Forschungszentrum Karlsruhe
Technik und Umwelt

Wissenschaftliche Berichte
FZKA 6115

A Maxwell-Lorentz Solver for Self-Consistent Particle-Field Simulations on Unstructured Grids

M. Fedoruk, C.-D. Munz, P. Omnes, R. Schneider

Institut für Neutronenphysik und Reaktortechnik

Juli 1998

Forschungszentrum Karlsruhe

Technik und Umwelt

Wissenschaftliche Berichte

FZKA 6115

A Maxwell-Lorentz Solver For Self-Consistent Particle-Field Simulations On Unstructured Grids

M. Fedoruk[1], C.-D. Munz[2], P. Omnes, R. Schneider

Institut für Neutronenphysik und Reaktortechnik

[1]Institute of Computational Technolgies, Lavrentjev ave.6, and Novosibirsk State University, Pirogov st. 2, Novosibirsk 630090, Russia

[2]Institut für Aerodynamik und Gasdynamik der Universität Stuttgart, Pfaffenwaldring 21, D-70550 Stuttgart

Forschungszentrum Karlsruhe GmbH, Karlsruhe
1998

Als Manuskript gedruckt

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

Mitglied der Hermann von Helmholtz-Gemeinschaft
Deutscher Forschungszentren (HGF)

ISSN 0947-8620

Abstract

The conceptual and algorithmic framework solving numerically the time-dependent Maxwell-Lorentz equations on an unstructured mesh in two and three space dimensions is presented. Beyond a brief review of the applied charged particle handling based on advanced particle-in-cell techniques, a modern finite-volume scheme for the numerical approximation of the three-dimensional, time-dependent Maxwell equations is introduced using unstructured grid arrangements. Furthermore, the algorithmic realization of the resulting numerical schemes is described in great detail. Apart from this, simulation results for typical benchmark problems computed with the particle treatment and Maxwell solver are presented, demonstrating the quality and properties as well as the relevance and reliability of the applied numerical methods.

EIN MAXWELL-LORENTZ LÖSER ZUR SELBSTKONSISTENTEN TEILCHEN-FELD SIMULATION AUF UNSTRUKTURIERTEN RECHENGITTERN

Überblick

Das Konzept und der algorithmische Rahmen zur numerischen Behandlung der zeitabhängigen Maxwell-Lorentzgleichungen auf unstrukturierten Rechennetzen in zwei und drei Raumdimensionen wird vorgestellt. Die wesentlichen Ideen der auf unstrukturierte Gitter erweiterten Methoden zur Teilchenbehandlung, die auf fortgeschrittenen Particle-in-Cell Techniken beruhen, werden in knapper Form beschrieben. Weiterhin wird ein moderner Zugang zur numerischen Approximation der dreidimensionalen, zeitabhängigen Maxwellgleichungen eingeführt, der in einem zeitgemäßen, hochauflösenden Finite-Volumen Verfahren mündet. Darüber hinaus werden die Kernstücke der algorithmischen Umsetzung der abgeleiteten numerischen Schemata ausführlich erläutert und die vom Gittererzeugungsmodul bestimmte Datenstruktur zusammenfassend dargestellt. Simulationsergebnisse für typische Testprobleme, die mit den Teilchenbehandlungsmodulen und mit dem Maxwell-Löser erzielt wurden, runden den vorliegenden Bericht ab, und geben deutliche Auskunft über die Eigenschaft, Güte, Relevanz und Verläßlichkeit der benutzten numerischen Methoden.

Contents

1 Introduction **3**

2 Numerical Schemes and Approximation Methods **5**
2.1 Computational Grid 6
2.2 A Finite-Volume Method for the Maxwell Equations in the Time Domain on Unstructured Grids 8
2.2.1 Numerical Scheme 9
2.2.2 Calculation of the Numerical Flux 11
2.2.3 Solution of the Riemann Problem 12
2.2.4 Computation of the Gradients 14
2.2.5 Boundary Conditions 15
2.3 Particle Treatment 18
2.3.1 Particle Pushing 18
2.3.2 Particle Localization 19
2.3.3 Particle Assignment and Interpolation 23
2.3.4 Particle Handling at the Border of the Computational Domain 24

3 Numerical Results **29**
3.1 Two and Three-Dimensional Electromagnetic Test Problems 29
3.1.1 Two-dimensional Test Examples 29
3.1.2 Cylinder Symmetrical Test Problem 33
3.1.3 Coaxial Wave Guide in Three Dimensions 34
3.2 Two and Three-Dimensional Benchmark Problems for Particle Treatment 38
3.2.1 Localization and Assignment 38
3.2.2 Localization and Interpolation 41
3.2.3 Interpolation, Particle Pushing and Localization 42

4 Algorithms and Data Structure **49**
4.1 Preparatory Step 51
4.2 Time Loop 51
4.3 Post-Processing 52
4.4 Details of the Maxwell Solver 52

4.4.1 Computation of the Sources 52
4.4.2 Computation of the Gradients 53
4.4.3 Field Values at a Common Border of Two Elements 54
4.4.4 Riemann Problem 55
4.4.5 Homogeneous Equations 55
4.4.6 Final Values of the Fields 56
4.5 Grid Informations and Structure of the Data 56

5 Conclusional Remarks and Outlook **59**

A Miscellaneous Properties of the Matrix used for the FV Approach **65**

B Formulas based on the Solution of the Riemann Problem **68**

C Numerical Scheme for the Maxwell Equations in Cylinder Symmetrical Geometry **71**

D Two-dimensional Particle Localization with the Assous Approach **78**

E Example of a Command File **80**

Chapter 1

Introduction

The goal of the present technical report is to provide the user of the Maxwell-Lorentz simulation program KADI$_3^2$D developed at the Institut für Neutronenphysik und Reaktortechnik (INR) with more detailed informations about the numerical methods, the approximation techniques and the implementation framework used in the program. The proposed Finite-Volume Particle-in-Cell (FV-PIC) Maxwell-Lorentz solver calculates the time-dependent solution of the Maxwell-Lorentz equations in two and three space dimensions on unstructured meshes, respectively, and is up to second-order accurate in both space and time.

The considered Maxwell-Lorentz model is constituted by the Maxwell equations in the vacuum and the usual laws of classical mechanics known as the Lorentz equations [13, 3, 22]. The evolution of the electromagnetic fields inside a domain Ω is given by the full set of the Maxwell equations in the vaccum

$$\partial_t \boldsymbol{E} - c^2 \nabla_x \times \boldsymbol{B} = -\frac{\boldsymbol{j}}{\epsilon_0}\,, \tag{1.1a}$$

$$\partial_t \boldsymbol{B} + \nabla_x \times \boldsymbol{E} = 0\,, \tag{1.1b}$$

$$\nabla_x \cdot \boldsymbol{E} = \frac{\rho}{\epsilon_0}\,, \tag{1.1c}$$

$$\nabla_x \cdot \boldsymbol{B} = 0\,, \tag{1.1d}$$

where $\boldsymbol{E}, \boldsymbol{B}, \rho$ and $\boldsymbol{j}$ respectively denote the electric field, the magnetic induction, the charge density and the current density. The electric permittivity ϵ_0 and magnetic permeability μ_0 of the vacuum are related to the speed of light according to $\epsilon_0 \mu_0 c^2 = 1$.

The dynamics of the charged macro particle distribution inside the computational domain is determined according to the Lorentz equations

$$\dot{\boldsymbol{x}}_k(t) = \boldsymbol{v}_k(t)\,, \tag{1.2a}$$

$$\dot{\boldsymbol{p}}_k(t) = \boldsymbol{F}(\boldsymbol{x}_k, \boldsymbol{v}_k, t)\,, \tag{1.2b}$$

where the particle index k runs over the total number N_p of charges. The Lorentz

force

$$\boldsymbol{F}(\boldsymbol{x}_k, \boldsymbol{v}_k, t) = Q_k \left[\boldsymbol{E}(\boldsymbol{x}_k(t), t) + \boldsymbol{v}_k(t) \times \boldsymbol{B}(\boldsymbol{x}_k(t), t)\right] \tag{1.2c}$$

on the charge $Q_k = N_k Q$ with the mass $M_k = N_k M$ depends on the electromagnetic fields $\boldsymbol{E}$ and $\boldsymbol{B}$ at the actual position $\boldsymbol{x}_k$ and on the velocity $\boldsymbol{v}_k$ of the kth macro particle, calculated from the momentum $\boldsymbol{p}_k$ according to $\boldsymbol{v}_k(t) = \frac{\boldsymbol{p}_k(t)}{M_k \gamma_k(t)}$ with $\gamma_k^2 = 1 + \left(\frac{\boldsymbol{p}_k}{M_k c}\right)^2$. Although M_k and Q_k depend on the number N_k of elementary constituents of a macro particle, it is noteworthy, that this number cancels out in (1.2c) and, consequently, the motion of a macro charge is determined by Q/M, the ratio of charge and mass of a single constituent.

The interaction of the charged particle distribution with the electromagnetic fields inside Ω is computed in a self-consistent manner: The charge and current density

$$\rho(\boldsymbol{x}, t) = \sum_{k=1}^{N_p} Q_k \delta\left[\boldsymbol{x} - \boldsymbol{x}_k(t)\right] , \tag{1.3a}$$

$$\boldsymbol{j}(\boldsymbol{x}, t) = \sum_{k=1}^{N_p} Q_k \boldsymbol{v}_k(\boldsymbol{p}_k) \delta\left[\boldsymbol{x} - \boldsymbol{x}_k(t)\right] , \tag{1.3b}$$

are obtained from the phase space coordinates ($\boldsymbol{x}_k$, $\boldsymbol{v}_k$) of the entire ensemble of the macro charges. The electromagnetic fields calculated from the sources (1.3) redistribute the charged particles within the domain Ω via the Lorentz force (1.2c), yielding changed phase space coordinates, and from these, the new densities are obtained. This complex interplay between fields and particles described by (1.1)-(1.3) is known as the non-linear Maxwell-Lorentz problem, the starting point for further numerical approximations.

The organization of the present report is as follows: In Chapter 2, some remarks concerning the unstructured computational grids usually used as test meshes for the code development are first given. Afterwards, we briefly recall the basic concepts and approximation techniques to come in useful for the numerical solution of the Maxwell-Lorentz system. In Chapter 3, numerical results for some typical test problems are presented, demonstrating the quality and property as well as the relevance and reliability of the applied numerical methods. In Chapter 4, an overview of the implemented principal algorithms and resulting subroutines is given and the determinative data structure needed for the realization of these algorithms is sketched out in more detail. Finally, conclusive remarks and a short outlook of the further activities are made in Chapter 5.

Chapter 2

Numerical Schemes and Approximation Methods

Traditional techniques for solving the Maxwell equations in the time domain rely on finite-difference (FD) methods [32]. Staggered grid FD schemes are used in different electromagnetic PIC simulation codes which are successfully applied to a multitude of investigations relevant for the understanding of electrical and plasma devices (see, e.g., [26, 27]). The FD approach is based on a Cartesian grid and may be extended to a structured boundary-fitted mesh consisting of quadrilateral grid zones by applying the FD scheme to the transformed equations [14]. To get more flexibility, especially, for the simulation of complexer geometries, in the present report, a FV approach for the Maxwell equations is proposed. This method based on high-resolution schemes originally developed for hyperbolic equations combines robustness at steep gradients with accurate resolution [17]. The coupling of the high-resolution FV Maxwell solver with the PIC method is a new way of approximation in the context of self-consistent particle simulation in electromagnetic fields [22]. The numerical concept in its entirety forming the basis of the FV-PIC approach is schematically depicted in Figure 2.1.

At each time step, the electromagnetic fields obtained from the numerical solution of the Maxwell equations on the computational mesh are interpolated to the actual locations of the charged particles (Interpolation). According to the Lorentz force the charges are redistributed and the new phase space coordinates are computed by solving numerically the usual laws of dynamics known as the Lorentz equations (Particle Pushing). To close the chain of self-consistent interplay between particles and fields, the particles have to be located with respect to the computational mesh (Localization) in order to determine their contributions to the changed charge and current densities (Assignment). These updated densities are then the sources for the Maxwell equations in the subsequent iteration cycle. For more detailed informations concerning the FV-PIC approach we refer to [6, 22, 23, 24].

