an algorithm truly is usable for real-time use.

Chapter 10

Conclusion

We had three goals in this thesis: develop and implement a real-time method estimating position of microparticles in 3D; find a model of *dielectrophoretic (DEP)* force suitable for control purposes and design an algorithm that computes potentials which needs to be set on electrodes in order to develop desired DEP forces at one or several places above the electrode array. The first two goals were fully achieved, the third goal was achieved to a satisfactory level.

In the first part of this thesis, we dealt with the position estimation method. Since we did not find a real-time method that would be directly usable for our case in the literature, we reviewed several methods that could possible be modified for real-time use. Specifically, we spent some time examining *digital holography* as a promising principle for real-time position estimation. Based on the review, we singled out the twin-beam method. Because the method was originally designed for off-line use, we modified it so that it became usable also for real-time use. We proved the applicability of the method by an experiment where we compared the estimated position by the method with ground truth measurements.

Then we focused on development of a control oriented model of DEP force field. The ability to compute a DEP force field boils down to the ability to compute the first and second derivatives of the potential field and that boils down to the ability to solve a specific boundary value problem—Laplace equation with *Dirichlet boundary conditions* imposed by the electrodes and zero charge *Neumann conditions* everywhere else. We found a way how to approximate the mixed boundary conditions purely by Dirichlet conditions and how to change the domain of the problem so that one can use Green's functions to solve this modified boundary value problem. This way, we obtained an analytical description of the potential field that enable us to compute the DEP force in real-time.

Finally, we delved into the design of an algorithm computing the potentials on electrodes for one or more prescribed DEP force(s). We treated separately 2D and higher dimensional case. In the former, we proposed an algorithm solving the problem, both based on *the numerical range* and its properties. For higher dimensions, we modified the problem to a *rank minimization problem* and solved it approximately by a heuristic. Unfortunately, in both cases, the algorithms are unable to find the potentials for all physically feasible forces, they are able to do so only for more limited sets. But we showed by experiments, that the sets seem to be sufficiently large to control arbitrarily the position of microparticles.⊗

Appendices

Appendix A

Digital holography

A.1 Theoretical treatment of back-propagation

An image with interference patterns is something like a snapshot of intensity values of the complex wave field incident upon the image sensor. If we assume that the complex wave field propagates through a linear and space invariant imaging system then the complex wave field at the image sensor plane can be described by the following two dimensional convolution

$$u_0(x_{im}, y_{im}) = h_z(x_{im}, y_{im}) \otimes u_z(x_{im}, y_{im}), \tag{A.1}$$

where x_{im}, y_{im} are the image coordinates, $u_0(x_{im}, y_{im})$ is complex wave field at the image sensor plane, $u_z(x_{im}, y_{im})$ is complex wave field at axial distance z from the image sensor plane and $h_z(x_{im}, y_{im})$ is the impulse response of the system for propagation distance z. This description allows us to easily propagate the captured complex wave field to an arbitrary distance. Therefore, if a captured image contains interference pattern from an object with a known axial distance z from the image sensor, the complex wave field captured by the image can be back-propagated by convolution with impulse response $h_{-z}(x_{im}, y_{im})$:

$$u_z(x_{im}, y_{im}) = h_{-z}(x_{im}, y_{im}) \otimes u_0(x_{im}, y_{im}). \tag{A.2}$$

Then, if we take intensity of the back-propagated complex wave field we obtain an image where the object is sharp and looks like it would be seen by a microscope.

There are more choices for the impulse response $h_z(x_{im}, y_{im})$ of the system. In this thesis, we used the most general one, the Rayleigh-Sommerfeld propagator:

$$h_z(x_{im}, y_{im}) = \frac{1}{2\pi} \frac{\partial}{\partial z} \frac{e^{ikR}}{R}, \tag{A.3}$$

where $R^2 = x_{im}^2 + y_{im}^2 + z^2$, $k = \frac{2\pi n}{\lambda}$, n is refractive index of the medium the light propagates through and λ is wavelength of the illumination.

