Institute for Theoretical Physics

University of Utrecht

Master's Thesis

Computational Charged Particle Optics

Author:
ing. Thomas Verduin

Supervisors:
prof. G.T. Barkema
prof. dr. ir. P. Kruit
dr. ir. C.W. Hagen

August 21, 2013

This thesis is dedicated to the memory of my father,

who valued education more than anything.

Contents

1. Introduction

Electrostatic electron lenses consist of several rotationally symmetric electrodes at different potentials. Deviations from rotational symmetry, whether caused by unroundness of the electrodes or by misalignments between the electrodes, usually cause undesired beam aberrations and are avoided as much as possible. In some situations, of course, non-rotational symmetry is created on purpose in order to create deflection, stigmation [1] or other multi-pole effects. To create multi-pole effects, the construction usually contains multiple electrodes around the optical axis of the system, multiple examples are given in the book of Hawkes et al. [2]. A new concept proposed by Zonnevylle et al. [3] intentionally breaks the rotational symmetry of an electrostatic lens into a mirror symmetric configuration. The proposed lens consists of five electrodes where the center electrode is displaced in the lateral direction. This is what we call a 'shift lens'. This shift lens moves the focused electron beam in the same lateral direction as the displaced electrode. Applying a voltage difference between the center electrode and the neighbor electrodes displaces the focused electron beam. This configuration could be implemented for alignment purposes in a multi-beam system. This kind of beam alignment is not applied for a single beam system, because beam deflectors are much more suited and well understood. Implementing this idea in a microfabricated multi-beam system should enable the correction of misalignment in a multi-beam MEMS system by only adjusting the voltage on its central array electrode, as described by Zonnevylle et al. [3]. We remark that in a multi-beam system such as for example the MAPPER [4] machine, each beamlet column can be modeled as an individual system. To the best of our knowledge, there are no experimental or theoretical studies on the proposed shift lens. However, introductions to non-rotational systems are found in literature, for example in chapter 23 of Hawkes and Kasper [2] and Whitmer et al. [5].

There are a couple of simulation programs available for theoretical studies in electron optics. For example, EOD [6], SIMION [7], MEBS [8] and GPT [9] are commonly used in this field. The interested reader is referred to the official website[1] for a list of applications per program. An overview of simulation programs including some specifications is given in table 1.

It is Sturrock [10] who investigated the aberrations induced by small misalignments using a first order perturbative method on rotationally symmetric systems. This approach to non-rotational symmetric designs is known in the field as Sturrock's method. The proposed model by Sturrock is valid as long as the perturbation is small, giving only a first order approximation of the perturbed system with respect to the rotationally symmetric design. This method is implemented in special versions of the simulation programs developed by Munro [8] and Zĺamal [6] for the purpose of tolerance calculations. In standard rotationally symmetric systems these tolerances are the shifts and tilts caused by machine tolerances and fabrication inaccuracies. In practice, the first

[1] `http://www.lencova.com` (EOD), `http://www.simion.com` (SIMION), `http://www.mebs.co.uk` (MEBS), `http://www.pulsar.nl/gpt` (GPT).

Tool	Method	2D/3D	Tracer	Relativistic	Interactions
EOD	FEM	2D, (3D)	Runge-Kutta	?	No
SIMION	FDM	2D, 3D	Runge-Kutta	?	No
CPO3	BEM	2D, 3D	Runge-Kutta	?	No
MEBS	FEM	2D, (3D)	Runge-Kutta	Yes	Yes
GPT	external	2D, 3D	Runge-Kutta	Yes	Yes
OPERA	FEM	2D, 3D	Runge-Kutta	Yes	Yes

Table 1 – Overview of commonly used programs in the field of charged particle optics. We identified for every program the method of field calculation, which can be the finite difference method (FDM), finite element method (FEM) or boundary element method (BEM). The only program that does not support native field calculations is GPT (the user can import externally calculated fields). Programs that support perturbative approximations to symmetric solutions are labeled in parenthesis (3D).

order approximation proposed by Sturrock is sufficient, because these shifts and tilts are generally less than, or in the order of one percent of the diameter of the aperture of the modeled lenses. However, Sturrock's method is not applicable to our shift lens, because the displacement of the center electrode is much larger.

The first few attempts in simulating the shift lens are due to A.C. Zonnevylle, who discovered that most of the programs barely produce the second order aberrations, and the programs that do, take an unreasonable amount of computation time. Unfortunately, the computation time cannot be addressed, because none of the programs are designed to run natively on distributed computer systems. Conclusion: **The shift lens cannot be calculated by conventional methods.**

The challenge of this project is to design a simulator capable of producing the desired accuracy with the shift lens as an application. We have set up a list of requirements for that purpose,

- The electric field and electron trajectories must be accurate enough to produce at least the second order aberrations.
- Reduce the computation time by using massive concurrent ray-tracing of charged particles.

With respect to future theoretical studies in electron optics, the list of requirements is extended by,

- Possibility to simulate embedded systems with varying scales. For example, the design of a nanometer sized emission tip embedded in a system of centimeter size [11, 12].
- Possibility to include Coulomb-interactions (all-pairs, slice-method [13, 11, 12], Barnes-Hut [14, 15] or multi-pole methods [16]).

In order to meet the proposed requirements, we decided to implement the following ingredients to our custom built simulator,

- **Boundary element method** for calculating electrostatic fields. Our logic for this type of field calculations is based on five pillars,
  1. Superior convergence in three-dimensional electrostatic simulations, see for example Cubric et al. [17].
  2. Direct evaluation of the electric field. This is to be compared to *deriving* the electric field from the scalar potential field, which introduces an additional error due to discrete differentiation.
  3. The electric field is a *continuous function* in space, i.e. no need to interpolate the electric field at the position of the particle.
  4. This method allows a high-performance concurrent implementation.
  5. Efficient solution to embedded systems with varying scales, see for example [2].
- **Geometrical integrators** for solving the equations of motion. Our logic for this particular class of integrators is based on the advantages due to preservation of the symplectic structure, see for example Donnelly et al. [18] for a general introduction.
- **Vector parallelism** for massive concurrent ray-tracing of charged particles. We specifically focus at an implementation for modern graphics cards by NVIDIA, which can be used as low-cost high-performance vector processors. The interested reader is referred to Garland et al. [19] for a first impression with respect to scientific simulations using graphics cards by NVIDIA.

This thesis is organized as follows:

In chapter two we introduce the basics of ray-tracing relativistic charged particles in static electromagnetic fields, at first using a relativistic Lagrangian and later, via the canonical Legendre-transformation, using a conserved Hamiltonian. We simplify the Hamiltonian to meet the requirements of the shift lens, which only includes electrostatics. The electrostatic Hamiltonian is identified as separable and, based on the symplectic structure, we discuss (1) time-reversal symmetry, (2) conservation of phase-space and (3) the fourth order geometrical integrator by Blanes et al. [20] for solving the equations of motion.

The subject of chapter three is to present a theoretical framework for calculating the three-dimensional electrostatic field due to a system of electrodes. The idea is to approximate the exact boundary integral by discretizing a system of electrodes into triangles with a linearized surface charge density function per triangle. We introduce a systematic approach for solving the unknown boundary values using vertex response functions and Green's functions. This systematic approach for solving the boundary values is – for the

sake of clarity – demonstrated by an example. After the example we focus on the numerical evaluation of the electric field by transforming the boundary integral into an effective point charge representation. In the last part of this chapter we introduce boundary error analysis, which gives a measure for the quality of the discretized boundary.

In chapter four we focus on the implementation of the boundary element method specifically for vector processors. In this chapter we briefly touch the topics of memory optimizations, which includes the caching of elements, memory coalescing and ordering of data in arrays. The performance of our boundary element method is tested against two graphics cards from NVIDIA and two multi-core scalar processors from INTEL. The chapter concludes with a couple of diagrams explaining the different stages of the simulator. The first stage relates to the process of obtaining a triangulated boundary in terms of faces and vertices. The boundary values are obtained in the second stage by solving the Green's equation. Finally, two diagrams are used to explain the parallel tracing of electron trajectories using a vector processor.

In chapter five we present the design, triangulation and results of the shift lens as an example to illustrate the power of our simulation tool. At first we present our plan of calculation, which basically answers the question: ,,How are we going to calculate the aberrations?". The second section illustrates how a given design can be triangulated by hand. After that we present the results of the shift lens including an error analysis.

In the final chapters we discuss the boundary element kernel of the simulator and the results with respect to the shift lens. We also discuss the simulation of emission tips and the effect of including Coulomb interactions into the theory. After giving the conclusion of this project we present some follow up studies in the outlook based on our ideas of improvements, (1) integrating electron trajectories using explicit symplectic adaptive time-step methods, (2) improved triangulation using the multipole expansion and (3) quality mesh generation and a method for automated mesh refinement.

The thesis ends with a final remark on the presented work.

2. Basics of Relativistic Ray-tracing

The purpose of this chapter is to formulate a mathematical framework for tracing relativistic charged particles in static electromagnetic fields. At the heart of ray-tracing charged particles in time-independent fields are Maxwell's equations for electromagnetism. Instead of directly using Newton's law for the trajectories, our approach is to express the theory as a Hamiltonian where the electromagnetic fields are coupled to relativistic particles. The reason for this approach is that the Hamiltonian view is associated with symmetries, such as energy conservation and conservation of phase-space. We require these symmetries to be present in our framework such that the resulting trajectories are indeed Hamiltonian, even at the level of numerics.

Although we formulate a general theory to capture a broad range of future applications, the shift lens does not include magnetostatics. The relativistic Hamiltonian involving only electrostatics is separable and this simplifies the equations of motion. Our strategy for solving the equations of motion is to use the class of explicit geometrical integrators. This class of intregrators is special because it respects the symmetries of the Hamiltonian. We conclude that the resulting trajectories are superior to non-symplectic solutions. In particular we discuss the fourth order geometrical integrator by Blanes et al. [20]. By our requirements, we verify that this integrator indeed respects the symmetries of the Hamiltonian.

Most of the content of this chapter is part of the education of bachelor and master studies in physics and mathematics. For example, the dynamics of Hamiltonian systems are described in Goldstein [21] and – with respect to charged particles in electromagnetic fields – in Jackson [22]. More advanced theoretical topics on Hamiltonian systems, such as the symplectic structure and the conservation of phase-space (Liouville's theorem), can be found in Landau [23], Goldstein [21], Rose [24] and Donnelly [18].

2.1. Maxwell's Time-independent Equations

We briefly review Maxwell's equations for time-independent electromagnetism. Maxwell's time-independent equations decouple into two seperate realms. The realm of electrostatics is given by,

$$\epsilon_0 \nabla \cdot \mathbf{E} = \rho \tag{1}$$

$$\nabla \times \mathbf{E} = 0 \tag{2}$$

Where ρ is the charge density and ϵ_0 the permittivity of free space. Equation (2) shows that the electrostatic field is free of curl and implies that it can be written as a scalar potential,

$$\mathbf{E} = -\nabla \Phi \tag{3}$$

The remaining equations form the realm of magnetostatics and are given by,

$$\nabla \cdot \mathbf{B} = 0 \tag{4}$$
$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} \tag{5}$$

Where μ_0 is the permeability of free space. Equation (4) shows that the magnetostatic field is free of divergence and implies that it can be written as a vector potential,

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{6}$$

The equation for the trajectory of a single particle in a time-independent electromagnetic field is a textbook equation in relativistic Newtonian mechanics (see for example Jackson [22]),

$$\frac{\mathrm{d}}{\mathrm{d}t} \frac{m\dot{\mathbf{q}}}{\sqrt{1 - \frac{\dot{\mathbf{q}}^2}{c^2}}} = -\lambda \nabla \left(\Phi - \dot{\mathbf{q}} \cdot \mathbf{A} \right) \tag{7}$$

Where $\mathbf{q}$ is the position of the particle, λ the charge of the particle, m the rest mass of the particle and c the speed of light in vacuo. Although this view is physically correct, we choose to formulate the framework of ray-tracing based on the principle of energy conservation. This means that we ultimately present the theory as a relativistic Hamiltonian. Our strategy is to follow Goldstein [21] and Jackson [22] by starting with a Lagrangian for N relativistic particles coupled to the electromagnetic field. The relativistic Hamiltonian is obtained by canonical transformation.

2.2. Relativistic Lagrangian

The dynamics of a system in the language of Lagrangian-mechanics is based on the action principle,

$$\mathcal{S} = \int \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) \, \mathrm{d}t \tag{8}$$

Where $\mathcal{L} = \mathrm{K} - \mathrm{V}$ is the Lagrangian with $\mathbf{q}$ the canonical coordinate. The equations of motion are obtained by functional minimization of the action ($\delta \mathcal{S} = 0$). Our goal is to find the Lagrangian for N relativistic particles coupled to the electromagnetic field. We start with the textbook equation for the relativistic kinetic energy of a single particle,

$$\mathrm{K} = -mc^2 \sqrt{1 - \frac{\dot{\mathbf{q}}^2}{c^2}} + mc^2 = \frac{1}{2} m \dot{\mathbf{q}}^2 + \frac{3}{8} m \frac{\dot{\mathbf{q}}^4}{c^2} + \mathcal{O}\left(\dot{\mathbf{q}}^6\right) \tag{9}$$

The mc^2-term in the kinetic energy can be left out because the equations of motion are invariant to constants in the Lagrangian. We obtain the full non-interacting Lagrangian by including the potential function from the force law (7) and generalizing to a system of N particles,

$$\mathcal{L} = -\sum_{i=1}^{N} m_i c^2 \sqrt{1 - \frac{\dot{\mathbf{q}}_i^2}{c^2}} - \sum_{i=1}^{N} \lambda_i \left(\Phi(\mathbf{q}_i) - \dot{\mathbf{q}}_i \cdot \mathbf{A}(\mathbf{q}_i) \right) \tag{10}$$

Where the summation runs over N particles. Conclusively, the trajectories of N relativistic charged particles coupled to a time-independent electromagnetic field follow from functional variation,

$$\delta\int\left(\sum_{i=1}^{N} m_i c^2\sqrt{1-\frac{\dot{\mathbf{q}}_i^2}{c^2}}+\sum_{i=1}^{N}\lambda_i\left(\Phi(\mathbf{q}_i)-\dot{\mathbf{q}}_i\cdot\mathbf{A}(\mathbf{q}_i)\right)\right)\mathrm{d}t=0 \tag{11}$$

This results as expected into the force-law (7) applied to *all* particles,

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{m\dot{\mathbf{q}}_i}{\sqrt{1-\frac{\dot{\mathbf{q}}_i^2}{c^2}}}=-\lambda\nabla\left(\Phi(\mathbf{q}_i)-\dot{\mathbf{q}}_i\cdot\mathbf{A}(\mathbf{q}_i)\right) \tag{12}$$

This concludes the dynamics of relativistic ray-tracing in the Lagrangian formalism.

2.3. Relativistic Hamiltonian

We use the Lagrangian of equation (10) for constructing the relativistic Hamiltonian. At first we need to introduce the canonical momenta, which are obtained from the Lagrangian as follows,

$$\mathbf{p}_i=\frac{\partial\mathcal{L}}{\partial\dot{\mathbf{q}}_i}=\frac{m_i\dot{\mathbf{q}}_i}{\sqrt{1-\frac{\dot{\mathbf{q}}_i^2}{c^2}}}+\lambda_i\mathbf{A}(\mathbf{q}_i) \tag{13}$$

The Hamiltonian is formally obtained by applying the Legendre transformation [21],

$$\mathcal{H}=\sum_{i=1}^{N}\dot{\mathbf{q}}_i\frac{\partial\mathcal{L}}{\partial\dot{\mathbf{q}}_i}-\mathcal{L} \tag{14}$$

The equation for canonical momentum (13) is inverted and substituted into equation (14). The net result is the following non-interacting Hamiltonian for N relativistic particles coupled to a time-independent electromagnetic field,

$$\mathcal{H}=\sum_{i=1}^{N} m_i c^2\sqrt{1+\left(\frac{\mathbf{p}_i-\lambda_i\mathbf{A}(\mathbf{q}_i)}{m_i c}\right)^2}+\sum_{i=1}^{N}\lambda_i\Phi(\mathbf{q}_i) \tag{15}$$

This Hamiltonian only depends on canonical position and canonical momentum instead of the time-derivative of the canonical position. The corresponding equations of motion for this Hamiltonian are,

$$\dot{\mathbf{q}}_i=\frac{\partial\mathcal{H}}{\partial\mathbf{p}_i} \;=\; \frac{1}{\sqrt{1+\left(\frac{\mathbf{p}_i-\lambda_i\mathbf{A}(\mathbf{q}_i)}{m_i c}\right)^2}}\frac{\mathbf{p}_i-\lambda_i\mathbf{A}(\mathbf{q}_i)}{m_i} \tag{16}$$

$$\dot{\mathbf{p}}_i=-\frac{\partial\mathcal{H}}{\partial\mathbf{q}_i} \;=\; \lambda_i\dot{\mathbf{q}}_i\cdot\nabla\cdot\mathbf{A}(\mathbf{q}_i)-\lambda_i\nabla\Phi(\mathbf{q}_i) \tag{17}$$

This is identified as a system of N coupled first order differential equations. Note that the equations of motion are independent *per* particle. Although this theory includes magnetostatics, we neglect it because the shift lens does not involve magnetism. The only reason that it is mentioned here is for future applications. The general Hamiltonian without magnetostatics reduces to,

$$\mathcal{H} = \sum_{i=1}^{N} m_i c^2 \sqrt{1 + \frac{\mathbf{p}_i^2}{(m_i c)^2}} + \sum_{i=1}^{N} \lambda_i \Phi(\mathbf{q}_i) \tag{18}$$

This Hamiltonian has the following equations of motions,

$$\dot{\mathbf{q}}_i = \frac{1}{\sqrt{1 + \frac{\mathbf{p}_i}{(m_i c)^2}}} \frac{\mathbf{p}_i}{m_i} \tag{19}$$

$$\dot{\mathbf{p}}_i = -\lambda_i \nabla \Phi(\mathbf{q}_i) \tag{20}$$

Note that this electrostatic Hamiltonian (18) is separable in canonical position and momentum. In other words, it can be written as $\mathcal{H} = \mathcal{F}(\mathbf{p}_1, \ldots, \mathbf{p}_N) + \mathcal{G}(\mathbf{q}_1, \ldots, \mathbf{q}_N)$. Because the Hamiltonian is seperable, the equation for position (19) depends *only* on canonical momentum. Similarly, the equation for momentum (20) depends *only* on canonical position.

This concludes the dynamics of relativistic ray-tracing in the Hamiltonian formalism. What remains to be discussed (and exploited) are the symmetries, which are *inherently* part of Hamiltonian systems. This means that the symmetries exist, whether the equations of motion take the simplified form (19, 20) or the more generic form (16, 17).

2.4. Hamiltonian Symmetries

2.4.1. Conservation of Energy

At first we observe that the Hamiltonian is an explicit time-independent expression for the total energy of the system,

$$\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t} = \sum_{i=1}^{N} \frac{\partial \mathcal{H}}{\partial \mathbf{q}_i} \dot{\mathbf{q}}_i + \sum_{i=1}^{N} \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} \dot{\mathbf{p}}_i = \sum_{i=1}^{N} \frac{\partial \mathcal{H}}{\partial \mathbf{q}_i} \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} - \sum_{i=1}^{N} \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} \frac{\partial \mathcal{H}}{\partial \mathbf{q}_i} = 0 \tag{21}$$

Where we have used the equations of motion, $\dot{\mathbf{q}}_i = \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i}$ and $\dot{\mathbf{p}}_i = -\frac{\partial \mathcal{H}}{\partial \mathbf{q}_i}$. The second obervation is that the equations of motion are invariant with respect to the following transformation,

$$\begin{aligned} t &\rightarrow -t \\ \mathbf{p}_i &\rightarrow -\mathbf{p}_i \end{aligned} \tag{22}$$

This invariant transformation is called time-reversal symmetry and is related to conservation of energy in the following way. Suppose that we have a Hamiltonian and that we

determine the trajectory of a particle in spacetime from some initial time t to $t + \Delta t$. This trajectory is a solution to the equations of motion and is illustrated in figure 1.

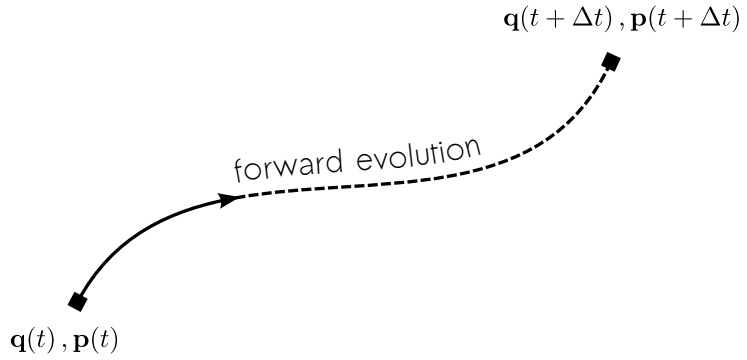

Figure 1 – Illustration of a forward evolution where the trajectory starts with a state defined at time t and advances to the state at time $t+\Delta t$. The displayed trajectory is a solution of the equations of motion for a hypothetical Hamiltonian.

We apply the time-reversal transformation rule (22) and determine the trajectory starting at time $t + \Delta t$ reversing to time t for the same Hamiltonian. This idea of reversing the trajectory is illustrated in figure 2

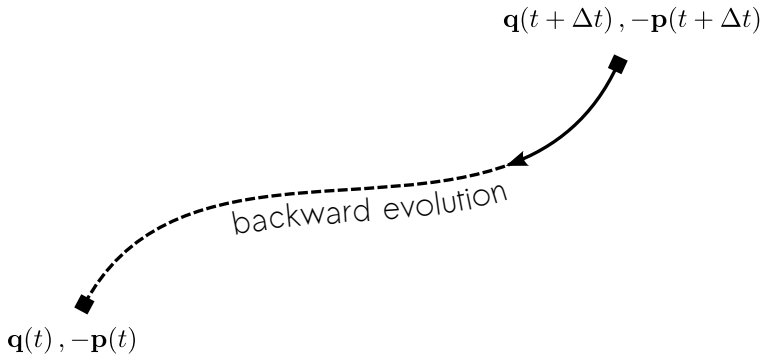

Figure 2 – Illustration of a backward evolution where the trajectory starts with the state at time $t+\Delta t$ and reverses to the state at time t. The path of the trajectory is again a solution to the equations of motion for the same hypothetical Hamiltonian as in figure 1.

The trajectories in figures 1 and 2 are *identical* because the equations of motion are invariant to the given transformation (quoting R.P. Feynman: „same equation, same solution"). This also holds for trajectories over larger spans of time, because these trajectories can be segmented into trajectories involving smaller time steps Δt.

We now argue that this time-reversal symmetry is broken by non-conservative systems. Suppose that there is a non-conservative force acting along the trajectory, which is a force that is always directed opposite to the direction of motion ($\hat{\mathbf{F}} = -\hat{\mathbf{p}}$). This means that the non-conservative force is dissipating energy in *both* the forward and backward

evolution. In other words, the dissipated energy in the forward evolution, can never be compensated for by the backward evolution. Therefore, trajectories of forward and backward evolution are not identical for non-conservative systems.

This time-reversal symmetry is not immediately apparant from the Newtonian view, because the Newtonian force-law is in general also applicable to non-conservative systems. The fact that we *can* express the theory in terms of a (conserved) Hamiltonian implies that the trajectories are symmetric with respect to time-reversal symmetry.

2.4.2. Conservation of Phase-Space

In this section we would like to illustrate the conservation property of phase-space. Suppose that we integrate the equations of motion by a *finite* amount of time. That is, we evolve from the state $\{\mathbf{q}, \mathbf{p}\}$ to $\{\mathbf{q}', \mathbf{p}'\}$ in phase-space. In general, this transformation can be written as a *map*,

$$\mathbf{q}' = \mathcal{Q}(\mathbf{q}, \mathbf{p}) \tag{23}$$
$$\mathbf{p}' = \mathcal{P}(\mathbf{q}, \mathbf{p}) \tag{24}$$

Where the functions $\mathcal{Q}$ and $\mathcal{P}$ are in general complicated non-linear functions. This mapping of states is 'area' preserving for Hamiltonian systems[2]. It is not our intention to give a *full* proof of this symmetry, which is also known as Liouville's theorem[3]. Instead, our goal is to give an intuitive first order proof for the two dimensional case, which is illustrated in figure 3.

The coordinates in a surface integral over phase-space transforms as follows,

$$\iint \cdots \mathrm{d}q\mathrm{d}p = \iint \cdots \det \mathrm{J}\, \mathrm{d}q'\mathrm{d}p' \tag{25}$$

Where J is the Jacobian matrix. The transformation is area preserving if and only if $\det \mathrm{J} = 1$. In order to show this, consider an infinitessimal area $\mathrm{d}A = \mathrm{d}q\mathrm{d}p$ in phase-space. We want to determine the area $\mathrm{d}A' = \mathrm{d}q'\mathrm{d}p'$ after an infinitessimal advancement. The linearized transformation of this element is obtained from a first order Taylor expansion of the phase-space variables and time,

$$\mathrm{d}q' = \mathrm{d}q + \frac{\partial^2 \mathcal{H}(q,p)}{\partial p \partial q}\,\mathrm{d}t\,\mathrm{d}q + \frac{\partial^2 \mathcal{H}(q,p)}{\partial p \partial p}\,\mathrm{d}t\,\mathrm{d}p \tag{26}$$
$$\mathrm{d}p' = \mathrm{d}p - \frac{\partial^2 \mathcal{H}(q,p)}{\partial q \partial q}\,\mathrm{d}t\,\mathrm{d}q - \frac{\partial^2 \mathcal{H}(q,p)}{\partial q \partial p}\,\mathrm{d}t\,\mathrm{d}p \tag{27}$$

[2] The area is conserved in two-dimensional phase-space. In multi-dimensional phase-space it is the hyper-volume that is preserved. We generically define the element $\mathrm{d}\mathbf{q}_1\,\mathrm{d}\mathbf{p}_1 \cdots \mathrm{d}\mathbf{q}_N\,\mathrm{d}\mathbf{p}_N$ in phase-space by 'area' with single quotation marks.

[3] The interested reader is referred to Landau [23] for example.

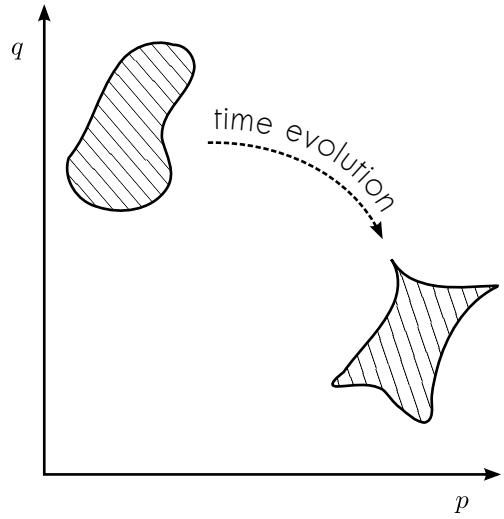

Figure 3 – This figure illustrates the conservation of phase-space in two dimensions. Phase-space reduces in two dimensions to a plot where the axes are relating to canonical momentum and canonical position. Phase-space has the structure of a symplectic manifold, which means that the mappings from one state to another are 'area' preserving. The shape and orientation of the element in this figure may change, but the area remains invariant for Hamiltonian dynamics.

Where we have used the equations of motion. This infinitessimal transformation can be written in condensed notation as follows,

$$\begin{bmatrix} \mathrm{d}q' \\ \mathrm{d}p' \end{bmatrix} = \mathrm{J} \cdot \begin{bmatrix} \mathrm{d}q \\ \mathrm{d}p \end{bmatrix} \tag{28}$$

Where J is the Jacobian matrix relating the transformation from $\mathrm{d}A$ to $\mathrm{d}A'$,

$$\mathrm{J} = \begin{bmatrix} 1 + \frac{\partial^2 \mathcal{H}(q,p)}{\partial p \partial q}\,\mathrm{d}t & \frac{\partial^2 \mathcal{H}(q,p)}{\partial p \partial p}\,\mathrm{d}t \\ -\frac{\partial^2 \mathcal{H}(q,p)}{\partial q \partial q}\,\mathrm{d}t & 1 - \frac{\partial^2 \mathcal{H}(q,p)}{\partial q \partial p}\,\mathrm{d}t \end{bmatrix} \tag{29}$$

The determinant of the Jacobian matrix evaluates to,

$$\det \mathrm{J} = 1 + \mathcal{O}\big(\delta t^2\big) \tag{30}$$

This illustrates that the linearized transformation is area preserving. In mathematical terms, phase-space has the structure of a symplectic manifold and the interested reader is referred to Hofer et al. [25] for more details on symplectic topology with respect to Hamiltonian dynamics.

2.5. Geometrical Integrators

All that remains to be discussed is a numerical strategy for solving the equations of motion (19, 20). If we consider the exact symmetries of the equations (time-reversal symmetry and conservation of phase-space), then it is desirable that a numerical approximation respects it. Such integrators exist[4], see for example the integrators of Forest [26], Yoshida [27], Omelyan [28] and Blanes [20].

Out of all possible integrators, we decided to implement the fourth order symplectic integrator by Blanes et al. because that integrator outperforms with respect to effective error [20]. There is also a sixth order version in that article, however the number of force evaluations grows rapidly and we consider the sixth order integrator not worth the extra complexity. We show that this explicit integrator respects the exact symmetries of the Hamiltonian as discussed before. We give, for the convenience of the reader, the integrator by Blanes et al. in table 2.