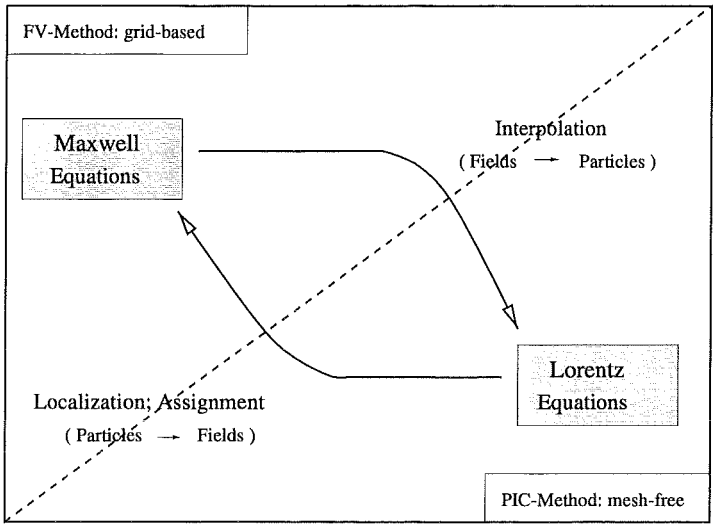

Figure 2.1: Particle-in-Cell iteration cycle.

2.1 Computational Grid

The necessity of a computational mesh is founded on the nature of the PIC approach itself [3, 13, 22]. In essence, this approach circumvents the direct force calculation between charged particles by introducing a grid-based and a mesh-free numerical model: On the computational grid the spatial and temporal evolution of the electromagnetic fields generated by all charges is determined, whereas the charged particles themselves are advanced in the continuous computational domain.

When solving numerically the time-dependent Maxwell problem, it is very important to possess an adequate computational mesh, which covers the geometry under consideration very properly. Especially, high quality simulations of electrical devices require an appropriate replica of the border of the device, where several kinds of physically and computationally motivated conditions can be applied. For our computational endeavor solving numerically the Maxwell-Lorentz problem in two and three space dimensions, we choose the most flexible concept, namely, unstructured meshing techniques, which possess the property of the highest degree of freedom in mapping the relevant geometry to the discrete image. Furthermore, grid generators based on triangulization (2D) and tetrahedrization (3D) of the domain are widespread and therefore usually (commercially) available.

Throughout the course of code development and validation, we usually use a simple unit square for the two and a unit cube for the three-dimensional case as computational domains. To discretize these domains, we apply the mesh generator Modulef (see, for example, [10]). As an example, a typical unstructured mesh

used for two-dimensional test calculations is depicted in Figure 2.2.

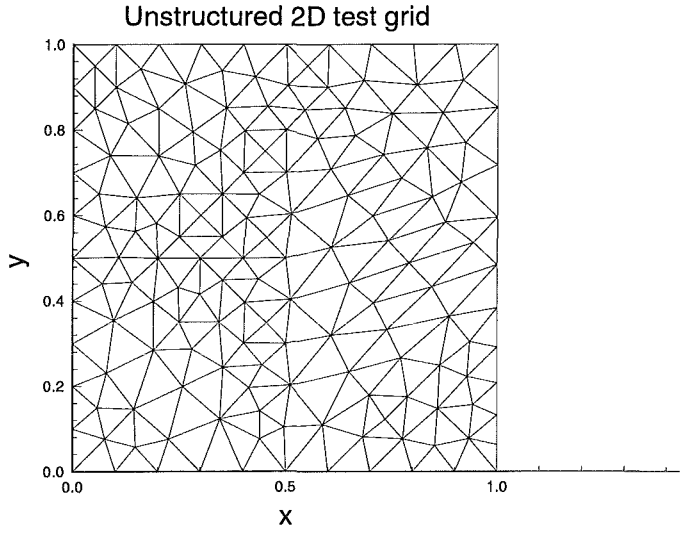

Figure 2.2: The standard two-dimensional computational domain covered by an unstructured mesh.

2.2 A Finite-Volume Method for the Maxwell Equations in the Time Domain on Unstructured Grids

In this part of the report, we consider the vacuum Maxwell equations for Cartesian $x = (x_1, x_2, x_3) = (x, y, z)$ coordinates. The sometimes important cylinder symmetrical case $x = (x_1, x_2, x_3) = (z, r, \theta)$ is treated for completeness in Appendix C.

Different forms of the Maxwell equations are usually used in computational electromagnetics (CEM) and are reviewed, for instance, in [4, 29, 23] and thus will not be repeated here. The relevant formulation for the construction of FV methods is the conservation form of the Maxwell equations:

$$\frac{\partial u}{\partial t} + \sum_{i=1}^{3} \frac{\partial f_i(u)}{\partial x_i} = q, \tag{2.1}$$

which is based on the time-dependent equations (1.1a-b) only. Here, the vector u of the electromagnetic quantities is composed by the electrical field E and the magnetic induction B and reads as

$$u(x,t) = (E,B)^T = (E_1, E_2, E_3, B_1, B_2, B_3)^T \,. \tag{2.2}$$

The physical fluxes f_i are given by

$$f_i(u) = \mathcal{K}_i u \;; \quad \text{for } i = 1,2,3 \,, \tag{2.3a}$$

where the block-structured matrices $\mathcal{K}_i \in \mathbb{R}^{6\times 6}$ are defined according to

$$\mathcal{K}_i = \begin{pmatrix} 0 & -c^2\mathcal{M}_i \\ \mathcal{M}_i & 0 \end{pmatrix} \,, \tag{2.3b}$$

with

$$\mathcal{M}_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} , \mathcal{M}_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} , \mathcal{M}_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} . \tag{2.3c}$$

The source term q of the time-dependent conservation equation (2.1) is independent of u and may be written as

$$q = -\frac{1}{\epsilon_0}(j_1, j_2, j_3, 0, 0, 0)^T \,, \tag{2.4}$$

where $j = (j_1, j_2, j_3)^T$ is the current density.

2.2.1 Numerical Scheme

The domain of computation Ω is partitioned into N non-overlapping cells denoted by C_i: $\Omega = \bigcup_{i=1}^{N} C_i$. The actual time level is computed according to $t^n = n\Delta t$, where Δt is determined with respect to the CFL condition. The average value over the cell C_i at time $t = t^n$ of any integrable function $h(x,t)$ is denoted in the following by h_i^n. Explicitly, this cell average is calculated from

$$h_i^n = \frac{1}{V_i} \int_{C_i} h(x, t^n)\, dV \ , \tag{2.5}$$

where V_i is the volume of C_i.

In order to solve (2.1), we apply a splitting ansatz, which consists in computing the following sequence of equations: First, we solve

$$\frac{\partial u^{(1)}}{\partial t} = q \ , \tag{2.6a}$$

with initial value $u^{(1)}(t^n) = u^n$. Then the solution of the homogeneous conservation equation

$$\frac{\partial u^{(2)}}{\partial t} + \sum_{i=1}^{3} \frac{\partial f_i(u^{(2)})}{\partial x_i} = 0 \tag{2.6b}$$

is determined for the full time step size Δt with initial the initial data $u^{(2)}(t^n) = u^{(1)}\left(t^n + \frac{\Delta t}{2}\right)$. In the next step the ordinary differential equation

$$\frac{\partial u^{(3)}}{\partial t} = q \ , \tag{2.6c}$$

is solved once again but now with initial value $u^{(3)}\left(t^n + \dfrac{\Delta t}{2}\right) = u^{(2)}(t^{n+1})$. By setting

$$u^{n+1} = u^{(3)}(t^{n+1}) \ , \tag{2.6d}$$

the value of u at the new time level t^{n+1} is finally obtained. The integration of the ordinary equation (2.6a) over the space-time volume $C_i \times [t^n, t^n + \dfrac{\Delta t}{2}]$ yields the exact equation

$$V_i \left[u_i^{(1)}\left(t^n + \frac{\Delta t}{2}\right) - u_i^{(1)}(t^n) \right] = \int_{t^n}^{t^n + \frac{\Delta t}{2}} \int_{C_i} q\, dV\, dt \ ,$$

which will be approximated according to

$$u_i^{(2)}(t^n) = u_i^{(1)}\left(t^n + \frac{\Delta t}{2}\right) = u_i^n + \frac{\Delta t}{2V_i}\int\limits_{C_i} q(x, t^n + \frac{\Delta t}{2})dV \ . \tag{2.7a}$$

In the same way, the approximation of equation (2.6c) can be performed, yielding

$$u_i^{n+1} = u_i^{(3)}(t^{n+1}) = u_i^{(2)}(t^{n+1}) + \frac{\Delta t}{2V_i}\int\limits_{C_i} q(x, t^n + \frac{\Delta t}{2})dV \ . \tag{2.7b}$$

Now, we consider the integration of the homogeneous conservation equation (2.6b) over the space-time element $C_i \times [t^n, t^{n+1}]$. Applying Gauß's theorem we obtain the exact evolution equation

$$V_i\left[(u^{(2)})_i^{n+1} - (u^{(2)})_i^n\right] = -\sum_{\alpha=1}^{\sigma_i}\int\limits_{t^n}^{t^{n+1}}\int\limits_{S_{i,\alpha}}\left(\sum_{j=1}^{3}(n_j)_{i,\alpha}\, f_j(u^{(2)})\right) dS\, dt \ ,$$

where $S_{i,\alpha}$ is the face α of C_i and σ_i denotes the total number of faces of C_i. Furthermore, $(n_j)_{i,\alpha}$ is the j^{th} component of the outwards directed unit normal at the face $S_{i,\alpha}$. The direct approximation of this integral formulation yields the explicit FV scheme, usually written in the form

$$(u^{(2)})_i^{n+1} = (u^{(2)})_i^n - \frac{\Delta t}{V_i}\sum_{\alpha=1}^{\sigma_i} G_{i,\alpha}^{n+1/2} \ . \tag{2.7c}$$

Clearly, the FV scheme is completely defined if the numerical flux $G_{i,\alpha}^{n+1/2}$ is specified. The numerical flux itself is a suitable approximation of the physical flux through the boundary face $S_{i,\alpha}$, which means:

$$G_{i,\alpha}^{n+1/2} \approx \frac{1}{\Delta t}\int\limits_{t^n}^{t^{n+1}}\int\limits_{S_{i,\alpha}} \mathcal{A}_{i,\alpha}\, u^{(2)}(x,t)\, dS\, dt \ , \tag{2.8a}$$

where the matrix $\mathcal{A}_{i,\alpha} \in \mathbb{R}^{6\times 6}$ is given by

$$\mathcal{A}_{i,\alpha} = \sum_{j=1}^{3}(n_j)_{i,\alpha}\mathcal{K}_j \ , \tag{2.8b}$$

a linear combination of the constant matrices $\mathcal{K}_j$ defined by (2.3b-c). The determination of this flux as a function of the averaged quantities $(u^{(2)})_i^n$ is the item of the next Section.