Appendix B

2D constrained inverse quadratic problem

B.1 Univariate optimization

As a reminder, the boundary points f_{θ_k} are obtained as the rightmost (or possibly also leftmost) points of the rotated numerical range $\mathcal{W}(e^{i\theta_k}\mathbf{A}_{\text{xy}})$ (see Section 8.1.1). It is intuitive to formulate the search for $\tilde{f}^{\text{des}}$ as the following optimization problem:

$$\begin{aligned} \theta^* = \underset{\theta\in(\theta_i,\theta_j)}{\arg\min}\ & |\text{Arg}(f^{\text{des}}) - \text{Arg}(f(\theta))|, \\ \text{s.t.:}\ & f(\theta) = \mathbf{u}_\theta^{\intercal}\mathbf{A}_{\text{xy}}\mathbf{u}_\theta, \\ & \mathbf{u}_\theta \in \mathbf{E}_1(\mathbf{H}(e^{i\theta}\mathbf{A}_{\text{xy}})), \quad \|\mathbf{u}\|_2 = 1. \end{aligned} \tag{B.1}$$

To show that this optimization task is easily solvable, we inspect the function $\text{Arg}(f(\theta))$. If we increase angle θ, we find out that function Arg(f(θ)) decreases monotonously. The only exception is when the step between $-\pi$ and π (or 0 and 2π in dependence on the definition of Arg()) occurs, but this step can be removed by addition of 2π appropriately. We can visually verify this fact with the aid of Fig. B.1. We start with $\theta = 0$. Then the value of $\text{Arg}(f(\theta))$ corresponds to the angle of the rightmost point of $\mathcal{W}(\mathbf{A}_{\text{xy}})$. Now as we increase θ we rotate $\mathcal{W}(\mathbf{A}_{\text{xy}})$ in the counterclockwise direction. Taking the rightmost point of the rotated numerical range, we get a boundary point of the original numerical range shifted in the clockwise direction. Therefore, $\text{Arg}(f(\theta))$ decreases as θ increases. Since $\text{Arg}(f(\theta))$ is monotonous, function

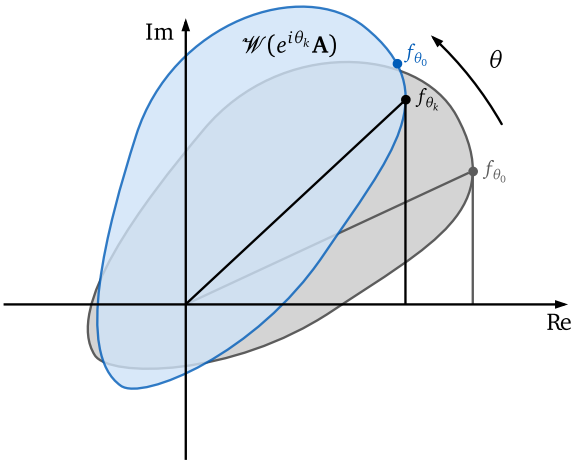


Figure B.1: *Computation of boundary points of a numerical range* $\mathcal{W}(\mathbf{A})$.

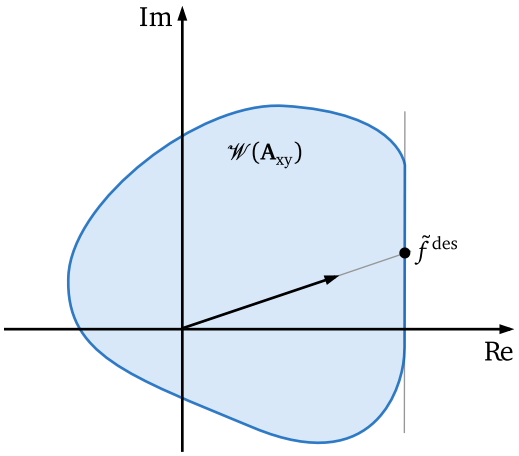


Figure B.2: *An example of numerical range with non-stricly convex boundary.*

$|\mathrm{Arg}(f^{des}) - \mathrm{Arg}(f(\theta))|$ has only one local minimum because it is given by the absolute value of a monotonous function. Thus, the optimization problem is easily solvable by, for instance, *golden search method* [38].