First of all, note that the coefficients of the integrator in table 2 are symmetric. The steps from top to bottom are identical to the reverse direction. This implies that the integrator respects time-reversal symmetry. What about conservation of phase-space? To illustrate this conservation principle we consider a small 'area' in phase-space. A small 'area' at iteration i is given by the 'square' $(\mathbf{Q}_i, \mathbf{P}_i) \rightarrow (\mathbf{Q}_i + \delta\mathbf{Q}_i, \mathbf{P}_i + \delta\mathbf{P}_i)$. An iteration involving a change in canonical position $(\mathbf{Q}_1, \ldots, \mathbf{Q}_7)$ transforms this 'area' as follows,

$$\mathbf{Q}_{i+1} = \mathbf{Q}_i + \alpha_i \left.\frac{\partial \mathcal{H}}{\partial \mathbf{p}}\right|_{\mathbf{P}_i} \delta t \tag{31}$$

$$\mathbf{Q}_{i+1} + \delta\mathbf{Q}_{i+1} = \mathbf{Q}_i + \delta\mathbf{Q}_i + \alpha_i \left.\frac{\partial \mathcal{H}}{\partial \mathbf{p}}\right|_{\mathbf{P}_i + \delta\mathbf{P}_i} \delta t \tag{32}$$

Subtracting and taking the limit of $\delta\mathbf{Q}$ and $\delta\mathbf{P}$ to zero gives,

$$\begin{bmatrix} \delta\mathbf{Q}_{i+1} \\ \delta\mathbf{P}_{i+1} \end{bmatrix} = \mathrm{J}_{\mathbf{Q}_i} \cdot \begin{bmatrix} \delta\mathbf{Q}_i \\ \delta\mathbf{P}_i \end{bmatrix} = \begin{bmatrix} 1 & \alpha_i \frac{\partial^2 \mathcal{H}}{\partial \mathbf{p} \partial \mathbf{p}} \delta t \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{Q}_i \\ \delta\mathbf{P}_i \end{bmatrix} \tag{33}$$

This explicitly shows the Jacobian matrix $\mathrm{J}_{\mathbf{Q}_i}$ relating the transformation from time t to $t + \alpha_i \delta t$, where α_i is the coefficient in the integrator for that particular step. Repeating this analysis for an iteration involving a change in canonical momentum $(\mathbf{P}_1, \ldots, \mathbf{P}_6)$ gives,

$$\begin{bmatrix} \delta\mathbf{Q}_{i+1} \\ \delta\mathbf{P}_{i+1} \end{bmatrix} = \mathrm{J}_{\mathbf{P}_i} \cdot \begin{bmatrix} \delta\mathbf{Q}_i \\ \delta\mathbf{P}_i \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \beta_i \frac{\partial^2 \mathcal{H}}{\partial \mathbf{q} \partial \mathbf{q}} \delta t & 1 \end{bmatrix} \cdot \begin{bmatrix} \delta\mathbf{Q}_i \\ \delta\mathbf{P}_i \end{bmatrix} \tag{34}$$

Note that the determinant of the Jacobian due to an update involving a change in canonical position (33) as well as canonical momentum (34) is 1 for all the individual

[4] For a general introduction to numerical symplectic methods, see Donnelly [18].

$$(\mathbf{Q}_0, \mathbf{P}_0) \;=\; (\mathbf{q}(t), \mathbf{p}(t))$$

$$\begin{aligned}
& \mathbf{Q}_1 &=\;& \mathbf{Q}_0 + 7.92036964311957\times10^{-2}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_0}}\delta t \\
\longrightarrow\quad & \mathbf{P}_1 &=\;& \mathbf{P}_0 + 2.09515106613362\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{q}}\right|_{\mathbf{Q_1}}\delta t \\
& \mathbf{Q}_2 &=\;& \mathbf{Q}_1 + 3.53172906049774\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_1}}\delta t \\
\longrightarrow\quad & \mathbf{P}_2 &=\;& \mathbf{P}_1 - 1.43851773179818\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{q}}\right|_{\mathbf{Q_2}}\delta t \\
& \mathbf{Q}_3 &=\;& \mathbf{Q}_2 - 4.20650803577195\times10^{-2}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_2}}\delta t \\
\longrightarrow\quad & \mathbf{P}_3 &=\;& \mathbf{P}_2 + 4.34336666566456\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{q}}\right|_{\mathbf{Q_3}}\delta t \\
& \mathbf{Q}_4 &=\;& \mathbf{Q}_3 + 2.19376955753499\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_3}}\delta t \\
\longrightarrow\quad & \mathbf{P}_4 &=\;& \mathbf{P}_3 + 4.34336666566456\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{q}}\right|_{\mathbf{Q_4}}\delta t \\
& \mathbf{Q}_5 &=\;& \mathbf{Q}_4 - 4.20650803577195\times10^{-2}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_4}}\delta t \\
\longrightarrow\quad & \mathbf{P}_5 &=\;& \mathbf{P}_4 - 1.43851773179818\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{q}}\right|_{\mathbf{Q_5}}\delta t \\
& \mathbf{Q}_6 &=\;& \mathbf{Q}_5 + 3.53172906049774\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_5}}\delta t \\
\longrightarrow\quad & \mathbf{P}_6 &=\;& \mathbf{P}_5 + 2.09515106613362\times10^{-1}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{q}}\right|_{\mathbf{Q_6}}\delta t \\
& \mathbf{Q}_7 &=\;& \mathbf{Q}_6 + 7.92036964311957\times10^{-2}\;\left.\frac{\partial\mathcal{H}}{\partial\mathbf{p}}\right|_{\mathbf{P_6}}\delta t
\end{aligned}$$

$$(\mathbf{q}(t+\delta t), \mathbf{p}(t+\delta t)) \;=\; (\mathbf{Q}_7, \mathbf{P}_6)$$

Table 2 – The fourth order symplectic integrator by Blanes et al. [20] with coefficients up to 15 significant digits. This integrator iterates by successive steps from some initial state in phase-space at time t to time $t + \delta t$. Each step involves either a change in canonical position or canonical momentum. The arrows identify the steps involving a force calculation. There are in total six force evaluations required to update a state in phase-space using this integrator. Remarkably, but typical for higher order symplectic methods, some of the steps go backwards in time.

iterations. The transformation for a complete update involving *all* iterations is written as the product of the individual transformations,

$$\begin{bmatrix}\delta\mathbf{Q}_i \\ \delta\mathbf{P}_i\end{bmatrix} = \mathrm{J}_{\mathbf{Q}_7}\cdot\mathrm{J}_{\mathbf{P}_6}\cdots\mathrm{J}_{\mathbf{P}_1}\cdot\mathrm{J}_{\mathbf{Q}_1}\cdot\begin{bmatrix}\delta\mathbf{Q}_{i-1} \\ \delta\mathbf{P}_{i-1}\end{bmatrix} \tag{35}$$

The determinant of the complete update evaluates to a product of determinants,

$$\det\left(\mathrm{J}_{\mathbf{Q}_7}\cdot\mathrm{J}_{\mathbf{P}_6}\cdots\mathrm{J}_{\mathbf{P}_1}\cdot\mathrm{J}_{\mathbf{Q}_1}\right)=\det\mathrm{J}_{\mathbf{Q}_7}\det\mathrm{J}_{\mathbf{P}_6}\cdots\det\mathrm{J}_{\mathbf{P}_1}\det\mathrm{J}_{\mathbf{Q}_1}=1 \tag{36}$$

Where we have used the multiplicative mapping property of the determinant. We conclude that this integrator also respects conservation of phase-space.

The reader might have noticed that conservation of phase-space does *not* depend on the specific coefficients in tabel 2. This means that we could have picked *any* random set of symmetric coefficients without destroying the symmetries. Although that is true, we emphasize that the coefficients by Blanes et al. are special because they give a *fourth order* accurate integration with a particular low *effective error*. The interested reader is referred to the original article by Blanes et al. [20] for more information on how these coefficients are obtained. Actually, the trajectories of a numerical symplectic integrator are equal to the exact dynamics of a *shadow* Hamiltonian [18]. For example, the dynamics obtained by a fourth order symplectic integrator are *exact* with respect to a Hamiltonian of the following form,

$$\widetilde{\mathcal{H}}=\mathcal{H}+(\cdots)\delta t^4+\mathcal{O}\big(\delta t^5\big) \tag{37}$$

Where $\widetilde{\mathcal{H}}$ is the shadow Hamiltonian, $\mathcal{H}$ the actual Hamiltonian and the term in parenthesis is generically a function of phase-space variables.

The advantage of symplectic methods is that they offer global stability. The area is bounded by adjacent trajectories, and thus the coordinates (and hence the energy) cannot increase (or decrease) without bound. Even in better non-symplectic approximations, such as the classical Runge-Kutta integrator, the energy deviates significantly from the initial value at sufficiently long times.

3. Calculation of Electrostatic Fields

The equations of motion (19, 20) can only be calculated if the scalar potential function Φ is known. The subject of this chapter is to introduce the reader to calculating (approximations to) electrostatic fields using the boundary element method. Electrostatics fields are derived from the potential function, which in the abscence of space-charge is a solution to the Laplace-equation,

$$\nabla^2\Phi = 0 \tag{38}$$

The solution to this equation is unique for a given Dirichlet boundary condition. There are three ways to numerically approximate the potential function, which we briefly discuss.

- **Finite discretization methods** are based on approximating the Laplace operator (∇^2) on a grid using discrete derivatives. The on-grid potentials are typically found by relaxation methods, such as Gauss-Seidel iterations. This type of field calculation using spherical coordinates was used in a premature development version of our ray-tracing simulator, see for example the article of Cook et al. [11].
- **Finite element methods** discretize the solution domain into a finite number of geometrical elements, typically triangles (in 2D) or tetrahedrons (in 3D). Each element is associated with an approximating function, for example an n-th order polynomial. The coefficients are obtained by minimizing the potential energy functional.
- **Boundary element methods** use the integral form of the Laplace equation, which is an exact solution to the differential form. The idea is to fit boundary values into the integral equation, rather than potentials or coefficients throughout space. Once this is done, the integral equation is used again to numerically calculate the potential (or electric field) anywhere in the solution domain. This type of field calculation is used in the current development version of the ray-tracing simulator, see for example the article of Verduin et al. [12].

We have implemented the boundary element because of the following reasons (already mentioned in the introduction), (1) superior convergence in three dimensional electrostatic simulations, (2) direct evaluation of the electric field (instead of numerically approximating the derivative of the potential), (3) the field solution is continuous throughout space, (4) allows efficient concurrent implementation for vector processors, and (5) potential to address embedded systems, i.e. systems with components at different scales.

This section is organized as follows:

We start by introducing the exact boundary integral equation for a generic system of n electrodes. The boundary integral is discretized into triangles and we express the surface charge density function per triangle as a linear interpolation in terms of boundary values. We present the functions for evaluating the boundary and discuss a strategy for evaluating the integrals numerically. The boundary functions are used to construct

Green's matrix which relates the boundary values to the boundary potentials. This leads to Green's equation, which is solved in order to obtain the boundary values. We give, after the Green's equation, an elementary example to demonstrate the boundary element method. After the example we focus on the numerical evaluation of the electric field by transforming the boundary integral into an effective point charge representation. Finally, we introduce boundary error analysis in order to estimate the quality of a given discretized boundary.

3.1. Boundary Integral Equation

Suppose that we have N electrodes in space, where each electrode has a static charge distribution. The potential field in space due to one electrode is determined by integrating Coulomb's law, which is illustrated in figure 4. The exact scalar potential due to an arbitrary system of electrodes is determined by the superposition principle,

$$\Phi(\mathbf{R}) = \sum_{i=1}^{n} \iiint_{\mathcal{V}_i} \frac{\rho_i(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}V \tag{39}$$

Where n is the number of electrodes and ρ is the volumic charge density. The index i enumerates the individual electrodes in the system.

In electrostatics, however, the charges of interest are on the boundary [22], which is illustrated in figure 5. The integral equation can therefore be written as a boundary integral,

$$\Phi(\mathbf{R}) = \sum_{i=1}^{n} \phi_i(\mathbf{R}) = \sum_{i=1}^{n} \iint_{\mathcal{S}_i} \frac{\sigma_i(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A \tag{40}$$

Where σ is the surface charge density and ϕ is the potential due to an individual electrode. Note that Coulomb's law (except for $\mathbf{r} = \mathbf{0}$) is an exact solution to the Laplace equation,

$$\nabla^2 \frac{1}{|\mathbf{R}|} = -\nabla \frac{\mathbf{R}}{|\mathbf{R}|^3} = \frac{|\mathbf{R}|^2}{|\mathbf{R}|^5} - \frac{1}{|\mathbf{R}|^3} = 0 \tag{41}$$

By construction, any surface integral with Coulomb's law is also an exact solution to the Laplace equation,

$$\nabla^2 \iint_{\mathcal{S}} \frac{1}{|\mathbf{R}|}\,\mathrm{d}A = \iint_{\mathcal{S}} \nabla^2 \frac{1}{|\mathbf{R}|}\,\mathrm{d}A = 0 \tag{42}$$

The boundary integral given by equation (40) is therefore also an exact solution to the Laplace equation $\nabla^2\Phi = 0$.

The problem of the boundary element method is to determine (approximate) the surface charge density function in the boundary integral (40). This surface charge density function is constrained, because the boundary integral must evaluate the boundary to

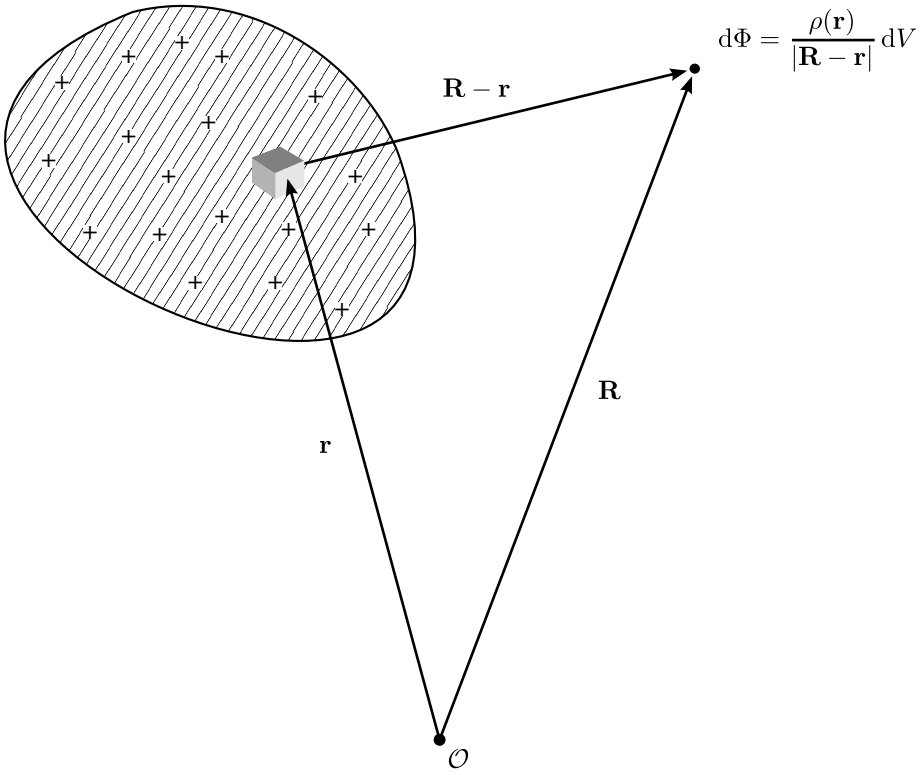

Figure 4 – Schematic diagram for calculating the electrostatic potential field in general. The figure illustrates a solid body in space relative to the origin $\mathcal{O}$. This body has some static charge distribution, which is given as a volumic charge density. The potential field at $\mathbf{R}$ is obtained by integrating the differential for the entire body.

the potentials of the electrodes. In general, obtaining an analytical expression for the surface charge density function is far too complicated. In order to handle *any* electrode configuration we split the boundary into smaller and more managable geometrical elements. Without loss of generality, we can express the boundary integral as a limit of infinite triangles[5].

$$\phi_i(\mathbf{R}) = \lim_{n_i\to\infty} \sum_{j=1}^{n_i} \varphi_{i,j}(\mathbf{R}) = \lim_{n_i\to\infty} \sum_{j=1}^{n_i} \iint_{\triangle_{i,j}} \frac{\sigma_{i,j}(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A \tag{43}$$

Where n_i is the number of triangles and the index j enumerates the triangles for the electrode of consideration. The potential due to an individual triangle is given by the function φ. To be precise, $\varphi_{i,j}(\mathbf{R})$ means the potential at $\mathbf{R}$ due to the j-th triangle from

[5] The reader might wonder why triangles? The reason is three folded, (1) triangulation of surfaces is a well-established field of research in computational geometry, (2) any surface can be approximated by triangles and (3) triangular elements are computationally managable objects.

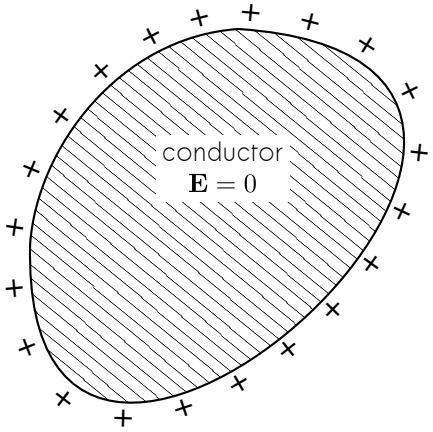

Figure 5 – This figures illustrates that the charges of interest are on the boundary in electrostatics. The charge distribution on the surface is given as a surface charge density. This means that the potential is obtained by integrating over the surface instead of integrating the entire body.

the i-the electrode. This limit of triangles converges because the integration measure in equation (40) can be splitted into two triangles. The exact boundary integral in discrete numerics is approximated by a *finite* number of triangles. An explicit example of triangulation by hand is given in the chapter of the shift lens.

Cook

3.2. Vertex Response Functions

Although the boundary is discretized into triangles, the surface charge density function $\sigma(\mathbf{r})$ of each element is still an analytical function. We focus on one element and, for the sake of simplicity, neglect the electrode enumeration and use the simplified notation $\varphi_{i,j} \to \varphi$. Consider a triangular element of the boundary, which is illustrated in figure 6. The vertices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ of the triangle are associated with scalar potentials Φ_A, Φ_B and Φ_C and unknown boundary values σ_A, σ_B and σ_C.

We follow Nielson [29] and define the linear interpolated charge density on the surface of the triangle in terms of the boundary values,

$$\sigma(\mathbf{r}) \approx \sigma_A \gamma_A(\mathbf{r}) + \sigma_B \gamma_B(\mathbf{r}) + \sigma_C \gamma_C(\mathbf{r}) \tag{44}$$

The functions γ_A, γ_B and γ_C are maps from spatial coordinates to barycentric coordinates. These coordinate mappings have the following property,

$$\begin{array}{rclrclrcl} \gamma_A(\mathbf{A}) & = & 1, & \gamma_B(\mathbf{A}) & = & 0, & \gamma_C(\mathbf{A}) & = & 0 \\ \gamma_A(\mathbf{B}) & = & 0, & \gamma_B(\mathbf{B}) & = & 1, & \gamma_C(\mathbf{B}) & = & 0 \\ \gamma_A(\mathbf{C}) & = & 0, & \gamma_B(\mathbf{C}) & = & 0, & \gamma_C(\mathbf{C}) & = & 1 \end{array} \tag{45}$$

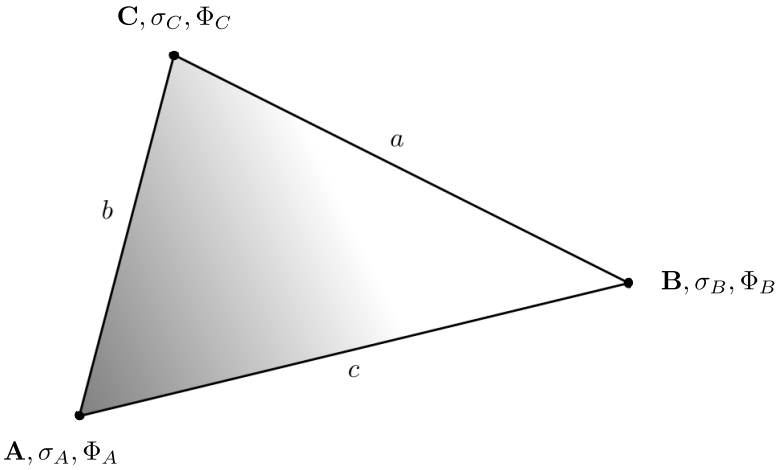

Figure 6 – Definition of triangular elements with a linearly interpolated charge density function. The vertices **A**, **B** and **C** are associated with a predefined potential Φ and an unknown boundary value σ. The charge density function of this triangle is obtained by linear interpolation of the boundary values. Each side has a length given by the small roman letter a, b or c.

Such that $\sigma(\mathbf{A}) = \sigma_A$, $\sigma(\mathbf{B}) = \sigma_B$ and $\sigma(\mathbf{C}) = \sigma_C$. The scalar potential at position $\mathbf{R}$ due to a triangle with linear interpolated charge density is obtained by substituting the linearized charge density function into the surface integral,

$$\varphi(\mathbf{R}) \approx \sigma_A \iint_{\triangle} \frac{\gamma_A(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A + \sigma_B \iint_{\triangle} \frac{\gamma_B(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A + \sigma_C \iint_{\triangle} \frac{\gamma_C(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A \tag{46}$$

This can be written in condensed notation as follows,

$$\varphi(\mathbf{R}) = \sigma_A g_A(\mathbf{R}) + \sigma_B g_B(\mathbf{R}) + \sigma_C g_C(\mathbf{R}) \tag{47}$$

Where the boundary values are seperated from and multiplied by vertex response functions,

$$g_A(\mathbf{R}) = \iint_{\triangle} \frac{\gamma_A(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A \tag{48}$$

$$g_B(\mathbf{R}) = \iint_{\triangle} \frac{\gamma_B(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A \tag{49}$$

$$g_C(\mathbf{R}) = \iint_{\triangle} \frac{\gamma_C(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A \tag{50}$$

These functions are interpreted as the scalar potential response at position $\mathbf{R}$ due to a unit boundary value at one of the vertices **A**, **B** or **C** and the other boundary values zero.

The general practice of these response functions is to evaluate the potential at the vertices of triangles, because that is where we have defined the boundary potential. We have

tried to determine general closed-form expressions for the response functions. Although we think that it is possible to do, we consider it not worth the extra complexity when compared to naive adaptive numerical integration. The only problem in numerics arises when we evaluate the *self* potential at one of the vertices, because the integrand is divergent for that case. The only satisfying solution[6] is to find analytical closed-form expressions. This turns out to be a tedious calculation (the interested reader is referred to appendix A). The resulting (coordinate invariant) closed-form expressions for the potential at vertex **A** evaluate to,

$$g_A(\mathbf{A}) = \frac{\triangle}{a}\log\frac{s}{s-a} \tag{51}$$

$$g_B(\mathbf{A}) = \frac{\triangle}{a}\left(\frac{1}{2}\frac{a^2+b^2-c^2}{a^2}\log\frac{s}{s-a} - \frac{b-c}{a}\right) \tag{52}$$

$$g_C(\mathbf{A}) = \frac{\triangle}{a}\left(\frac{1}{2}\frac{a^2-b^2+c^2}{a^2}\log\frac{s}{s-a} + \frac{b-c}{a}\right) \tag{53}$$

Where $\triangle$ is the area and s the semiperimeter of the triangle. The potential at any of the other vertices is obtained by relabeling the vertices and edges. To our best knowledge, these closed-form expressions (51), (52) and (53) are not known and especially not for the linearly interpolated charge density function. For completeness we mention the response functions for the uniform distributed triangle,

$$g(\mathbf{R}) = g_A(\mathbf{R}) + g_B(\mathbf{R}) + g_C(\mathbf{R}) = \iint_{\triangle}\frac{\mathrm{d}A}{|\mathbf{R}-\mathbf{r}|} \tag{54}$$

$$g(\mathbf{A}) = 2\frac{\triangle}{a}\log\frac{s}{s-a} = 2g_A(\mathbf{A}) \tag{55}$$

This concludes the element analysis, which is used to determine the unknown boundary values.

3.3. Construction of Green's Matrix

We now present a method to determine the unknown boundary values, such that the vertices of the entire boundary have the desired potential. First of all, note that the boundary integral (40) after triangulation reads,

$$\Phi(\mathbf{R}) = \sum_{i=1}^{n}\sum_{j=1}^{n_i}\varphi_{i,j}(\mathbf{R}) \tag{56}$$

[6] Other solutions involve evaluating at a height h, which regularizes the divergent part of the integral [2]. This introduces artificial boundaries: Instead of one physical boundary, we would have two boundaries. The first boundary defines the charge density function such that on *another* boundary (at height h) the potential is fixed.

Where the first summation runs over all electrodes and the second summation runs over all triangles of that particular electrode. We drop, without loss of generality, the electrode enumeration and instead enumerate over the whole collection of triangles,

$$\Phi(\mathbf{R}) = \sum_{i=1}^{\tilde{n}} \varphi_i(\mathbf{R}) \tag{57}$$

Where the total number of triangles $\tilde{n}$ equals $\sum_{i=1}^{n} n_i$. Remember that for the potential contribution per triangle (47), the boundary values are multiplied linearly by vertex response functions. This means that we can seperate out the boundary values from the response functions for *all* elements and group shared boundary values together, i.e.,

$$\Phi(\mathbf{R}) = \sum_{i=1}^{M} G_i(\mathbf{R})\sigma_i = \begin{bmatrix} G_1(\mathbf{R}) & \cdots & G_M(\mathbf{R}) \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_M \end{bmatrix} \tag{58}$$

Where we have introduced Green's function $G_i(\mathbf{R})$ into the equation. The purpose of the Green's function is to evaluate the potential at $\mathbf{R}$ due to *all* elements with a unit boundary value located at vertex i and all other boundary values are zero. Note that the summation runs up to M, which defines the number of unique vertices[7]. The pseudocode for evaluating the Green's function is given in algorithm 1.

Algorithm 1 Pseudocode for calculating Green's function.

```
σ_i = 1
σ_j = 0   ∀j ≠ i

G_i(R) = 0
for all boundary elements do
    G_i(R) += potential at R due to boundary element.
end for
```

Now we construct a system of equations by evaluating the Green's function (58) at *all* unique vertices,

$$\begin{aligned} \Phi(\mathbf{R}_1) &= \textstyle\sum_{i=1}^{M} G_i(\mathbf{R}_1)\sigma_i \\ &\vdots \\ \Phi(\mathbf{R}_M) &= \textstyle\sum_{i=1}^{M} G_i(\mathbf{R}_M)\sigma_i \end{aligned} \tag{59}$$

We evaluate at the vertices because that is precisely where we have defined the potential. This system of equations can be written in condensed notation as follows,

$$\begin{bmatrix} G_{1,1} & \cdots & G_{1,M} \\ \vdots & \ddots & \\ G_{M,1} & & G_{M,M} \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_M \end{bmatrix} = \begin{bmatrix} \Phi_1 \\ \vdots \\ \Phi_M \end{bmatrix} \tag{60}$$

[7] Connected triangles have shared vertices and thus also shared boundary values.

The square matrix is the Green's matrix and relates the boundary values to the vertex potentials. Here $G_{i,j}$ is short notation for $G_j(\text{vertex}i)$. Note that this is a system of M equations with M unknowns. We obtain the unknown boundary values by solving the matrix equation[8],

$$\mathbf{G} \cdot \mathbf{S} = \mathbf{P} \tag{61}$$

This equation can be solved if the determinant is unequal to zero and has the following solution,

$$\mathbf{S} = \mathbf{G}^{-1} \cdot \mathbf{P} \tag{62}$$

This concludes the abstract construction of Green's matrix and solving for the unknown boundary values.

3.3.1. Elementary Example

There is nothing that clarifies more than an elementary example to illustrate the boundary element method. We show how to construct Green's matrix for the case of a square boundary expressed as two connected triangles. Suppose that we discretized a square boundary into two connected triangles with four unique vertices as illustrated in figure 7. Every vertex is associated with a predefined potential Φ and an unknown boundary value σ.

The boundary integral (57) for this particular case evaluates to a sum of two triangles,

$$\Phi(\mathbf{R}) = \varphi^{(1)}(\mathbf{R}) + \varphi^{(2)}(\mathbf{R}) \tag{63}$$

Where upperscript in parenthesis identifies the triangle. The next step is to group the boundary values from the boundary integral,

$$\begin{aligned}
\Phi(\mathbf{R}) &= \varphi^{(1)}(\mathbf{R}) + \varphi^{(2)}(\mathbf{R}) \\
&= g_1^{(1)}(\mathbf{R})\sigma_1 + g_2^{(1)}(\mathbf{R})\sigma_2 + g_3^{(1)}(\mathbf{R})\sigma_3 + g_2^{(2)}(\mathbf{R})\sigma_2 + g_3^{(2)}(\mathbf{R})\sigma_3 + g_4^{(2)}(\mathbf{R})\sigma_4 \\
&= g_1^{(1)}(\mathbf{R})\sigma_1 + \Big(g_2^{(1)}(\mathbf{R}) + g_2^{(2)}(\mathbf{R})\Big)\sigma_2 + \Big(g_3^{(1)}(\mathbf{R}) + g_3^{(2)}(\mathbf{R})\Big)\sigma_3 + g_4^{(2)}(\mathbf{R})\sigma_4
\end{aligned}$$

This expression evaluates in accordance with equation (58) to,

$$\Phi(\mathbf{R}) = G_1(\mathbf{R})\sigma_1 + G_2(\mathbf{R})\sigma_2 + G_3(\mathbf{R})\sigma_1 + G_4(\mathbf{R})\sigma_4 \tag{64}$$

Where the Green's functions are expressed as follows,

$$G_1(\mathbf{R}) = g_1^{(1)}(\mathbf{R}) \tag{65}$$

$$G_2(\mathbf{R}) = g_2^{(1)}(\mathbf{R}) + g_2^{(2)}(\mathbf{R}) \tag{66}$$

$$G_3(\mathbf{R}) = g_3^{(1)}(\mathbf{R}) + g_3^{(2)}(\mathbf{R}) \tag{67}$$

$$G_4(\mathbf{R}) = g_4^{(2)}(\mathbf{R}) \tag{68}$$

[8] Matrix equations can be solved numerically using multi-core implementations of LAPACK, for example, ATLAS (`http://math-atlas.sourceforge.net/`) by Whaley et al. [30] and the Math Kernel Library by Intel (`http://software.intel.com/en-us/intel-mkl`).

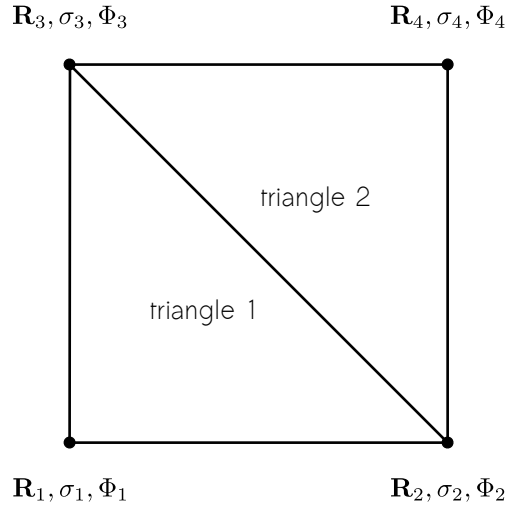

Figure 7 – This figure illustrates a square boundary discretized as two triangles. The corners of this square are fixed at a particular potential. Note that the boundary values σ_2 and σ_3 are shared by both triangles. This system is used as an example to demonstrate the construction of Green's matrix. In that example, the boundary values σ_1, σ_2, σ_3 and σ_4 are determined such that the potential at the corners $\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}_3$ and $\mathbf{R}_4$ evaluate to the corresponding potentials.

This shows that the Green's functions for the shared vertices (66) and (67) are constructed by vertex response functions from *both* triangles. We obtain the Green's matrix equation for this sytem by evaluating (64) at *every* unique vertex,

$$\begin{bmatrix} G_1(\mathbf{R}_1) & G_2(\mathbf{R}_1) & G_3(\mathbf{R}_1) & G_4(\mathbf{R}_1) \\ G_1(\mathbf{R}_2) & G_2(\mathbf{R}_2) & G_3(\mathbf{R}_2) & G_4(\mathbf{R}_2) \\ G_1(\mathbf{R}_3) & G_2(\mathbf{R}_3) & G_3(\mathbf{R}_3) & G_4(\mathbf{R}_3) \\ G_1(\mathbf{R}_4) & G_2(\mathbf{R}_4) & G_3(\mathbf{R}_4) & G_4(\mathbf{R}_4) \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \end{bmatrix} = \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \end{bmatrix} \tag{69}$$

This equation can be solved for the unknown boundary values σ_1, σ_2, σ_3 and σ_4.

3.4. Point Charge Representation

What we have not discussed so far is *how* to actually numerically calculate the electric field from the vertex response functions. First of all, note that the electric field is determined from the vertex response functions by applying the gradient operator,

$$\mathbf{E}(\mathbf{R}) = -\sigma_A \nabla g_A(\mathbf{R}) - \sigma_B \nabla g_B(\mathbf{R}) - \sigma_C \nabla g_C(\mathbf{R}) \tag{70}$$

Where the *vectorial* response functions evaluate to,

$$\nabla g_A(\mathbf{R}) = -\iint_{\triangle} \gamma_A(\mathbf{r}) \frac{\mathbf{R}-\mathbf{r}}{|\mathbf{R}-\mathbf{r}|^3} \,\mathrm{d}A \tag{71}$$

$$\nabla g_B(\mathbf{R}) = -\iint_{\triangle} \gamma_B(\mathbf{r}) \frac{\mathbf{R}-\mathbf{r}}{|\mathbf{R}-\mathbf{r}|^3} \,\mathrm{d}A \tag{72}$$

$$\nabla g_C(\mathbf{R}) = -\iint_{\triangle} \gamma_C(\mathbf{r}) \frac{\mathbf{R}-\mathbf{r}}{|\mathbf{R}-\mathbf{r}|^3} \,\mathrm{d}A \tag{73}$$

At first glance, we are tempted to integrate (at least) one dimension out of the response functions. This implies that the boundary response functions change from a definite double integral into a difference of single definite integrals,

$$\iint_{\mathcal{S}} f \,\mathrm{d}A \rightarrow \int_{\ldots}^{\ldots} F \,\mathrm{d}l - \int_{\ldots}^{\ldots} F \,\mathrm{d}l \tag{74}$$

There is a catch however. Suppose that we want to evaluate the response functions for a point far away from the triangle. In that case, the two contributions from the single integrals are close to equal. Subtracting two numbers close to equal with finite precision results in a loss of significant digits [31]. We do not prefer this approach because this is the most common application for evaluating the vertex response functions. Instead, we suggest to use direct two dimensional quadrature to approximate the double integrals[9]. This means that, by using Gaussian-qudrature[10], the electric field per triangle is approximated as,

$$-\nabla\varphi(\mathbf{R}) = \iint_{\triangle} \sigma(\mathbf{r}) \frac{\mathbf{R}-\mathbf{r}}{|\mathbf{R}-\mathbf{r}|^3} \,\mathrm{d}A \approx \sum_{i=1}^{n}\sum_{j=1}^{m} w_{i,j}\sigma(\mathbf{r}_{i,j}) \frac{\mathbf{R}-\mathbf{r}_{i,j}}{|\mathbf{R}-\mathbf{r}_{i,j}|^3} \tag{75}$$

Where the double integrals are replaced by double summation, the discrete vectors $\mathbf{r}_{i,j}$ are the abscissae of integration and $w_{i,j}$ are the corresponding weights [32, 33]. The *order* of integration is fixed[11] and is related to the summation counts n and m.

[9] By no means do we imply that this is the best methodology. We consider this approach convenient and sufficient for this thesis. Another way to tackle the integrals is to use the more sophisticated multipole expansion, which is discussed in the outlook.

[10] This is a very commonly applied technique in the field of numerical analysis for approximating integrals. See for example Babolian et al. [32] and Laurie [33].

[11] Why not adapative integration? The reason is that the integrals of different triangles are not treated equally by adapative integration. For instance, one integral may require more evaluations than another and therefore adapative integration is not really well-suited for vector processors.