2.2.2 Calculation of the Numerical Flux

In this section we outline the path of approximations to obtain the numerical flux $G_{i,\alpha}^{n+1/2}$. For that purpose, we apply the second-order accurate midpoint rule to the integrals (2.8a), yielding the first approximation

$$G_{i,\alpha}^{n+1/2} \approx L_{i,\alpha}\, \mathcal{A}_{i,\alpha}\, u(M_{i,\alpha}, t^{n+\frac{1}{2}}) \,, \tag{2.9}$$

where $M_{i,\alpha}$ and $L_{i,\alpha}$ denote the midpoint and area of the face $S_{i,\alpha}$, respectively (here and in the following we drop the superscript $^{(2)}$ for readability reasons). Obviously, the central point to establish the numerical scheme is the computation of a suitable estimation of $u(M_{i,\alpha}, t^{n+\frac{1}{2}})$. For that, the typical space increment is denoted by Δx and, furthermore, it is supposed that a first-order accurate approximation of the gradient of the k^{th} component of u at the barycenter B_i of C_i is known at time t^n:

$$(s_i^n)_{kj} \approx \frac{\partial u_k}{\partial x_j}(B_i, t^n) \;\text{ with } 1 \le k \le 6\,,\; 1 \le j \le 3\,. \tag{2.10}$$

A truncated Taylor expansion with respect to t yields the sought function u_k at $M_{i,\alpha}$ and $t^{n+1/2}$

$$u_k(M_{i,\alpha}, t^{n+\frac{1}{2}}) = u_k(M_{i,\alpha}, t^n) + \frac{\Delta t}{2}\frac{\partial u_k}{\partial t}(M_{i,\alpha}, t^n) + O(\Delta t^2)\,. \tag{2.11a}$$

Here, the approximation of the time derivative can be replaced by the approximation of the space derivatives of u_k as given by equation (2.6b):

$$\frac{\partial u_k}{\partial t}(M_{i,\alpha}, t^n) = -\sum_{j=1}^{3}\left[\mathcal{K}_j \frac{\partial u}{\partial x_j}(M_{i,\alpha}, t^n)\right]_k \,. \tag{2.11b}$$

Since the quatities u_k are associated per definition with the barycenter B_i, a further Taylor expansion, now, with respect to x has to be performed, yielding

$$u_k(M_{i,\alpha}, t^n) = u_k(B_i, t^n) + \sum_{j=1}^{3}(\overrightarrow{B_i M_{i,\alpha}})_j \frac{\partial u_k}{\partial x_j}(B_i, t^n) + O(\Delta x^2)\,, \tag{2.11c}$$

where $(\overrightarrow{B_i M_{i,\alpha}})_j$ is the j^{th} component of the distance vector $\overrightarrow{B_i M_{i,\alpha}}$ seen in Figure 2.3 for the two-dimensional analogue of the considered three-dimensional case. With the equalities

$$[u_k]_i^n = u_k(B_i, t^n) + O(\Delta x^2)\,, \tag{2.11d}$$

(see Appendix C) and

$$\frac{\partial u_k}{\partial x_j}(M_{i,\alpha}, t^n) = (s_i^n)_{kj} + O(\Delta x)\,, \tag{2.11e}$$

we finally get an approximation $[u_k]_{i,\alpha}^{n+}$ of the k^{th} component of $u(M_{i,\alpha}, t^{n+\frac{1}{2}})$

$$[u_k]_{i,\alpha}^{n+} = [u_k]_i^n + \sum_{j=1}^{3} (\overrightarrow{B_i M_{i,\alpha}})_j \, (s_i^n)_{kj} - \frac{\Delta t}{2} \sum_{j=1}^{3} [\mathcal{K}_j \, (s_i^n)_j]_k \ , \tag{2.12a}$$

which is second-order accurate in both time and space. Due to the fact, that $S_{i,\alpha}$

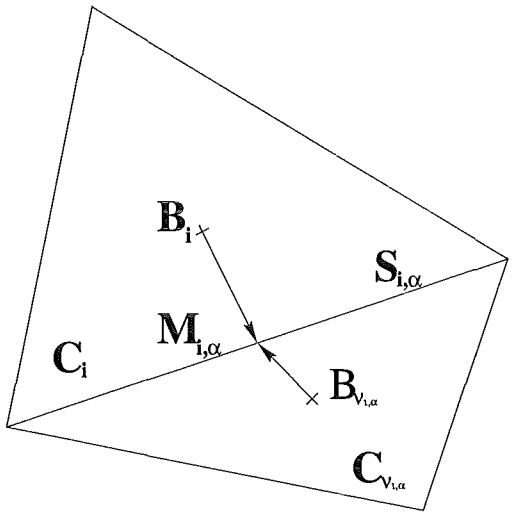

Figure 2.3: Two neighboring cells C_i and $C_{\nu_{i,\alpha}}$ with the barycenters B_i and $B_{\nu_{i,\alpha}}$ and their common side $S_{i,\alpha}$ with the midpoint $M_{i,\alpha}$ for the two-dimensional geometry.

is also a face of the neighboring grid cell $C_{\nu_{i,\alpha}}$ of C_i (see Figure 2.3), a further second-order estimation $[u_k]_{i,\alpha}^{n-}$ of $u_k(M_{i,\alpha}, t^{n+\frac{1}{2}})$ can be found:

$$[u_k]_{i,\alpha}^{n-} = [u_k]_{\nu_{i,\alpha}}^n + \sum_{j=1}^{3} (\overrightarrow{B_{\nu_{i,\alpha}} M_{i,\alpha}})_j \, (s_{\nu_{i,\alpha}}^n)_{kj} - \frac{\Delta t}{2} \sum_{j=1}^{3} \left[\mathcal{K}_j \, (s_{\nu_{i,\alpha}}^n)_j\right]_k \ . \tag{2.12b}$$

These two second-order accurate approximations (2.12a)-(2.12b) are the initial values of the Riemann problem which is discussed in more detail in the next section. However, it is obvious that if the slopes $(s_i^n)_{kj}$ in (2.12a)-(2.12b) are set equal to zero, piecewise constant states are obtained and the order of the numerical scheme is reduced to one.

2.2.3 Solution of the Riemann Problem

The further step to obtain the numerical scheme for the Maxwell equations is a combination of the two approximations (2.12a-2.12b) and the solution of the

Riemann problem (RP). The local RP is an initial-value problem of the form

$$\frac{\partial u}{\partial t} + \mathcal{A}\frac{\partial u}{\partial \xi} = 0 \,, \tag{2.13a}$$

with the initial data

$$u^{(0)}(\xi) = u(\xi, 0) = \begin{cases} u^+ & \text{if } \xi < 0 \\ u^- & \text{if } \xi > 0 \end{cases} \,, \tag{2.13b}$$

where the coordinate ξ is associated with the orientation of the normal vector at the face $S_{i,\alpha}$ (see Figure 2.4). For the sake of clarity, we have dropped in the present RP formulation the superscript n and the subscripts i and α (cf. (2.8b) and (2.12)). For the interesting case of the vacuum Maxwell equations the RP can be solved exactly by applying the theory of characteristics (see, e.g., [17]).

In order to illustrate the solution path of the RP, we introduce the characteristic variable

$$v(\xi, t) = \mathcal{R}^{-1}\, u(\xi, t) \,, \tag{2.14a}$$

where $\mathcal{R}^{-1}$ is the matrix of the left eigenvectors of $\mathcal{A}$ (see Appendix A), and recast (2.13a) according to

$$\frac{\partial v}{\partial t} + \Lambda\frac{\partial v}{\partial \xi} = 0 \,. \tag{2.14b}$$

Since Λ is a diagonal matrix (see Appendix A) we obtain six uncoupled linear transport equations whose solutions are given by

$$v_k(\xi, t) = v_k^{(0)}(\xi - \lambda_k t) \;; \quad k = 1, ..., 6 \,, \tag{2.14c}$$

with the initial values $v^{(0)}(\xi) = \mathcal{R}^{-1}\, u^{(0)}(\xi)$. At $\xi = 0$ the solution of the RP in characteristic variables reads as (cf. Figure 2.4)

$$v(0, t) = \left(v_1^-(u^-), v_2^-(u^-), \overline{v}_3(\overline{u}), \overline{v}_4(\overline{u}), v_5^+(u^+), v_6^+(u^+)\right)^T \,, \tag{2.15a}$$

where $v^\pm$ is calculated from $v^\pm = \mathcal{R}^{-1}\, u^\pm$. Because two eigenvalues of the matrix $\mathcal{A}$ are zero ($\lambda_{3/4}$, see Appendix A), the solution of (2.15a) depends also on the initial value $\overline{u} = u(0, 0)$. However, later on we will see that this value does not influence the numerical flux computation, and hence, it is not necessary to specify $\overline{u}$ [29] in this context. Multiplying now (2.15a) with the matrix $\mathcal{R}$ of the right eigenvectors (see Appendix A), we obtain the solution of the RP at $\xi = 0$

$$u(0, t) = \mathcal{R}\, v(0, t) \,, \tag{2.15b}$$

which is explicitly written down in Appendix B. Applying the matrix $\mathcal{A}$ to the last equation, it can be easily proved (see Appendix B) that the relation

$$\mathcal{A}\, u(0, t) = \mathcal{A}^+ u^+ + \mathcal{A}^- u^- \tag{2.15c}$$

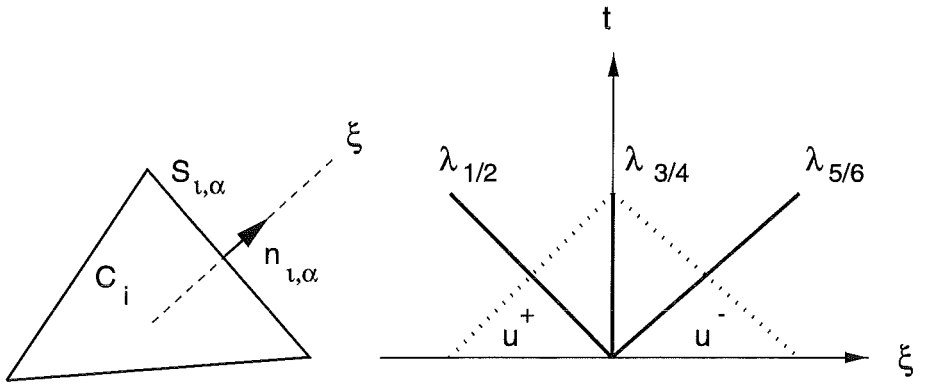

Figure 2.4: Riemann problem at the face $S_{i,\alpha}$ of the cell C_i and its schematical solution in the (ξ, t)-plane.

holds. Inserting this result into the equation for the numerical flux (2.9), the compact flux-splitting form

$$G_{i,\alpha}^{n+1/2} = L_{i,\alpha}\left(\mathcal{A}_{i,\alpha}^{+} u_{i,\alpha}^{n+} + \mathcal{A}_{i,\alpha}^{-} u_{i,\alpha}^{n-}\right) \tag{2.16}$$

is obtained. This formulation reveals that the total flux $G_{i,\alpha}^{n+1/2}$ is balanced by a flux to the "right" having positive eigenvalues only, and a flux to the "left" having negative eigenvalues only associated with $\mathcal{A}_{i,\alpha}^{+}$ and $\mathcal{A}_{i,\alpha}^{-}$, respectively.

2.2.4 Computation of the Gradients

There are different ways to compute approximations of the gradient of a regular function $v(x,t)$ at time $t = t^n$. Let us consider the tetrahedral cell C_i with the barycenter B_i, which totally lies inside the domain Ω and is, hence, no boundary cell. This cell has four neighbors $C_{\nu_{i,\alpha}}$, with $\alpha \in [1,4]$, having the barycenters $B_{i,\alpha}$.

The first strategy to define the gradient vector $(s_i^n)^{(1)} \in \mathbb{R}^3$ of any regular function v is to consider the following system of linear equations:

$$\begin{cases} v(B_i,t^n) - v(B_{i,2},t^n) = \overrightarrow{B_{i,2}B_i} \cdot (s_i^n)^{(1)} \\ v(B_i,t^n) - v(B_{i,3},t^n) = \overrightarrow{B_{i,3}B_i} \cdot (s_i^n)^{(1)} \\ v(B_i,t^n) - v(B_{i,4},t^n) = \overrightarrow{B_{i,4}B_i} \cdot (s_i^n)^{(1)} \end{cases} . \tag{2.17}$$

It is easy to prove that this system of equations has a (unique) solution if the four barycenters B_i, $B_{i,2}$, $B_{i,3}$ and $B_{i,4}$ form a non-degenerate tetrahedron which, however, might not always be the case. Then, $(s_i^n)^{(1)}$ is a first-order approximation of the gradient of the function v in B_i. Similary, we can define a further gradient approximation $(s_i^n)^{(2)}$, calculated from the values of v at the locations $\{B_i; B_{i,3}; B_{i,4}; B_{i,1}\}$; this approximation is obtained by a cyclical permutation of

the indices $\{1;2;3;4\}$ in (2.17). Obviously, two supplementary approximations $(s_i^n)^{(3)}$ and $(s_i^n)^{(4)}$ for the gradient of the function v in B_i can be found by further permutations of the indices.