However, there is a hitch in the preceding discussion. Fig. B.1 is a little misleading because it shows a strictly convex numerical range. The situation gets a bit more complicated in the case of a non-strictly convex numerical range, because then different unit vectors $\mathbf{u}_\theta \in \mathbf{E}_1(\mathbf{H}(e^{i\theta}\mathbf{A}_{xy}))$ can generate different boundary points. Have a look at Fig. B.2 where there are infinitely many rightmost boundary points. Now many questions arises. Do we have to detect the case where the choice of $\mathbf{u}_\theta$ matters? Potentially, how to detect it? How to choose a unit $\mathbf{u}_\theta$ from $\mathbf{E}_1(\mathbf{H}(e^{i\theta}\mathbf{A}_{xy}))$? We will treat the questions one after the other.

Luckily, the answer to the first question is negative and the second one thus meaningless. We do not have to detect such cases. If $\tilde{f}^{des}$ lies on a strictly convex part of the boundary then, trivially, it is immaterial how we choose $\mathbf{u}$ on the rest of the boundary. The optimization will always end up with $f(\theta^*) = \tilde{f}^{des}$. If $\tilde{f}^{des}$ lies on a non-strictly convex part of the boundary, then the choice of $\mathbf{u}$ influences the value of $f(\theta^*)$ but not the optimal value θ^*. The optimization will always end up with θ^* for which a unit generating vector of $\tilde{f}^{des}$ lies in $\mathbf{E}_1(\mathbf{H}(e^{i\theta^*}\mathbf{A}_{xy}))$. This gives importance to the third question.

Now, we will answer the question how to find a unit vector $\mathbf{u}$ from $\mathbf{E}_1(\mathbf{H}(e^{i\theta^*}\mathbf{A}_{xy}))$ generating $\tilde{f}^{des}$. Let us have a look how the rightmost points of a rotated numerical range $\mathcal{W}(e^{i\theta^*}\mathbf{A}_{xy})$ are generated and hopefully we will find a hint of the answer. They are generated by the following relation

$$\mathbf{u}^\intercal \left(e^{i\theta^*}\mathbf{A}_{xy}\right)\mathbf{u}, \quad \mathbf{u} \in \mathbf{E}_1(\mathbf{H}(e^{i\theta^*}\mathbf{A}_{xy})), \quad \|\mathbf{u}\|_2 = 1. \tag{B.2}$$

We decompose matrix $e^{i\theta^*}\mathbf{A}_{xy}$. First of all, we express the rotation by the famous Euler formula:

$$e^{i\,\theta^*}\,\mathbf{A}_{xy} = \mathbf{A}_{xy}\cos\theta^* + i\,\mathbf{A}_{xy}\sin\theta^*. \tag{B.3}$$

Next, we make use of the fact, that the matrix $\mathbf{A}$ has a special form; it is given by $\mathbf{A}_{xy} = \mathbf{A}_x + i\mathbf{A}_y$. Substitution of this relation to the previous one yields

$$\begin{aligned} e^{i\,\theta^*}\,\mathbf{A}_{xy} &= (\mathbf{A}_x + i\mathbf{A}_y)\cos\theta^* + i\,(\mathbf{A}_x + i\mathbf{A}_y)\sin\theta^* && \text{(B.4)}\\ &= (\mathbf{A}_x\cos\theta^* - \mathbf{A}_y\sin\theta^*) + i\,(\mathbf{A}_x\sin\theta^* + \mathbf{A}_y\cos\theta^*). && \text{(B.5)} \end{aligned}$$

Now we can see that the decomposition is

$$e^{i\theta^*}\mathbf{A}_{\mathrm{xy}} = \mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}) + i\,\mathbf{S}'(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}), \tag{B.6}$$

where matrices $\mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}})$ and $\mathbf{S}'(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}})$ are

$$\mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}) = \mathbf{A}_{\mathrm{x}}\cos\theta^* - \mathbf{A}_{\mathrm{y}}\sin\theta^*, \tag{B.7}$$
$$\mathbf{S}'(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}) = \mathbf{A}_{\mathrm{x}}\cos\theta^* - \mathbf{A}_{\mathrm{y}}\sin\theta^*. \tag{B.8}$$

Note, that this decomposition is not standard Hermitian decomposition; matrix $\mathbf{S}'(e^{i\theta^*}\mathbf{A})$ is not skew-Hermitian.