Observe that the integral in (75) is reduced to nothing but the evaluation of a superposition of point charges located at the abscissae where each point charge has an effective charge of,

$$\tilde{\lambda}_{i,j} = w_{i,j}\sigma(\mathbf{r}_{i,j}) \tag{76}$$

We emphasize that this approach replaces *every* triangle by point charges. For instance, the 4-point Gauss-Legendre rule applied to both dimensions (summation count $n = m$) replaces each triangle by $4 \times 4 = 16$ effective point charges. The complete collection of effective point charges can be determined ab initio for the entire boundary. In other words, the net electric field is determined by,

$$\mathbf{E}(\mathbf{R}) = -\nabla\Phi(\mathbf{R}) \approx \sum_{k=1}^{\tilde{M}} \tilde{\lambda_k} \frac{\mathbf{R} - \tilde{\mathbf{r}}_k}{|\mathbf{R} - \tilde{\mathbf{r}}_k|^3} \tag{77}$$

Where the summation runs over *all* effective point charges. The total number of point charges $\tilde{M}$ equals the number of triangles $\times$ the number of effective point charges per triangle.

3.5. Boundary Error Analysis

In the last part of the chapter on calculating electric fields we would like to introduce the subject of error analysis. To that order, consider the elementary example presented in figure 7. The boundary values obtained from equation (69) are such that the scalar potential is fixed *at* the vertices. What about the potential *away* from the vertices anywhere in the interior? In general, if we define $\Phi_1 = \Phi_2 = \Phi_3 = \Phi_4 = \Phi$ then the potential in the interior is *not* equal to Φ. Therefore, the idea that we pursue is that the calculated potential in respect to the expected potential on the surface is a direct measure for the quality of the boundary solution. To accomplish this, we need to evaluate the vertex response functions for points *inside* a triangle because that is what our boundary is made of. If we resort on numerics, then we face the problem of divergent integrals. The other way is to derive an analytical solution, which in general gives expressions too complicated to manage.

Although it appears that we are stuck, we do have a solution to this problem: we can find the potential for interior points by *construction*. This method works as follows. Suppose that we want to calculate the potential at the incenter of the triangle[12]. We insert a point iat the incenter such that the total area of the triangle is splitted into three triangles with equal area (figure 8).

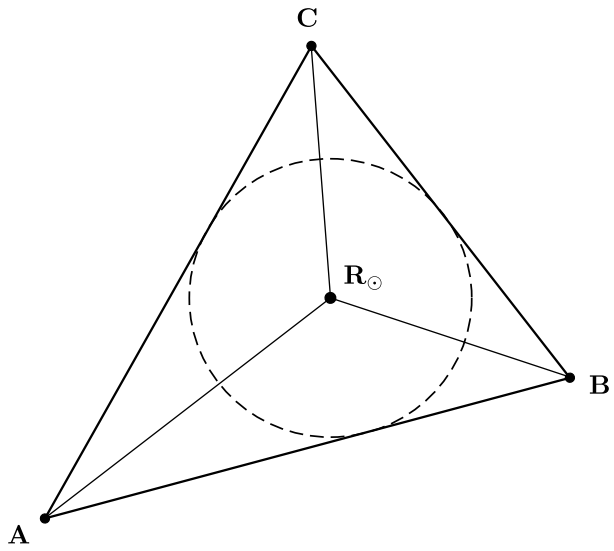

Figure 8 – Illustration of splitting a triangle into three connected triangles. This triangle is splitted by inserting a point at the incenter. Note that this point is shared as a vertex by all three triangles. Although we choose the incenter, the presented method works for any point in the interior of the triangle. The incenter is convenient because it equally divides the area.

[12] This method works for *any* point in the interior of the triangle. The incenter is *convenient* because it equally divides the area.

Observe that the incenter of the original triangle is located *at* the vertices of the smaller triangles. The problem is solved because we do know how to calculate analytically the potential at the vertex of a triangle, see equations (51), (52) and (53). The potential at the incenter is obtained by the principle of superposition (figure 9).

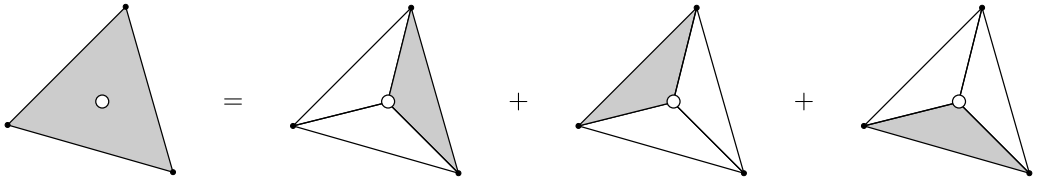

Figure 9 – This figure illustrates that the potential for an internal point can be obtained by construction. The triangle of interest is splitted into three new triangles, where each triangle shares the point of interest as a vertex. The potential at this vertex is evaluated analytically per triangle. The net potential is obtained by the superposition principle, i.e. adding the potential contribution from all three triangles at this vertex.

All that we need is a *pseudo* boundary value for the incenter, which can be obtained from equation (44). For the special case of the incenter, the pseudo boundary value evaluates to,

$$\sigma_{\odot} = \frac{1}{2}\frac{\sigma_A a + \sigma_B b + \sigma_C c}{s} \tag{78}$$

In practice, the boundary error is measured by obtaining the calculated potential (after solving) for *all* triangles and determine the deviation from the expected potential. The triangulation of the boundary needs to be reconsidered if the boundary solution deviates substantially from a predefined tolerance[13]. At the moment we do this by trial and error[14]. In the outlook we discuss automated mesh refinement.

[13] In principle, this can be defined per triangle such that special regions of the boundary are subjected to stricter tolerances.

[14] We know for instance that the largest errors are found in the *sharp* edges of the boundary. Therefore we manually apply mesh refinement to those areas in particular.

4. Implementation

In this chapter we focus on the implementation of the boundary element method. At fist we show that boundary element method in general can be stated in the natural parallel language of a vector processor, such as graphics cards. Roughly speaking, the power of the vector processor comes from the fact that there are *many* threads running concurrently, essentially maximizing parallism at the cost of light weight arithmetic. The implemenation is optimized further by considering several memory strategies, such as (1) caching structures, (2) utilizing memory coalescing and (3) optimized ordering of data in arrays. We implement the boundary element method using the effective point charge representation and demonstrate the performance by testing against two graphics cards from NVIDIA and two multi-core scalar processors from INTEL.

The chapter concludes with the three processing stages of the simulator. The first stage exports a triangulated boundary in terms of faces and vertices. These faces and vertices are the input of the second stage, which constructs the Green's matrix and solves for the unknown boundary values. The net result of that module is a boundary solution and is used in the third stage which calculates the trajectories of predefined particles using the vector processor as a device for evaluating the electric-field.

4.1. Boundary Element Kernel

4.1.1. Vector Processing

At first we investigate the arithmetic structure of the boundary element method. Suppose that we have N particles distributed in space. This system of N particles can only be updated if we calcualte the electric field at *every* particle. We do this by evaluating equation (20) applied to all particles. The problem of calculating the electric field at every particle can be stated in matrix format as follows,

Field		Element 1		Element 2		$\cdots$		Element M
$\mathbf{E}(\mathbf{q}_1)$	$=$	$-\nabla\varphi_1(\mathbf{q}_1)$	$+$	$-\nabla\varphi_2(\mathbf{q}_1)$	$+$	$\cdots$	$+$	$-\nabla\varphi_M(\mathbf{q}_1)$
$\mathbf{E}(\mathbf{q}_2)$	$=$	$-\nabla\varphi_1(\mathbf{q}_2)$	$+$	$-\nabla\varphi_2(\mathbf{q}_2)$	$+$	$\cdots$	$+$	$-\nabla\varphi_M(\mathbf{q}_2)$
	$\vdots$							
$\mathbf{E}(\mathbf{q}_N)$	$=$	$-\nabla\varphi_1(\mathbf{q}_N)$	$+$	$-\nabla\varphi_2(\mathbf{q}_N)$	$+$	$\cdots$	$+$	$-\nabla\varphi_M(\mathbf{q}_N)$

(79)

Where φ is the potential due to an individual element of the boundary. There are two

significant observations. First of all, note that every row can be calculated independently. This means that parallelism is applicable with respect to individual particles. The second observation is that every column represents a single boundary element, which is evaluated for multiple particles. In terms of arithmetic, the evaluation of a boundary element is a *static* algorithm and involves in our case[15] a constant number of operations (see algorithm 2). This type of parallelism is formally classified as single instruction, multiple data.

Algorithm 2 Pseudocode for parallel calculation of electric fields on a vector processor.

```
1: for all particles do
2:     E(particle) ← 0
3: end for
4:
5: for all boundary elements do
6:     for all particles do
7:         E(particle) += field at particle due to boundary element
8:     end for
9: end for
```

Vector processors (such as graphics cards) are dedicated to such parallelism, see for example Garland et al. [19] and Luebke et al. [34]. Therefore, the problem of evaluating the electric field at N particles due to M boundary elements can be stated in the natural language of a vector processor.

4.1.2. Caching of Boundary Elements

While looking at algorithm 2, we can see that the inner loop involves one boundary element at a time and is reused by all particles. Boundary elements have a representation in memory (in terms of parameters) and the most trivial optimization is to avoid excessive memory requests as much as possible. This can be achieved by requesting the parameters of the boundary element only once and store the parameters temporarily inside a cache, where it can be reused at low cost.

In practice we divide the matrix representation (79) into sub-matrices defined on a grid where each sub-matrix has a seperate cache. This principle is illustrated in figure 10. The reason for this particular approach in terms of sub-matrices is that vector processors usually have limited amount of available cache. On top of that, the caching structure in figure 10 matches with the design of graphics cards, where sub-matrices are called 'blocks' on a 'grid' (see Garland et al. [19]). The number of blocks in figure 10 is determined as follows,

$$n_x = 1 + \lfloor \frac{\text{total number of boundary elements}}{\text{number of boundary elements per block}} \rfloor \tag{80}$$

[15] This is why we avoid adaptive integration of boundary integrals.

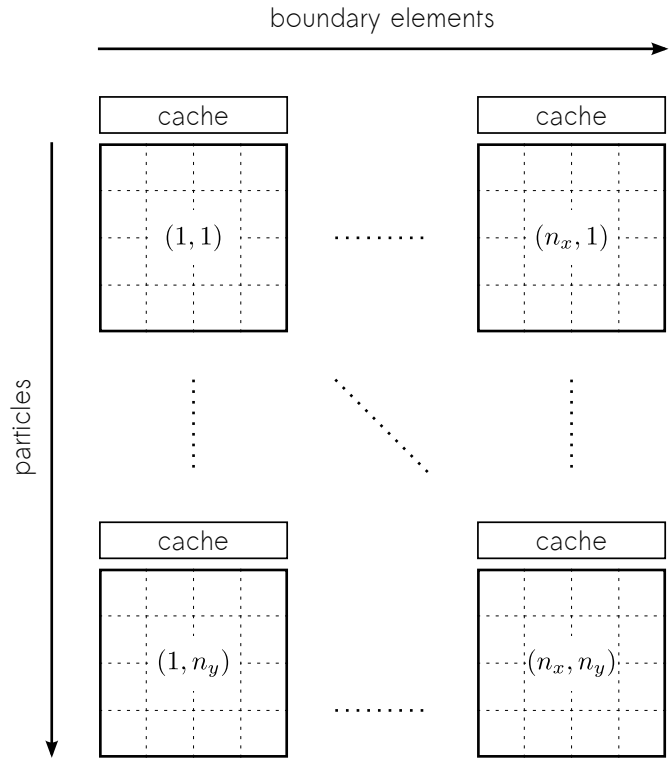

Figure 10 – Illustration of parallel design for vector processors with element caching. This figure divides the matrix of (79) into sub-matrices defined on a grid, where each sub-matrix has a seperate cache. The parallism is in the direction of the particles and the blocks are addressed from left to right. The calculation flow is explained in the text.

$$n_y = 1 + \lfloor \frac{\text{total number of particles}}{\text{number of particles per block}} \rfloor \tag{81}$$

Where the actual size of a block is limited by the hardware resources of the vector processor. The size of the local cache for each block is computed as follows,

$$\text{cache} = \text{bytes per boundary element} \times \text{number of boundary elements per block} \tag{82}$$

Let us explain the calculation flow of figure 10 in words. Consider the top left block with n threads. At first the cache is loaded with the first m elements, after which the the block calculates n particles concurrently. All threads in that block are effectively reusing the m elements in the cache. The first block completes after the contributions from all m elements of that block have been added to the n particles. Then, the program jumps to the next block on the right and repeats the procedure. This means that the cache of the new block is loaded with the next elements in line and the field contributions are calculated until all elements of that block are issued. This process is repeated until the final block of that row is reached, after which the program jumps to the first column

again of a new row. The vector processor initiates the calculation not with one block at a time, but with many blocks distributed over the first column concurrently. In the extreme case, the entire first column of figure 10 is launched for parallel execution.

The original algorithm 2 is modified to include caching of boundary elements and is given in algorithm 3.

Algorithm 3 Pseudocode for parallel calculation of electric fields using element caching on a vector processor.

```
1: for all particles do
2:     E(particle) ← 0
3: end for
4:
5: for all vertical blocks do
6:     for all horizontal blocks do
7:         for all boundary elements in block do
8:             copy boundary element to local cache
9:         end for
10:        for all boundary elements in local cache do
11:            for all particles in block do
12:                E(particle) += field at particle due to boundary element
13:            end for
14:        end for
15:    end for
16: end for
```

The estimated performance increase with respect to memory requests is determined by,

$$\frac{\text{number of particles per block} \times \text{number of elements per block}}{\text{number of elements per block}} \tag{83}$$

Note that this is only an estimation, because we have not taken into account (if available) the instrinsic memory caching structures (L1, L2, ...) of the hardware.

4.1.3. Coalesced Memory Access

We have minimized the amount of memory requests, but we have not yet looked into the actual memory calls themselves. How is memory transferred and can we optimize that further? The answer to the latter is yes, because the memory bus usually fetches chunks of data (multiple bits), which can be optimized and is called coalesced memory access (see Davidson et al. [35]). We explain this idea by the use of a simplified model of the memory bus, which is given in figure 11.

Suppose that we request four non-aligned elements (labelled as 1, 2, 3 and 4) from memory, as illustrated in figure 12. The memory bus needs to be addressed three times

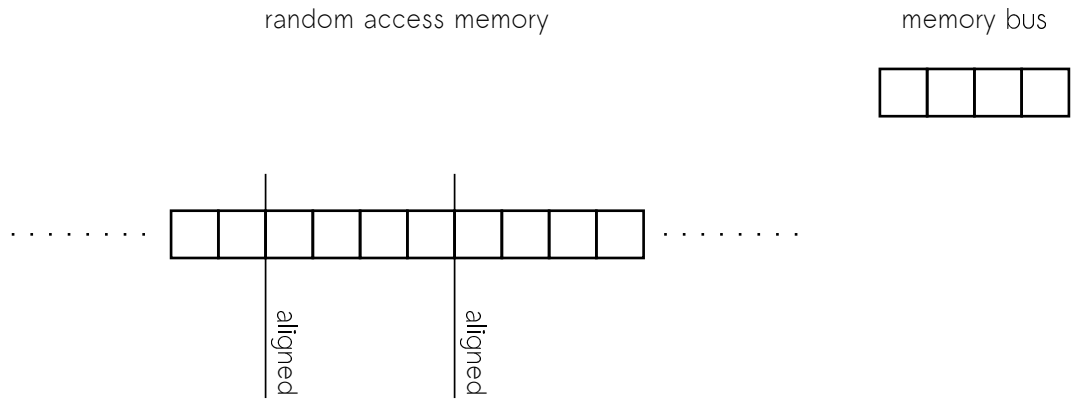

Figure 11 – Illustration of a simple memory model with random access memory and a memory bus. Random access memory is aligned and every request involves the transfer of elements in random access memory via the memory bus.

in order to fulfill that requrest, because the memory bus *only* fetches chunks of aligned data and the data for that particular request is not sequentially ordered and aligned.

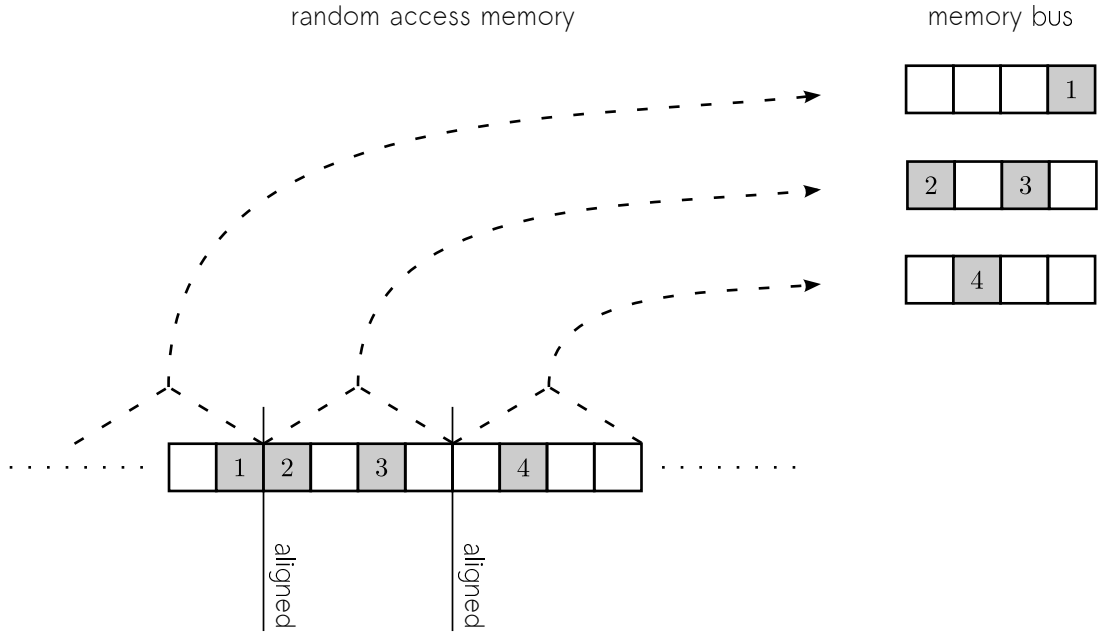

Figure 12 – Illustration of uncoalesced memory access. The elements 1, 2, 3 and 4 in random access memory are requested. In this example, the memory bus needs to be addressed three times, because the elements are not sequentially ordered and aligned.

The resolution to the uncoalesced access pattern of figure 12 is to properly align the data, such that the memory bus is only addressed once when the four elements are requested. This is called coalesced memory access and is illustrated in figure 13.

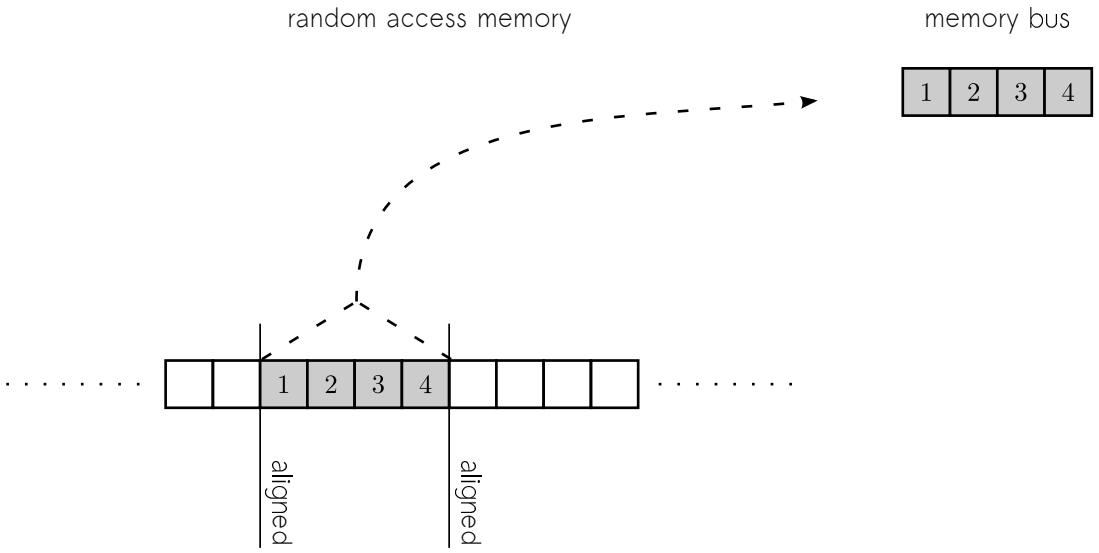

Figure 13 – Illustration of coalesced memory access. The elements 1, 2, 3 and 4 in random access memory are requested. The memory bus, in comparison to figure 12, needs to be addressed only once, because the elements are sequential and properly aligned.

Suppose that our boundary is defined as an effective point charge representation given by equation (77). This means that the boundary is represented in memory as point charges with parameters defining the location x_i, y_i, z_i and the effective charge λ_i per particle. The ordering of the parameters allows two options,

- **Array of structure**, where the parameters per boundary element are grouped together and pushed sequentially to a queue. This is illustrated in figure 14.
- **Structure of arrays**, where the parameters are distributed over four distinct arrays and each array relates to one parameter. This is illustrated in figure 15.

As a final remark on memory optimizations, we emphasize that the uncoalesced access of memory via the memory bus is not expected to be as extreme as illustrated in figures 12 and 14. The reason is that processors usually have multiple intrinsic caches denoted by L1, L2, and so on. Where L1 is the fastest and closest to the working threads, but usually the smallest of all. These caches, however, cannot be controlled by the user directly. This is viewed in contrast to the *programmable* caches of the vector processor illustrated in figure 10

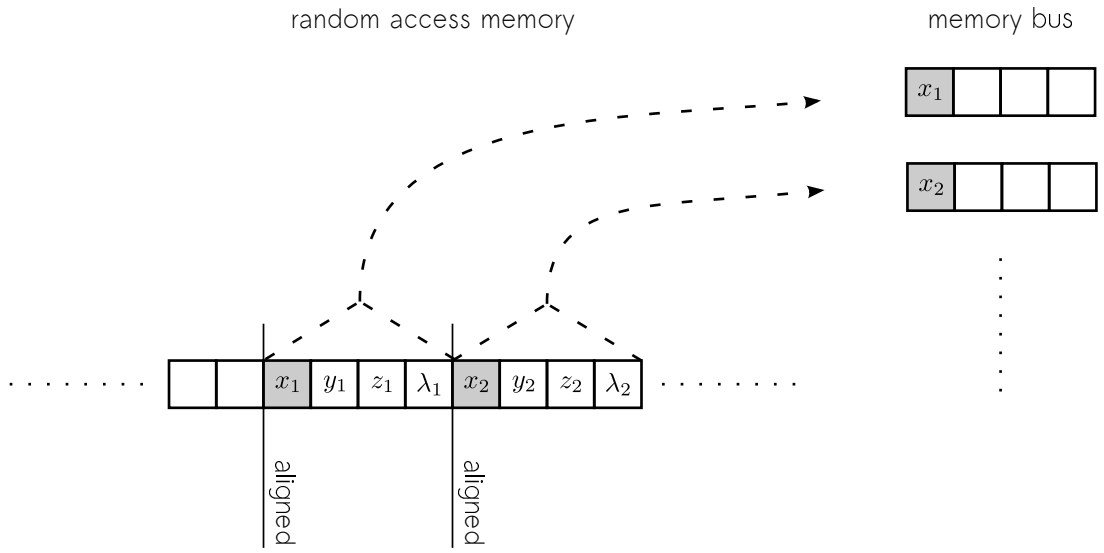

Figure 14 – Illustration of memory ordered as an array of structure. This figure domonstrates the request of the x positions of the boundary charges, which can be a typical operation in the arithmetic flow of the vector processor. Note that this type of memory ordering does not allow coalesced memory access.

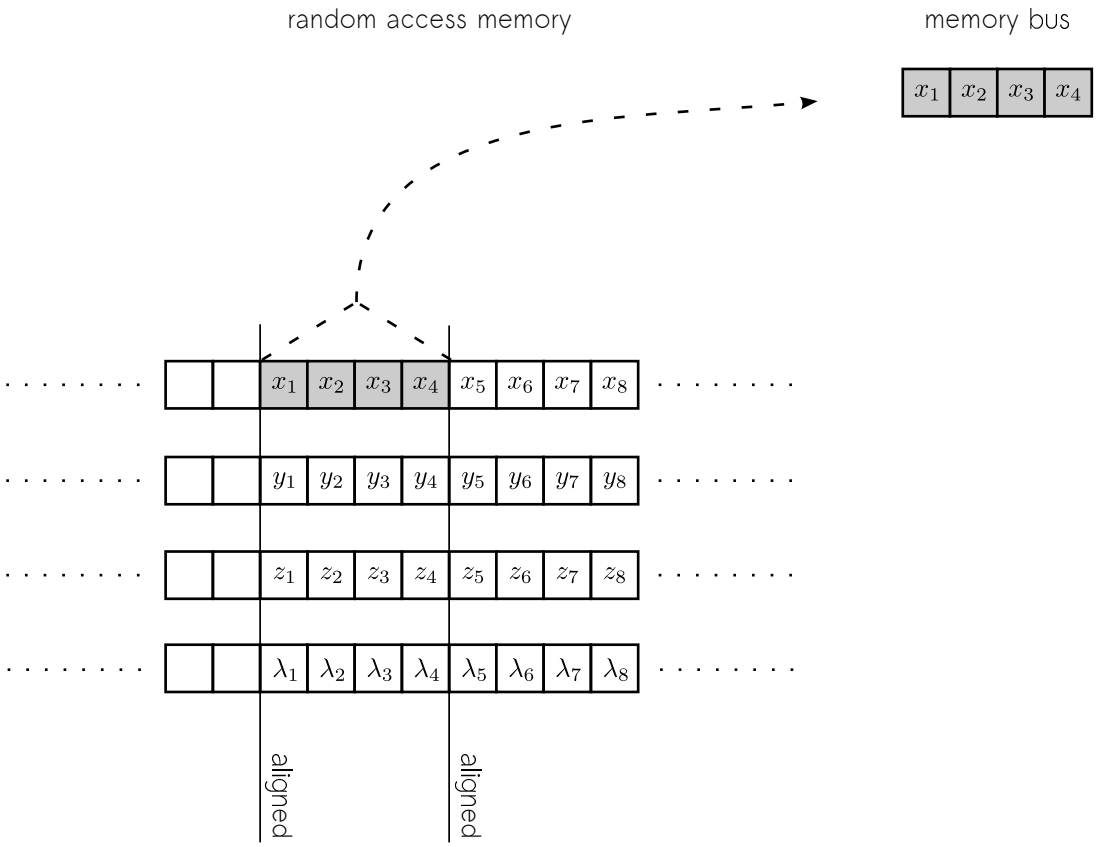

Figure 15 – Illustration of memory ordered as a structure of array. This figure domonstrates the request of the x positions of the boundary charges, which can be a typical operation in the arithmetic flow of the vector processor. Note that this type of memory ordering enables coalesced access of memory.

4.1.4. Kernel Performance

We demonstrate the performance of our boundary element kernel with respect to scalar processors, because that is what the conventional simulation tools in electron optics are designed for. We have at our disposal the following hardware,

- **GeForce GTX-480** by NVIDIA with 448 CUDA cores running at 1215 MHz (processor clock). This graphics card was introduced in the first quarter of 2010.
- **Quadro FX-4800** by NVIDIA with 192 CUDA cores running at 1204 MHz (processor clock). This high-end graphics card was introduced in the fourth quarter of 2008.
- **INTEL Xeon-5650** with 6 physical cores (12 threads using hyperthreading) running at 2.66 GHz with 12 MB of cache. This processor was introduced in the first quarter of 2010.
- **INTEL i7-860** with 4 physical cores (8 threads using hyperthreading) running at 2.8 GHz with 8 MB of cache. This processor was introduced in the third quarter of 2009.

The boundary element kernel evalutes the effective point charges given by equation (77). The actual distribution of point charges does not affect computation time, only the number of boundary elements and the choice of hardware are relevant. We are therefore free to use a randomized set of boundary point charges. In order to maximize parallelism, we choose the number of points in space (where the 'boundary' is evaluated) to be much larger than the number of boundary elements. This choice saturates the hardware with respect to parallelism and the performance is measured as the number of boundary point charges per second.

At first we run performance simulations on the scalar processors by INTEL[16]. The results are given in figure 16, where we have plotted the number of boundary charges per second versus the number of threads. The next run of performance simulations are targeting the graphics cards from NVIDIA. The results are given in figure 17, where we have plotted the number of boundary charges per second versus the block size. The peak performances from figures 16 and 17 are summarized per device in figure 18.

[16] We have used OpenMP (`http://openmp.org`) for implementing scalar parallelism.

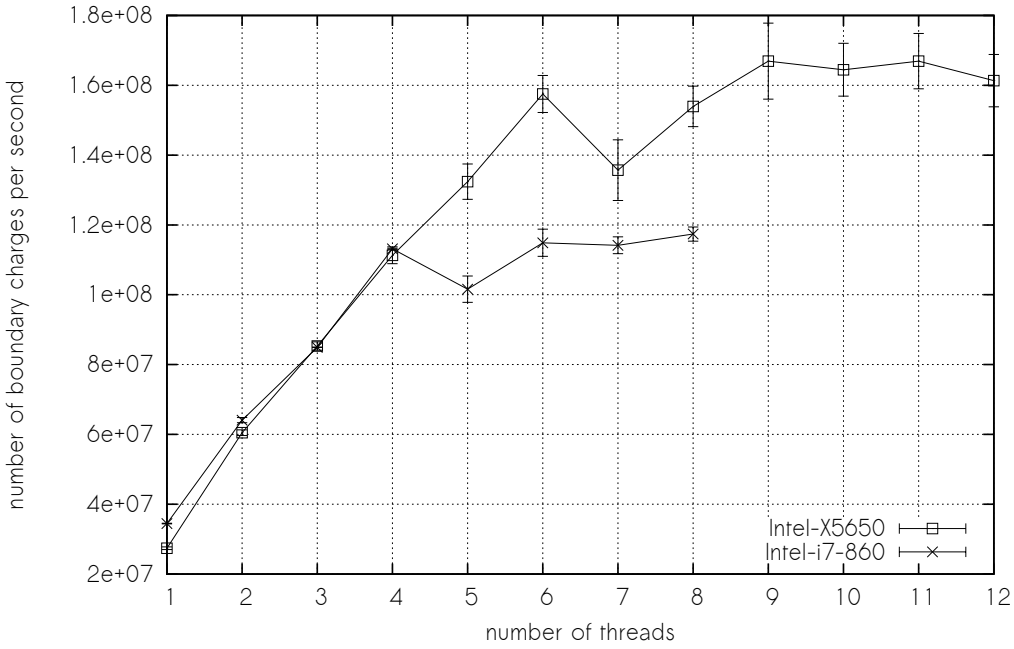

Figure 16 – Maximum performance of the boundary element kernel measured for scalar processors from INTEL. This figure gives the number of evaluated boundary charges per second as a function of the number of threads. The boundary charges are randomly distributed. This test maximizes parallism, because the number of particles is much larger than the number of boundary charges. The two curves correspond to the INTEL X5650 and INTEL i7-860 processor. Note that the curves initially grow linear and finally flatten out. This happens when the number of threads becomes larger than the physical core count. The errors bars are obtained from repeating this test many times.

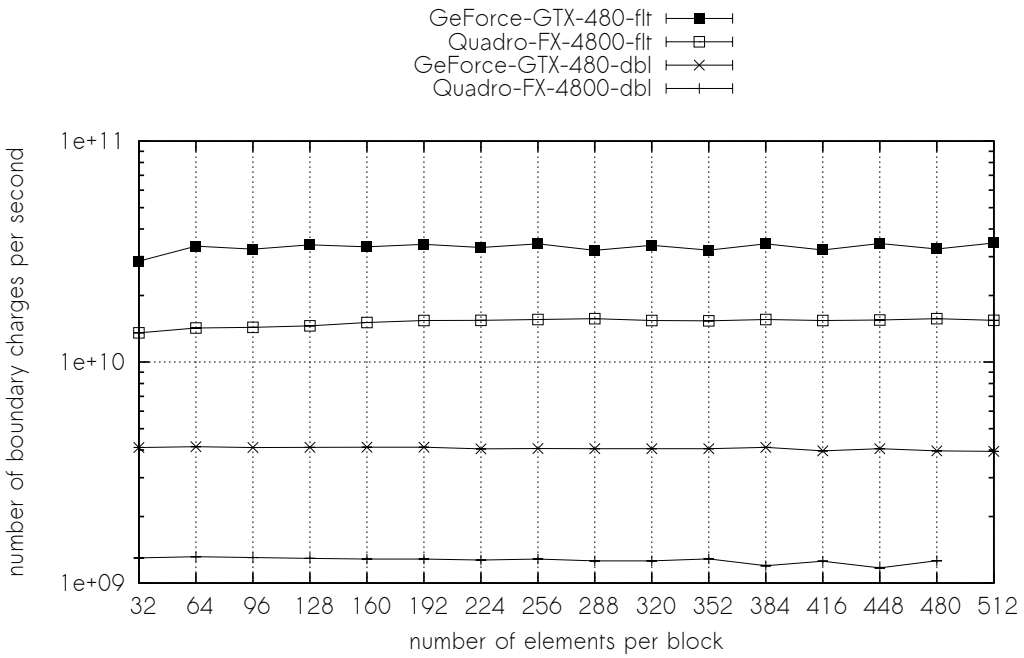

Figure 17 – Maximum performance of the boundary element kernel measured for graphics cards from NVIDIA. This figure gives the number of evaluated boundary charges per second as a function of the number of elements per block (see figure 10). The boundary charges are randomly distributed. This test maximizes parallism, because the number of particles is much larger than the number of boundary charges. The performance is measured for two graphics cards, GeForce GTX-480 and Quadro FX-4800. The four curves correspond to single floating point precision (GeForce-GTX-480-flt and Quadro-FX-4800-flt) and double floating point precision (GeForce-GTX-480-dbl and Quadro-FX-4800-dbl). The errors bars are obtained from repeating this many times.