To avoid spurious oscillations near steep gradients, it is convenient to use the slope-limited gradient, being the one in the set $\left\{(s_i^n)^{(l)}\right\}_{l\in[1,4]}$ with the smallest norm. For more details of this item we refer to [7, 21, 23].

The second way to define the gradient of $v(x,t^n)$ in B_i is established by the solution of the system

$$\begin{cases} v(B_{i,1},t^n) - v(B_{i,4},t^n) = \overrightarrow{B_{i,4}B_{i,1}} \cdot (s_i^n)^{(5)} \\ v(B_{i,2},t^n) - v(B_{i,4},t^n) = \overrightarrow{B_{i,4}B_{i,2}} \cdot (s_i^n)^{(5)} \\ v(B_{i,3},t^n) - v(B_{i,4},t^n) = \overrightarrow{B_{i,4}B_{i,3}} \cdot (s_i^n)^{(5)} \end{cases}, \tag{2.18}$$

where the value $v(B_i,t^n)$ is not explicitly used. The advantage of this definition of $(s_i^n)^{(5)} \in \mathbb{R}^3$ is that the barycenters $B_{i,1}$, $B_{i,2}$, $B_{i,3}$ and $B_{i,4}$ always form a non-degenerate tetrahedron and, hence, the equation (2.18) possesses a unique solution. Furthermore, for regular grids, this approximation is a second-order accurate one, which is, in general, not valid in the case $\left\{(s_i^n)^{(l)}\right\}_{l\in[1,4]}$.

After performing numerical experiments with both possibilities of the gradient calculation, it has been found out that the $(s_i^n)^{(5)}$ gradient approximation yields more accurate and stable results than those obtained by using $(s_i^n)^{(l)}$ with $l \in [1,4]$ or by using a slope-limited approximation.

2.2.5 Boundary Conditions

In this section we describe the numerical realization and implementation of physically occuring as well as computationally motivated boundary conditions. Boundary conditions and their implementation are only well-posed if locally the characteristic-based wave propagation is taken into account. In general, initial-boundary-value (IBV) instead of RP have to be solved at the grid cells adjacent to the border of the computational domain. However, it is possible to reformulate these IBV as Riemann problems by introducing fictitious grid cells surrounding the real computational domain and specifying in these cells suitable values in such a way that the solution of the RP at the border provides with the proper boundary conditions [23].

For our purposes, we proceed as follows: First, we determine from $u(0,t)$ (see equation (2.15b) and Appendix B) the values in the fictitious dummy or ghost cells denoted, without loss of generality, by u^-. Then, we compute the numerical boundary flux (cf. (2.16)), which have to be prescribed in our implemetation for the boundary conditions of the electromagnetic field.

Perfect Conductor

First, we consider the boundary condition of a perfectly conducting face, where the tangential electrical field vanishes at the surface:

$$\mathbf{E}_0 \times \mathbf{n} = 0 \; ; \quad \text{with} \quad \mathbf{n} = (n_1, n_2, n_3)^T \; . \tag{2.19a}$$

In our notation, this condition is equivalent to (see Appendix B)

$$\begin{pmatrix} u_{0_2} n_3 - u_{0_3} n_2 \\ u_{0_3} n_1 - u_{0_1} n_3 \\ u_{0_1} n_2 - u_{0_2} n_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \; . \tag{2.19b}$$

Inserting the values for u_{0_1}, u_{0_2} and u_{0_3} given in Appendix B, we find

$$\begin{aligned} & n_3(u_2^+ + u_2^-) - n_2(u_3^+ + u_3^-) \\ & -c(n_2^2 + n_3^2)(u_4^+ - u_4^-) + cn_1n_2(u_5^+ - u_5^-) + cn_1n_3(u_6^+ - u_6^-) \;\; = \;\; 0 \; , \end{aligned} \tag{2.20a}$$

$$\begin{aligned} & -n_3(u_1^+ + u_1^-) + n_1(u_3^+ + u_3^-) \\ & +cn_1n_2(u_4^+ - u_4^-) - c(n_1^2 + n_3^2)(u_5^+ - u_5^-) + cn_2n_3(u_6^+ - u_6^-) \;\; = \;\; 0 \; , \end{aligned} \tag{2.20b}$$

and

$$\begin{aligned} & n_2(u_1^+ + u_1^-) - n_1(u_2^+ + u_2^-) \\ & +cn_1n_3(u_4^+ - u_4^-) + cn_2n_3(u_5^+ - u_5^-) - c(n_1^2 + n_2^2)(u_6^+ - u_6^-) \;\; = \;\; 0 \; . \end{aligned} \tag{2.20c}$$

Obviously, for (2.20a)-(2.20c) there exists no uniquely determined solution, since we have three equations for six unknows. This arbitrariness reflects the fact that in the dummy cell the characteristic variables corresponding to waves which do not enter the computational domain have no influence on the solution at the boundary. A sufficient condition given by

$$(u_1^-, u_2^-, u_3^-, u_4^-, u_5^-, u_6^-) = (-u_1^+, -u_2^+, -u_3^+, u_4^+, u_5^+, u_6^+) \; , \tag{2.21}$$

fulfils (2.20a)-(2.20c) identically, and is used to compute the boundary flux through the considered surface cell according to (2.16).

Silver-Müller Conditions

With the normal $\mathbf{n}$ (2.19a) and the abbreviations

$$u_e^{(0)} = (u_{0_1}, u_{0_2}, u_{0_3})^T \quad \text{and} \quad u_b^{(0)} = (u_{0_4}, u_{0_5}, u_{0_6})^T \; , \tag{2.22a}$$

the Silver-Müller boundary condition reads as

$$\left(u_e^{(0)} - c u_b^{(0)} \times \mathbf{n}\right) \times \mathbf{n} = (e_0 - c b_0 \times \mathbf{n}) \times \mathbf{n} \; , \tag{2.22b}$$

where e_0 and b_0 are given vectors in $\mathbb{R}^3$. In the case where these two vectors are equal to zero, it can be shown that the equality (2.22b) is a first-order accurate absorbing boundary condition (see, for example, [25]). Combining the condition (2.22b) with $u_0 = u(0,t)$ given in the Appendix B, we get

$$\left(u_e^- - cu_b^- \times \mathbf{n}\right) \times \mathbf{n} = (e_0 - cb_0 \times \mathbf{n}) \times \mathbf{n} \,, \tag{2.22c}$$

where $u_e^- = \left(u_1^-, u_2^-, u_3^-\right)^T$ and $u_b^- = \left(u_4^-, u_5^-, u_6^-\right)^T$. A sufficient condition that fulfils (2.22c) is

$$u_e^- = e_0 \quad \text{and} \quad u_b^- = b_0 \,. \tag{2.23}$$

Cylinder Symmetrical Axis

In the case where the problem under consideration possesses a cylindrical symmetry, a two-dimensional description in the (z,r)-plane is satisfactory whereat the third component of the normal vector $\mathbf{n}$ is equal to zero. However, for such a problem it is necessary to specify an additional condition at the axis $r = 0$. Because $u_3 = u_6 = 0$, on this axis we impose the boundary condition

$$u_{0_3} = u_{0_6} = 0 \,. \tag{2.24a}$$

By using the solution of the RP (see Appendix B), we furthermore obtain the following requirements

$$(u_3^+ + u_3^-) + cn_2(u_4^+ - u_4^-) - cn_1(u_5^+ - u_5^-) = 0 \,, \tag{2.24b}$$

$$-n_2(u_1^+ - u_1^-) + n_1(u_2^+ - u_2^-) + c(u_6^+ + u_6^-) = 0 \,. \tag{2.24c}$$

on the z-axis. A sufficient condition to fulfil the last two equalities is given by

$$(u_1^-, u_2^-, u_3^-, u_4^-, u_5^-, u_6^-) = (u_1^+, u_2^+, -u_3^+, u_4^+, u_5^+, -u_6^+) \,. \tag{2.25}$$

2.3 Particle Treatment

A further central building block in order to find the numerical solution of the Maxwell-Lorentz model (1.1)-(1.3) is the particle treatment, consisting of an accumulation of different approximation techniques for the interpolation, particle pushing, localization and charge assignment [3, 13]. Basically, the goal of particle treatment is to obtain the redistributed charged macro particle distribution under the action of the applied external as well as self-generated electromagnetic fields in order to compute the changed charge and current densities, the sources for the Maxwell equations.

2.3.1 Particle Pushing

The discretization of the relativistic equations of motion (1.2) as well as its non-relativistic counterpart has been described extensively in the literature [5, 3, 13, 30]. However, for the sake of completeness we briefly recall the basic features of the used leapfrog-scheme introduced by Boris [5], taking into account the special structure of the Lorentz force (1.2c). For that purpose, we consider in the following only one macro particle and, hence, drop from now on the index k and rewrite (1.2a-b) according to

$$\boldsymbol{x}^{n+1} - \boldsymbol{x}^n = \Delta t \boldsymbol{v}^{n+1/2} , \tag{2.26a}$$

$$\left(\boldsymbol{U}^{n+1/2} - \alpha \boldsymbol{E}^n\right) - \left(\boldsymbol{U}^{n-1/2} + \alpha \boldsymbol{E}^n\right) = \frac{2\alpha}{\gamma^n} \boldsymbol{U}^n \times \boldsymbol{B}^n , \tag{2.26b}$$

where $\alpha = \frac{Q\Delta t}{2M}$ and the relativistic velocity is computed from $\boldsymbol{U} = \gamma \boldsymbol{v}$ with $\gamma^2 = \frac{1}{1-|\boldsymbol{v}|^2/c^2} = 1 + |\boldsymbol{U}|^2/c^2$. Furthermore, Δt is the time step size and n denotes the actual time level where the electromagnetic fields at the position $\boldsymbol{x}^n$ are given. Obviously, the right-hand side of (2.26a) is time-centered around $t^{n+1/2} = (n + 1/2)\Delta t$ while that of (2.26b) has to be computed at $t^n = n\Delta t$, leading to a second-order accurate integration scheme. For the further proceeding, we now introduce the quantities

$$\boldsymbol{u}^- = \boldsymbol{U}^{n-1/2} + \alpha \boldsymbol{E}^n , \tag{2.27a}$$

$$\boldsymbol{u}^+ = \boldsymbol{U}^{n+1/2} - \alpha \boldsymbol{E}^n , \tag{2.27b}$$

replace $\boldsymbol{U}^n$ by its average value $\frac{1}{2}\left(\boldsymbol{U}^{n-1/2} + \boldsymbol{U}^{n+1/2}\right)$ and approximate $\gamma^n \approx \gamma^- = 1 + |\boldsymbol{u}^-|^2/c^2$ with the velocity $\boldsymbol{u}^-$ obtained after the first "half-acceleration" described by (2.27a). Then, equation (2.26b) can be recast into the form

$$\boldsymbol{u}^+ - \boldsymbol{u}^- = \left(\boldsymbol{u}^+ + \boldsymbol{u}^-\right) \times \boldsymbol{t} , \tag{2.27c}$$

where we introduced the auxilary vector $\boldsymbol{t} = \frac{\alpha}{\gamma^-}|\boldsymbol{B}^n|\boldsymbol{b}^n$ with $\boldsymbol{b}^n = \frac{\boldsymbol{B}^n}{|\boldsymbol{B}^n|}$. With this relation it is easy to prove that $\boldsymbol{t} \cdot \boldsymbol{u}^+ = \boldsymbol{t} \cdot \boldsymbol{u}^-$ and $|\boldsymbol{u}^+| = |\boldsymbol{u}^-|$. In order to

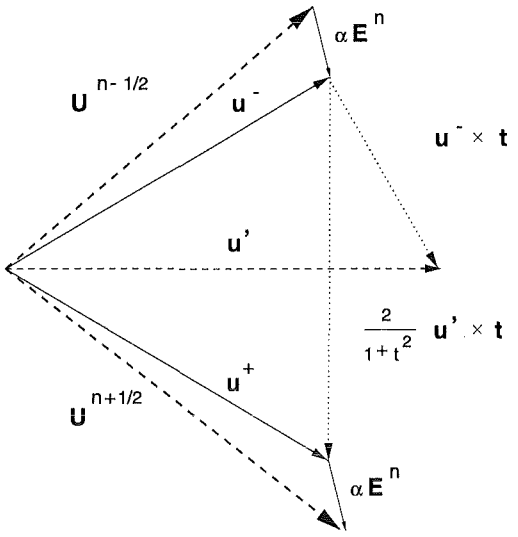

Figure 2.5: Geometrical illustration of the second-order accurate Boris scheme.