Substituting the decomposition to (B.2), we get

$$\mathbf{u}^{\mathsf{T}}\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{u} = \mathbf{u}^{\mathsf{T}}\mathbf{H}\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{u} + i\,\mathbf{u}^{\mathsf{T}}\mathbf{S}'\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{u}. \tag{B.9}$$

Since matrices $\mathbf{H}\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)$ and $\mathbf{S}'\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)$ are real and symmetric, the first and the second term determine the real and the imaginary part of the generated boundary points, respectively. Vectors $\mathbf{u}$ from $\mathbf{E}_1(\mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}))$ generate the rightmost boundary points of the rotated numerical range hence the real part is necessarily constant, and by the assumption that $\tilde{f}^{\mathrm{des}}$ is among the rightmost boundary points of the rotated numerical range, the real part equals to $\mathrm{Re}(e^{i\theta^*}\tilde{f}^{\mathrm{des}})$. Thus, our problem reduces to

$$\mathrm{Im}(e^{i\theta^*}\tilde{f}^{\mathrm{des}}) = \mathbf{u}^{\mathsf{T}}\mathbf{S}'\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{u}. \tag{B.10}$$

Let $\lambda_1(\mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}))$ be of multiplicity k and $\{\mathbf{u}_1,\ldots,\mathbf{u}_k\}$ be the real orthonormal basis of $\mathbf{E}_1(\mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}))$. There always is such a basis because $\mathbf{H}(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}})$ is real and symmetric. Let us express a unit vector $\mathbf{u} \in \mathrm{span}\{\mathbf{u}_1,\ldots,\mathbf{u}_k\}$ as

$$\mathbf{u} = \begin{bmatrix}\mathbf{u}_1 & \ldots & \mathbf{u}_k\end{bmatrix}\begin{bmatrix}\alpha_1\\ \vdots\\ \alpha_k\end{bmatrix} = \mathbf{U}\boldsymbol{\alpha}, \qquad \sum_{i=1}^{k}\alpha_i^2 = \|\boldsymbol{\alpha}\|_2 = 1. \tag{B.11}$$

Substituting this relation to (B.10) yields

$$\mathrm{Im}(e^{i\theta^*}\tilde{f}^{\mathrm{des}}) = \boldsymbol{\alpha}^{\mathsf{T}}\mathbf{U}^{\mathsf{T}}\mathbf{S}'\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{U}\boldsymbol{\alpha}, \qquad \|\boldsymbol{\alpha}\|_2 = 1. \tag{B.12}$$

If we define a new matrix $\mathbf{B} = \mathbf{U}^{\mathsf{T}}\mathbf{S}'\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{U}$, we get

$$\mathrm{Im}(e^{i\theta^*}\tilde{f}^{\mathrm{des}}) = \boldsymbol{\alpha}^{\mathsf{T}}\mathbf{B}\boldsymbol{\alpha}, \qquad \|\boldsymbol{\alpha}\|_2 = 1. \tag{B.13}$$

Since matrix $\mathbf{B}$ is also real and symmetric, vector $\boldsymbol{\alpha}$ can be expressed in the orthonormal basis $\{\mathbf{b}_1,\ldots\mathbf{b}_k\}$ of $\mathbf{B}$, that is $\boldsymbol{\alpha} = \sum_{i=1}^{k}\beta_i\mathbf{b}_i$. Then we can rewrite (B.13) to

$$\mathrm{Im}(e^{i\theta^*}\tilde{f}^{\mathrm{des}}) = \boldsymbol{\alpha}^{\mathsf{T}}\mathbf{B}\boldsymbol{\alpha} = \sum_{i=1}^{k}\beta_i^2\lambda_i(\mathbf{B}), \qquad \sum_{i=1}^{k}\beta_i^2 = 1. \tag{B.14}$$

This problem is nothing else than a possibly under-determined system of linear equations,

that, by construction, always has a positive solution:

$$\begin{bmatrix} \lambda_1(\mathbf{B}) & \dots & \lambda_k(\mathbf{B}) \\ 1 & \dots & 1 \end{bmatrix} \begin{bmatrix} \beta_1^2 \\ \vdots \\ \beta_k^2 \end{bmatrix} = \begin{bmatrix} \mathrm{Im}(e^{i\theta^*}\tilde{f}^{\mathrm{des}}) \\ 1 \end{bmatrix}. \tag{B.15}$$