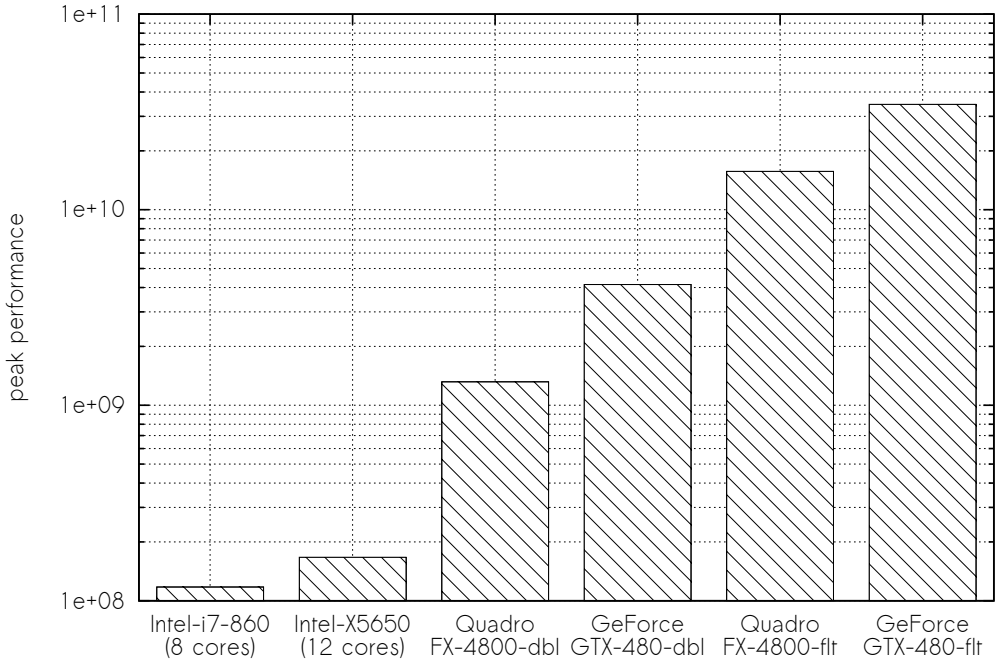

Figure 18 – Peak performance of the boundary element kernel measured for all test devices. This figure gives the peak performance as the number of evaluated boundary charges per second. The peak performance of the INTEL processor is the maximum found in figure 16. The peak performance of the graphics card from NVIDIA is the maximum found in figure 17.

4.2. Processing Stages

Suppose that we have an electron optical problem and that we want to simulate this problem using the program of this thesis. What precisely needs to be done? And what does the typical workflow look like? We now focus on answering those questions. The program of this thesis divides the simulation into three sequentially connected stages, where the output of one stage is (part of) the input of the next stage,

1. **Triangulation of the boundary.** This is a prelimary stage and can be done either by hand or an external program capable of discretizing a boundary into triangular elements. The only requirement is that the output of this stage produces two files, (1) a vertex file and (2) a corresponding faces file. The structure of those files is discussed in detail later on.

2. **Solving for boundary values.** This stage constructs the Green's matrix (see section 3.3) for the boundary obtained from the first stage. The unknown boundary values are solved for and the complete boundary, which now includes the boundary values, is exported to the next stage.

3. **Ray-tracing of particles.** The boundary is transformed into an effective point charge representation (see section 3.4 and in particular equation (77)). Particles with a predefined initial state are traced through the electric field using the geometrical integrator described in section 2.5. The states (canonical position and canonical momentum) are exported per iteration and for every particle to a file.

There is, in reality, a fourth stage which is not mentioned here. In the fourth stage, the trajectories are interpreted and is usually combined with post-analysis in order to extract the results of interest. We now discuss for the remainder of this chapter the processing stages in more detail.

4.2.1. Stage 1: Triangulation of the Boundary

This is the preliminary stage and involves the design of the electrostatic boundary. The actual steps of the first stage are schematically given in figure 19.

- **Design of electrostatic boundary.** The electron optical problem of interest involves an electrostatic boundary, which must be designed at first. The design can be made by hand or by using a program as long as the boundary is given in a format which can be discretized into triangles. An example of electrostatic boundary design is given in the chapter of the shift lens.

- **Triangulation.** There are two ways to do this. The boundary can be triangulated by hand using a home-built program or script. The other option is to use an external programs capable of discretizing the boundary into triangles[17]. An exam-

[17] For example, MATLAB (which has built-in Delaunay functions) or the program "Triangle" from J.R. Shewchuk (`http://www.cs.cmu.edu/~quake/triangle.html`).

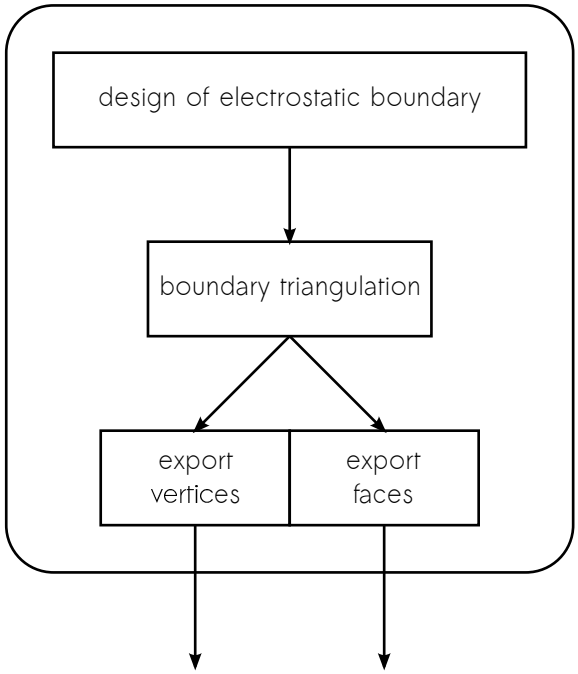

Figure 19 – Schematic diagram illustrating the steps of the first stage. The first step is to design the electrostatic boundary for the problem at hand. The second step discretizes the boundary into triangles, which can be done by hand or by an external program. The triangles of the boundary are exported to two seperate files, of which the "vertices file" describes the vertices of all triangles and the "faces file" describes the faces by connecting the vertices.

ple of triangulation is given in the chapter of the shift lens, where we completely discretize the system by hand.

❑ **Exporting the boundary.** We explain the export format of this stage by using the elementary example given in figure 7. The vertices of that example are exported in the following comma seperated format,

$$\begin{array}{lcllll} \text{VERTEX } 1 & \rightarrow & (\mathbf{R}_1)_x, & (\mathbf{R}_1)_y, & (\mathbf{R}_1)_z, & \Phi_1 \\ \text{VERTEX } 2 & \rightarrow & (\mathbf{R}_2)_x, & (\mathbf{R}_2)_y, & (\mathbf{R}_2)_z, & \Phi_2 \\ \text{VERTEX } 3 & \rightarrow & (\mathbf{R}_3)_x, & (\mathbf{R}_3)_y, & (\mathbf{R}_3)_z, & \Phi_3 \\ \text{VERTEX } 4 & \rightarrow & (\mathbf{R}_4)_x, & (\mathbf{R}_4)_y, & (\mathbf{R}_4)_z, & \Phi_4 \end{array}$$

The corresponding faces of that boundary are exported as follows,

$$\begin{array}{lcccc} \text{FACE } 1 & \rightarrow & \text{VERTEX } 1, & \text{VERTEX } 2, & \text{VERTEX } 3 \\ \text{FACE } 2 & \rightarrow & \text{VERTEX } 2, & \text{VERTEX } 3, & \text{VERTEX } 4 \end{array}$$

The primary reason for this particular format is that the *shared* vertices, such as the ones in the elementary example, can easily be encoded.

Note that it does not matter *how* the triangulated boundary is obtained. The only requirement is that the output of this stage gives the triangulated boundary with the vertices and faces exported to seperate files, because that is the input for the next stage.

4.2.2. Stage 2: Solving for Boundary Values

The purpose of the second stage is to solve for the boundary values, such that the boundary has the desired potential upon evaluation. The steps of this stage are given in figure 20. The input is directly connected to the output of the previous stage given in figure 19.

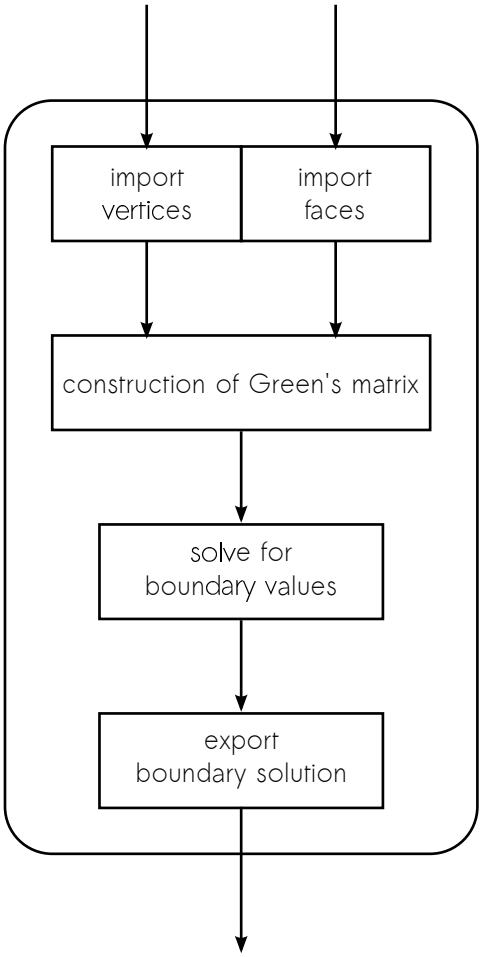

Figure 20 – Schematic diagram illustrating the steps of the second stage. The input of this stage is the boundary given in the format of vertices and faces. The second step is to construct Green's matrix for that particular boundary. The matrix equation is solved for the boundary values and the complete boundary, which now includes the boundary values, is exported in the final step as a single file.

- **Importing the boundary.** The triangulated boundary of the first stage is imported, which at the moment does not include boundary values. We verify that no duplicate vertices exist, because otherwise the Green's matrix becomes singular. We also verify that all referenced vertices in the faces file indeed exist in the vertices file.
- **Construction of Green's matrix.** The matrix equation is constructed by following the details of section 3.3, where the vertex response functions are calculated using adapative Gaussian quadrature. We have implemented the nested Gauss-Kronrod quadrature rules from which also an error estimate can be derived [33].
- **Solving for boundary values.** Now that Green's matrix is constructed, we can start solving the matrix equation (60) for the unknown boundary values. We use the parallel DGESV (Double GEneral SolVer) function from the INTEL Math Kernel Library[18] to do so.
- **Exporting the boundary solution.** The boundary of the elementary example is now exported in the following format,

TRIANGLE 1	$\rightarrow$	$(\mathbf{R}_1)_x$,	$(\mathbf{R}_1)_y$,	$(\mathbf{R}_1)_z$,	Φ_1,	σ_1
		$(\mathbf{R}_2)_x$,	$(\mathbf{R}_2)_y$,	$(\mathbf{R}_2)_z$,	Φ_2,	σ_2
		$(\mathbf{R}_3)_x$,	$(\mathbf{R}_3)_y$,	$(\mathbf{R}_3)_z$,	Φ_3,	σ_3
TRIANGLE 2	$\rightarrow$	$(\mathbf{R}_2)_x$,	$(\mathbf{R}_2)_y$,	$(\mathbf{R}_2)_z$,	Φ_2,	σ_2
		$(\mathbf{R}_3)_x$,	$(\mathbf{R}_3)_y$,	$(\mathbf{R}_3)_z$,	Φ_3,	σ_3
		$(\mathbf{R}_4)_x$,	$(\mathbf{R}_4)_y$,	$(\mathbf{R}_4)_z$,	Φ_4,	σ_4

  Where the components of the triangle are explicitly encoded together with the potentials and boundary values per vertex. This single file completely describes the boundary and is called the "boundary solution file". Note that in this format the shared vertices are duplicated, because there is no need anymore for keeping track of shared vertices.

The boundary solution obtained by this stage is the input of the next stage, where the particles with a predefined state are traced through the electric field of the boundary.

4.2.3. Stage 3: Ray-tracing of Particles

The purpose of this stage is to solve the equations of motion for N particles, where the particles are traversing through the electric field of the boundary from the previous stage. This stage is divided into two parts (see figure 21 and 22), where the first part is devoted to initialization. The boundary is transformed into an effective point charge representation and copied to the vector processor. In the second part (trace-loop), the trajectories are determined by using the vector processor for calculating the electric field

[18] `http://software.intel.com/en-us/intel-mkl`

(detail view given in figure 22). The states of the particles are exported to a file, which are then used for post-analysis specifically for the problem at hand.

- **Import of boundary solution.** The boundary solution of the previous stage is imported without modifications.
- **Import of particle definitions.** The initial state of all charged particles involved in the simulation are described in the particle definition file. The format used for describing the states is as follows,

PARTICLE 1	$\rightarrow$	$(\mathbf{q}_1)_x,$	$(\mathbf{q}_1)_y,$	$(\mathbf{q}_1)_z$
		$(\mathbf{p}_1)_x,$	$(\mathbf{p}_1)_y,$	$(\mathbf{p}_1)_z$
		$m_1,$	λ_1	
PARTICLE 2	$\rightarrow$	$(\mathbf{q}_2)_x,$	$(\mathbf{q}_2)_y,$	$(\mathbf{q}_2)_z$
		$(\mathbf{p}_2)_x,$	$(\mathbf{p}_2)_y,$	$(\mathbf{p}_2)_z$
		$m_2,$	λ_2	
$\vdots$			$\vdots$	

- **Transfrom boundary into point charge representation.** The boundary solution is transformed to an effective point charge representation as described in section 3.4. We have implemented the fourth order Gauss-Legendre quadrature rule [32] for evaluating the electric field. This means that every triangle is replaced by $4 \times 4 = 16$ effective point charges. The parameters of the boundary charges are ordered in accordance with figure 15 such that memory access is coalesced.
- **Trace-loop.** This is the core business of the tracer, which we discuss in detail seperately from the rest of the steps. The detailed view is given in figure 22.

We now discuss the trace-loop give by figure 22. The actions in the trace-loop are targeting the host (the computer that controls the vector processor) and the vector processor. The positions of the particles are copied to the vector processor at the beginning of the iteration. The vector processor then calculates the electric field for all the particles in accordance with the implementation given in section 4.1. When the vector processor has completed calculating the fields, the results are copied to the host such that the electric field at the particles is available to program running on the host. This concludes the actions of the vector processor.

In the mean time (parallel to the field calculations of the vector processor), the host can do several tasks. The first task is to export the current state of the particles to a file. Althoug the Coulomb interactions are not used at the moment, we mention that they can be calculated by the host, for example by using Barnes-Hut. The host and vector processor are synchronized after all tasks have been completed. What remains to be done is to update the states of the particles (see section 2.5), After this update, the next iteration starts and the whole procedure is repeated until a terminating condition is satisfied.

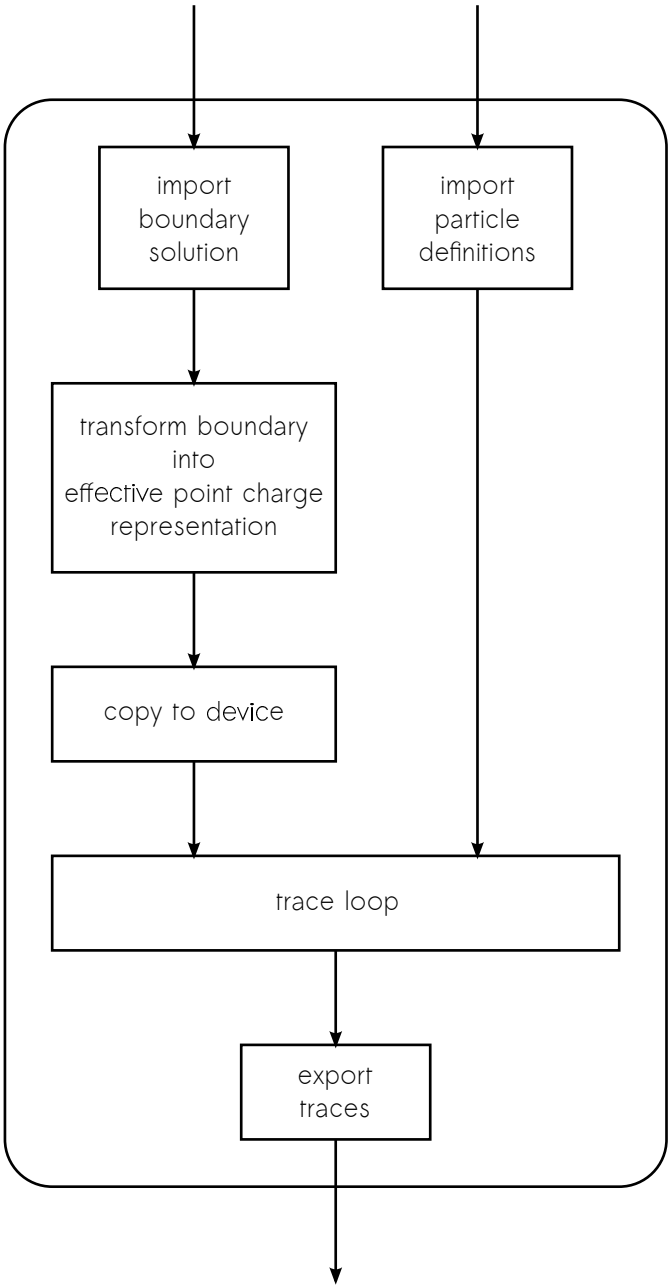

Figure 21 – Schematic diagram illustrating the steps of the third stage. The input of this stage involves two definitions, (1) the boundary solution of the previous stage and (2) the initial particle definitions. The boundary solution is transformed into an effective point charge representation, which is then copied to the vector processor. This concludes the initialization. The actual tracing takes place in the trace-loop of which a detailed view is given in figure 22.

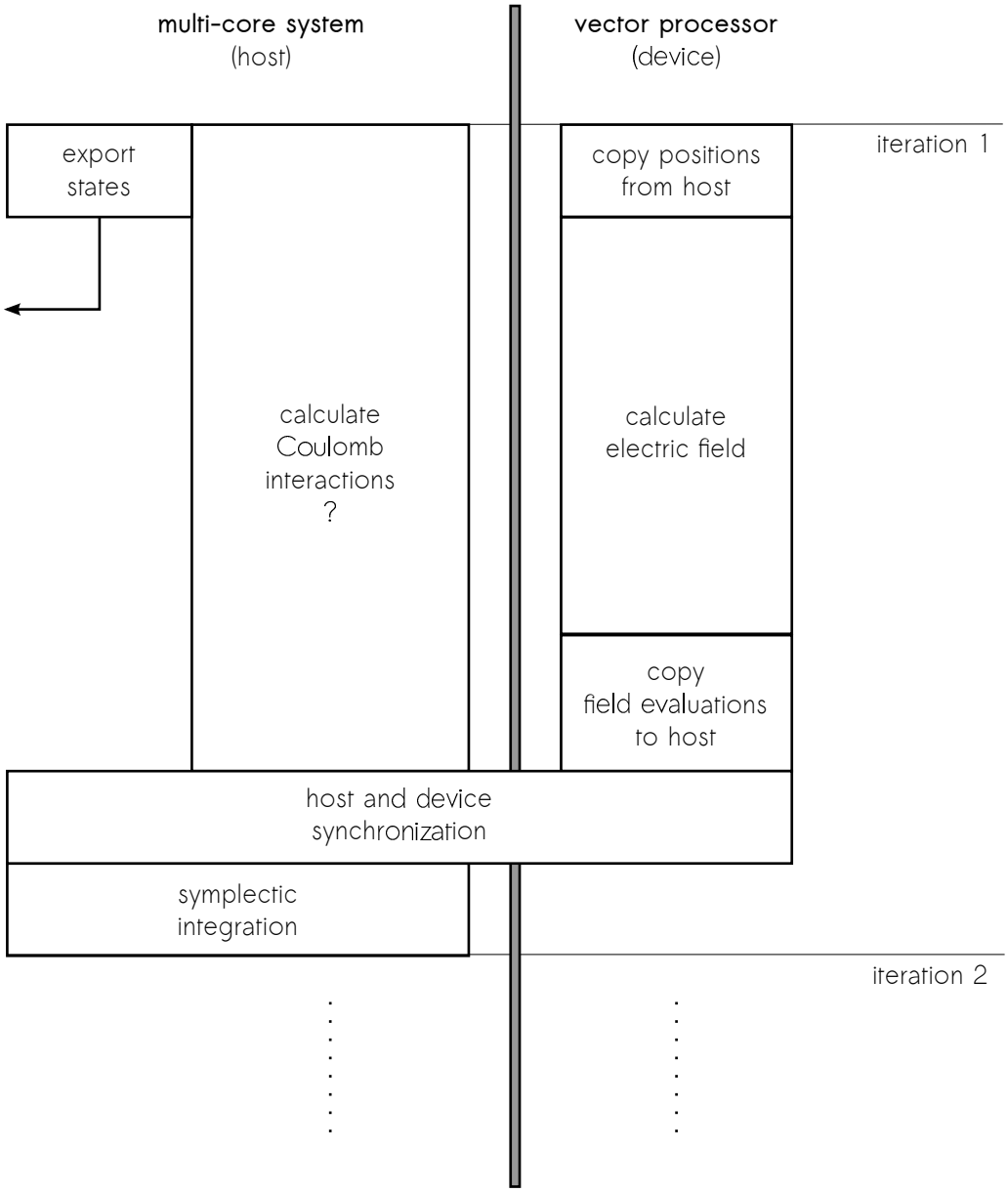

Figure 22 – Schematic diagram illustrating the steps of the trace-loop. This diagram is nested into the diagram of figure 21. The left side of the diagram represents the actions for the host (the computer that is controlling the vector processor) and the right side represents the actions targeting the vector processor. The operations of the host and vector processor are running in parallel and are synchronized before the next iteration starts. The Coulomb interactions are not used at the moment, but are mentioned for future applications (this is discussed in the outlook). Note that the diagram is repeated for successive iterations of which only two are illustrated. The details of the iterations are given in the text.

5. Application: Shift Lens

In this chapter we apply the simulator of this thesis to a specific application: The shift lens. The shift lens constists of a system of five electrodes of which the center electrode is displaced in the lateral direction [3]. The motivation for this lens is mentioned in the introduction. Our goal is to calculate the higher order aberrations of this lens. This is a difficult task, because most of the convential programs mentioned in table 1 barely produce the second order aberrations, and the programs that do, take an unreasonable amount of time. This chapter also serves as a template for simulating an electron optical problem using the simulator this thesis.

The ultimate goal is to calculate the higher order aberrations of the shift lens and our plan of calculation for doing so is given in the first section. We propose to calculate the aberrations using the thin lens model and fit the final deflection angles in the plane of the thin lens to a Taylor expansion involving several orders. The aberrations are related to the orders in the Taylor expansion and this approach is discussed in detail.

In the next section we focus on the electrostatic design of the boundary of the shift lens. The boundary is discretized by hand using two dimensional triangulated primitives and projections. This triangulation is not uniform, but has more triangles distributed close to the sharp edges in order to accomodate the strong electric field.

The chapter concludes with the results of the simulation including an error analysis.

5.1. Plan of Calculation

At first we examine the design and function of the shift lens. The specifications of the shift lens are given in detail in figure 23. The function of this lens is to deflect a parallel incoming electron in a particular direction. This deflection is in the direction of the displacement of the center electrode (see figure 24).

The idea is not only to calculate this deflection, but also the *aberrations* of this shift lens. Because in the end, we want to have the smallest focused spot with the largest possible current. The problem is that the aberrations cause the size of focused spot to increase. So the theoretical study of this lens is actually the study of the aberrations caused by this lens. The interested reader is referred to Hecht [36] for details on different types of aberrations.

The way we are going to study this lens is by using the model of a thin lens (see figure 25). This thin lens captures the deflection properties of the shift lens and therefore also the aberrations. All that we need to do is to trace electrons through the shift lens and determine the final deflection angle after the lens[19]. This final deflection angle is then related to the initial position, such that the deflection of an incoming electron is

[19] This must be far enough such that the lens no longer has any significant influence on the trajectory.

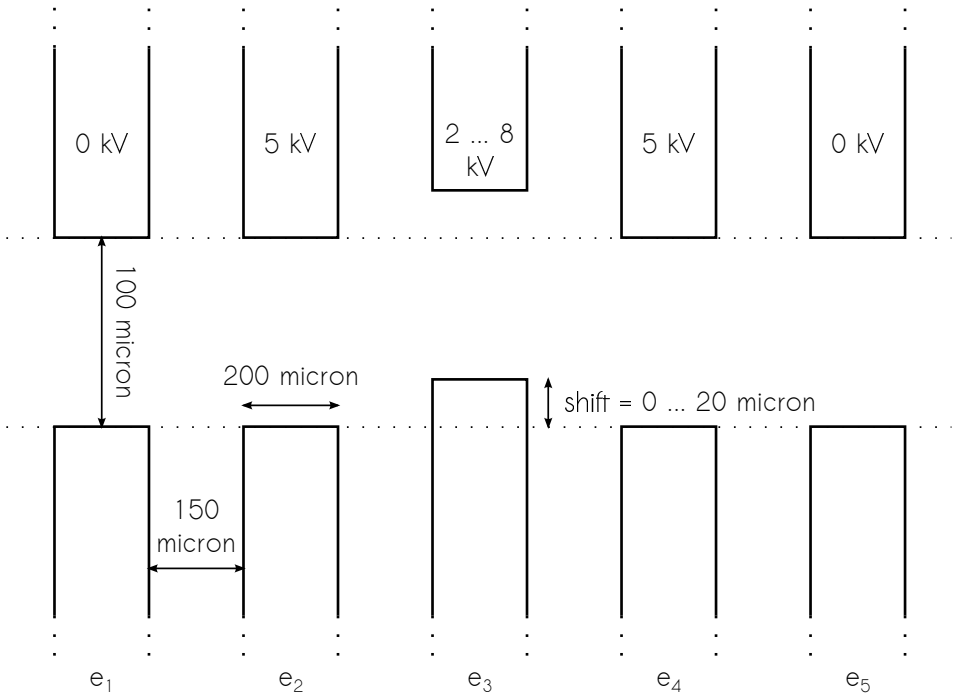

Figure 23 – Schematic two dimensional view of the shift lens [3]. This system consists of five electrodes of which the center electrode (e_3) is displaced laterally in the range of 0 to 20 micron. All five electrodes must be seen as rotationally symmetric. This means that each electrode in the diagram is in reality a solid disc with a hole in the center. The potential of the center electrode is varied from 2 kV to 8 kV. The neightbor electrodes (e_2 and e_4) are at the fixed potential of 5 kV. The outermost electrode (e_1 and e_5) are at the fixed potential of 0 V.

related to that particular position in the plane. For each particle we determine the final deflection angle in the plane as follows,

$$\alpha(x_0, y_0) = \frac{p_x}{p_z} \tag{84}$$

$$\beta(x_0, y_0) = \frac{p_y}{p_z} \tag{85}$$

Where α is the deflection in the x-direction and β the deflection in the y-direction. The coordinates x_0 and y_0 are the initial coordinates of the electron before entering the lens[20]. The final momenta of the particles are taken with respect to a measurement plane and thus need to be interpolated. The deflection functions α and β can be measured by tracing many electrons with different initial positions. We express these functions in

[20] This is the position where the parallel incoming ray hits the plane of the thin lens approximation.

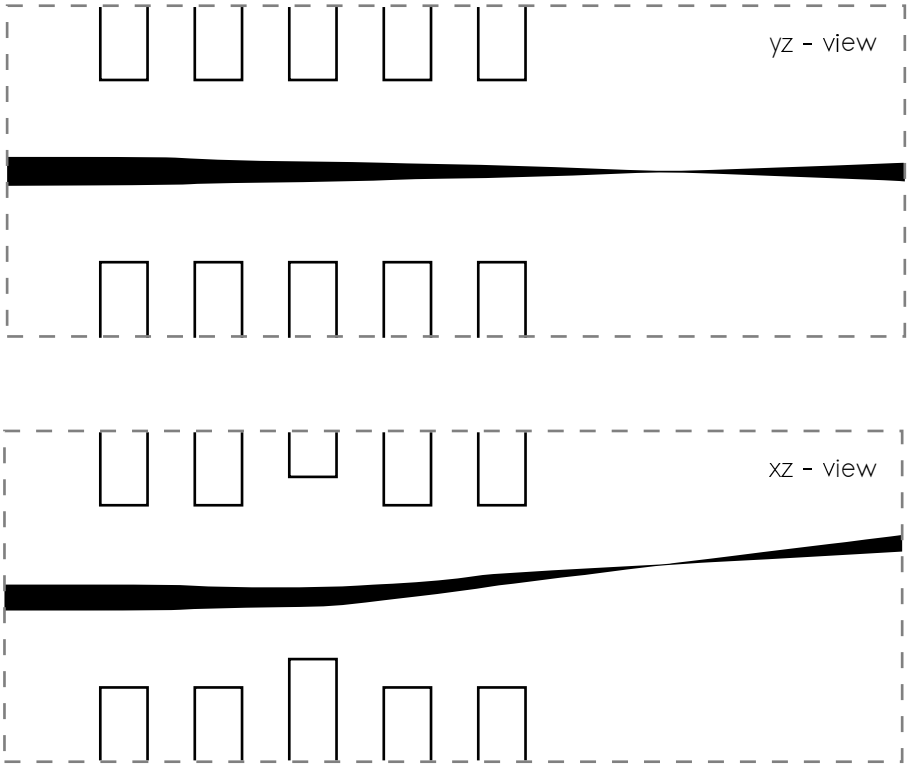

Figure 24 – Illustration of the beam deflection due to the displacement of the center electrode. In the top image we see that the beam shows no deflection in the yz-plane, because the electrode is not shifted in that direction. The bottom image shows the xz-plane and that is the plane in which the center electrode is displaced. The beam is deflected in the xz-plane, because the deflection is in the same direction as the displacement of the electrode.

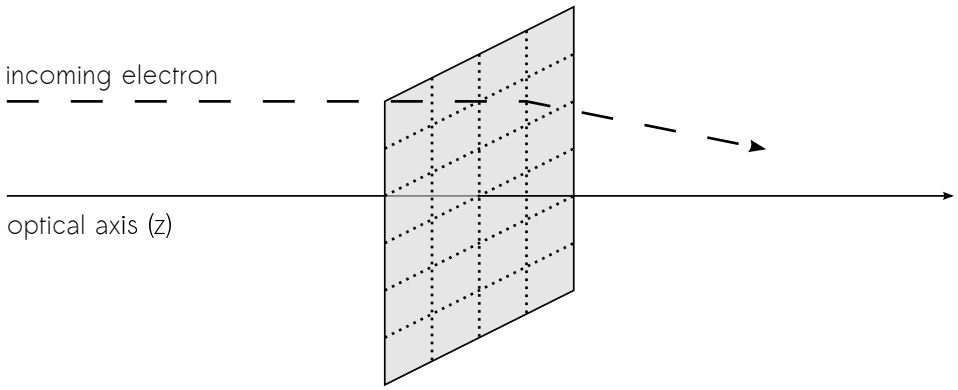

Figure 25 – Illustration of the thin lens approximation. The shift lens can be studied as a thin lens approximation, where the plane of the thin lens deflects parallel incoming electrons according to the actual deflection function of the shift lens.

general as a Taylor expansion up to and including order n,

$$\alpha(x,y) = \alpha_{0,0} + \sum_{i=1}^{n}\sum_{j=0}^{i} \alpha_{i-j,j} x^{i-j} y^{j} \tag{86}$$

$$\beta(x,y) = \beta_{0,0} + \sum_{i=1}^{n}\sum_{j=0}^{i} \beta_{i-j,j} x^{i-j} y^{j} \tag{87}$$

This expansion is too general because the shift lens has mirror symmetry, which is best seen in figure 24. The consequence of the mirror symmetry is that odd powers of y drop out of the $\alpha(x,y)$ function and even powers of y drop out of the $\beta(x,y)$ function. The resulting Taylor expansion for the shift lens evaluates to,

$$\alpha(x,y) = \alpha_{0,0} + \sum_{i=1}^{n}\sum_{j=0,2,4,\ldots}^{i} \alpha_{i-j,j} x^{i-j} y^{j} \tag{88}$$

$$\beta(x,y) = \sum_{i=1}^{n}\sum_{j=1,3,5,\ldots}^{i} \beta_{i-j,j} x^{i-j} y^{j} \tag{89}$$

The coefficients in this Taylor expansion are related to the different types of aberrations [37, 36, 24, 13]. The plan of calculation is to trace many electrons through the shift lens at random and fit the Taylor expansions (88) and (89) against a collection of deflection angles in the measurement plane. We repeat the calculation of the coefficients by varying the displacement and voltage of the center electrode. The aberrations of the shift lens can then be related to the amount of displacement and voltage of the center electrode.