determine $\boldsymbol{u}^+$ from (2.27c), we compute an additional velocity vector given by

$$\boldsymbol{u}' = \boldsymbol{u}^- + \boldsymbol{u}^- \times \boldsymbol{t}\,, \tag{2.27d}$$

which is the sum of $\boldsymbol{u}^-$ and the "half-rotation" of $\boldsymbol{u}^-$ around the magnetic induction $\boldsymbol{B}^n$, having the length of $|\boldsymbol{u}'|^2 = \left(1+|\boldsymbol{t}|^2\right)|\boldsymbol{u}^-|^2 - \left(\boldsymbol{u}^- \cdot \boldsymbol{t}\right)^2$. A further "half-rotation" but now of $\boldsymbol{u}'$ around $\boldsymbol{B}^n$ yields

$$\begin{aligned} \boldsymbol{u}' \times \boldsymbol{t} &= \boldsymbol{u}^+ \times \boldsymbol{t} + |\boldsymbol{t}|^2 \boldsymbol{u}^+ - \left(\boldsymbol{u}^+ \cdot \boldsymbol{t}\right)\boldsymbol{t} \\ &= \boldsymbol{u}^- \times \boldsymbol{t} - |\boldsymbol{t}|^2 \boldsymbol{u}^- + \left(\boldsymbol{u}^- \cdot \boldsymbol{t}\right)\boldsymbol{t}\,. \end{aligned} \tag{2.27e}$$

From this relation and equality (2.27c) we find that $\boldsymbol{u}^+$ is obtained from

$$\boldsymbol{u}^+ = \boldsymbol{u}^- + \frac{2}{1+|\boldsymbol{t}|^2}\boldsymbol{u}' \times \boldsymbol{t}\,. \tag{2.27f}$$

After a second "half-acceleration" by $\Delta t/2$ with the electrical field $\boldsymbol{E}^n$, we finally get the solution of (2.26b), namely, the velocity at the time level $t = t^{n+1/2}$

$$\boldsymbol{U}^{n+1/2} = \boldsymbol{u}^+ + \alpha \boldsymbol{E}^n\,, \tag{2.27g}$$

and from that and (2.26a) the new particle position at $t = t^{n+1}$. For the special case where $\boldsymbol{t}$ is orthogonal to $\boldsymbol{u}^-$ the outlined Boris scheme can be illustrated geometrically as it is shown in Figure 2.5.

2.3.2 Particle Localization

For our purposes, we adopt the intensively investigated particle localization techniques proposed by Löhner et al. [19, 18] and Assous et al. [2] for unstructured

mesh zones spanning triangles in two and tetrahedra in three space dimensions. Especially, we use the Löhner approach for the two-dimensional situation while we adopt the Assous strategy for the three-dimensional case, because it is found out, that the Assous algorithm is faster than the one proposed by Löhner.

Two-dimensional Localization Algorithm

In the following, we describe briefly the basic ideas of the two-dimensional Löhner approach. This algorithm based on the calculation of shape-functions $\mathcal{S}_{i,\alpha} = \mathcal{S}_{i,\alpha}(\boldsymbol{x}^n)$ for the macro particle located at time $t = t^n$ inside the mesh zone C_i of the computational grid at $\boldsymbol{x}^n$. From these functions, criteria are obtained for a sophisticated searching strategy [19, 18]. Closely related to this approach is the one proposed by Westermann [31], where the ideas of convex hulls are used.

In the two-dimensional case, C_i is a triangle with the vertices $\mathcal{M}_{i,\alpha}$ having the grid coordinates $\boldsymbol{a}_{i,\alpha} = (a_\alpha, b_\alpha)^T$, where α runs from one to $\sigma_i = 3$. The situation on hand is depicted in Figure 2.6, where we recognize that the triangle opposite to the vertex $\mathcal{M}_{i,\alpha}$ is denoted by C_{i_α}. To decide now whether a particle $\boldsymbol{x}^n = (x^n, y^n)^T$ lies inside the element C_i at $t = t^n$ the following strategy is applied:

Step 1:

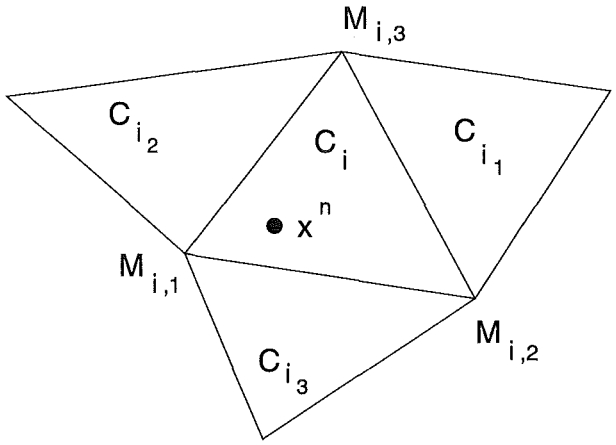

Figure 2.6: Particle localization in the two-dimensional case.

Calculate the three real numbers $\mathcal{S}_{i,1}$, $\mathcal{S}_{i,2}$, $\mathcal{S}_{i,3} \in \mathbb{R}$ according to

$$\sum_{\alpha=1}^{\sigma_i} \mathcal{S}_{i,\alpha} \boldsymbol{a}_{i,\alpha} = \boldsymbol{x}^n \;, \quad \text{with} \quad \sum_{\alpha=1}^{\sigma_i} \mathcal{S}_{i,\alpha} = 1 \;. \tag{2.28a}$$

Explicitly, these linear shape-functions $\mathcal{S}_{i,\alpha}$ with respect to the particle coordinate

$\boldsymbol{x}^n$ are given by

$$\mathcal{S}_{i,2} = \frac{1}{J}\left[(b_3 - b_1)(x^n - a_1) - (a_3 - a_1)(y^n - b_1)\right] , \tag{2.28b}$$

$$\mathcal{S}_{i,3} = \frac{1}{J}\left[-(b_2 - b_1)(x^n - a_1) + (a_2 - a_1)(y^n - b_1)\right] , \tag{2.28c}$$

$$\mathcal{S}_{i,1} = 1 - \mathcal{S}_{i,2} - \mathcal{S}_{i,3} , \tag{2.28d}$$

with

$$J = (b_3 - b_1)(a_2 - a_1) - (b_2 - b_1)(a_3 - a_1) . \tag{2.28e}$$

Step 2:

If

$$\text{MIN}\left(\mathcal{S}_{i,1}; \mathcal{S}_{i,2}; \mathcal{S}_{i,3}\right) \geq 0 \quad \text{and} \quad \text{MAX}\left(\mathcal{S}_{i,1}; \mathcal{S}_{i,2}; \mathcal{S}_{i,3}\right) \leq 1 , \tag{2.29a}$$

then $\boldsymbol{x}^n$ is located inside the mesh zone C_i at time $t = t^n$.
Otherwise, if

$$\text{MIN}\left(\mathcal{S}_{i,1}; \mathcal{S}_{i,2}; \mathcal{S}_{i,3}\right) < 0 \tag{2.29b}$$

$\boldsymbol{x}^n \notin C_i$, and we have to continue the search in the element adjacent to C_i and lying opposite to the vertex $\mathcal{M}_{i,\alpha}$ possessing the smallest value for the shape-function $\mathcal{S}_{i,\alpha}$.

Three-dimensional Localization Algorithm

The basic ideas of the three-dimensional scheme proposed by Assous et al. [2] can be summarized as follows: Consider the tetrahedric grid zone C_i as depicted in Figure 2.7 having the vertices $\mathcal{M}_{i,\alpha}$ with the coordinates $\boldsymbol{a}_{i,\alpha} = (a_\alpha, b_\alpha, c_\alpha)^T$ where $1 \leq \alpha \leq \sigma_i = 4$. Then, we first calculate the determinants $\Delta_{i,\alpha}$ with respect to the particle position $\boldsymbol{x}^n$ at $t = t^n$ from

$$\Delta_{i,\alpha_1} = (-1)^{\alpha_1 + 1}\left(\boldsymbol{A}_{\alpha_2} \times \boldsymbol{A}_{\alpha_3}\right) \cdot \boldsymbol{A}_{\alpha_4} , \tag{2.30a}$$

where the $\{\alpha_1; \alpha_2; \alpha_3; \alpha_4\}$ are cyclical permutations of $\{1; 2; 3; 4\}$ and $\boldsymbol{A}_\alpha$ is the abbreviation of the difference vector $\boldsymbol{A}_\alpha = \boldsymbol{a}_{i,\alpha} - \boldsymbol{x}^n$. Explicitly, we find for the four determinants of C_i

$$\Delta_{i,1} = (\boldsymbol{A}_2 \times \boldsymbol{A}_3) \cdot \boldsymbol{A}_4 , \tag{2.30b}$$

$$\Delta_{i,2} = (\boldsymbol{A}_1 \times \boldsymbol{A}_4) \cdot \boldsymbol{A}_3 , \tag{2.30c}$$

$$\Delta_{i,3} = (\boldsymbol{A}_4 \times \boldsymbol{A}_1) \cdot \boldsymbol{A}_2 , \tag{2.30d}$$

$$\Delta_{i,4} = (\boldsymbol{A}_3 \times \boldsymbol{A}_2) \cdot \boldsymbol{A}_1 . \tag{2.30e}$$

Afterwards, the following possible cases have to be considered during the searching procedure, where ϵ denotes the machine zero:

Case 1:
If

$$\forall k \in \{1,2,3,4\} \ , \ \ \Delta_{i,k} \geq -\epsilon \ , \tag{2.31a}$$

then the particle is located in the mesh zone C_i at $t = t^n$, i.e., $\boldsymbol{x}^n \in C_i$.
Case 2:
If

$$\exists k \in \{1,2,3,4\} \ , \ \ \Delta_{i,k} < -\epsilon \ , \tag{2.31b}$$

holds, continue the searching procedure in the cell C_{i_k} located opposite of the vertex $\mathcal{M}_{i,k}$.
Case 3:
If

$$\exists (k,l) \in \{1,2,3,4\}^2 \ , \ k \neq l \ , \Delta_{i,k} \ , \Delta_{i,l} < -\epsilon \ , \tag{2.31c}$$

or if even three determinants become negative, the decision process is more complex. To discuss this item, we assume that $\Delta_{i,1}$ and $\Delta_{i,2}$ are lower than $-\epsilon$ and denote by H the intersection point of the particle trajectory with the plane spanned by $(\mathcal{M}_{i,2}\ \mathcal{M}_{i,3}\ \mathcal{M}_{i,4})$ (cf. right picture of Figure 2.7). Then, we introduce

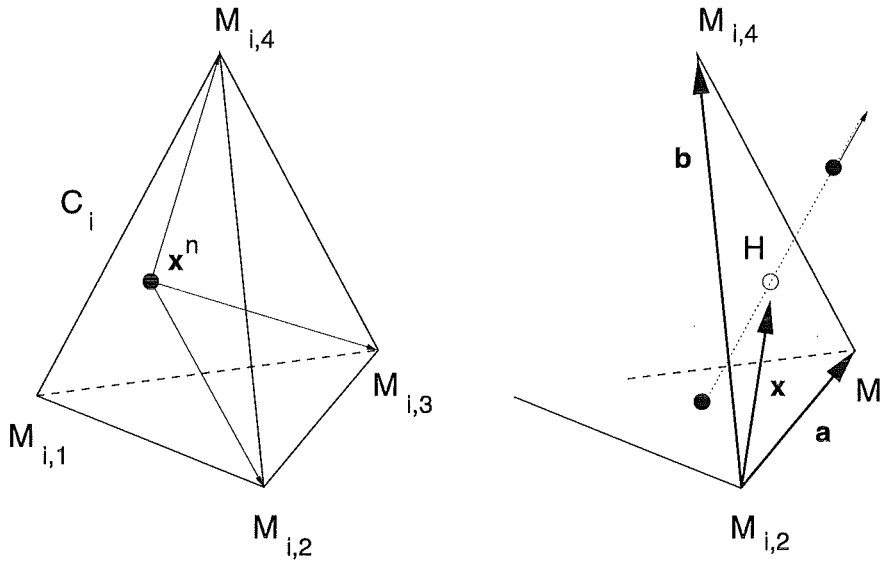

Figure 2.7: Particle localization within the tetrahedron C_i according to Assous et al. [2].