Therefore, by solution of (B.15) we obtain coefficients $\beta_{1,\dots,k}$. These coefficients enable us to calculate vector $\boldsymbol{\alpha} = \sum_{i=1}^{k} \beta_i \mathbf{b}_i$ which enables us to determine $\mathbf{u} = \mathbf{U}\boldsymbol{\alpha}$. Vector $\mathbf{u}$ is the sought solution, because we have

$$\mathbf{u}^{\intercal}\left(e^{i\theta^*}\mathbf{A}_{\mathrm{xy}}\right)\mathbf{u} = e^{i\theta}\tilde{f}^{\mathrm{des}} \tag{B.16}$$

and thus

$$\mathbf{u}^{\intercal}\mathbf{A}_{\mathrm{xy}}\mathbf{u} = \tilde{f}^{\mathrm{des}}. \tag{B.17}$$

In Section 7.2.2, we showed how the system of equations (7.4), can be formulated as an optimization problem. Here we ended up with an optimization problem as well so what is the difference? Which of the optimization problems is more tractable? It is not difficult to see that the optimization problem stated here is significantly more tractable. First of all, it has one local minimum. Moreover, the optimization runs over θ, which is a scalar. In contrast, the optimization problem (7.5) can have more local minimums and it runs over n-dimensional vector $\mathbf{u}$.

Bibliography

[1] C. Zhang, K. Khoshmanesh, A. Mitchell, and K. Kalantar-zadeh, "Dielectrophoresis for manipulation of micro/nano particles in microfluidic systems," *Analytical and Bioanalytical Chemistry*, vol. 396, no. 1, pp. 401–420, 2009.

[2] R. Pethig, "Review article—dielectrophoresis: Status of the theory, technology, and applications," *Biomicrofluidics*, vol. 4, no. 2, p. 022811, 2010.

[3] U. Schnars, C. Falldorf, J. Watson, and W. Jüptner, *Digital Holography and Wavefront Sensing*. Springer Berlin Heidelberg, 2015.

[4] L. Repetto, E. Piano, and C. Pontiggia, "Lensless digital holographic microscope with light-emitting diode illumination," *Optics letters*, vol. 29, no. 10, pp. 1132–1134, 2004.

[5] O. Mudanyali, D. Tseng, C. Oh, S. O. Isikman, I. Sencan, W. Bishara, C. Oztoprak, S. Seo, B. Khademhosseini, and A. Ozcan, "Compact, light-weight and cost-effective microscope based on lensless incoherent holography for telemedicine applications," *Lab on a chip*, vol. 10, no. 11, pp. 1417–1428, 2010.

[6] U. Schnars and W. Jüptner, "Direct recording of holograms by a CCD target and numerical reconstruction," *Applied optics*, vol. 33, no. 2, pp. 179–181, 1994.

[7] U. Schnars and W. P. Jüptner, "Digital recording and numerical reconstruction of holograms," *Measurement science and technology*, vol. 13, no. 9, p. R85, 2002.

[8] A. Greenbaum, W. Luo, T.-W. Su, Z. Göröcs, L. Xue, S. O. Isikman, A. F. Coskun, O. Mudanyali, and A. Ozcan, "Imaging without lenses: achievements and remaining challenges of wide-field on-chip microscopy," *Nature methods*, vol. 9, no. 9, pp. 889–895, 2012.

[9] J. W. Goodman, *Introduction to Fourier optics*. McGraw-Hill, 1996.

[10] X. Yu, J. Hong, C. Liu, and M. K. Kim, "Review of digital holographic microscopy for three-dimensional profiling and tracking," *Optical Engineering*, vol. 53, no. 11, pp. 112 306–112 306, 2014.

[11] T. Wriedt, "Mie theory: A review," in *The Mie Theory*, ser. Springer Series in Optical Sciences, W. Hergert and T. Wriedt, Eds. Springer Berlin Heidelberg, 2012, no. 169, pp. 53–71.

[12] E. Le Ru and P. Etchegoin, *Principles of Surface-Enhanced Raman Spectroscopy*. Elsevier, 2008.

[13] J. A. Guerrero-Viramontes, D. Moreno-Hernández, F. Mendoza-Santoyo, and M. Funes-Gallanzi, "3d particle positioning from CCD images using the generalized lorenz–mie and

huygens–fresnel theories," *Measurement Science and Technology*, vol. 17, no. 8, p. 2328, 2006.