5.2. Definition of Electrodes

In this section we illustrate how to triangulate the design of figure 23 by hand. From figure 23 we see that the shift lens is made out of five similar designed electrodes. The idea is to triangulate only one electrode and use that triangulation as a template for the others. We split the electrode into two seperate parts, (1) the inner cylinder and (2) the enclosing discs at the left and right of this cylinder. The inner cylinder is constructed by triangulating a two dimensional plane (see figure 26).

The triangulated plane of figure 26 is projected onto the surface of a cylinder by using the following transformation,

$$\begin{aligned} x &\rightarrow x_0 + R\cos\theta \\ y &\rightarrow y_0 + R\sin\theta \\ z &\rightarrow z_0 + z \end{aligned}$$

Where R is the radius of the cylinder, x_0, y_0 and z_0 determines the position of the cylinder in three dimensional space. The resulting projection onto the cylinder is illustrated in

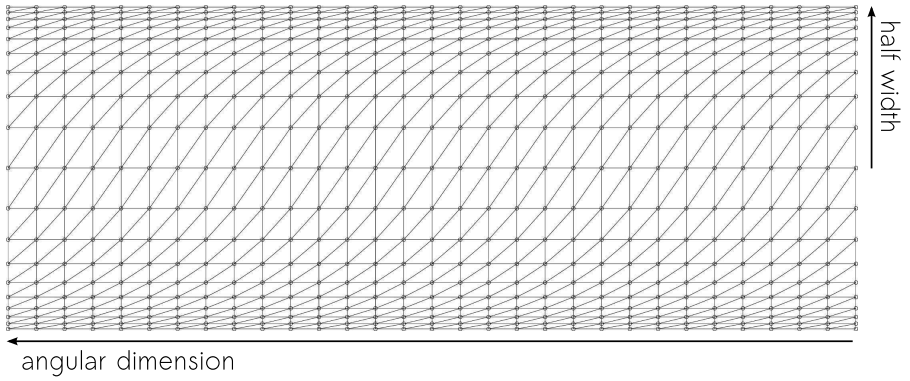

Figure 26 – Illustration of a triangulated plane where the size of the triangles vary in the vertical direction. This plane is used for the construction of an electrode of the shift lens. The triangulated cylinder is obtained by projecting this plane onto the surface of a cylinder. In the horizontal direction we have the angular dimension (ranging from 0 to 2π) and the vertical axis represents the axial direction of the cylinder. Note that the vertices in the vertical direction are not equally distributed. The distance between the vertices decreases as we get closer to the ends. The vertices in the vertical direction are distributed with a log scale.

figure 27. Note that the triangles in figure 27 decrease in size as we approach the ends of the cylinder. This is a consequence of the vertical vertex distribution in the plane of figure 26, where the vertices are distributed with a log scale. The ends of the cylinder corresponds to the sharp edges of the electrode and the potential field lines close to that surface must follow this sharp edge. This is where the electric field is the strongest and we therefore decrease the size of the triangles approaching this edge in order to accurately model this effect.

The second component of the electrode is constructed by triangulating the surface of a disc with a hole in the center, which is shown in figure 28. The number of angular nodes at the inner radius of this disc is equal to the number of angular nodes in the cylinder map of figure 26. This ensures that the two objects can be merged into one single object without artefacts. The electrode is constructed by merging two of the discs to the ends of the cylinder. The net result of that is given in figure 29.

The shift lens of figure 23 is constructed by placing five copies of the electrode in figure (29) together in accordance with the schematic design of figure 23. The net result is the triangulated boundary of the shift lens, which is given as a wireframe in figure 30

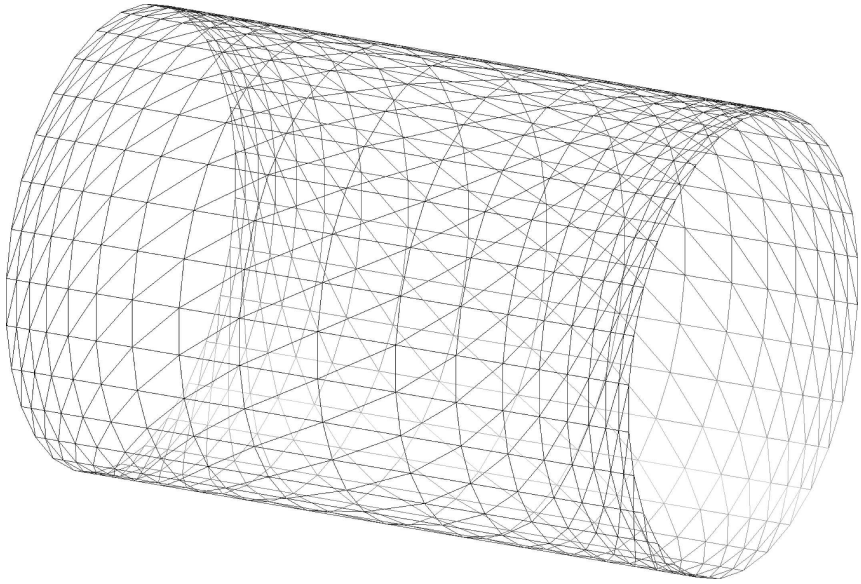

Figure 27 – Illustration of a triangulated cylinder. This object is obtained by triangulating a two dimensional plane and projecting that plane onto the surface of a cylinder. Note that there are more triangles distributed at the ends of the cylinder to accomodate the strong field.

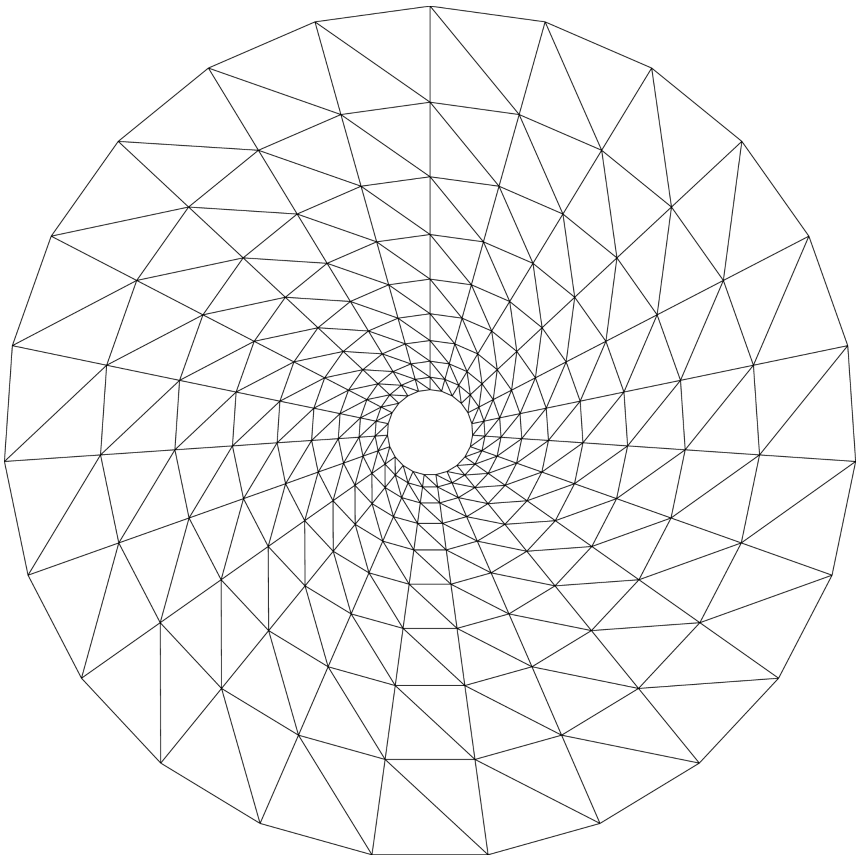

Figure 28 – Illustration of a triangulated disc with a hole in the center. This is a component of the electrode of the shift lens. An electrode of the shift lens is constructed by merging two of these discs to the ends of the cylinder in figure 27. Note that most of the triangles are distributed close to the inner radius.

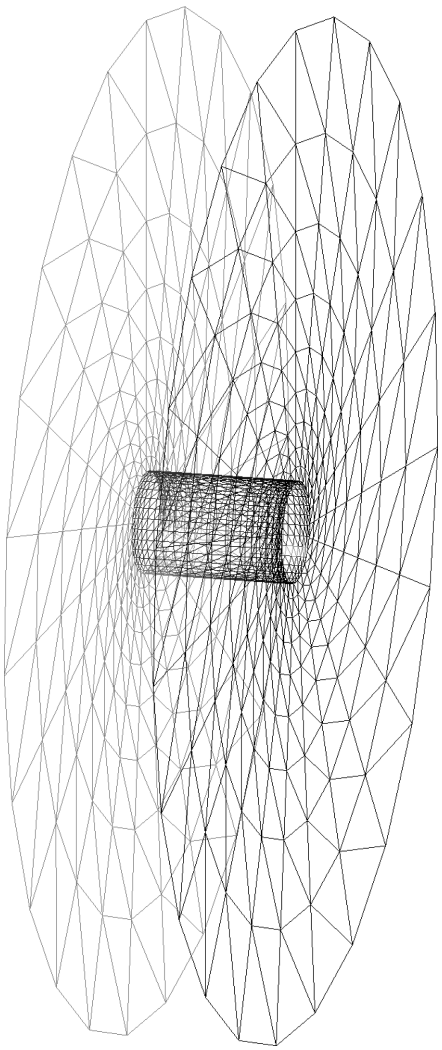

Figure 29 – Illustration of a triangulated electrode of the shift lens. This electrode is obtained by merging two discs (figure 28) to the ends of the cylinder (figure 27). The triangles at the ends of the cylinder are exactly matching with the triangles at the inner radius of the disc, such that the vertices at the intersection are shared. The number of triangles close to the ends of the cylinder is larger because that is where the sharp edges of the electrode are located.

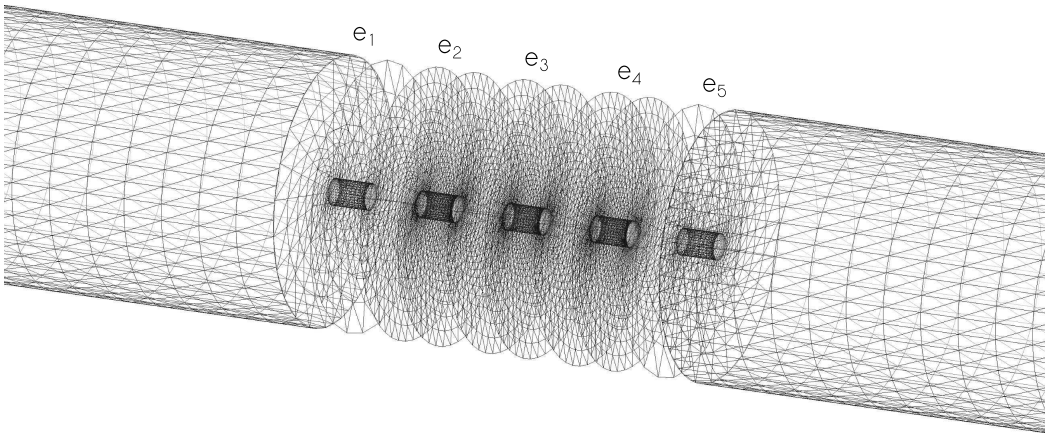

Figure 30 – Wireframe model of the triangulated boundary of the shift lens. This is the triangulated boundary representing the schematic design of figure 23. This boundary is obtained by triangulating one electrode as a template and repeat that electrode five times.

5.3. Simulation Results

The coefficients of the Taylor expansion of the deflection angle in the x-direction are described by equation (88) and are called alpha-coefficients. Similarly, the coefficients for the Taylor expansion in the y-direction are described by equation (89) and are called beta-coefficients. In general, these coefficients are *functions* of the displacement (D) and voltage (V) of the center electrode, i.e. $\alpha(D, V)$ and $\beta(D, V)$. We evaluate these coefficients by simulating 5 predefined displacements (0, 5, 10, 15 and 20 micron) and 50 predefined voltages ranging from 2 kV to 8 kV. This means that every coefficient in the Taylor expansion is determined for $5 \times 50 = 250$ different settings. The electrons are launched parallel to the z-axis with an initial energy of 5 kV. The details of the simulation are summarized in table 3 and table 4.

Number of ...	**Quantity**
Triangles per simulation	25.700
Effective point charges per triangle	16
Effective point charges per simulation	411.200
Trajectories per simulation	4096
Iterations per trajectory per simulation	2.000
Simulations	250
Trajectories	1×10^6
Force evaluations	2×10^9
Boundary point charge evaluations	8×10^{14}

Table 3 – Table showing the quantities involved in the simulation of the shift lens.

Time of ...	**Quantity**
Solving boundary per simulation	15 minutes
Tracing per simulation	14 minutes
Tracing per particle (derived)	200 milliseconds
Total simulation time	61 hours

Table 4 – Table showing the computation times for simulating the shift lens.

The results for the zeroth order coefficient are given in figure 32. The first order results are given in figures 33 and 34. The second order results are given in figures 35, 36 and 37. The results up to fifth order are given in appendix B.

The first order coefficients in figures 33 and 34 relate to the first order deflection in the x and y direction respectively. These coefficients relate to the focal distance as, $f_x = -\frac{1}{\alpha_{1,0}}$ and $f_y = -\frac{1}{\beta_{0,1}}$. The effect of astigmatism is obtained by subtracting the focal distance in the x direction from the focal distance in the y direction. The net result is given in figure 31.

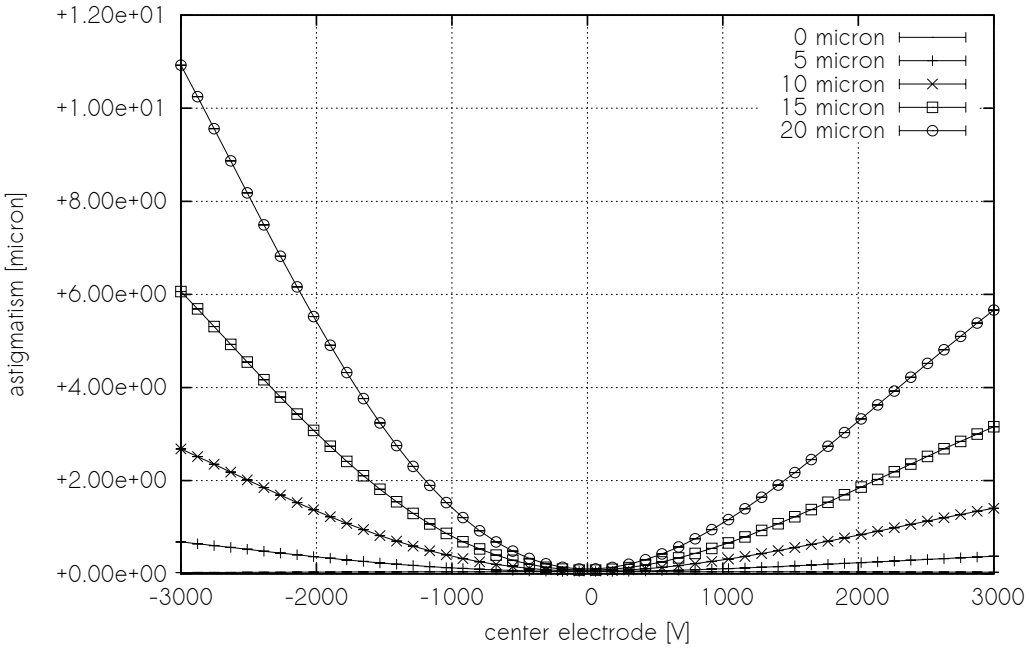

Figure 31 – Illustration of the effect of astigmatism in the shift lens. This figure is obtained by subtracting the focal distance in the x direction from the focal distance in the y direction. Note that the error bars in this figure are too small to see.

5.3.1. Zeroth-order Coefficient

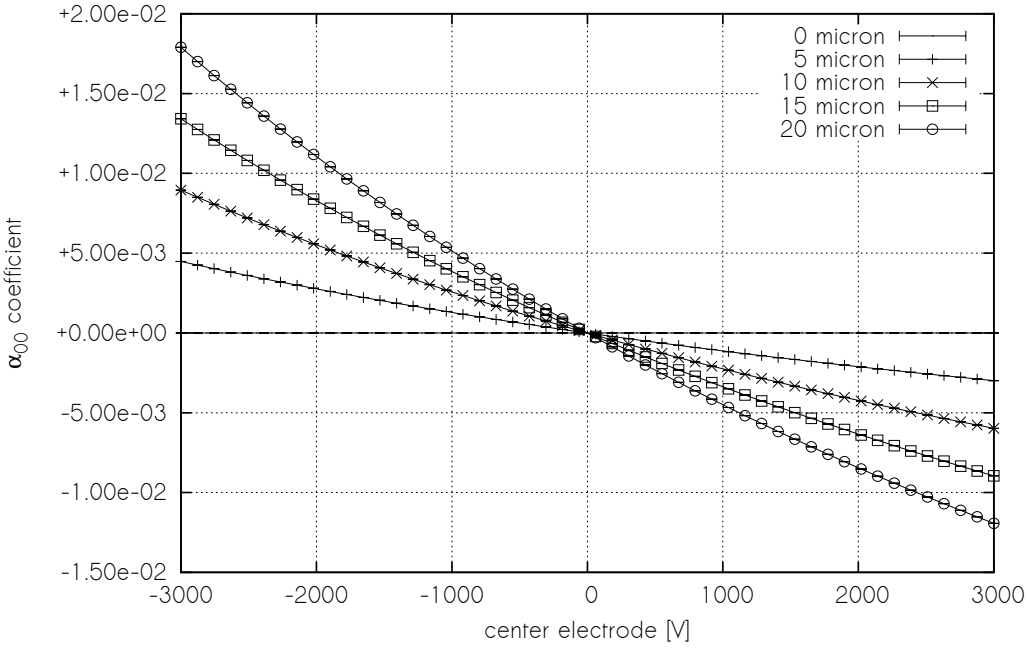

Figure 32 – Graphical representation of the zeroth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

5.3.2. First-order Coefficient

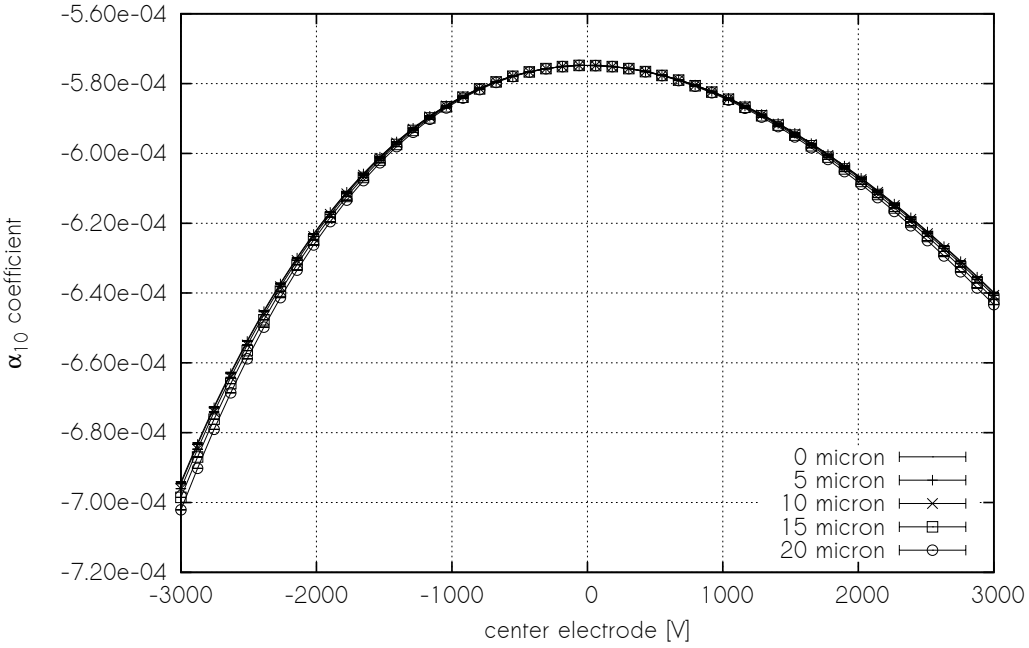

Figure 33 – Graphical representation of the first order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

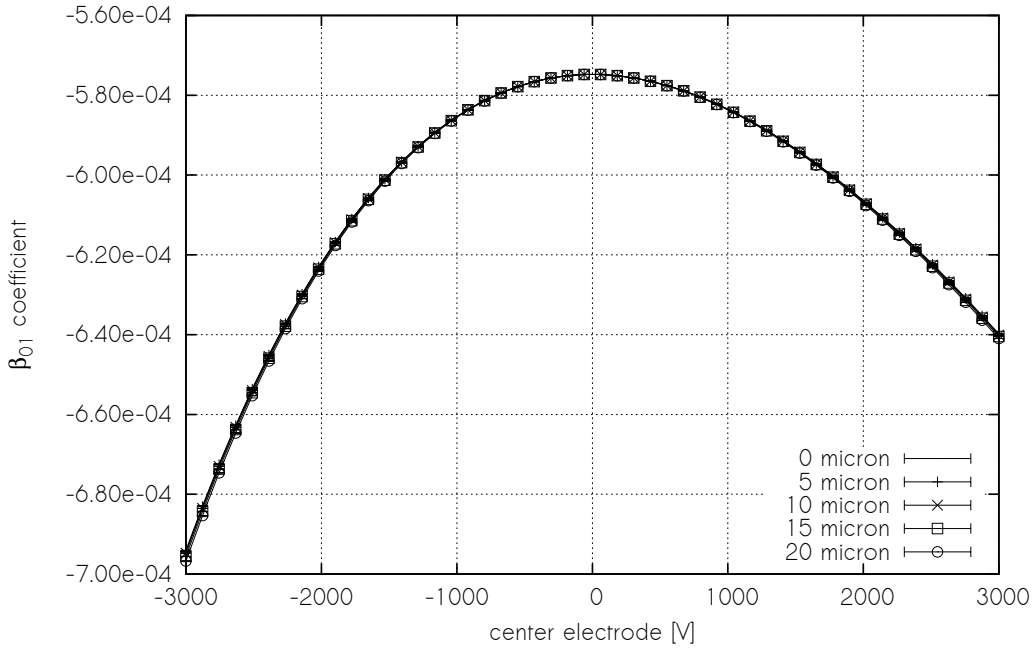

Figure 34 – Graphical representation of the first order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

5.3.3. Second-order Coefficient

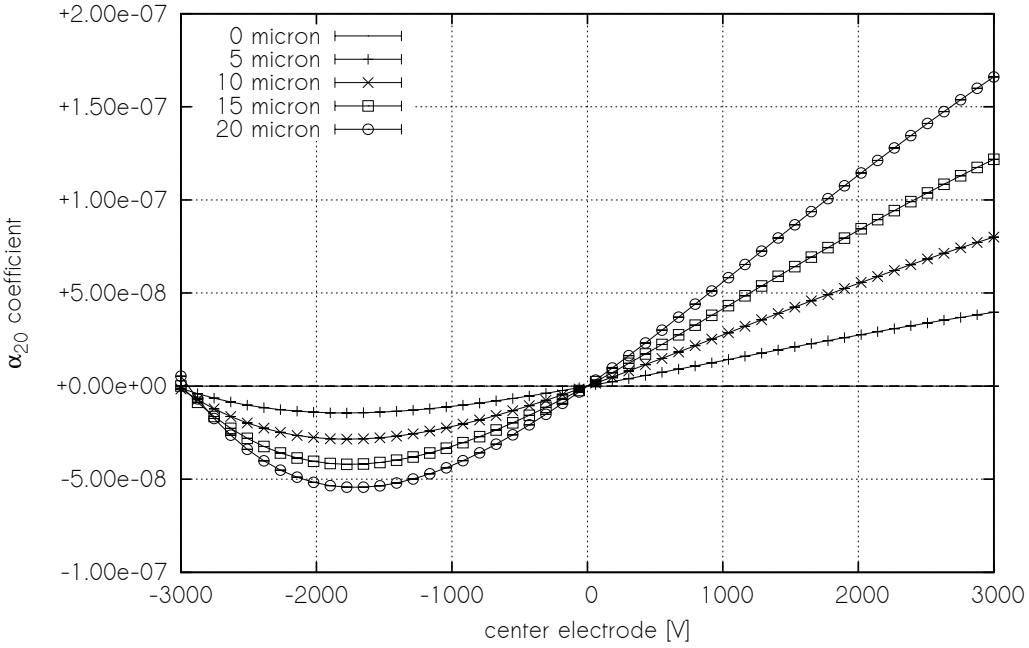

Figure 35 – Graphical representation of the second order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^2y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

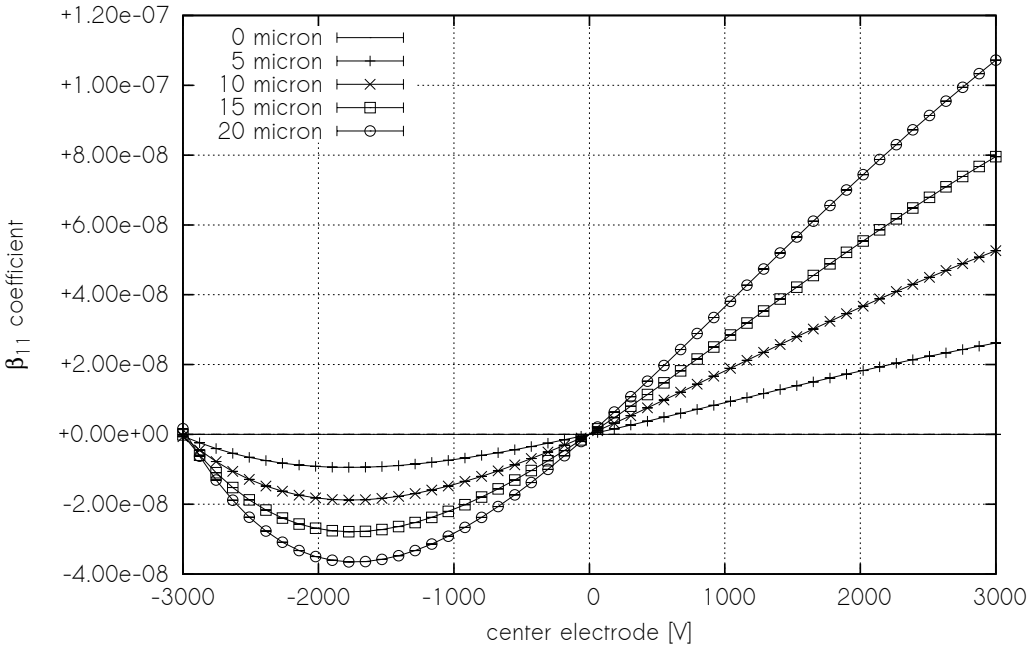

Figure 36 – Graphical representation of the second order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

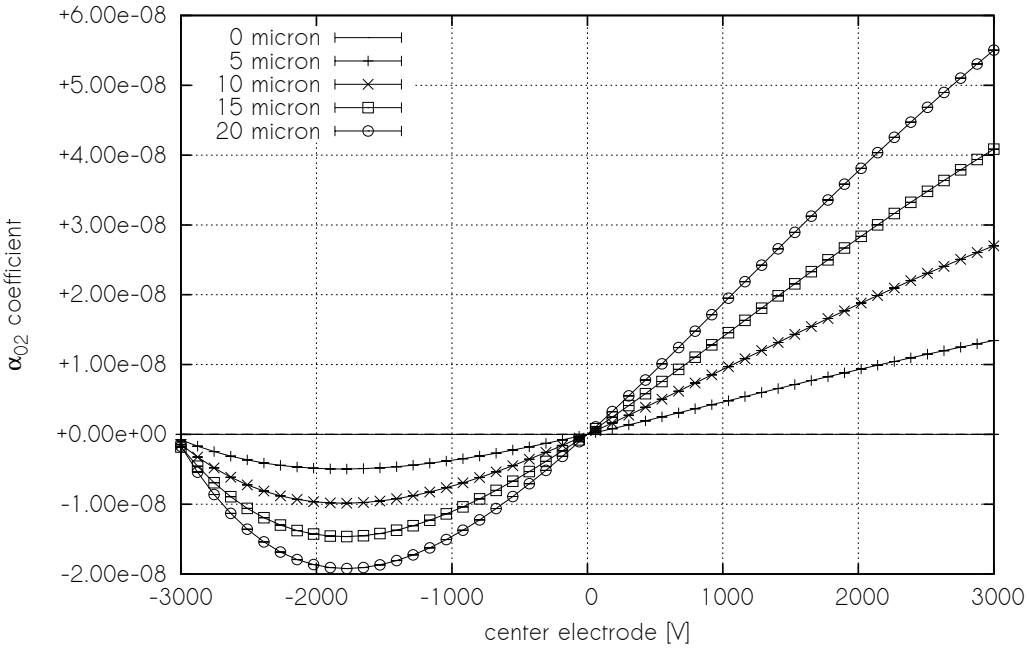

Figure 37 – Graphical representation of the second order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^2-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

5.3.4. Error Analysis

Results have a meaning only when they are given with an error estimate. With respect to our simulation, we can think of five possible causes for the error in the coefficients $\alpha(D, V)$ and $\beta(D, V)$,

1. Truncation errors due to finite precision.
2. Error in trajectory due to inexact solving of boundary values.
3. Error in trajectory due to discrete boundary triangulation.
4. Error in trajectory due to time-step of integration.
5. Statistical error in fitting the coefficients due to number of trajectories.

We assume that error contributions from the first two items are negligible when compared to the others. The total error due to items 3 to 5 combined is estimated by repeating the simulation with a 10% increase of the number of triangles (25.700 $\rightarrow$ 28.270), number of iterations (2000 $\rightarrow$ 2200) and number of particles (4096 $\rightarrow$ 4505). The estimated error is then determined by taking twice the difference with respect to the initial measurement. This concept is illustrated in figure 38. The underlying assumption is that such an increase of the parameters is a process that converges. In other words, the result of a third run with another 10% increase with respect to the second run is expected somewhere inside the domain of convergence.

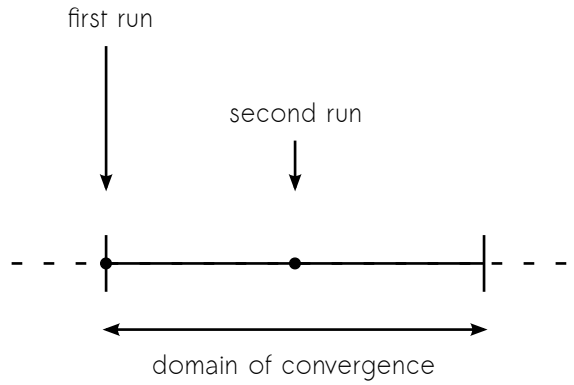

Figure 38 – Illustration of how the error is analyzed. The simulation is repeated with a higher accuracy (second run). The estimated error is determined by taking twice the difference with respect to the initial measurement.

This analysis is applied to *all* coefficients and results in the error bars displayed in figures 32 till 37 and all figures in appendix B.

Although we have an error estimate in the figures, what we are really looking for is an estimate for the error *per* order, because that is more representative in terms of electron optical analysis. Instead of maintaining all errors in shift and voltage sweeps per coefficient, we consider the maximum relative error per order which is given in table

5. This error is obtained by taking the maximum of the relative error of all points in the plot of a coefficient. Actually, this gives us two tables, one for the α-coefficients and one for the β-coefficients,

$$\epsilon^{\alpha}_{i,j} = \max \epsilon^{\alpha}_{i,j}(D, V) \tag{90}$$

$$\epsilon^{\beta}_{i,j} = \max \epsilon^{\beta}_{i,j}(D, V) \tag{91}$$

The resulting error in table 5, however, is the maximum of that as well,

$$\epsilon_{i,j} = \max\left(\epsilon^{\alpha}_{i,j}, \epsilon^{\beta}_{i,j}\right) \tag{92}$$

Where i is related to the order and j is related to the power of y in the Taylor expansion, such that $\epsilon_{i,j}$ is the maximum relative error corresponding to the terms in the expansion with $x^{i-j}y^{j}$. This means that the relative error in the α-coefficient and β-coefficient is always less than or equal to what is given in table 5 for a particular order.