the local basis $\boldsymbol{a} = \overrightarrow{\mathcal{M}_{i,2}\mathcal{M}_{i,3}}$, $\boldsymbol{b} = \overrightarrow{\mathcal{M}_{i,2}\mathcal{M}_{i,4}}$ and expand the vector $\boldsymbol{x} = \overrightarrow{\mathcal{M}_{i,2}H}$ according to

$$\boldsymbol{x} = \alpha\boldsymbol{a} + \beta\boldsymbol{b} \ . \tag{2.32a}$$

Moreover, this vector is also given by

$$\boldsymbol{x} = \overrightarrow{\mathcal{M}_{i,2}\mathcal{P}} - \overrightarrow{H\mathcal{P}} \,, \tag{2.32b}$$

where $\mathcal{P}$ denotes the particle position at time $t = t^n$. Because $\overrightarrow{H\mathcal{P}}$ is parallel to $\boldsymbol{v}^{n-1/2}$, we find that

$$\boldsymbol{x} \times \boldsymbol{v}^{n-1/2} = \overrightarrow{\mathcal{M}_{i,2}\mathcal{P}} \times \boldsymbol{v}^{n-1/2} \tag{2.32c}$$

holds, which means, that it is not necessary to compute H explicitly. Inserting (2.32a) into the last relation, we get

$$\overrightarrow{\mathcal{M}_{i,2}\mathcal{P}} \times \boldsymbol{v}^{n-1/2} = \alpha \boldsymbol{a} \times \boldsymbol{v}^{n-1/2} + \beta \boldsymbol{b} \times \boldsymbol{v}^{n-1/2} \,, \tag{2.32d}$$

yielding two equations for the unknown parameters α and β. Now, we can decide if

$$\alpha, \beta \in [0, 1] \quad \text{and} \quad \alpha + \beta \leq 1 \,,$$

then the particle crossed the plane $(\mathcal{M}_{i,2}\ \mathcal{M}_{i,3}\ \mathcal{M}_{i,4})$ and has to be searched in C_{i_1}. If this condition is not met, the searching has to be continued in the tetrahedron C_{i_2}.

2.3.3 Particle Assignment and Interpolation

Applying the outlined localization strategy (in two or three space dimensions), a macro particle with the phase space coordinates $(\boldsymbol{x}_k(t), \boldsymbol{v}_k(t))$ may be found in the grid cell C_i. Now, we have to calculate the contribution of this particle to the charge and current densities at the node i with the coordinates $\boldsymbol{a}_i$. This node is surrounded by a certain number ν_i of elements forming the local node domain Ω_i as depicted in Figure 2.8. In order to do this, we compute the particle

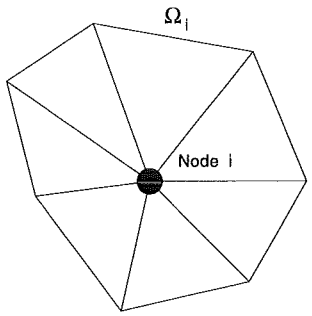

Figure 2.8: Local node domain Ω_i established by ν_i elements.

linked shape-function $\mathcal{S}_i = \mathcal{S}_i(\boldsymbol{x}(t))$ according to (2.28a) (for the two or three-dimensional case). This function possesses the property that at the considered

node i the relation $\mathcal{S}_i(\boldsymbol{a}_j) = \delta_{i,j}$ holds, where $\delta_{i,j}$ denotes the Kronecker symbol. Then, we perform an averaging process over the local node domain Ω_i, defined for any integrable function $h(\boldsymbol{x}, t)$ according to

$$h(\boldsymbol{a}_i, t) = \frac{1}{\mathcal{V}_i} \int_{\Omega_i} h(\boldsymbol{x}, t) \mathcal{S}_i(\boldsymbol{x}, t)\, dV \;, \tag{2.33a}$$

where $\mathcal{V}_i$ denotes the volume associated with the node i given by

$$\mathcal{V}_i = \int_{\Omega_i} \mathcal{S}_i(\boldsymbol{x}) dV \;. \tag{2.33b}$$

Applying this averaging to (1.3), the contribution of the kth charged macro particle to the charge and current densities at the node i is obained from

$$\rho_k(\boldsymbol{a}_i, t) = \frac{Q_k}{\mathcal{V}_i} \mathcal{S}_i^{(k)} \;, \tag{2.34a}$$

$$\boldsymbol{j}_k(\boldsymbol{a}_i, t) = \frac{Q_k \boldsymbol{v}_k}{\mathcal{V}_i} \mathcal{S}_i^{(k)} \;, \tag{2.34b}$$

where the abbreviation $\mathcal{S}_i^{(k)} = \mathcal{S}_i(\boldsymbol{x}_k(t))$ is used.

After the total charge and current densities at the nodes of the computational mesh are determined, the new electromagnetic fields at these nodes are computed by solving the Maxwell equations (2.1)) with the discussed FV scheme. To advance the charged particles from the time level t^n to t^{n+1} in this new fields, we once again apply the concept based on the shape-function approach, but now, in order to determine the fields at the actual particles positions at $t = t^n$. The electromagnetic fields $u(\boldsymbol{x}_k(t^n), t^n)$ (cf. equation (2.2)) acting at the kth charged particle position are calculated from the formula

$$u(\boldsymbol{x}_k(t^n), t^n) = \sum_{\alpha=1}^{\sigma_i} \mathcal{S}_{i,\alpha}(\boldsymbol{x}_k(t^n)) u(\boldsymbol{a}_{i,\alpha}, t^n) \;. \tag{2.35}$$

Here, $u(\boldsymbol{a}_{i,\alpha}, t^n)$ are the electromagnetic fields given at the nodes $\mathcal{M}_{i,\alpha}$ of the grid cell C_i and $\mathcal{S}_{i,\alpha}(\boldsymbol{x}_k(t^n))$ denotes the particle linked shape-function in this cell.

Now, a typical PIC cycle is closed and the entire particle treatment starts again with particle pushing as described in Section 2.3.1.

2.3.4 Particle Handling at the Border of the Computational Domain

Two important boundary conditions in the context of particle treatment, namely, absorption and reflection on a certain border of the computational domain will be discussed in the following for the two-dimensional situation. For that, we assume

that the macro particle $\mathcal{P}^n$ with the coordinates $\boldsymbol{x}^n$ is situated in the grid zone C_i at time $t = t^n$ (cf. Appendix D). This grid element has one common side with the border of the computational domain and is established by the vertices $\mathcal{M}_{i,\alpha}$ with the coordinates $\boldsymbol{a}_{i,\alpha} = (a_\alpha, b_\alpha, 0)^T$, where $1 \leq \alpha \leq \sigma_i = 3$. Let us suppose that the particle leaves the grid cell C_i within the time interval $\Delta t = t^{n+1} - t^n$. The first task is now to find out the side $S_{i,\alpha}$ through which the particle moves during Δt. For that, we determine subsequently the following determinants

$$d_1 = \det(\boldsymbol{A}_1, \boldsymbol{p}) = (\boldsymbol{A}_1 \times \boldsymbol{p}) \cdot \boldsymbol{e}_3 \,, \tag{2.36a}$$

$$d_2 = \det(\boldsymbol{A}_2, \boldsymbol{p}) = (\boldsymbol{A}_2 \times \boldsymbol{p}) \cdot \boldsymbol{e}_3 \,, \tag{2.36b}$$

$$d_3 = \det(\boldsymbol{A}_3, \boldsymbol{p}) = (\boldsymbol{A}_3 \times \boldsymbol{p}) \cdot \boldsymbol{e}_3 \,, \tag{2.36c}$$

with the abbreviations $\boldsymbol{A}_\alpha = \overrightarrow{\mathcal{P}^n \mathcal{M}_{i,\alpha}} = \boldsymbol{a}_{i,\alpha} - \boldsymbol{x}^n$ and $\boldsymbol{p} = \overrightarrow{\mathcal{P}^n \mathcal{P}^{n+1}}$ and where the unit vector $\boldsymbol{e}_3$ is given by $\boldsymbol{e}_3 = (0, 0, 1)^T$. With the agreement that the side $S_{i,\alpha}$ which the particle crossed lies between $\mathcal{M}_{i,\alpha}$ and $\mathcal{M}_{i,\alpha+1}$ (see Figure 2.9) and the convention that $\mathcal{M}_{i,4} = \mathcal{M}_{i,1}$, we have to check the following possible alternatives, subsequently:

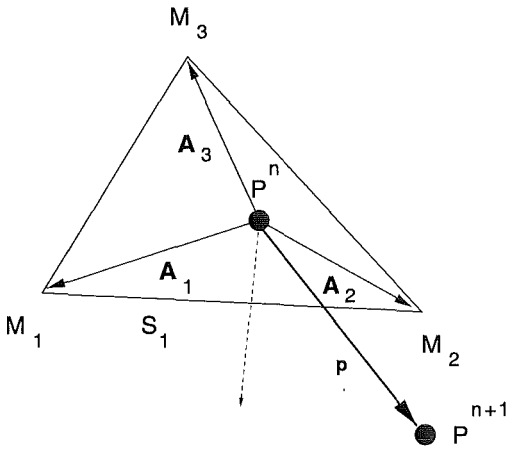

Figure 2.9: Particle passed side $S_{i,1}$ of the border grid cell C_i.

- **Particle passed side $S_{i,1}$**

This case is schematically depicted in Figure 2.9 . The signs of the determinants (2.36) for this situation are given by

$$d_1 > 0 \text{ and } d_2 < 0 \,;\; d_3 = \begin{cases} \geq 0, & \text{if } \angle(\boldsymbol{p}, \boldsymbol{A}_3) \leq \pi \\ < 0, & \text{if } \angle(\boldsymbol{p}, \boldsymbol{A}_3) > \pi \end{cases} . \tag{2.37a}$$

- **Particle passed side $S_{i,2}$**

This situation is seen in Figure 2.10 and characterized with the determinants

(2.36) according to

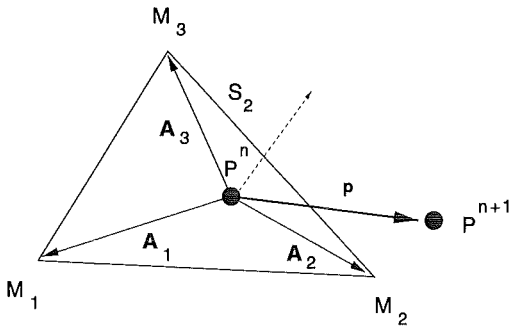

Figure 2.10: Particle crossed side $S_{i,2}$ during $\Delta t = t^{n+1} - t^n$ of the border cell C_i.

$$d_2 > 0 \text{ and } d_3 < 0\,; \;\; d_1 = \begin{cases} \geq 0, & \text{if } \angle(\boldsymbol{p}, \boldsymbol{A}_1) \leq \pi \\ < 0, & \text{if } \angle(\boldsymbol{p}, \boldsymbol{A}_1) > \pi \end{cases} . \tag{2.37b}$$

- **Particle passed side $S_{i,3}$**

Using once again the determinants (2.36), we find for this situation, schematically illustrated in Figure 2.11, the condition

$$d_3 > 0 \text{ and } d_1 < 0\,; \;\; d_2 = \begin{cases} \geq 0, & \text{if } \angle(\boldsymbol{p}, \boldsymbol{A}_2) \leq \pi \\ < 0, & \text{if } \angle(\boldsymbol{p}, \boldsymbol{A}_2) > \pi \end{cases} . \tag{2.37c}$$

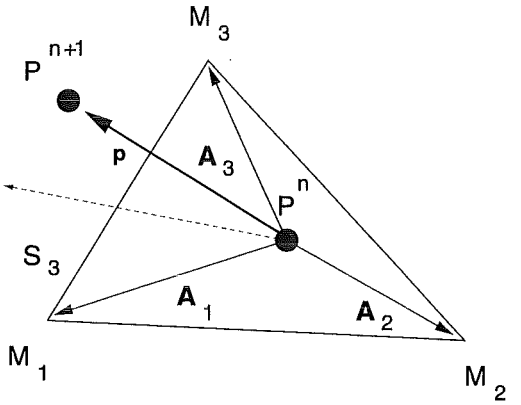

Figure 2.11: Side $S_{i,3}$ of the border cell C_i is crossed by the macro particle.