[14] S.-H. Lee, Y. Roichman, G.-R. Yi, S.-H. Kim, S.-M. Yang, A. van Blaaderen, P. van Oostrum, and D. G. Grier, "Characterizing and tracking single colloidal particles with video holographic microscopy," *Optics Express*, vol. 15, no. 26, pp. 18 275–18 282, 2007.

[15] F. C. Cheong, B. Sun, R. Dreyfus, J. Amato-Grill, K. Xiao, L. Dixon, and D. G. Grier, "Flow visualization and flow cytometry with holographic video microscopy," *Optics express*, vol. 17, no. 15, pp. 13 071–13 079, 2009.

[16] Y. Park, G. Popescu, K. Badizadegan, R. R. Dasari, and M. S. Feld, "Fresnel particle tracing in three dimensions using diffraction phase microscopy," *Optics letters*, vol. 32, no. 7, pp. 811–813, 2007.

[17] J. F. Restrepo and J. Garcia-Sucerquia, "Automatic three-dimensional tracking of particles with high-numerical-aperture digital lensless holographic microscopy," *Optics Letters*, vol. 37, no. 4, pp. 752–754, 2012.

[18] F. Dubois, C. Schockaert, N. Callens, and C. Yourassowsky, "Focus plane detection criteria in digital holography microscopy by amplitude analysis," *Optics Express*, vol. 14, no. 13, pp. 5895–5908, 2006.

[19] P. Langehanenberg, G. Bally, and B. Kemper, "Autofocusing in digital holographic microscopy," *3D Research*, vol. 2, no. 1, 2011.

[20] F. C. Cheong, B. J. Krishnatreya, and D. G. Grier, "Strategies for three-dimensional particle tracking with holographic video microscopy," *Opt. Express*, vol. 18, no. 13, pp. 13 563–13 573, 2010.

[21] L. Dixon, F. C. Cheong, and D. G. Grier, "Holographic deconvolution microscopy for high-resolution particle tracking," *Optics express*, vol. 19, no. 17, pp. 16 410–16 417, 2011.

[22] T. Latychevskaia, F. Gehri, and H.-W. Fink, "Depth-resolved holographic reconstructions by three-dimensional deconvolution," *Optics express*, vol. 18, no. 21, pp. 22 527–22 544, 2010.

[23] P. Bouchal and Z. Bouchal, "Non-iterative holographic axial localization using complex amplitude of diffraction-free vortices," *Optics Express*, vol. 22, no. 24, p. 30200, 2014.

[24] T.-W. Su, S. O. Isikman, W. Bishara, D. Tseng, A. Erlinger, and A. Ozcan, "Multi-angle lensless digital holography for depth resolved imaging on a chip," *Optics express*, vol. 18, no. 9, pp. 9690–9711, 2010.

[25] T.-W. Su, L. Xue, and A. Ozcan, "High-throughput lensfree 3d tracking of human sperms reveals rare statistics of helical trajectories," *Proceedings of the National Academy of Sciences*, vol. 109, no. 40, pp. 16 018–16 022, 2012.

[26] I. R. Perch-Nielsen, P. J. Rodrigo, and J. Glückstad, "Real-time interactive 3d manipulation of particles viewed in two orthogonal observation planes," *Optics Express*, vol. 13, no. 8, p. 2852, 2005.

[27] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[28] J. Zemánek, "Feedback control for noise-aided parallel micromanipulation of several particles using dielectrophoresis," *Electrophoresis*, vol. 36, no. 13, pp. 1451–1458, 2015.

[29] T. Michálek, "Real-time optimization-based control for dielectrophoresis," Diploma thesis, Czech Technical University in Prague, 2015.

[30] J. Zemánek, "Noncontact parallel manipulation with micro- and mesoscale objects," Diploma thesis, Czech Technical University in Prague, 2009.

[31] J. Tomášek, "Optically controlled and sensed micromanipulation - basic setup," Bachelor's thesis, Czech Technical University in Prague, 2013.

[32] A. Ramos, *Electrokinetics and electrohydrodynamics in microsystems*. Springer, 2011, vol. 530.

[33] M. Kharboutly, M. Gauthier, and N. Chaillet, "Modeling the trajectory of a microparticle in a dielectrophoresis device," *Journal of Applied Physics*, vol. 106, no. 11, p. 114312, 2009.

[34] D. G. Duffy, *Green's Functions with Applications, Second Edition*, 2nd ed. CRC Press, 2015.