Order	y^0-terms	y^1-terms	y^2-terms	y^3-terms	y^4-terms	y^5-terms
0	2×10^{-10}					
1	7×10^{-10}	7×10^{-10}				
2	4×10^{-6}	7×10^{-6}	1×10^{-5}			
3	5×10^{-6}	7×10^{-6}	7×10^{-6}	5×10^{-6}		
4	4×10^{-3}	6×10^{-3}	5×10^{-3}	1×10^{-2}	4×10^{-3}	
5	3×10^{-3}	5×10^{-3}	4×10^{-3}	4×10^{-3}	6×10^{-3}	4×10^{-3}

Table 5 – Table showing the largest relative errors in the deflection coefficients. Each error in this table corresponds to a particular term in the expansion of equation (88) and (89). The i-th row (order) and j-th column (power of y) represents the term in the expansion with $x^{i-j}y^{j}$. The relative error in the α-coefficient and β-coefficient is always less than or equal to what is given in the table for a particular order. An explicit interpretation for the first few orders is given in the text.

The first few orders in table 5 are interpreted as follows. The zeroth order coefficient has a maximum relative error of 2×10^{-10} for the $\alpha_{0,0}$ coefficient, which is the largest maximum relative error found in figure 32. The first order coefficient has a maximum relative error of 7×10^{-10} for the $\alpha_{1,0}$ coefficient and 7×10^{-10} for the $\beta_{0,1}$ coefficient, which can be found in figures 33 and 34 respectively. The second order coefficient has a maximum relative error of 4×10^{-6} for the $\alpha_{2,0}$ coefficient (figure 35), 7×10^{-6} for the $\beta_{1,1}$ coefficient (figure 36) and 1×10^{-5} for the $\alpha_{0,2}$ coefficient (figure 37).

6. Discussion

We divide the discussion into three seperate subjects. At first we discuss the performance of the boundary element kernel with respect to the available hardware. We focus in particular on the results of figures 16, 17 and 18. The second topic relates to the shift lens simulation. We discuss the first few orders of the coefficients given in figures 33 till 37 and verify whether the results are consistent with our expectations. We also compare the program of this thesis against the conventional programs. Finally we discuss the use of this simulator specifically for the case of emission tips and the consequence of including Coulomb interactions to the theory.

6.1. Boundary Element Kernel

The first remarkable observation in figure 18 is that the performance ratio of the GPU's for single precision versus double precision is of order 10. This comes however as no surprise: The Quadro FX-4800 and GeForce GTX-480 have 8 times as many arithmetic logical units (ALU's) for single precision than for double precision[21]. What this means is that when the ALU's for double precision are saturated, the arithmetic operations are queued to the available ALU's. Fortunately, the performance ratio of single precision versus double precision is reduced to a factor of 2 for the newer generation of graphics cards from NVIDIA.

The second observation is that the performance measure in figure 17 is independent of the block size. This means that the execution time is completely dominated by arithmetic. This is good news, because this tells us that the performance of the boundary element kernel scales with newer graphics cards[22]. This also holds for INTEL processors in figure 16, where we can see that the performance increases linear with respect to the number of physical cores. Although the INTEL processors support hyperthreading, note that the performance plot flattens out when the number of threads is larger than the physical core count. This means that the arithmetic load per thread is too large for hyperthreading to make a difference.

The reader might wonder if the copying of states and field evaluations between host and device in the tracer (see figure 22) causes significant overhead and delays. The increase in computation time due to the memory transfers is negligible, because the calculation of the electric field practically consumes all computation time (99%). The shift lens, for example, involves approximately 10^6 memory transfers from host to device and vice versa. With respect to *total* computation time, the time consumed by these transfers is of order minutes. We consider that negligible in comparison to the 61 hours mentioned in table 4.

[21] Single precision is usually sufficient for image processing, analysis and graphical applications. This is the most common field of application for these graphics cards in particular.

[22] Newer graphics cards are more likely to offer *increased* parallelism instead of faster memory.

6.2. Shift Lens Simulation

6.2.1. Consistencies in the Aberration Curves

At first we discuss the curves of the Taylor coefficients given in figures 32 to 37. We do not want to go into a detailed physical explanation of those curves, because the goal of this project is the design of a simulator and not to extract physical conclusions from the aberration coefficients. The physical interpretation of those curves is discussed in the latest article by Zonnevylle[23]. Instead we discuss the results by verifying some expected consistencies.

At first we consider the case where the potential of the center electrode is equal to the potential of the neighbor electrodes ($\Delta V = 0$). This means that we can *ignore* the center electrode, because it does not contribute to the field inside the lens anymore. The results must therefore be unaffected by displacements of the center electrode. This is precisely what we see: The displacement curves of *every* figure collapse together at $\Delta V = 0$, meaning that the coefficients are independent of the amount of displacement.

The second consistency is related to the fact that when there is no displacement, the shift lens becomes completely rotationally symmetric. This has at least the following two consequences, (1) the zeroth order coefficient must vanish and (2) the lens must be free of astigmatism. The reason for (1) is that a global translation to the deflection angle in one particular direction destroys rotational symmetry. We see in figure 32 that the curve for no displacement is on top of the horizontal axis and thus vanishes as required. The reason for (2) is that the deflection in the x direction and y direction must be the same because of rotational invariance. We see in figure 31 that the curve for no displacement is also on top of the horizontal axis and thus vanishes as required.

For the last consistency, consider the case where the potential of the center electrode is either 2 kV, 5 kV or 8 kV. The electrons are *accelerating* if the potential is increasing and *decelerating* if the potential is decreasing. As we go from one electrode to the other, the potential increases or decreases and thus the particles are accelerating or decelerating in that region accordingly. If the potential of the center electrode is 8 kV, then from left to right we have the following sequence: accelerate, accelerate, accelerate, decelerate, accelerate. The sequence for 2 kV is: accelerate, decelerate, accelerate, decelerate. The 5 kV case is special because then the potential does not change from the second electrode to the fourth electrode and the sequence reduces to: accelerate (electrode 1 $\rightarrow$ 2), decelerate (electrode 4 $\rightarrow$ 5). Note that the *largest* momentum at the center is obtained for the 8 kV case. The *lowest* momentum at the center is obtained for the 2 kV case. This is directly related to the accuracy, because we use a fixed time step for the geometrical integrator. If the momentum is large, then the electron takes bigger steps in space. The consequence is that the electric field is less accurately probed and therefore increases the error of the trace. The errors of the shift lens are exceptionally

[23] The title of that article is ,,Deflection properties of an electrostatic electron lens with a shifted electrode." and has been submitted to JVST for publication.

small and this effect can only be seen in the figures of the highest order with the largest displacement. For example, the curve of 20 micron displacement in figure 53 (appendix B) shows that the error bars from left to right are increasing as expected.

6.2.2. Comparison to the Conventional Programs

The only *conventional* program in table 1 that also uses the boundary element method is CPO3. That program, however, does *not* produce the second order aberrations of the shift lens. The only sensible reason we can think of is because that program has a strict limitation on the number of triangles. It does not allow more than 5.000 triangles, probably to protect the user from too large computation times. And for good reason: CPO3 would take 'forever' with 25.700 triangles applied to 250 simulations involving 4.096 particles. If we roughly estimate that the performance of CPO3 becomes comparable to our boundary element kernel for INTEL processors, then the shift lens simulation would take more than three quarters of a year.

We do not have a hard encoded restriction with respect to element count, because there is no reason to do so. In the worst case we are *practically* limited by the amount of available memory. For example, the Green's matrix for the shift lens has about 600 million entries. The amount of memory required for such a matrix is already close to 4.5 GB. On the other hand, the trace of an electron with 2.000 iterations in an electric field due to 25.700 triangles takes effectively about 200 milliseconds using the GeForce GTX-480 running at double precision (see table 4). Therefore it is not computation time, but available memory that limits our boundary triangulation.

We think that CPO3 fails because it cannot handle 25.700 triangles. If CPO3 could, then the results are expected to become similar, because the method of field calculation is essentially the same[24] and both integrators are fourth order (maybe CPO3 needs a few more iterations due to non-symplectic integration). The remaining conventional tools (SIMION, MEBS and OPERA) either use the finite element method or the finite difference method for field calculations. We have excluded EOD from that list because it does not support full three dimensional electrostatics. At first we note that Cubric et al. [17] concluded that the BEM converges faster than FEM and FDM in three dimensional electrostatic simulations. That conclusion, however, does not consider the nature of parallelism which can drastically affect the convergence rate for a whole ensemble of trajectories. The field calculations of FEM and FDM are based on discretizing solution space and as consequence of that, suffer from sub-optimal vector parallelism in the following way. The electric field at the particles is in general obtained from different parts of solution space. One particle is here and another is there and in order to obtain the fields, we need to consider different parts of solution space[25]. This does not match

[24] We do not know how CPO3 computes the integrals.

[25] This leads to randomized memory access and pushes the caching efficiency to the lowest when many particles are distributed over solution space. In the worse case we also need the surrounding elements or grid nodes in order to numerically differentiate the scalar potential.

the design of a vector processor: particles are not treated equally. The best parallism in that case is offered by *scalar* processors.

We now argue that vector parallelism outperforms the scalar type with respect to three dimensional electrostatic field calculations. Consider the performance graph of the INTEL processors in figure 16. Furthermore, we *assume* that the performance increases linearly with the number of physical cores[26]. How many physical INTEL X5650 cores do we need in order to meet the double-precision performance of the GeForce GTX-480? The answer is approximately 135 physical cores. In order to illustrate that vector parallism for field calculations is superior in the *extreme* sense, consider the latest graphics card from NVIDIA with KEPLER architecture. The estimated double precision performance (based on TFLOPS) for this architecture is about 10^{11} boundary charges per second. About 3.500 physical INTEL X5650 cores are required to meet that performance. In other words, the performance is comparable to a *super computer* and comes already for the amazing price of approximately € 600 in the Netherlands. Cluster 10 of these devices into *one* big tower and you have the estimated performance replacement of 35.000 physical INTEL X5650 cores. There is no other conclusion: Vector parallelism in BEM is superior in three dimensional electrostatic field calculations.

6.3. Emission Tips and Coulomb Interactions

Electron optical simulations sometimes include the emission of electrons from a cathode, see for example Verduin et al. [12]. The structure of simulation changes for emission tips, because then the number of particles N is no longer a constant. Tips, whether they are cold-field emitters or thermal emitters, introduce a *current* to the simulation. This means that electrons are created at random at the emission surface and eliminated after the measurement plane. Although this introduces additional practical complexity, it does not change the theoretical framework. The conservation laws (time-reversal symmetry and conservation of phase-space) are still valid. The reason is that the equations of motion are *independent* per particle and therefore the conservation laws are applicable per particle as well. What is different with respect to the shift lens is that the summation of the total energy (involving *all* particles) does not add up to a constant. That, however, is not a necessary requirement of the conservation laws. Let us be more precise: the Hamiltonian of this system can be written as a sum of N independent one-particle Hamiltonians,

$$\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2 + \cdots + \mathcal{H}_N \tag{93}$$

The conservation laws are valid for Hamiltonian systems, regardless of how many particles are involved. Therefore the advantages of symplectic integrators are still of use with respect to creating and eliminating particles.

[26] This is an over-estimate, because in reality, the independent nature of the scalar processors introduces additional overhead.

The game seems to change when we start to include Coulomb interactions. At first we consider the interacting Hamiltonian with a constant number of particles,

$$\mathcal{H}_{e-e} = \mathcal{H} + \frac{1}{4\pi\epsilon_0} \sum_{i=1}^{N} \sum_{j \neq i}^{N} \frac{\lambda_i \lambda_j}{|\mathbf{q}_i - \mathbf{q}_j|} \tag{94}$$

This is only a classical approximation because the interactions are considered *instantaneous*, which is forbidden by relativity. This interaction term destroys conservation of phase-space, because Liouville's theorem states that flows in phase-space are incompressible and thus the divergence must be zero [25]. What has changed in the analysis is that the 'area' element in phase-space of figure 3 now includes particles contributing from the inside out[27]. This means that the flow is no longer incompressible (forces are acting from within the 'area'), hence conservation of phase-space is destroyed. To make things even worse, energy is no longer conserved on a *per* particle basis if we allow creation and elimination of particles to the interacting Hamiltonian. In order to see this, note that the interacting Hamiltonian cannot be written as a sum of independent one-particle Hamiltonians, similar to equation (93). This means that the interacting Hamiltonian can only be conserved if the total number of particles is constant. The advantages of symplectic integrators are rendered useless and perform in the case of an interacting Hamiltonian with varying number of particles as good as non-symplectic integrators of similar order.

7. Conclusion

The challenge of this project is to design a simulator for electrostatic electron optics satisfying the following requirements, (1) the electric field and electron trajectories must be accurate enough to produce at least the second order aberrations and (2) reduce the computation time by using massive concurrent ray-tracing of charged particles. We have demonstrated that these requirements are satisfied in an application that could not be simulated using conventional methods. The simulation of the shift lens produced the fifth order aberration coefficients with a relative error of less than one percent. This is an exceptional result, because (1) the fifth order aberrations are not produced by any of the conventional tools and (2) that amount of accuracy is usually accepted for the second order aberrations in electron optics. All coefficients up to fifth order are obtained in approximately 61 hours of simulation using the GeForce GTX-480 from NVIDIA running at double precision. If we include the error analysis, the total simulation time approximately doubles.

[27] The infinitessimal area of our first order derivation did not include forces from the inside out. Our analysis was based on external forces applied to the outer edge.

8. Outlook

Although the simulator has proven to be successful, we do have some ideas for improvement. Our ideas on improvement are related to (1) the symplectic integrators, (2) numerical integration of triangular elements and (3) boundary triangulation. We discuss these topics in corresponding order.

8.1. Adaptive Symplectic and Reversible Integrators

The number of field evaluations can be optimized by considering adaptive time stepping in the integration of the equations of motion. This basically means that the time step decreases whenever the particle traverses through strong fields, but becomes larger when the field is weak. In other words, with adaptive time stepping, the integrator is more accurate at strong fields than at weak fields. For symplectic integrators, the use of a variable time step significantly decreases the efficiency of the method. Actually, all of the advantages of symplectic integrators are lost in standard variable time step implementations [38, 39].

There are, however, several ways to overcome this problem. The first and most easy approach is to use multiple time stepping where the vector field is decoupled and integrated over different parts with constant time-steps [40, 41, 42]. With respect to the shift lens, the most basic implementation divides space into three regions: (1) before the lens, (2) within the lens and (3) after the lens. The smallest time step is then assigned to within the lens, where the field is the strongest. The downside is that this method is not generic, i.e. the multi-stepper needs to be defined for every specific problem.

The second approach is more generic and considers the strategy in which the adaptive step size is made reversible [43, 44, 45, 46]. This is basically achieved by introducing a differential equation for the variable time step, which is solved using symmetric methods.

The third approach is the deepest and transforms the Hamiltonian using a time reparametrization to a new system, which is either reversible or Hamiltonian [47]. This transformed system is then integrated using a constant time step, which provides a variable step implementation to the original system. Systems can be made Hamiltonian using a Poincaré transformation and depending on the characteristics of the transformation, symplectic methods can either become explicit [48, 49, 50, 51] or implicit [45, 52, 53].

We have a preference for the second (reversible time step) and third method (transforming the Hamiltonian), because those methods are independent of the problem at hand. Unfortunately, transforming a Hamiltonian using time reparametrization is complicated matter. To our best knowledge, reparametrizations of relativistic Hamiltonians in three spatial dimensions do not exist in literature. Therefore, we cannot tell whether a transformation of our Hamiltonian (18) is possible, let alone feasible. The best practical solution is probably to investigate the implementation of a reversible time step method.

8.2. Multipole Expansion of Triangular Elements

In contrast to numerical Gaussian quadrature, we illustrate another approach for evaluating the surface integrals of (48), (49) and (50). At first we rewrite the integral to the following form,

$$\varphi(\mathbf{R}) = \iint_{\triangle} \frac{\sigma(\mathbf{r})}{|\mathbf{R}-\mathbf{r}|}\,\mathrm{d}A = \frac{1}{|\mathbf{R}_{\mathrm{CM}}|}\iint_{\triangle} \frac{\sigma(\mathbf{r})}{\sqrt{1+\frac{\mathbf{r}_{\mathrm{CM}}^2-2\,\mathbf{R}_{\mathrm{CM}}\cdot\mathbf{r}_{\mathrm{CM}}}{\mathbf{R}_{\mathrm{CM}}^2}}}\,\mathrm{d}A \tag{95}$$

Where the vectors with subscript are taken with respect to the center of mass of the triangle. Taylor expanding the square root in the denominator of equation (95) gives us,

$$\varphi(\mathbf{R}) = \frac{1}{|\mathbf{R}_{\mathrm{CM}}|}\iint_{\triangle} \sigma(\mathbf{r})\left(1-\frac{1}{2}\frac{\mathbf{r}_{\mathrm{CM}}^2-2\,\mathbf{R}_{\mathrm{CM}}\cdot\mathbf{r}_{\mathrm{CM}}}{\mathbf{R}_{\mathrm{CM}}^2}+\cdots\right)\mathrm{d}A \tag{96}$$

The lowest order in this expansion evaluates to the monopole term,

$$\mathcal{Q}_0 = \frac{\iint_{\triangle}\sigma(\mathbf{r})\,\mathrm{d}A}{|\mathbf{R}_{\mathrm{CM}}|} \tag{97}$$

The next-to-lowest order in the expansion evaluates to the dipole term,

$$\mathcal{Q}_1 = \frac{\iint_{\triangle}\sigma(\mathbf{r})\mathbf{r}_{\mathrm{CM}}\,\mathrm{d}A\cdot\mathbf{R}_{\mathrm{CM}}}{|\mathbf{R}_{\mathrm{CM}}|^3} \tag{98}$$

First of all, note that all integrals in the expansion can be determined ab initio[28]. Also note that we can reuse the reciprocal square root with respect to the center of mass ($|\mathbf{R}_{\mathrm{CM}}|$), because that same term appears in every order of the expansion. This in comparison to the point charge representation, where *every* point charge of the triangle requires the evaluation of a square root. Therefore, the multipole expansion reduces the number of square root evaluations to only one per triangle. This is benficial because the evaluation of a reciprocal square root is the most expensive operation in the algorithm. The question is, however, does the multipole expansion converge faster than the point charge representation? We do not know and therefore cannot tell whether this really improves the integration. The only way is to try and see, which is unfortunately beyond the scope of time for this thesis.

[28] The integrals for higher order terms are coupled to powers of the components of the position vector $\mathbf{R}_{\mathrm{CM}}$. The dipole term, for instance, involves the following three moments: $(\mathbf{r}_{\mathrm{CM}})_x(\mathbf{R}_{\mathrm{CM}})_x$, $(\mathbf{r}_{\mathrm{CM}})_y(\mathbf{R}_{\mathrm{CM}})_y$ and $(\mathbf{r}_{\mathrm{CM}})_z(\mathbf{R}_{\mathrm{CM}})_z$. Where the integral is only affecting the components of $\mathbf{r}_{\mathrm{CM}}$ and can therefore be calculated ab initio.

8.3. Improved Triangulation

Perhaps the biggest improvement can be made in the triangulation of surfaces. The shift lens is triangulated completely by hand, this includes custom mesh refinement at sharp edges. Our approach is far from optimal, especially because the triangulation of surfaces is a well-established field in computational geometry. Consider for instance the guaranteed quality mesh generation for curved surfaces by Chew et al. [54] and the Delaunay mesh refinement algorithm by Ruppert [55].

Perhaps we can take this even one step further in the direction of automated mesh refinement based on our boundary error analysis. The idea that we have is as follows. We start with a bare triangulation of our surface, i.e. using the least amount of regular triangles[29] that captures the design. From then on we follow this procedure,

1. Determine the boundary values by solving Green's equation.
2. Evaluate the error as the deviation of the calculated potential from the expected potential for *all* triangles.
3. Find the triangle with the largest error.
4. If the largest error is greater than the maximum allowable tolerance, then refine that particular triangle (see figure 39) and restart with item 1. Otherwise we are done.

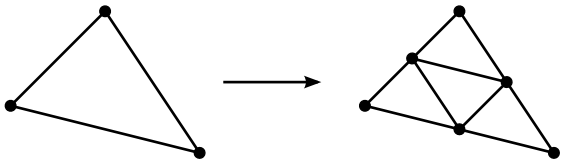

Figure 39 – Basic refinement rule of a triangular element. Three new nodes are inserted at the centers of each edge. The three nodes are connected such that the big triangle is now divided into four smaller triangles. Note that the smaller triangles are congruential by construction. The quality of the individual triangles are therefore as good as the original triangle.

The advantage of this type of refinement, when compared to standard methods in computational geometry, is that the boundary refinement is based on the error in potential. The disadvantage is that the computation time grows very fast, because each iteration involves the construction of Green's matrix and solving for the boundary values. This method is feasible as long as the computation time devoted to tracing particles is much larger than obtaining a triangulated boundary. Interesting to note is that the rule of refinement given by figure 39 breaks continuity of the surface charge density function. To see this, consider the refinement given in figure 40. Vertex B in this figure is not shared by the upper triangle and is therefore not related to the charge density function of

[29] The angles of a regular triangle are more or less similar. This is preferred because triangles become sharp when one of the angles is relatively small.

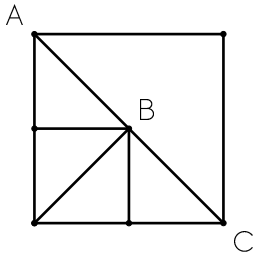

Figure 40 – Example of refinement leading to a discontinuous charge density function. Initially, a square is made out of two triangles of which only the lower triangle is refined. The surface charge density function at B is discontinuous, because that particular node is *not* a vertex of the upper triangle.

the upper triangle. This means that the surface charge density function over the entire square is no longer guaranteed to be continuous. This is not really a problem, because the only requirement is that each vertex of the boundary is unique[30].

[30] The Green's matrix becomes singular for degenerate vertices.

9. Final Remark

We have come a long way to develop this simulator. The first concept of the simulator was already created around the time of my bachelor in applied physics at Delft university of technology. The original idea was that building my own simulator gives more understanding of the underlying physics rather than using proprietary software, press a button and get the result.

The earliest simulator was based on the finite difference method in spherical coordinates and that already proved to be successful, because it was used in a article by Cook et al. [11]. At some point we started investigating other types of field calculations and in particular the boundary element method. We started playing with graphics cards for scientific calculations and figured out that the core of the simulator could be translated to the language of the graphics cards. This version allowed us to investigate an electron optical problem that could not be investigated otherwise and led to the publication of another article of which I am the main author [12]. We gave a presentation of the results of that article at the EIPBN conference in Las Vegas. After that I got involved into more advanced mathematical topics related to Hamiltonian dynamics and symplectic integration. Finally, after a long struggle with the vertex potential integrals, I managed to derive the closed-form expressions (51), (52) and (53). The total framework is presented in this thesis.

In the end I was introduced to the shift lens of A.C. Zonnevylle, who was struggling with the accuracy and computation time of the conventional programs. We decided to solve that problem with my custom built simulator. The results of the lower orders (figures 32 till 37 and some derived figures) are submitted to JVST for publication. This is the third article in line.

The higher order aberration curves in appendix B probably keep us occupied for the years to come, because it is very difficult to explain them physically. In the mean time, this simulator keeps developing and probably is going to be used for other problems in electron optics as well. The ultimate goal is to develop the fastest and most accurate simulator ever for charged particle optics. And by looking at the results of this thesis, we seem to have made a big step in the direction of achieving that.

A. Derivation of Vertex Potential

The purpose of this appendix is to reproduce the vertex response functions given by equations (51), (51) and (53). It is not our intention to give a *full* proof, instead we give the summary of each integration stage. The reason is that we value the roadmap of this derivation more than explicitly giving all integral steps.

Let us begin by choosing a particular coordinate system for a general triangle. We choose the origin of that system to coincide with vertex **A**, where the horizontal axis is in the direction of $\mathbf{B}-\mathbf{A}$. The triangle is now completely determined by the coordinates x_2, x_3 and y_3, which is illustrated in figure 41.

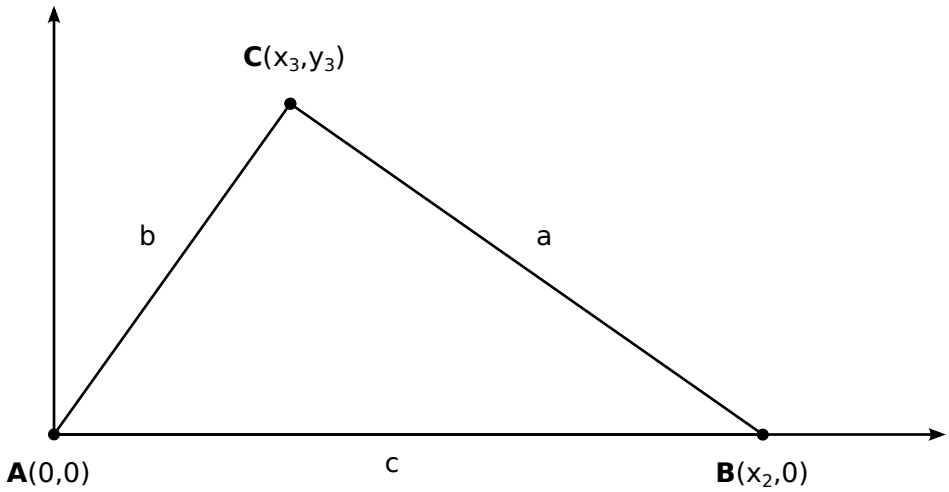

Figure 41 – Derivation.

We derive explicit closed-form expressions for the triangle given in figure 41. At first we derive the coordinate mappings and express the vertex response functions in terms of integrals for this particular coordinate system. The integrals of the vertex response functions are linear combinations of basic interpolation integrals, which are evaluated analytically. The resulting expressions, however, are given with respect to the coordinate system of figure 41. At last we generalize the interpolation integrals to coordinate-invariant expressions and transform the vertex response functions to coordinate-invariant form.

A.1. Interpolation Integrals

The coordinate mappings given in equation (44) for this particular coordinate system evaluate explicitly to,

$$\gamma_A(x,y) = 1 - \frac{1}{x_2}x + \frac{x_3 - x_2}{x_2 y_3}y \tag{99}$$

$$\gamma_B(x,y) = \frac{1}{x_2}x - \frac{x_3}{x_2 y_3}y \tag{100}$$

$$\gamma_C(x,y) = 1 - \gamma_A(x,y) - \gamma_B(x,y) = \frac{x_2}{x_2 y_3}y \tag{101}$$

The reader can verify that this mapping indeed satisfies the following conditions,

$$\begin{array}{lll} \gamma_A(0,0)=1, & \gamma_A(x_2,0)=0, & \gamma_A(x_2,y_3)=0 \\ \gamma_B(0,0)=0, & \gamma_B(x_2,0)=1, & \gamma_B(x_2,y_3)=0 \\ \gamma_C(0,0)=0, & \gamma_C(x_2,0)=0, & \gamma_C(x_2,y_3)=1 \end{array} \tag{102}$$

The charge density function (44) is a linear combination of the mapping functions and evaluates for this coordinate system to,

$$\begin{array}{c} \sigma(x,y) \\ = \\ \left(1 - \frac{1}{x_2}x + \frac{x_3-x_2}{x_2 y_3}y\right)\sigma_A + \left(\frac{1}{x_2}x - \frac{x_3}{x_2 y_3}y\right)\sigma_B + \left(\frac{x_2}{x_2 y_3}y\right)\sigma_C \end{array} \tag{103}$$

This means that the vertex response functions (48), (49) and (50) evaluate to,

$$g_A(\mathbf{A}) = \int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y} \frac{1 - \frac{1}{x_2}x + \frac{x_3-x_2}{x_2 y_3}y}{\sqrt{x^2+y^2}}\mathrm{d}x\mathrm{d}y \tag{104}$$

$$g_B(\mathbf{A}) = \int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y} \frac{\frac{1}{x_2}x - \frac{x_3}{x_2 y_3}y}{\sqrt{x^2+y^2}}\mathrm{d}x\mathrm{d}y \tag{105}$$

$$g_C(\mathbf{A}) = \int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y} \frac{\frac{x_2}{x_2 y_3}y}{\sqrt{x^2+y^2}}\mathrm{d}x\mathrm{d}y \tag{106}$$

Note that each integral is a linear combination involving integrals with $\frac{1}{\sqrt{x^2+y^2}}$, $\frac{x}{\sqrt{x^2+y^2}}$ and $\frac{y}{\sqrt{x^2+y^2}}$. These integrals are called 'interpolation integrals' and all that remains to be done is to find analytical expressions for those integrals. We start with the uniform distributed integral ($\frac{1}{\sqrt{x^2+y^2}}$), followed by the horizontal distributed integral ($\frac{x}{\sqrt{x^2+y^2}}$) and finally we address the vertical distributed integral ($\frac{y}{\sqrt{x^2+y^2}}$).