Defining the function k in the following way:

$$k(1,1) = 1\ ,\ k(1,2) = 2\ ,\ k(2,1) = 2\ ,\ k(2,2) = 3\ ,\ k(3,1) = 3\ ,\ k(3,2) = 1\ ,$$

we recognize from (2.37a) - (2.37c) that the particle crossed the side $S_{i,j}$ if and only if $d_{k(j,1)} > 0$ and $d_{k(j,2)} < 0$; this criterion is explicitly used in our computer program. If this side coincides with the border of the computational domain, the macro particle is absorbed, which means, it is taken out of the domain and of the further computation, and assessed for diagnostical reasons.

To discuss the particle reflection schematically depicted in Figure 2.12, we assume without loss of generality that the side $S_{i,1}$ of the triangle cell C_i coincides with the border of the computational domain where a reflection boundary condition is imposed. The considered macro particle crossed the side $S_{i,1}$ at the point

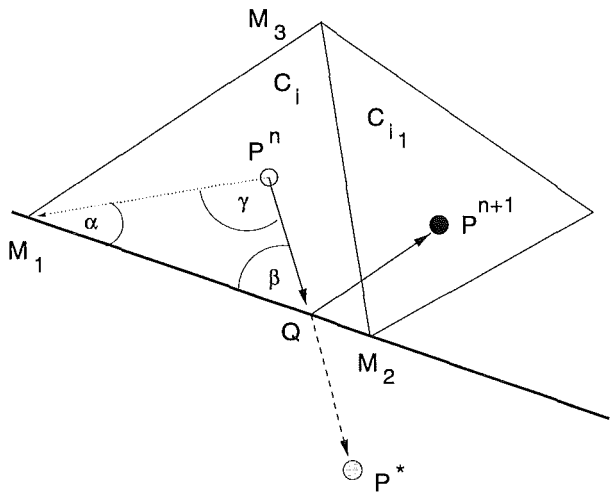

Figure 2.12: Reflection of a particle at the side $S_{i,1}$ of the border cell C_i.

Q during the time interval Δt and possesses the known position $\mathcal{P}^*$ at $t = t^{n+1}$, located outside the computational domain (see Figure 2.12). However, due to the reflection condition we have to put the particle to its true location $\mathcal{P}^{n+1}$ inside the domain which is still unknown. To determine this true position, we define the unit vectors according to

$$\boldsymbol{s}_1 = \frac{\overrightarrow{\mathcal{M}_{i,1}\mathcal{M}_{i,2}}}{|\overrightarrow{\mathcal{M}_{i,1}\mathcal{M}_{i,2}}|}\ ,\quad \boldsymbol{s}_2 = \frac{\overrightarrow{\mathcal{P}^n\mathcal{M}_{i,1}}}{|\overrightarrow{\mathcal{P}^n\mathcal{M}_{i,1}}|}\ ,\quad \boldsymbol{s}_3 = \frac{\overrightarrow{\mathcal{P}^n\mathcal{P}^*}}{|\overrightarrow{\mathcal{P}^n\mathcal{P}^*}|} \tag{2.38a}$$

and compute the scalar products

$$\boldsymbol{s}_2 \cdot \boldsymbol{s}_1 = \cos\alpha\ ,\quad \boldsymbol{s}_2 \cdot \boldsymbol{s}_3 = \cos\gamma\ ,\quad \boldsymbol{s}_3 \cdot \boldsymbol{s}_1 = \cos\beta\ , \tag{2.38b}$$

yielding the three angles α, β and γ of the triangle $(\mathcal{M}_{i,1}\, Q\, \mathcal{P}^n)$. With the given length $|\overrightarrow{\mathcal{P}^n \mathcal{M}_{i,1}}|$, we easily find that

$$|\overrightarrow{\mathcal{P}^n Q}| = \frac{\sin\alpha}{\sin\beta}\, |\overrightarrow{\mathcal{P}^n \mathcal{M}_{i,1}}| \,. \tag{2.39a}$$

Furthermore, it is obvious from Figure 2.12 that the vector $\overrightarrow{Q\mathcal{P}^*}$ can be expressed in terms of

$$\overrightarrow{Q\mathcal{P}^*} = \overrightarrow{\mathcal{P}^n\mathcal{P}^*} - \overrightarrow{\mathcal{P}^n Q} = \left(|\overrightarrow{\mathcal{P}^n\mathcal{P}^*}| - |\overrightarrow{\mathcal{P}^n Q}|\right)\, \boldsymbol{s}_3 \,. \tag{2.39b}$$

This vector may also be expanded in the orthogonal basis established by $\boldsymbol{s}$ and $\boldsymbol{m}$, with $\boldsymbol{s}\cdot\boldsymbol{m} = 0$ and $\boldsymbol{s}\times\boldsymbol{m} = \boldsymbol{e}_3$ where $\boldsymbol{e}_3 = (0,0,1)^T$, yielding

$$\overrightarrow{Q\mathcal{P}^*} = \left(\overrightarrow{Q\mathcal{P}^*}\cdot\boldsymbol{s}_1\right)\boldsymbol{s}_1 + \left(\overrightarrow{Q\mathcal{P}^*}\cdot\boldsymbol{m}\right)\boldsymbol{m} \,. \tag{2.39c}$$

Since $\boldsymbol{s}_3\cdot\boldsymbol{m} = -\sin\beta$, we obtain from the last two equations the following result:

$$\overrightarrow{Q\mathcal{P}^*} = \left(|\overrightarrow{\mathcal{P}_n\mathcal{P}^*}| - |\overrightarrow{\mathcal{P}_n Q}|\right)\, (\cos\beta\, \boldsymbol{s}_1 - \sin\beta\, \boldsymbol{m}) \,. \tag{2.39d}$$

A reflection of this vector at the side $S_{i,1}$ yields $\overrightarrow{Q\mathcal{P}^{n+1}}$, obtained from the relation

$$\overrightarrow{Q\mathcal{P}^{n+1}} = \left(|\overrightarrow{\mathcal{P}_n\mathcal{P}^*}| - |\overrightarrow{\mathcal{P}_n Q}|\right)\, (\cos\beta\, \boldsymbol{s}_1 + \sin\beta\, \boldsymbol{m}) \,. \tag{2.40}$$

Performing the final computation step

$$\overrightarrow{\mathcal{P}_n\mathcal{P}^{n+1}} = \overrightarrow{\mathcal{P}_n Q} + \overrightarrow{Q\mathcal{P}^{n+1}} \,, \tag{2.41}$$

we get the true position of the macro charge at $t = t^{n+1}$ reflected at the side $S_{i,1}$ of the triangle cell C_i.

Chapter 3

Numerical Results

In this chapter, we present results of pure electromagnetic as well as particle benchmark calculations computed with the Maxwell and particle solvers, respectively. These results underline the high quality and characteristic properties of the outlined numerical methods.

3.1 Two and Three-Dimensional Electromagnetic Test Problems

3.1.1 Two-dimensional Test Examples

First, let us consider a simple two-dimensional test problem without any symmetry, for which the use of Cartesian coordinates $x = (x_1, x_2) = (x, y)$ is appropriate. The physical domain $\Omega = [0, 1] \times [0, 1]$ consists of the unit square with a length of 1m. It is easy to check that the fields

$$E_1(x, y, t) = -\frac{k_\perp}{k_\parallel} \sin(k_\perp y) \cos(k_\parallel x - \omega t) \,, \tag{3.1a}$$

$$E_2(x, y, t) = \cos(k_\perp y) \sin(k_\parallel x - \omega t) \,, \tag{3.1b}$$

$$B_3(x, y, t) = \frac{\omega}{k_\parallel c^2} \cos(k_\perp y) \sin(k_\parallel x - \omega t) \,, \tag{3.1c}$$

are a set of solutions of the time-dependant Maxwell equations, where the longitudinal and transversal wave numbers $k_\parallel$ and $k_\perp$, respectively, are related to the pulsation ω according to

$$k_\parallel^2 + k_\perp^2 = \frac{\omega^2}{c^2} \,. \tag{3.1d}$$

In order to check simultaneously different kinds of boundary conditions, we limit

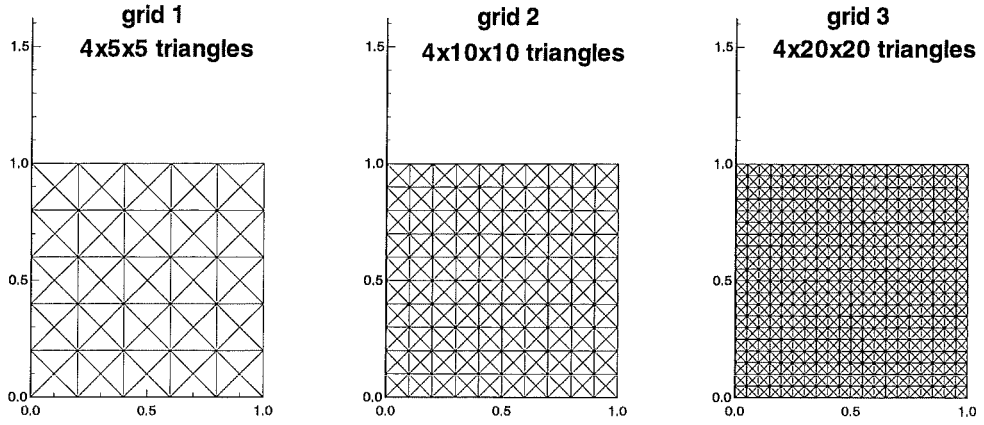

Figure 3.1: Discretization of the computational domain with different fineness in order to study the properties of the FV Maxwell solver.

the computational domain at $y = 0$ and $y = 1\text{m}$ by a perfectly conducting wall, resulting in

$$E_1(x, 0, t) = E_1(x, 1, t) = 0 \; ; \quad \forall (x, t) \; , \tag{3.2a}$$

from which we get the requirement that

$$k_\perp = p\pi \; , p \in \mathbb{Z} \; . \tag{3.2b}$$

Furthermore, we prescribe field boundary conditions at $x = 0$ and $x = 1$ m given by the analytical values of the incoming characteristics

$$E_2(0, y, t) + cB_3(0, y, t) = -\left(1 + \frac{\omega}{k_\parallel c}\right) \cos(k_\perp y) \sin(\omega t) \tag{3.3a}$$

and

$$-E_2(1, y, t) + cB_3(1, y, t) = \left(-1 + \frac{\omega}{k_\parallel c}\right) \cos(k_\perp y) \sin(k_\parallel - \omega t) \; , \tag{3.3b}$$

respectively, computed from the equations (3.1b) and (3.1c). Performing numerical experiments, we explicitly choose $k_\parallel = k_\perp = \pi/\text{m}$ and initialize the fields according to the analytical solution (3.1a)-(3.1c) at time $t = 0$. Afterwards, the Maxwell solver computes the evolution of the fields within the time interval of three periodes $\frac{2\pi}{\omega}$, during which the boundary conditons (3.2a), (3.3a) and (3.3b) are imposed. In order to study experimentally the effective order of the schemes under consideration, we perform the computations on three different grids with