[35] S. L. Dong Eui Chang, "Closed-form solutions in the electrical field analysis for dielectrophoretic and travelling wave inter-digitated electrode arrays," *Journal of Physics D: Applied Physics*, vol. 36, no. 23, p. 3073, 2003.

[36] L. C. Evans, *Partial Differential Equations: Second Edition*, 2nd ed. American Mathematical Society, 2010.

[37] E. Wegert, *Visual Complex Functions*. Springer Basel, 2012.

[38] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, 3rd ed. Cambridge University Press, 2007.

[39] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*. Cambridge University Press, 1994.

[40] P. J. Psarrakos and M. J. Tsatsomeros, "Numerical range: (in) a matrix nutshell," 2002. [Online]. Available: http://www.math.wsu.edu/faculty/tsat/files/short.pdf

[41] F. Uhlig, "Faster and more accurate computation of the field of values boundary for n by n matrices," *Linear and Multilinear Algebra*, vol. 62, no. 5, pp. 554–567, 2014.

[42] N. A. BEBIANO, J. da Providência, A. Nata, and J. DA PROVIDÊNCIA, "Revisiting the inverse field of values problem," *Electronic Transactions on Numerical Analysis*, vol. 42, pp. 1–12, 2014.

[43] C. Johnson, "Numerical determination of the field of values of a general complex matrix," *SIAM Journal on Numerical Analysis*, vol. 15, no. 3, pp. 595–602, 1978.

[44] T. Braconnier and N. J. Higham, "Computing the field of values and pseudospectra using the lanczos method with continuation," *BIT Numerical Mathematics*, vol. 36, no. 3, pp. 422–440, 1996.

[45] L. Brickman, "On the field of values of a matrix," *Proceedings of the American Mathematical Society*, vol. 12, no. 1, pp. 61–66, 1961.

[46] R. Carden, "A simple algorithm for the inverse field of values problem," *Inverse Problems*, vol. 25, no. 11, p. 115019, 2009.

[47] M. K. Fan and A. L Tits, "On the generalized numerical range," *Linear and Multilinear Algebra*, vol. 21, no. 3, pp. 313–320, 1987.

[48] Y. T. Poon, "On the convex hull of the multiform numerical range," *Linear and Multilinear Algebra*, vol. 37, no. 1, pp. 221–223, 1994.

[49] P. Binding and C.-K. Li, "Joint ranges of hermitian matrices and simultaneous diagonalization," *Linear Algebra and Its Applications*, vol. 151, pp. 157–167, 1991.

[50] Y. H. Au-Yeung and Y. T. Poon, "A remark on the convexity and positive definiteness concerning hermitian matrices," *Southeast Asian Bull. Math*, vol. 3, no. 2, pp. 85–92, 1979.

[51] E. Gutkin, E. A. Jonckheere, and M. Karow, "Convexity of the joint numerical range: topological and differential geometric viewpoints," *Linear Algebra and its Applications*, vol. 376, pp. 143–171, 2004.

[52] S. Boyd and L. Vandenberghe, *Convex Optimization*, 1st ed. Cambridge University Press, 2004.

[53] G. C. Calafiore and L. El Ghaoui, *Optimization Models*, 1st ed. Cambridge University Press, 2014.

[54] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM review*, vol. 38, no. 1, pp. 49–95, 1996.

[55] M. Fazel, H. Hindi, and S. Boyd, "Rank minimization and applications in system theory," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 4. IEEE, 2004, pp. 3273–3278.

[56] M. Fazel, "Matrix rank minimization with applications," PhD thesis, Stanford University, 2002.

[57] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Review*, vol. 52, no. 3, pp. 471–501, 2010.

[58] M. Mesbahi and G. P. Papavassilopoulos, "On the rank minimization problem over a positive semidefinite linear matrix inequality," *Automatic Control, IEEE Transactions on*, vol. 42, no. 2, pp. 239–243, 1997.

[59] M. Fazel, H. Hindi, and S. Boyd, "Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices," in *American Control Conference, 2003. Proceedings of the 2003*, vol. 3, 2003, pp. 2156–2162 vol.3.

[60] D. P. Palomar and Y. C. Eldar, *Convex Optimization in Signal Processing and Communications*, 1st ed. Cambridge University Press, 2010.