A.1.1. Uniform Distributed Integral

The uniform distributed integral involves the integration of $\frac{1}{\sqrt{x^2+y^2}}$. We start by integrating out one dimension,

$$\begin{gathered}\int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y}\frac{\mathrm{d}x\mathrm{d}y}{\sqrt{x^2+y^2}}\\ =\\ \int_0^{y_3}\log\frac{\sqrt{y^2+\left(x_2-\frac{x_2-x_3}{y_3}y\right)^2}+x_2-\frac{x_2-x_3}{y_3}y}{\sqrt{\left(\frac{x_3^2}{y_3^2}+1\right)y^2+\frac{x_3}{y_3}y}}\mathrm{d}y\end{gathered}\tag{107}$$

This gives us two new integrals, because $\log\frac{x}{y}=\log x-\log y$. The integral of the upper logarithm is evaluated as follows,

$$\begin{gathered}\int_0^{y_3}\log\left(\sqrt{y^2+\left(x_2-\frac{x_2-x_3}{y_3}y\right)^2}+x_2-\frac{x_2-x_3}{y_3}y\right)\mathrm{d}y\\ =\\ \frac{x_2y_3}{\sqrt{(x_2-x_3)^2+y_3^2}}\log\frac{\sqrt{(x_2-x_3)^2+y_3^2}\sqrt{y_3^2+x_3^2}+(x_2-x_3)^2+y_3^2-x_2(x_2-x_3)}{x_2\sqrt{(x_2-x_3)^2+y_3^2}-x_2(x_2-x_3)}\\ +\\ y_3\log\left(\sqrt{y_3^2+x_3^2}+x_3\right)-y_3\end{gathered}\tag{108}$$

The lower logarithm evaluates to,

$$\int_0^{y_3}\log\left(\sqrt{\left(\frac{x_3^2}{y_3^2}+1\right)y^2}+\frac{x_3}{y_3}y\right)\mathrm{d}y \;=\; y_3\log\left(\sqrt{x_3^2+y_3^2}+x_3\right)-y_3\tag{109}$$

The resulting expression for the uniform distributed triangle is obtained by subtracting the integrals of the upper and lower logarithm,

$$\begin{gathered}\int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y}\frac{\mathrm{d}x\mathrm{d}y}{\sqrt{x^2+y^2}}\\ =\\ \frac{x_2y_3}{\sqrt{(x_2-x_3)^2+y_3^2}}\log\frac{\sqrt{(x_2-x_3)^2+y_3^2}\sqrt{y_3^2+x_3^2}+(x_2-x_3)^2+y_3^2-x_2(x_2-x_3)}{x_2\sqrt{(x_2-x_3)^2+y_3^2}-x_2(x_2-x_3)}\end{gathered}\tag{110}$$

A.1.2. Horizontal Distributed Integral

The horizontal distributed integral involves the integration of $\frac{x}{\sqrt{x^2+y^2}}$. We start by integrating out one dimension,

$$\begin{array}{c} \int_0^{y_3} \int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y} \frac{x \mathrm{d}x \mathrm{d}y}{\sqrt{x^2+y^2}} \\ = \\ \int_0^{y_3} \sqrt{y^2+\left(x_2-\frac{x_2-x_3}{y_3}y\right)^2}\mathrm{d}y - \int_0^{y_3}\sqrt{\left(\frac{x_3^2}{y_3^2}+1\right)y^2}\mathrm{d}y \end{array} \tag{111}$$

The first term is an integral of the type $\sqrt{\alpha+\beta y+y^2}$ which evaluates to,

$$\begin{array}{c} \int_0^{y_3} \sqrt{y^2+\left(x_2-\frac{x_2-x_3}{y_3}y\right)^2}\mathrm{d}y \\ = \\ \frac{1}{2}\frac{x_2^2y_3^3}{\left((x_2-x_3)^2+y_3^2\right)^{\frac{3}{2}}}\log\frac{\sqrt{(x_2-x_3)^2+y_3^2}\sqrt{y_3^2+x_3^2}+(x_2-x_3)^2+y_3^2-x_2(x_2-x_3)}{x_2\sqrt{(x_2-x_3)^2+y_3^2}-x_2(x_2-x_3)} \\ - \\ \frac{1}{2}\frac{x_2(x_2-x_3)y_3}{(x_2-x_3)^2+y_3^2}\left(\sqrt{y_3^2+x_3^2}-\sqrt{x_2^2}\right) \\ + \\ \frac{1}{2}y_3\sqrt{y_3^2+x_3^2} \end{array} \tag{112}$$

The second term is a trivial one of the type $\int y\,\mathrm{d}y$ and evaluates to,

$$\int_0^{y_3}\sqrt{\left(\frac{x_3^2}{y_3^2}+1\right)y^2}\mathrm{d}y \;=\; \frac{1}{2}y_3\sqrt{x_3^2+y_3^2} \tag{113}$$

The resulting expression for the horizontal distributed triangle evaluates to,

$$\begin{array}{c} \int_0^{y_3} \int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y} \frac{x \mathrm{d}x \mathrm{d}y}{\sqrt{x^2+y^2}} \\ = \\ \frac{1}{2}\frac{x_2^2y_3^3}{\left((x_2-x_3)^2+y_3^2\right)^{\frac{3}{2}}}\log\frac{\sqrt{(x_2-x_3)^2+y_3^2}\sqrt{y_3^2+x_3^2}+(x_2-x_3)^2+y_3^2-x_2(x_2-x_3)}{x_2\sqrt{(x_2-x_3)^2+y_3^2}-x_2(x_2-x_3)} \\ - \\ \frac{1}{2}\frac{x_2(x_2-x_3)y_3}{(x_2-x_3)^2+y_3^2}\left(\sqrt{y_3^2+x_3^2}-\sqrt{x_2^2}\right) \end{array} \tag{114}$$

A.1.3. Vertical Distributed Integral

The vertical distributed integral involves the integration of $\frac{y}{\sqrt{x^2+y^2}}$. We start by integrating out one dimension,

$$\begin{gathered}\int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y}\frac{y\mathrm{d}x\mathrm{d}y}{\sqrt{x^2+y^2}} \\ = \\ \int_0^{y_3} y\log\frac{\sqrt{y^2+\left(x_2-\frac{x_2-x_3}{y_3}y\right)^2}+x_2-\frac{x_2-x_3}{y_3}y}{\sqrt{\left(\frac{x_3^2}{y_3^2}+1\right)y^2+\frac{x_3}{y_3}y}}\mathrm{d}y\end{gathered} \tag{115}$$

This is basically the same expression as for the uniform distributed triangle, but involves an additional y-term. The upper logarithm evalutes to,

$$\begin{gathered}\int_0^{y_3} y\log\left(\sqrt{y^2+\left(x_2-\frac{x_2-x_3}{y_3}y\right)^2}+x_2-\frac{x_2-x_3}{y_3}y\right)\mathrm{d}y \\ = \\ \frac{1}{2}\frac{x_2y_3^2}{(x_2-x_3)^2+y_3^2}\left(\sqrt{y_3^2+x_3^2}-\sqrt{x_2^2}\right) \\ + \\ \frac{1}{2}\frac{x_2^2(x_2-x_3)y^2}{\left((x_2-x_3)^2+y_3^2\right)^{\frac{3}{2}}}\log\frac{\sqrt{(x_2-x_3)^2+y_3^2}\sqrt{y_3^2+x_3^2}+(x_2-x_3)^2+y_3^2-x_2(x_2-x_3)}{x_2\sqrt{(x_2-x_3)^2+y_3^2}-x_2(x_2-x_3)} \\ + \\ \frac{1}{2}y_3^2\log\left(\sqrt{y_3^2+x_3^2}+x_3\right)-\frac{1}{4}y_3^2\end{gathered} \tag{116}$$

The lower logarithm evaluates to,

$$\int_0^{y_3} y\log\left(\sqrt{\left(\frac{x_3^2}{y_3^2}+1\right)y^2}+\frac{x_3}{y_3}y\right)\mathrm{d}y \;=\; \frac{1}{2}y_3^2\log\left(\sqrt{x_3^2+y_3^2}+x_3\right)-\frac{1}{4}y_3^2 \tag{117}$$

The resulting expression for the vertical distributed triangle evaluates to,

$$\int_0^{y_3} \int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y} \frac{y \mathrm{d}x \mathrm{d}y}{\sqrt{x^2+y^2}} = \tag{118}$$

$$\frac{1}{2}\frac{x_2 y_3^2}{(x_2-x_3)^2+y_3^2}\left(\sqrt{y_3^2+x_3^2}-\sqrt{x_2^2}\right) + \frac{1}{2}\frac{x_2^2(x_2-x_3)y_3^2}{\left((x_2-x_3)^2+y_3^2\right)^{\frac{3}{2}}}\log\frac{\sqrt{(x_2-x_3)^2+y_3^2}\sqrt{y_3^2+x_3^2}+(x_2-x_3)^2+y_3^2-x_2(x_2-x_3)}{x_2\sqrt{(x_2-x_3)^2+y_3^2}-x_2(x_2-x_3)}$$

A.2. Coordinate Invariant Representation

In principle we are done, because we now have closed-form expressions for the interpolation integrals. The vertex response functions are obtained by the linear combinations given by equations (104), (105) and (106). Note that the interpolation integrals are expressed in terms of the coordinate system given by figure 41. However, the vertex response functions of the triangle in figure 41 do not depend on *any* coordinate system. We can rotate the triangle or choose another coordinate system, but this does not affect the vertex potential.

We now transform the interpolation integrals to coordinate-invariant form. First of all, note that many of the terms in the interpolation integrals can be identified as invariants[31],

$$\sqrt{(x_3-x_2)^2+y_3^2} \rightarrow a \tag{119}$$

$$\sqrt{x_3^2+y_3^2} \rightarrow b \tag{120}$$

$$x_2 \rightarrow c \tag{121}$$

The remaining term (x_2x_3) seems to have no corresponding invariant at first sight. However, it turns out that this term can be expressed as an invariant using Heron's formula for the area of the triangle,

$$\begin{aligned} x_2x_3 &= \sqrt{b^2c^2-h^2c^2} = \sqrt{b^2c^2-4\triangle^2} \\ &= \sqrt{b^2c^2-b^2c^2+\frac{1}{4}\left(b^2+c^2-a^2\right)^2} \\ &= \frac{1}{2}\left(b^2+c^2-a^2\right) \end{aligned} \tag{122}$$

[31] Invariants, for example, are the length of the sides, the area and (semi)perimeter of the triangle.

Where $\triangle$ is the area of the triangle. The uniform distributed integral transforms to,

$$\begin{aligned}\int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y}\frac{\mathrm{d}x\mathrm{d}y}{\sqrt{x^2+y^2}} &= 2\frac{\triangle}{a}\log\frac{(a+b)^2-c^2}{b^2-(a-c)^2}\\ &= 2\frac{\triangle}{a}\log\frac{(a+b-c)(a+b+c)}{(b-a+c)(b+a-c)}\\ &= 2\frac{\triangle}{a}\log\frac{s}{s-a}\end{aligned} \tag{123}$$

The horizontal distributed integral transforms to,

$$\begin{aligned}\int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y}\frac{x\mathrm{d}x\mathrm{d}y}{\sqrt{x^2+y^2}} &= 4\frac{\triangle^3}{a^3c}\log\frac{(a+b)^2-c^2}{b^2-(a-c)^2}-\frac{1}{2}\frac{\triangle}{a^2c}\left(a^2-b^2+c^2\right)(b-c)\\ &= 4\frac{\triangle^3}{a^3c}\log\frac{s}{s-a}-\frac{1}{2}\frac{\triangle}{a^2c}\left(a^2-b^2+c^2\right)(b-c)\end{aligned} \tag{124}$$

The vertical distributed integral transforms to,

$$\begin{aligned}\int_0^{y_3}\int_{\frac{x_3}{y_3}y}^{x_2-\frac{x_2-x_3}{y_3}y}\frac{y\mathrm{d}x\mathrm{d}y}{\sqrt{x^2+y^2}} &= \frac{\triangle^2}{a^3c}\left(a^2-b^2+c^2\right)\log\frac{(a+b)^2-c^2}{b^2-(a-c)^2}+2\frac{\triangle^2}{a^2c}(b-c)\\ &= \frac{\triangle^2}{a^3c}\left(a^2-b^2+c^2\right)\log\frac{s}{s-a}+2\frac{\triangle^2}{a^2c}(b-c)\end{aligned} \tag{125}$$

Where s is the semiperimeter ($\frac{a+b+c}{2}$) of the triangle and $\triangle$ the area.

A.3. Vertex Response Functions

The vertex response functions are obtained by the linear combinations given by equations (104), (105) and (106). If we use the coordinate-invariant form of the interpolation integrals, then the resulting expressions for the vertex response functions are transformed into coordinate-invariant form as well[32]. The resulting expressions are,

$$g_A(\mathbf{A}) = \frac{\triangle}{a}\log\frac{s}{s-a} \tag{126}$$

$$g_B(\mathbf{A}) = \frac{\triangle}{a}\left(\frac{1}{2}\frac{a^2+b^2-c^2}{a^2}\log\frac{s}{s-a}-\frac{b-c}{a}\right) \tag{127}$$

$$g_C(\mathbf{A}) = \frac{\triangle}{a}\left(\frac{1}{2}\frac{a^2-b^2+c^2}{a^2}\log\frac{s}{s-a}+\frac{b-c}{a}\right) \tag{128}$$

Where the vertex response functions are expressed with invariants, such as the length of the sides (a, b or c), the area ($\triangle$) and the semiperimeter ($\frac{a+b+c}{2}$) of the triangle.

[32] This derivation involves additionally the interpretation of the constants in the linear combination as invariants.

B. Aberration Curves

B.1. Alpha Deflection Coefficients

Zeroth-order

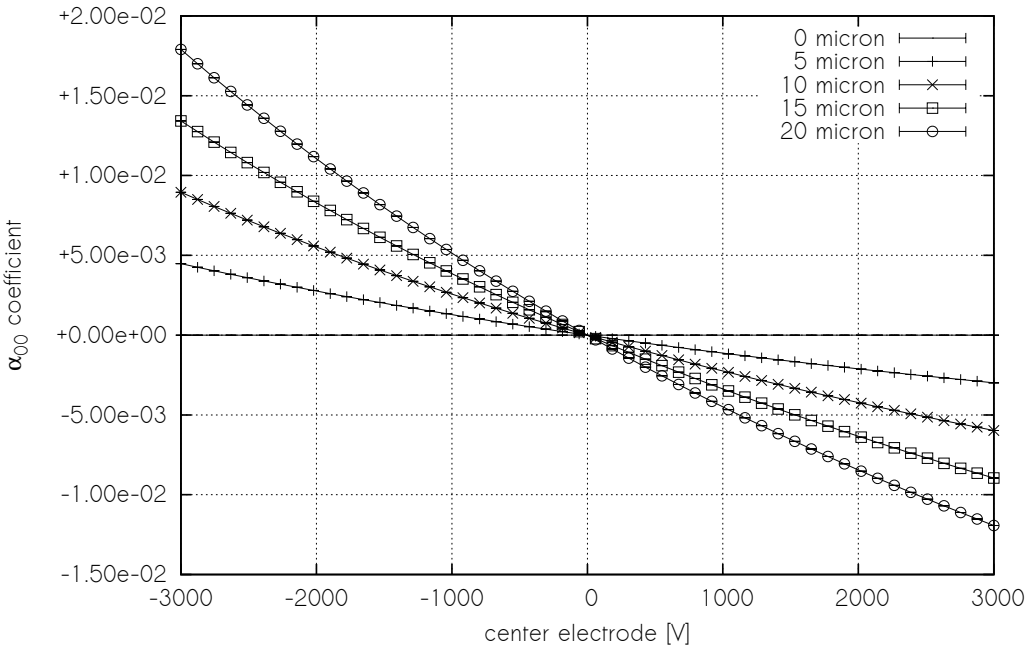

Figure 42 – Graphical representation of the zeroth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

First-order

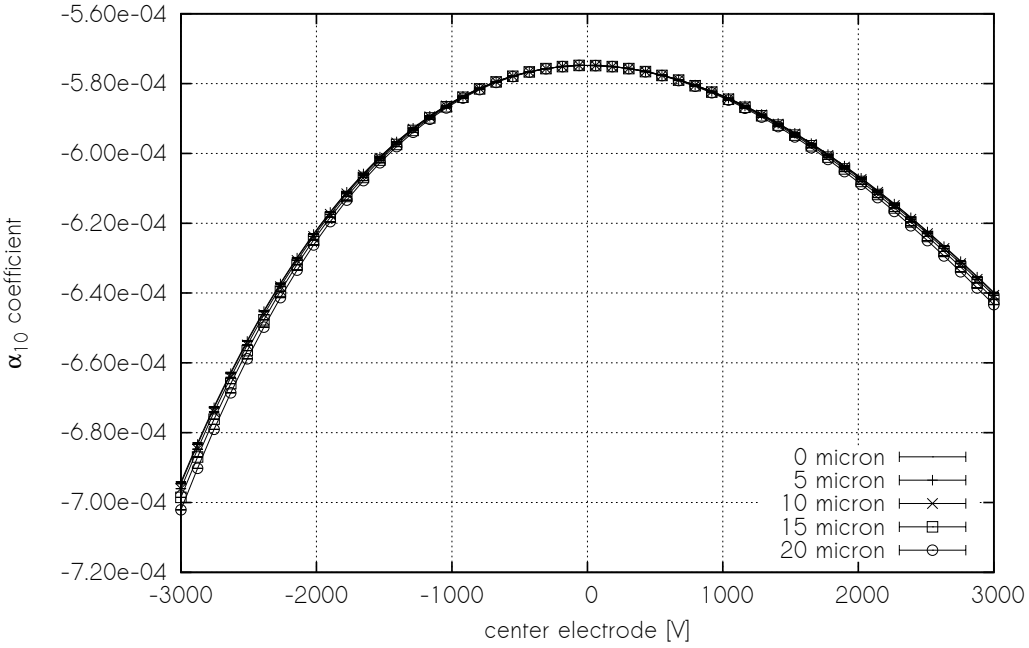

Figure 43 – Graphical representation of the first order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Second-order

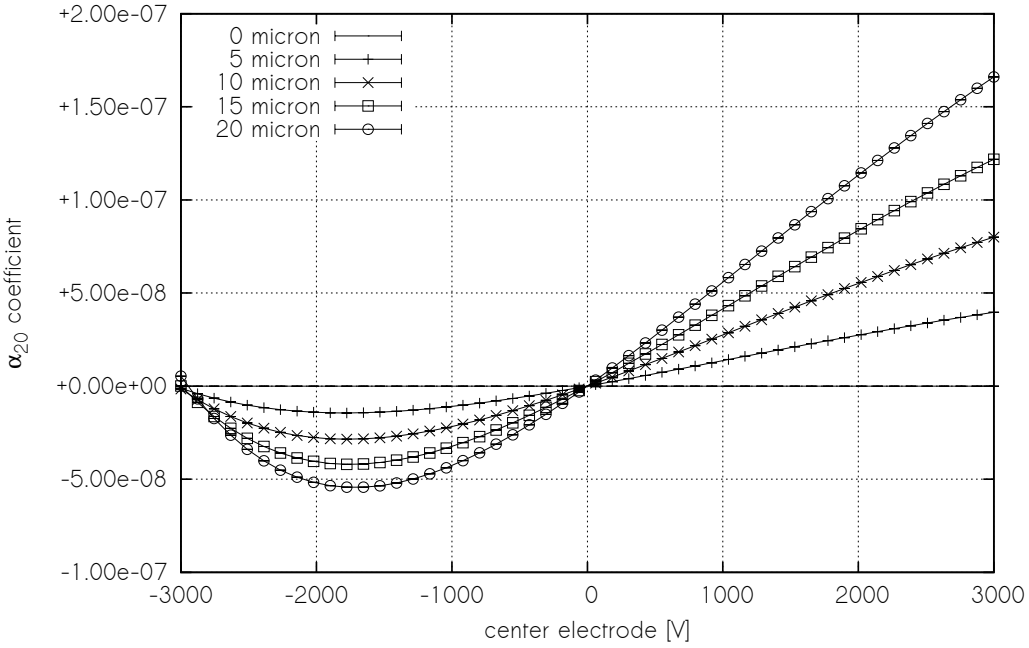

Figure 44 – Graphical representation of the second order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^2y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

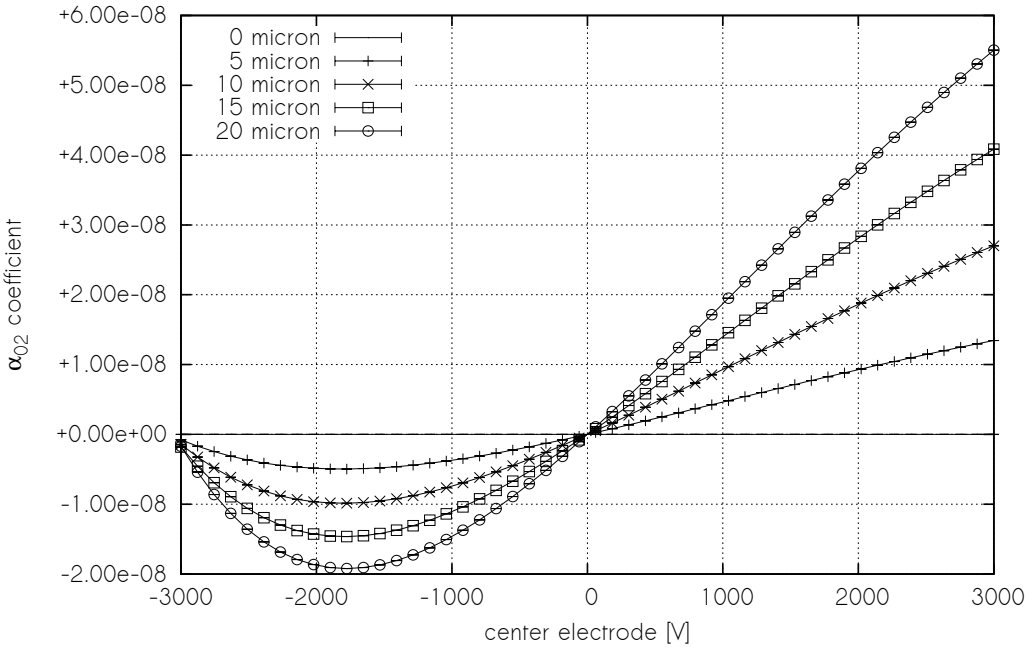

Figure 45 – Graphical representation of the second order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^2-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Third-order

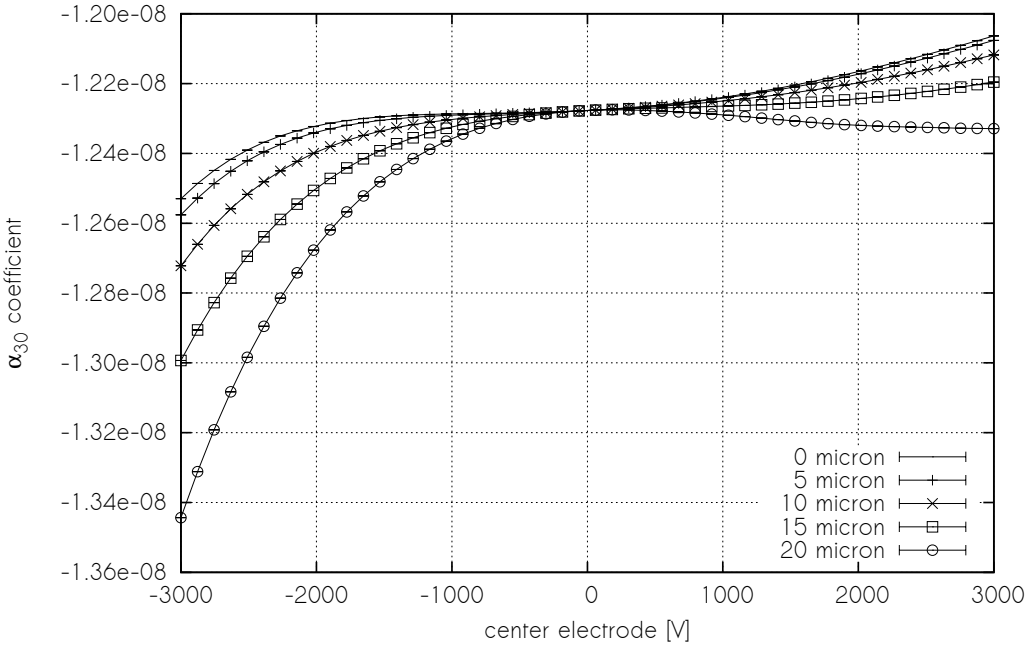

Figure 46 – Graphical representation of the third order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^3y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

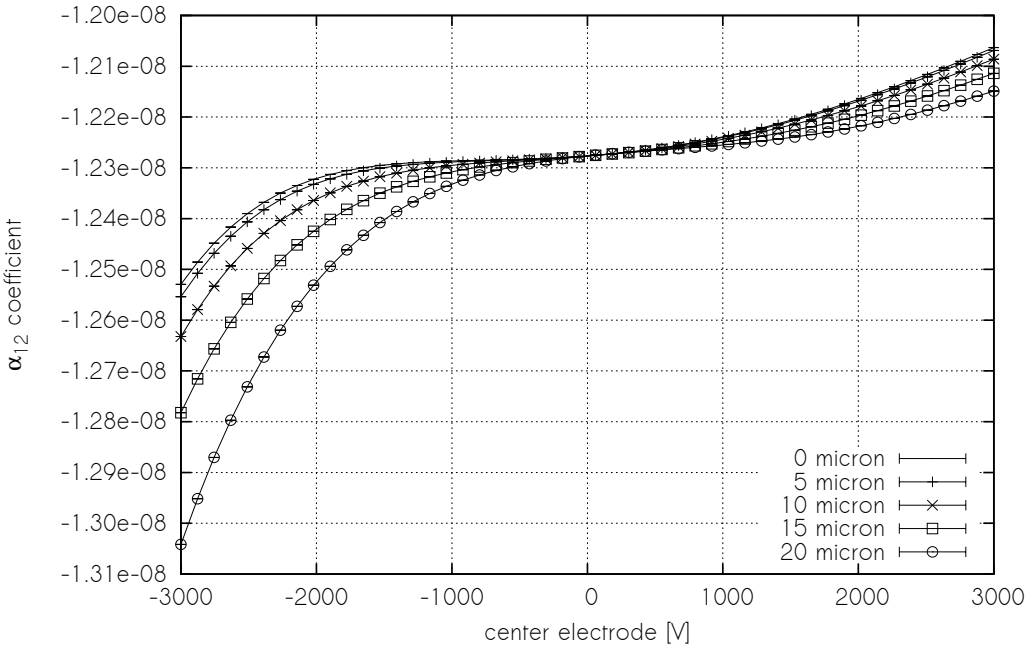

Figure 47 – Graphical representation of the third order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^2-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Fourth-order

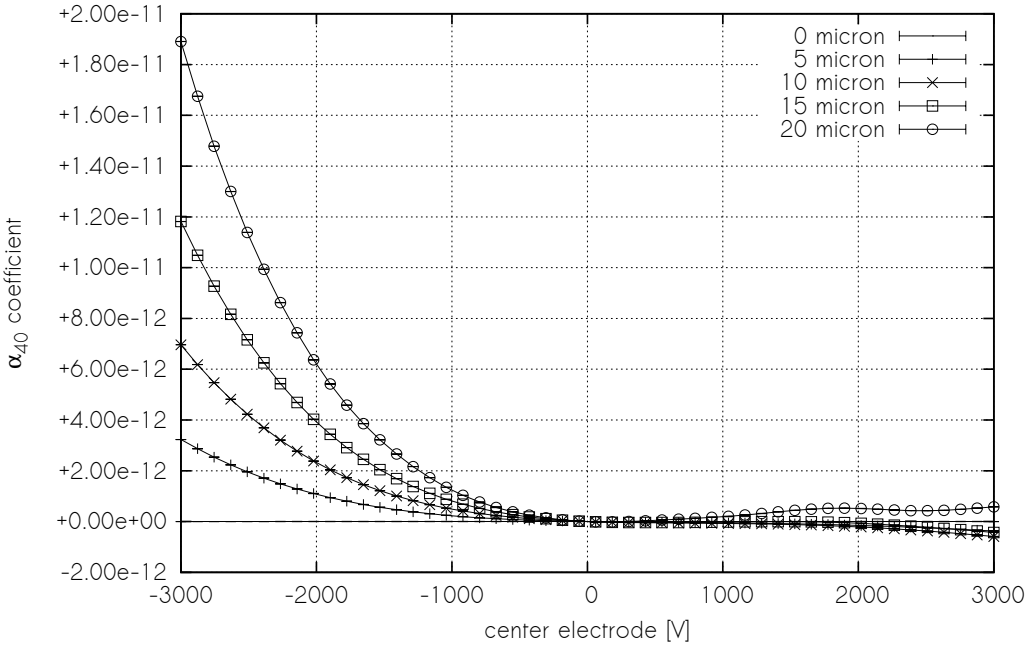

Figure 48 – Graphical representation of the fourth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^4y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

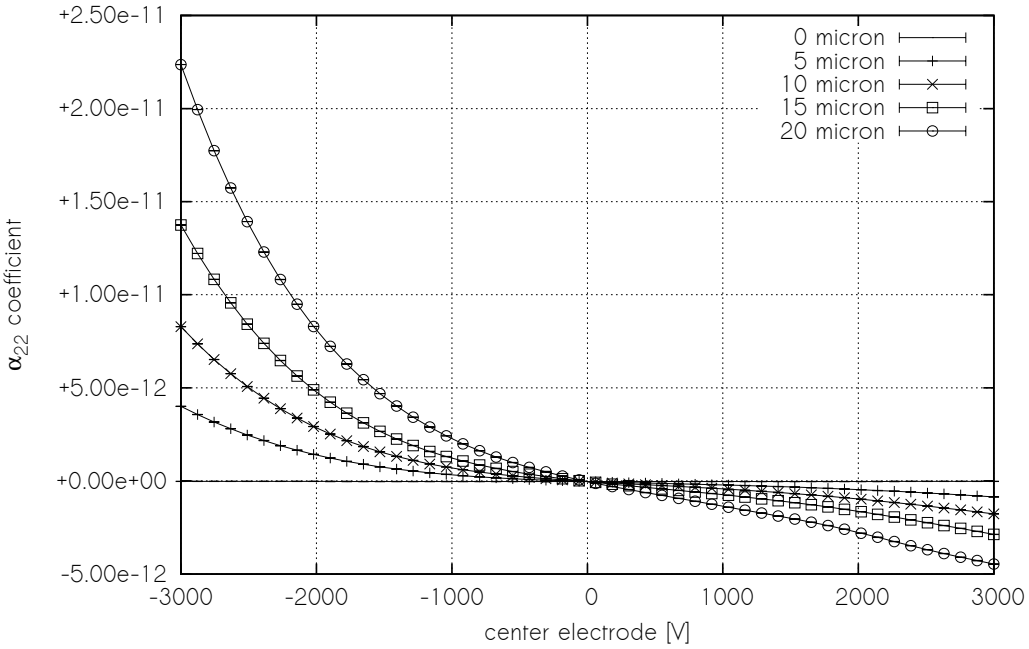

Figure 49 – Graphical representation of the fourth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^2y^2-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

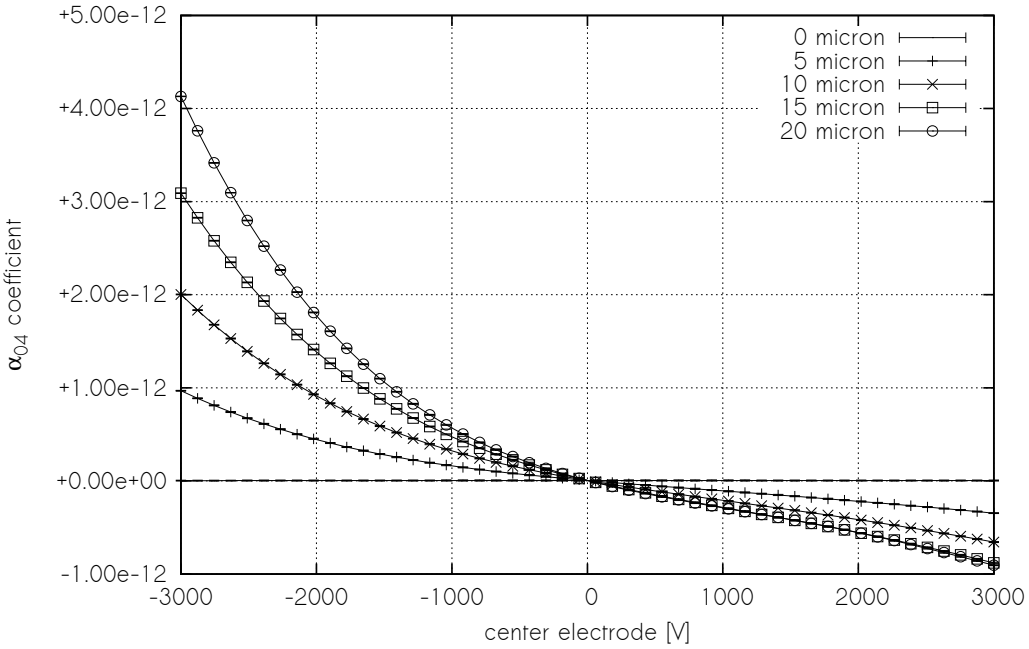

Figure 50 – Graphical representation of the fourth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^4-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Fifth-order

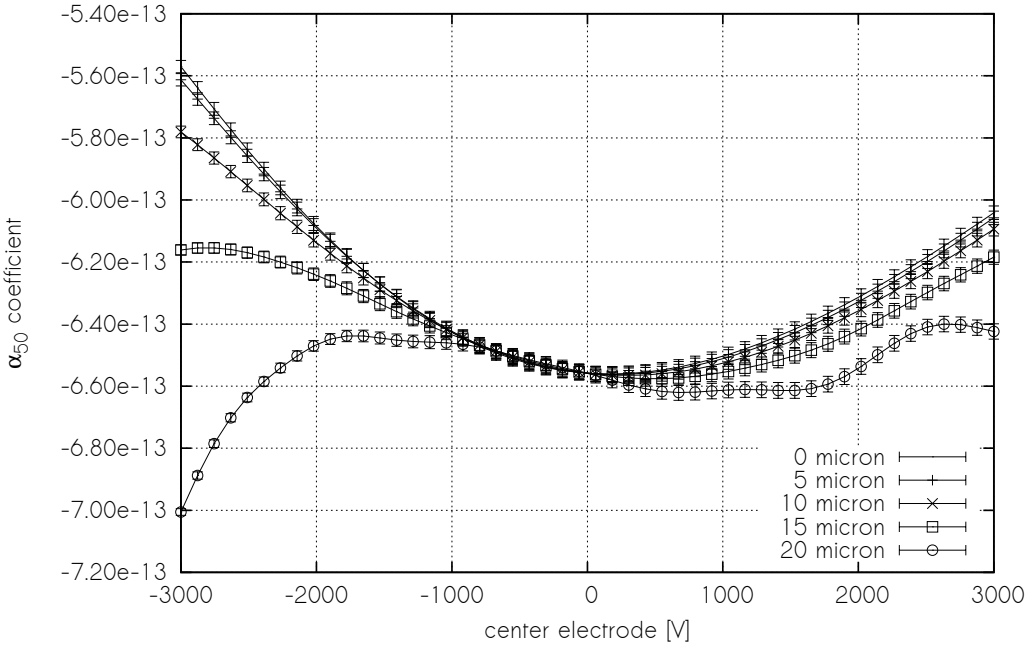

Figure 51 – Graphical representation of the fifth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^5y^0-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis.

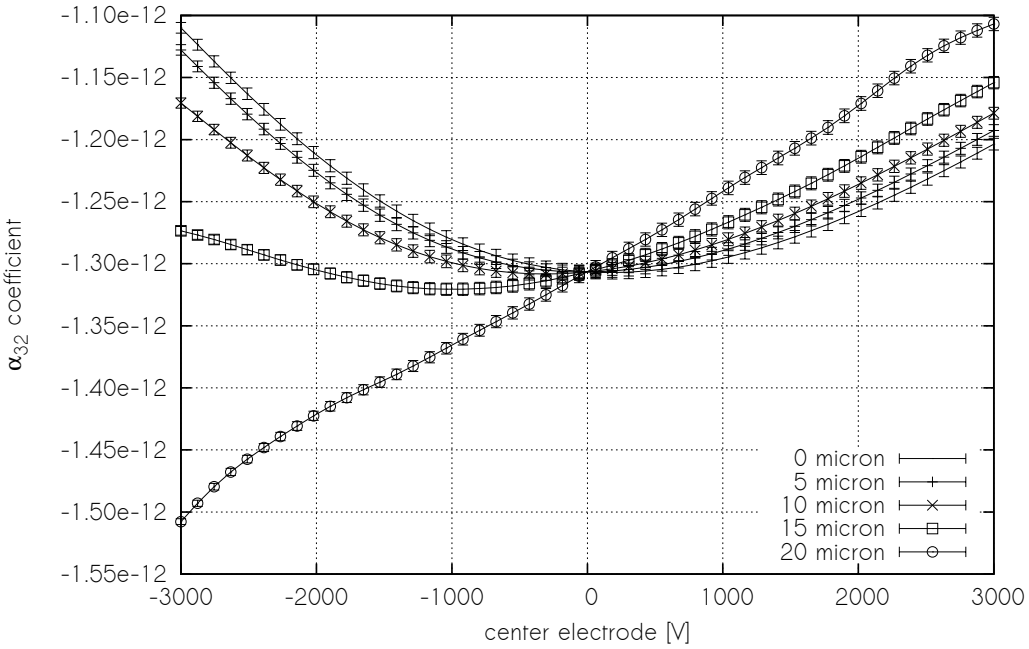

Figure 52 – Graphical representation of the fifth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^3y^2-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis.

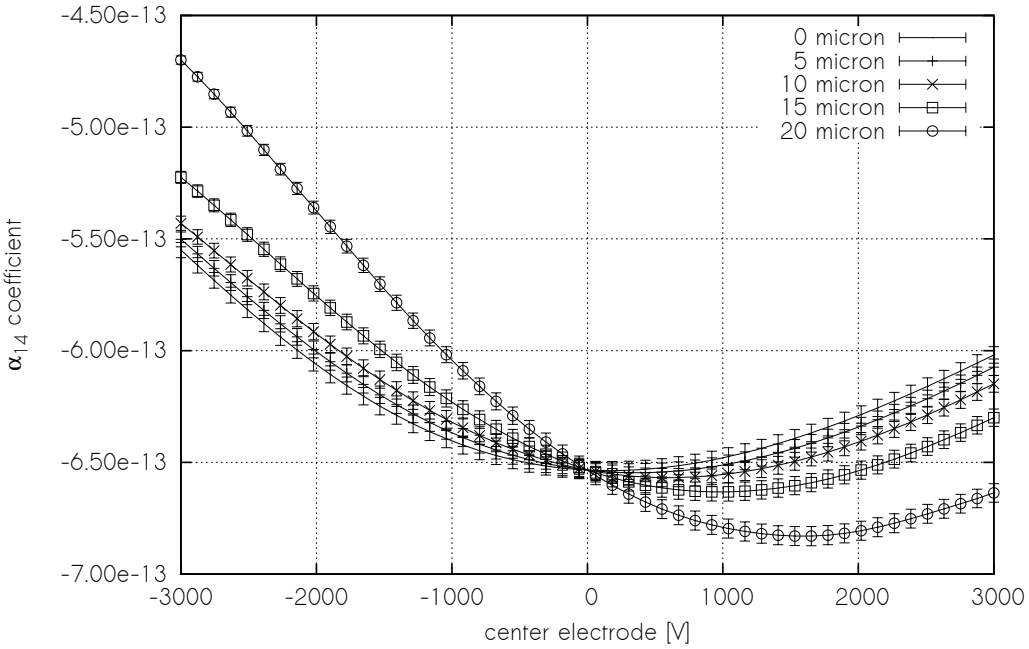

Figure 53 – Graphical representation of the fifth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^4-term in the Taylor expansion of the deflection angle given by equation (88). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis.

B.2. Beta Deflection Coefficients

First-order

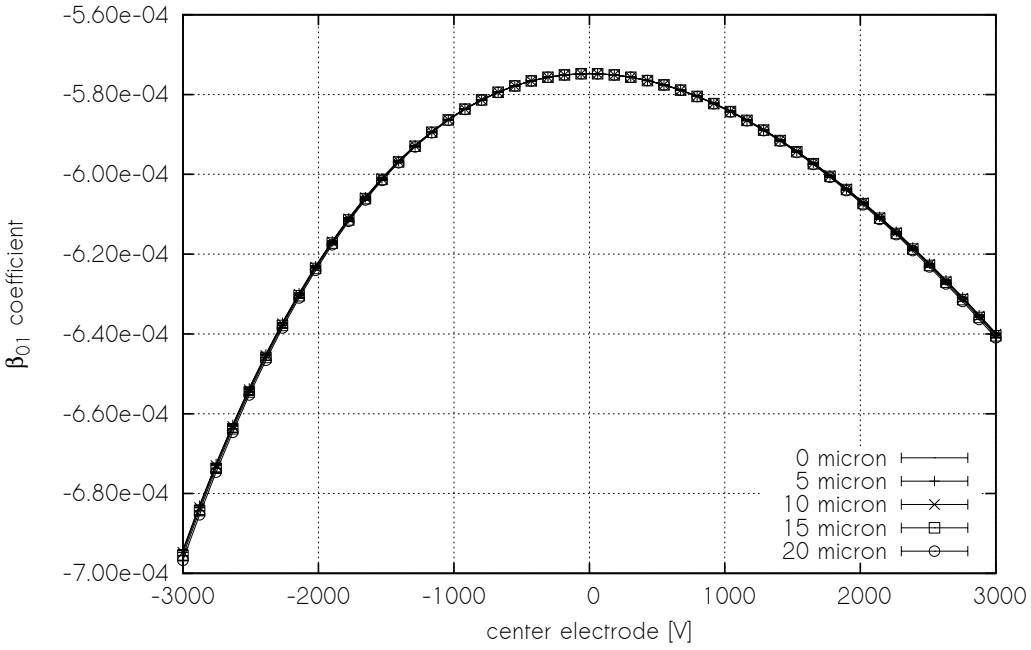

Figure 54 – Graphical representation of the first order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Second-order

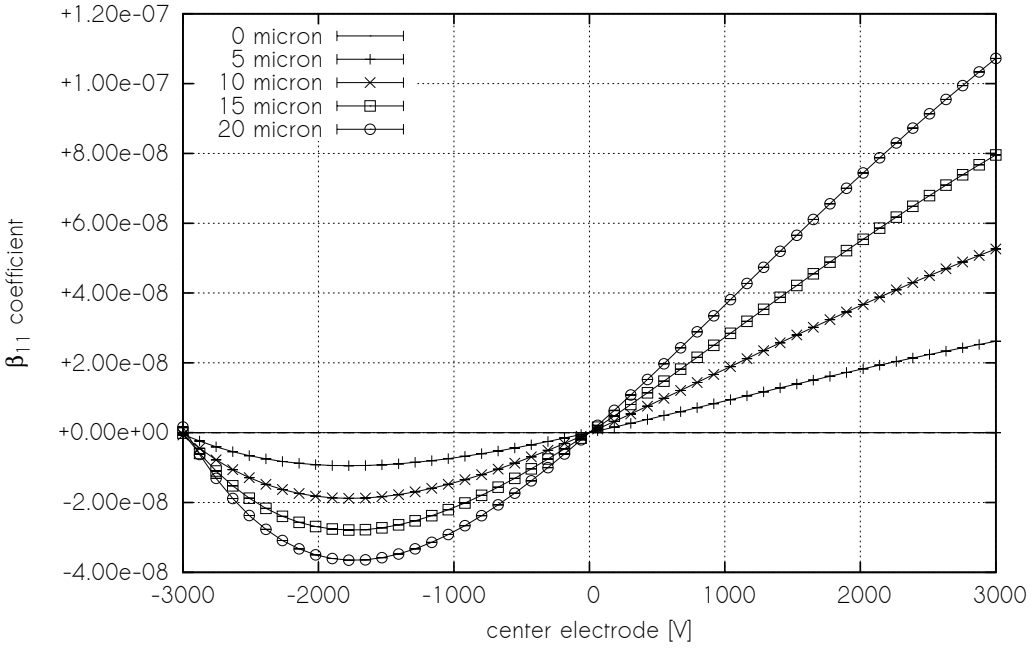

Figure 55 – Graphical representation of the second order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Third-order

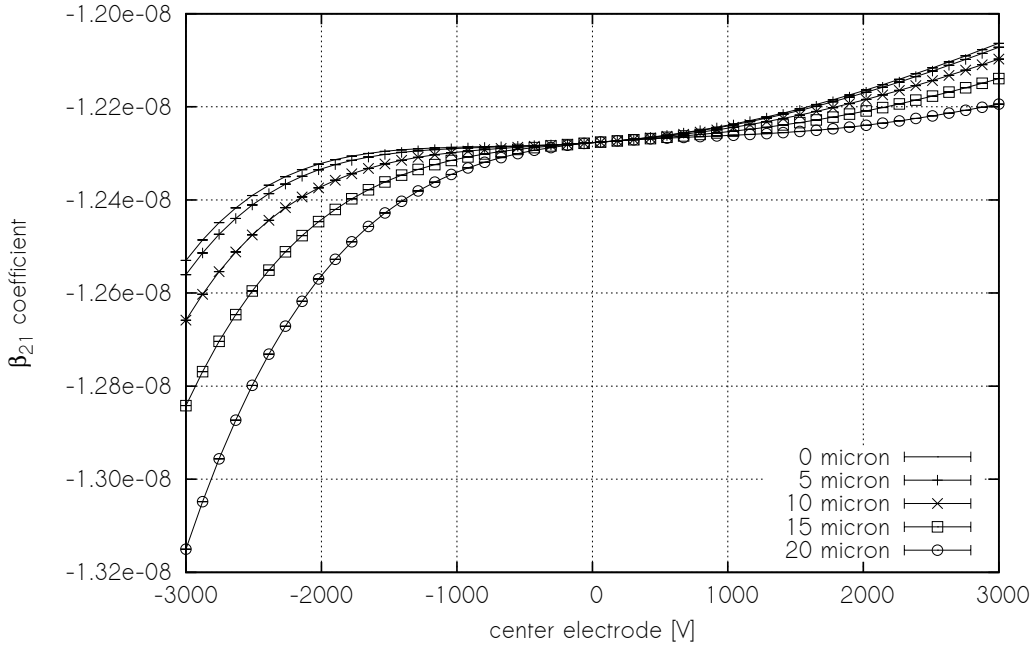

Figure 56 – Graphical representation of the third order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^2y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

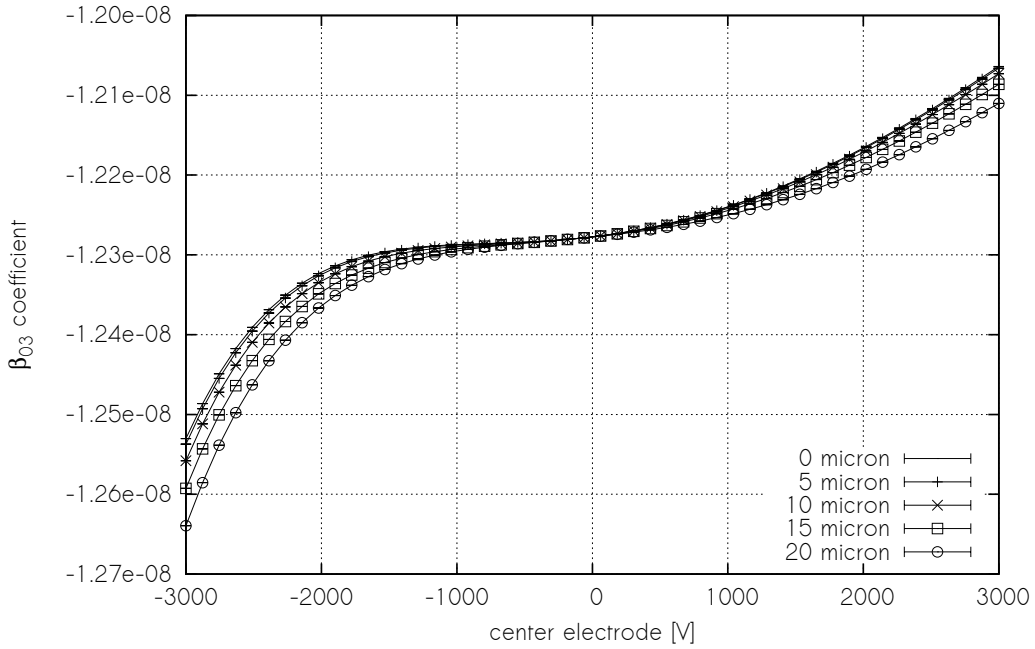

Figure 57 – Graphical representation of the third order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^3-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Fourth-order

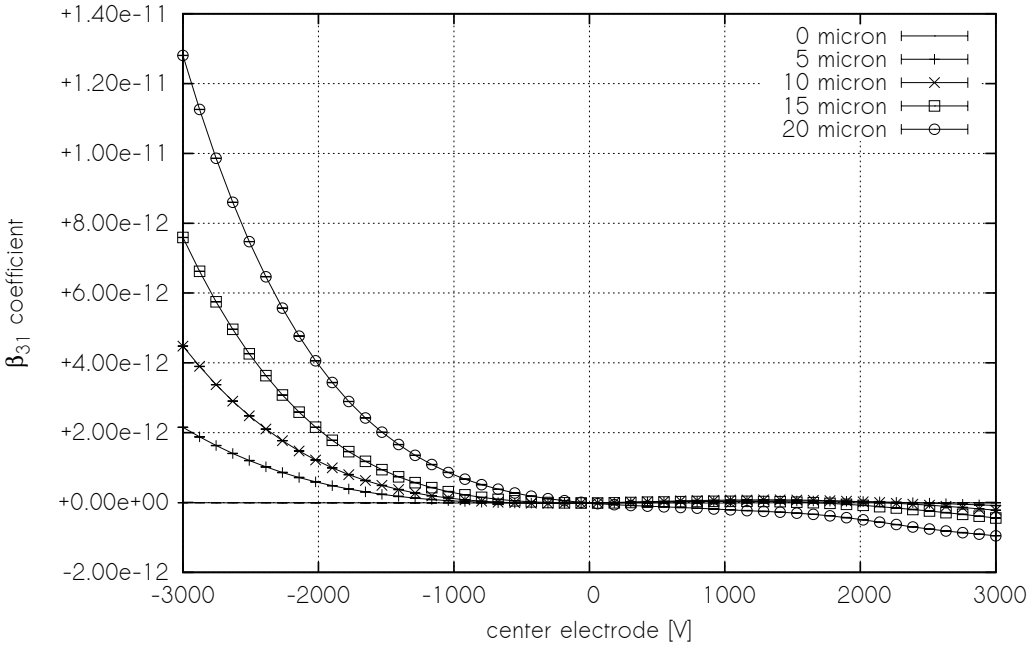

Figure 58 – Graphical representation of the fourth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^3y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

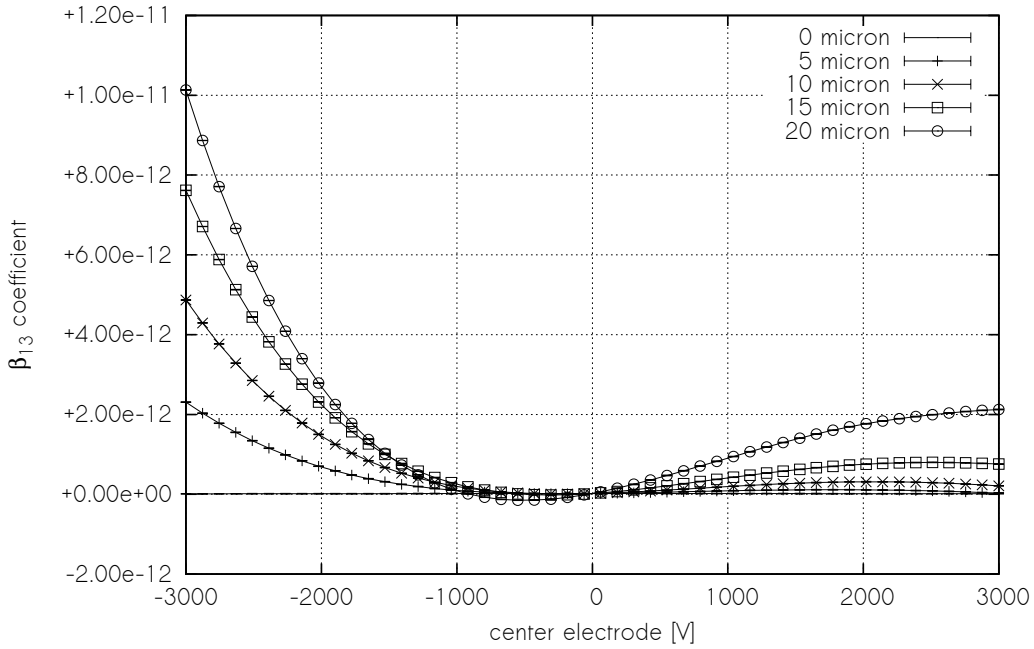

Figure 59 – Graphical representation of the fourth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^1y^3-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis. Note that the error bars in this figure are too small to see.

Fifth-order

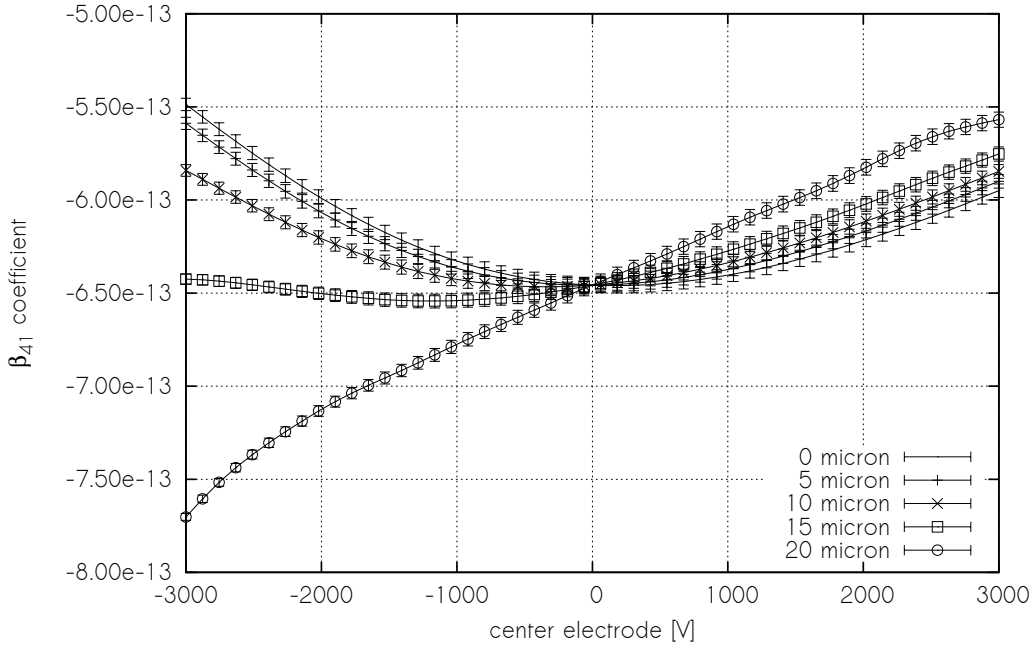

Figure 60 – Graphical representation of the fifth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^4y^1-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis.

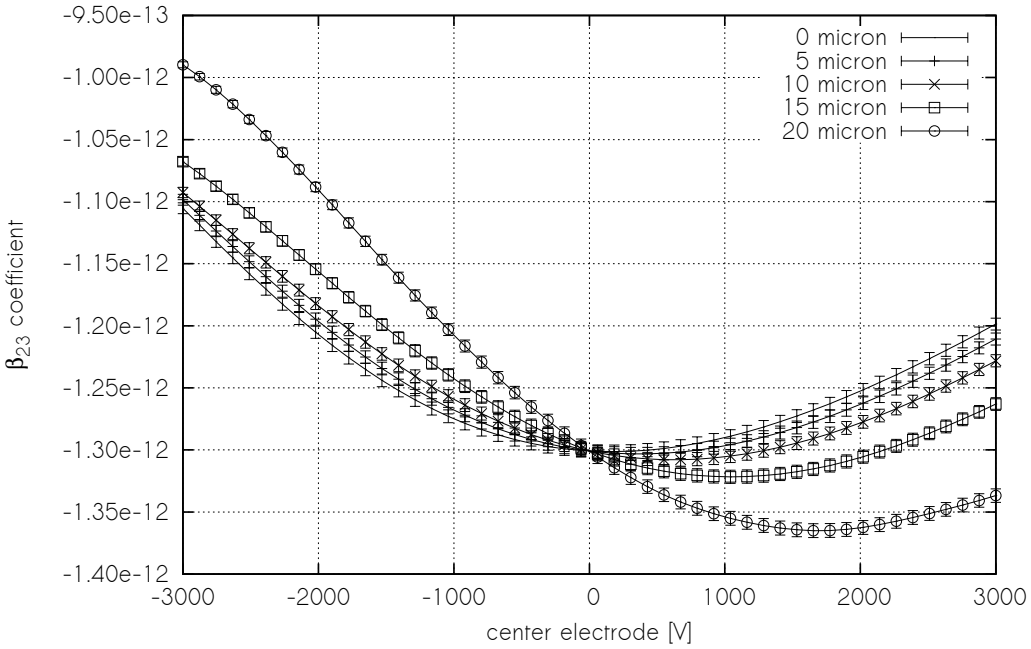

Figure 61 – Graphical representation of the fifth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^2y^3-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis.

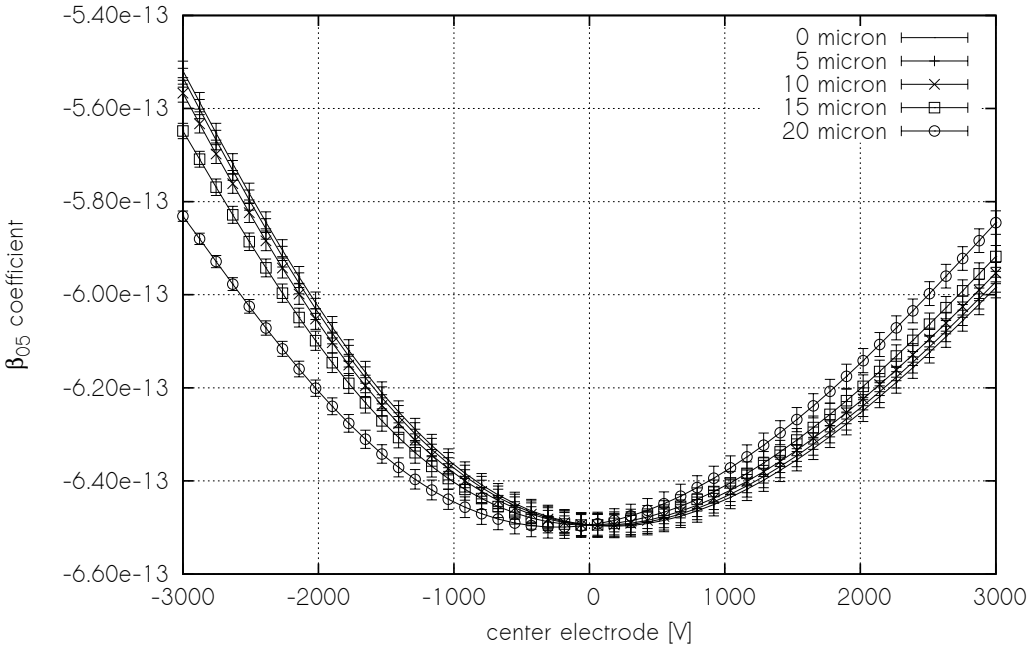

Figure 62 – Graphical representation of the fifth order coefficient obtained from simulating the shift lens of figure 23. This figure represents the x^0y^5-term in the Taylor expansion of the deflection angle given by equation (89). This figure illustrates how the coefficient changes when the displacement and potential of the center electrode is varied. Each seperate curve represents a displacement (0, 5, 10, 15 or 20 micron) of the center electrode. The horizontal axis gives the potential difference of the center electrode with respect to the neighbor electrodes. The actual value of the coefficient is given by the vertical axis.

Bibliography

[1] L. a. Baranova and F. H. Read, "Electrostatic electron-optical crossed lens with controlled astigmatism," *Review of Scientific Instruments*, vol. 65, no. 6, p. 1993, 1994.

[2] P. W. Hawkes and E. Kasper, *Principles of Electron Optics Series*. Academic Press, 1996.

[3] A. C. Zonnevylle, C. T. H. Heerkens, P. Kruit, M. L. Wieland, F. M. Postma, and S. W. K. H. Steenbrink, "Electrostatic rotator for alignment purposes in multi electron beam systems," *Microelectronic Engineering*, vol. 87, pp. 1095–1099, May 2010.

[4] E. Slot and M. Wieland, "MAPPER: high throughput maskless lithography," *... Lithography*, vol. 6921, pp. 69211P–69211P–9, Mar. 2008.

[5] R. F. Whitmer, "Investigation of Nonrotationally Symmetrical Electrostatic Electron Optical Lenses," *Journal of Applied Physics*, vol. 27, no. 7, p. 808, 1956.

[6] J. Zlámal and B. Lencová, "Development of the program EOD for design in electron and ion microscopy," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 645, pp. 278–282, July 2011.

[7] D. A. Dahl, J. E. Delmore, and A. D. Appelhans, "SIMION PC/PS2 electrostatic lens design program," *Review of Scientific Instruments*, vol. 61, no. 1, p. 607, 1990.

[8] E. Munro, "Finite difference programs for computing tolerances for electrostatic lenses," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 6, p. 941, May 1988.

[9] M. de Loos and S. van der Geer, "General Particle Tracer: A new 3D code for accelerator and beamline design," in *5th European Particle Accelerator Conference*, p. 1241, 1996.

[10] P. Sturrock, "The aberrations of magnetic electron lenses due to asymmetries," *Philosophical Transactions of the Royal Society of London*, vol. 243, no. 868, pp. 387–429, 1951.

[11] B. Cook, T. Verduin, C. Hagen, and P. Kruit, "Brightness limitations of cold field emitters caused by Coulomb interactions," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 28, no. 6, p. C6C74, 2010.

[12] T. Verduin, B. Cook, and P. Kruit, "Influence of gun design on Coulomb interactions in a field emission gun," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 29, no. 6, p. 06F605, 2011.

[13] J. Orloff, *Handbook of Charged Particle Optics*. CRC Press / Taylor & Francis, 2 ed., 2009.

[14] J. Barnes and P. Hut, "A hierarchical O(N log N) force-calculation algorithm," *Nature*, vol. 324, pp. 446–449, Dec. 1986.

[15] J. E. Barnes, "A modified tree code: Don't laugh; It runs," *Journal of Computational Physics*, vol. 87, pp. 161–170, Mar. 1990.

[16] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *Journal of Computational Physics*, vol. 73, pp. 325–348, Dec. 1987.

[17] D. Cubric, B. Lencova, F. Read, and J. Zlamal, "Comparison of FDM, FEM and BEM for electrostatic charged particle optics," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 427, pp. 357–362, May 1999.

[18] D. Donnelly and E. Rogers, "Symplectic integrators: An introduction," *American Journal of Physics*, vol. 73, no. 10, p. 938, 2005.

[19] M. Garland, S. Le Grand, J. Nickolls, J. Anderson, J. Hardwick, S. Morton, E. Phillips, Y. Zhang, and V. Volkov, "Parallel Computing Experiences with CUDA," *IEEE Micro*, vol. 28, pp. 13–27, July 2008.

[20] S. Blanes and P. Moan, "Practical symplectic partitioned RungeKutta and RungeKuttaNyström methods," *Journal of Computational and Applied Mathematics*, vol. 142, pp. 313–330, May 2002.

[21] H. Goldstein, C. P. Poole, and J. L. Safko, *Classical mechanics*. Addison-Wesley Longman, 3e ed., 2002.

[22] J. D. Jackson, *Classical Electrodynamics*. Wiley, 2 ed., 2010.

[23] L. Landau and E. Lifshitz, *Mechanics*. Elsevier, 3 ed., 2005.

[24] H. H. Rose, *Geometrical Charged-Particle Optics*. Springer, 2009.

[25] H. Hofer and E. Zehnder, *Symplectic Invariants and Hamiltonian Dynamics*. Springer Verlag, 1994.

[26] E. Forest and R. D. Ruth, "Fourth-order symplectic integration," *Physica D: Nonlinear Phenomena*, vol. 43, pp. 105–117, May 1990.

[27] H. Yoshida, "Construction of higher order symplectic integrators," *Physics Letters A*, vol. 150, no. 5, pp. 262–268, 1990.

[28] I. P. Omelyan, I. M. Mryglod, and R. Folk, "New optimized algorithms for molecular dynamics simulations," *Condensed Matter Physics*, vol. 5, no. 3, p. 369, 2002.

[29] G. M. Nielson, "The side-vertex method for interpolation in triangles," *Journal of Approximation Theory*, vol. 25, pp. 318–336, Apr. 1979.

[30] R. Clint Whaley, A. Petitet, and J. J. Dongarra, "Automated empirical optimizations of software and the ATLAS project," *Parallel Computing*, vol. 27, pp. 3–35, Jan. 2001.

[31] D. Goldberg, "What every computer scientist should know about floating-point arithmetic," *ACM Computing Surveys*, vol. 23, pp. 5–48, Mar. 1991.

[32] E. Babolian, M. MasjedJamei, and M. Eslahchi, "On numerical improvement of GaussLegendre quadrature rules," *Applied Mathematics and Computation*, vol. 160, pp. 779–789, Jan. 2005.

[33] D. P. Laurie, "Calculation of Gauss-Kronrod quadrature rules," *Mathematics of Computation*, vol. 66, pp. 1133–1146, July 1997.

[34] D. Luebke, M. Harris, N. Govindaraju, A. Lefohn, M. Houston, J. Owens, M. Segal, M. Papakipos, and I. Buck, "A Survey of GeneralPurpose Computation on Graphics Hardware," in *Proceedings of the 2006 ACM/IEEE conference on Supercomputing - SC '06*, (New York, New York, USA), p. 208, ACM Press, 2006.

[35] J. W. Davidson and S. Jinturkar, "Memory access coalescing," *ACM SIGPLAN Notices*, vol. 29, pp. 186–195, June 1994.

[36] E. Hecht, *Optics*. Addison-Wesley Longman, 4 ed., 2002.

[37] L. Reimer, *Scanning Electron Microscopy: Physics of Image Formation and Microanalysis*. Springer, 2 ed., 1998.

[38] B. Gladman, M. Duncan, and J. Candy, "Symplectic integrators for long-term integrations in celestial mechanics," *Celestial Mechanics and Dynamical Astronomy*, vol. 52, no. 3, pp. 221–240, 1991.

[39] M. P. Calvo and J. M. Sanz-Serna, "The Development of Variable-Step Symplectic Integrators, with Application to the Two-Body Problem," *SIAM Journal on Scientific Computing*, vol. 14, pp. 936–952, July 1993.

[40] R. Skeel and J. Biesiadecki, "Symplectic integration with variable stepsize," *Annals of Numerical Mathematics*, vol. 1, no. 1-4, pp. 191–198, 1994.

[41] M. H. Lee, M. J. Duncan, and H. F. Levison, "Variable Timestep Integrators for Long-Term Orbital Integrations," *arXiv preprint astro-ph/9703117*, p. 7, Mar. 1997.

[42] D. J. Hardy, D. I. Okunbor, and R. D. Skeel, "Symplectic variable step size integration for N-body problems," *Applied Numerical Mathematics*, vol. 29, pp. 19–30, Jan. 1999.

[43] P. Hut, J. Makino, and S. McMillan, "Building a better leapfrog," *The Astrophysical Journal*, vol. 443, p. L93, Apr. 1995.

[44] D. Stoffer, "Variable steps for reversible integration methods," *Computing*, vol. 55, pp. 1–22, Mar. 1995.

[45] E. Hairer and D. Stoffer, "Reversible Long-Term Integration with Variable Stepsizes," *SIAM Journal on Scientific Computing*, vol. 18, pp. 257–269, Jan. 1997.

[46] E. Hairer and G. Söderlind, "Explicit, Time Reversible, Adaptive Step Size Control," *SIAM Journal on Scientific Computing*, vol. 26, pp. 1838–1851, Jan. 2005.

[47] Y. Funato, P. Hut, S. McMillan, and J. Makino, "Time-Symmetrized Kustaanheimo-Stiefel Regularization," *The Astronomical Journal*, vol. 112, p. 1697, Oct. 1996.

[48] W. Huang and B. Leimkuhler, "The Adaptive Verlet Method," *SIAM Journal on Scientific Computing*, vol. 18, pp. 239–256, Jan. 1997.

[49] B. Leimkuhler, "Reversible adaptive regularization: perturbed Kepler motion and classical atomic trajectories," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 357, pp. 1101–1133, Apr. 1999.

[50] E. Barth, B. Leimkuhler, and S. Reich, "A Time-Reversible Variable-Stepsize Integrator for Constrained Dynamics," *SIAM Journal on Scientific Computing*, vol. 21, pp. 1027–1044, Jan. 1999.

[51] S. Blanes and C. J. Budd, "Explicit, adaptive, symplectic (EASY) integrators using scale invariant regularisations and canonical transformations," *Celestial Mechanics and Dynamical Astronomy*, vol. 89, no. 4, pp. 383–405, 2004.

[52] M. Calvo, M. López-Marcos, and J. Sanz-Serna, "Variable step implementation of geometric integrators," *Applied Numerical Mathematics*, vol. 28, pp. 1–16, Sept. 1998.

[53] S. Reich, "Backward Error Analysis for Numerical Integrators," *SIAM Journal on Numerical Analysis*, vol. 36, pp. 1549–1570, Jan. 1999.

[54] L. P. Chew, "Guaranteed-quality mesh generation for curved surfaces," in *Proceedings of the ninth annual symposium on Computational geometry - SCG '93*, (New York, New York, USA), pp. 274–280, ACM Press, 1993.

[55] J. Ruppert, "A Delaunay Refinement Algorithm for Quality 2-Dimensional Mesh Generation," *Journal of Algorithms*, vol. 18, pp. 548–585, May 1995.