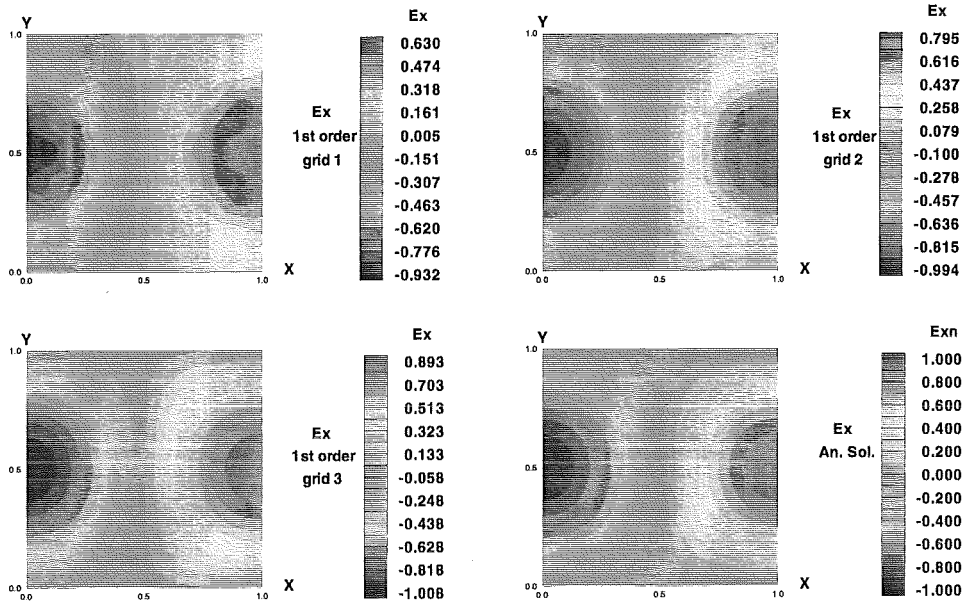

Figure 3.2: Numerical result of the E_1 field computed with the first-order scheme for three different discretizations. The lower right plot shows the analytical solution of this field component.

different finenesses (100, 400 and 1600 triangles, respectively) depicted in Figure 3.1. Obviously from this Figure, the lengths of the sides of the triangles, and hence, the time step size determined by the CFL condition is reduced by a factor two in switching from grid 1 to grid 2 and from grid 2 to grid 3, respectively. The numerical results for E_1 obtained with the first and second-order accurate Maxwell solver for the three different discretizations are shown in the Figures 3.2-3.3. Additionally, the analytical solution is given there for comparison. A closer inspection of these Figures reveals that the numerical solution on the finest mesh computed with the second-order accurate scheme is nearly in perfect agreement with the analytical result. To get a more quantitative picture of the approximation quality of the first and second-order schemes, we compute the relative discrete L^2-error between the analytical and numerical solution. This

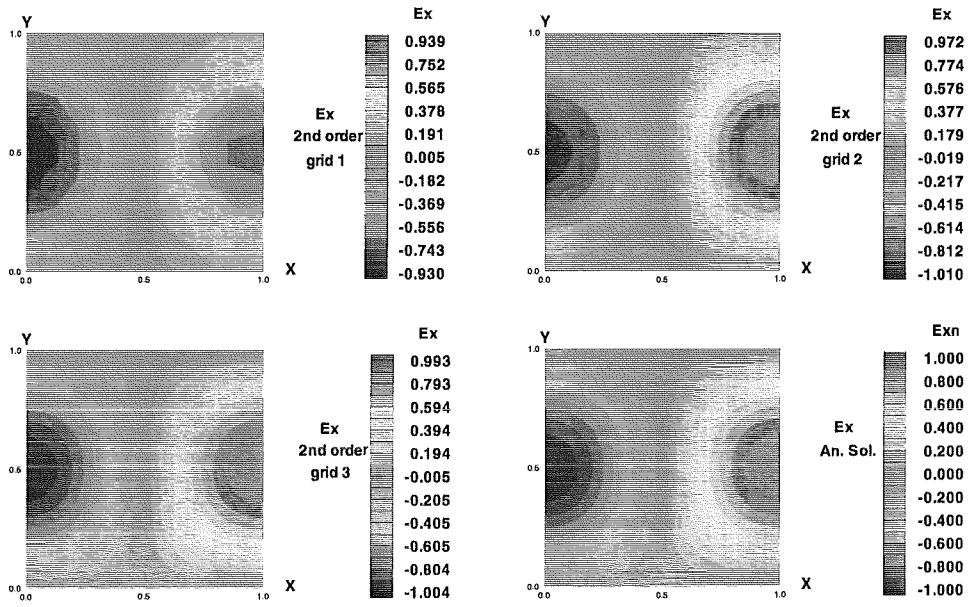

Figure 3.3: Numerical solution of the E_1 field for three different fine grids computed with the second-order scheme. The lower right plot shows the analytical solution of this component.

error norm is defined as

$$\frac{||u^n_{num} - u^n_{ana}||_{L^2}}{||u^n_{ana}||_{L^2}} = \sqrt{\frac{\sum_{i=1}^{N} \left[\left(u^n_{1,i} - E^n_{1,i}\right)^2 + \left(u^n_{2,i} - E^n_{2,i}\right)^2 + c^2 \left(u^n_{6,i} - B^n_{3,i}\right)^2 \right] V_i}{\sum_{i=1}^{N} \left[\left(E^n_{1,i}\right)^2 + \left(E^n_{2,i}\right)^2 + c^2 \left(B^n_{3,i}\right)^2 \right] V_i}}, \tag{3.4}$$

where the analytical solution is computed at the barycenter B_i of the cell C_i at time $t = t^n$, which means for example: $E^n_{1,i} = E_1(B_i, t^n)$. The relative discrete L^2-error for both the first and the second-order accurate schemes computed for the three different grids is plotted with respect to time in Figure 3.4. We verify from these plots the fact that the schemes we use are really of the order one and two: the L^2-error is (approximately) reduced by a factor two for the first-order and by a factor four for the second-order scheme, when switching from grid 1 to grid 2 and from grid 2 to grid 3, respectively.

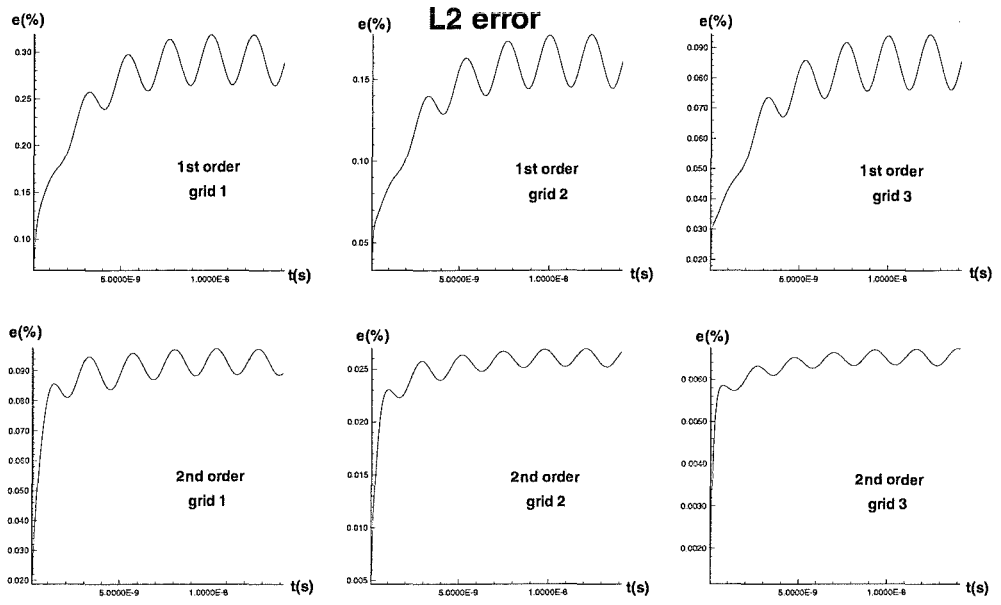

Figure 3.4: The L^2-error computed with the first and second-order accurate schemes for grids with different finenesses.

3.1.2 Cylinder Symmetrical Test Problem

Secondly, we are interested in a cylinder symmetrical problem with respect to the z-axis. For the natural description, it is convenient to use cylindrical coordinates and for the computation, we restrict ourselves to the finite domain $\Omega = [-0.5, 0.5] \times [0, 0.5]$ in the (z, r)-plane. Apart from the rotational axis, the boundaries of Ω are considered to be perfectly conducting walls. The electromagnetic fields are driven by the current density of the form [12]

$$\boldsymbol{j}(z,r,t) = \begin{cases} j_z \boldsymbol{e}_z \,, & \text{if } \; 0 \le t \le \dfrac{2\pi}{\omega} \;\text{ and }\; (z^2+r^2)^{\frac{1}{2}} \le R \\ 0 \,, & \text{otherwise} \end{cases} \,, \tag{3.5a}$$

where j_z is given by

$$j_z(z,r,t) = 1 - \frac{(z^2+r^2)^{\frac{1}{2}}}{R} \sin(\omega t) \tag{3.5b}$$

and $\boldsymbol{e}_z = (1,0)^T$, modelling the radiation of a dipole in an empirical manner. For the simulation, we fixed the parameters ω and R, respectively, equal to $\omega = 5\pi 10^9$ s^{-1} and $R = 0.04$ m. With a time step size of $\Delta t = 5$ ps , we perform 600 iteration cycles to reach the final simulation time of 3 ns . The computational mesh we use is composed of $4 \times 34 \times 68 = 9248$ triangles and shown in Figure 3.5. The numerical values of the B_3 $(= B_\theta)$, E_1 $(= E_z)$ and E_2 $(= E_r)$ fields are

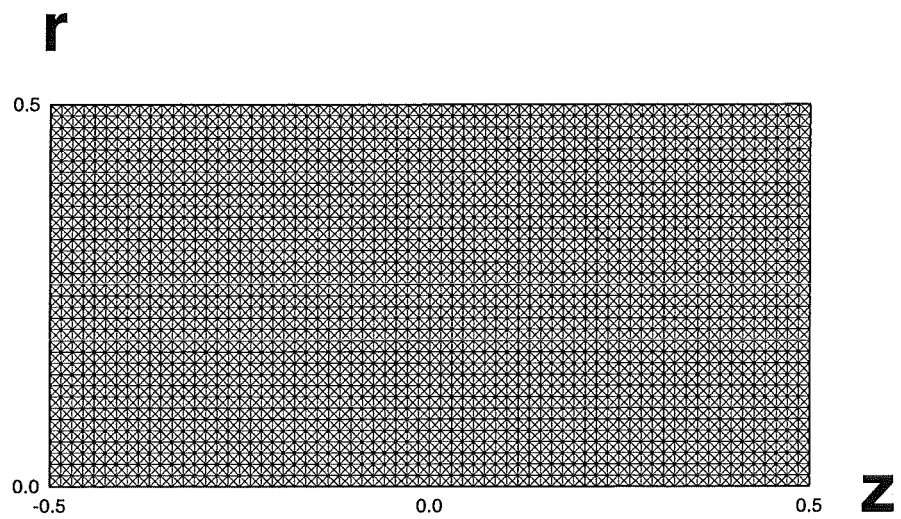

Figure 3.5: Computational mesh used for the dipole radiation test problem.

computed with the second-order accurate Maxwell solver and snapshots recorded at 1 ns, 2 ns and 3 ns are respectively presented in Figure 3.6. The observed structure of the wave propagation matches exactly with the one published by Hermeline [12], who proposed two finite-volume methods on Delaunay-Voronoï meshes for the Maxwell equations in the time domain.

3.1.3 Coaxial Wave Guide in Three Dimensions

As a last example, we consider here a waveguide of length L whose cross-section is a circular crown limited by two circles of radius R_1 and R_2. By z, we denote the common rotational axis of both cylinders, being also the direction of wave propagation. This means, that we are interested in the temporal evolution of a TEM mode, where the non-zero field components are transverse to the z-axis (i.e., $E_z = B_z = 0$). The analytical formulas of the TEM fields are given by

$$\begin{pmatrix} E_x \\ E_y \\ E_z \end{pmatrix} = \begin{pmatrix} \dfrac{x}{x^2+y^2} \\ \dfrac{y}{x^2+y^2} \\ 0 \end{pmatrix} \sin(kz - \omega t) \,, \tag{3.6a}$$

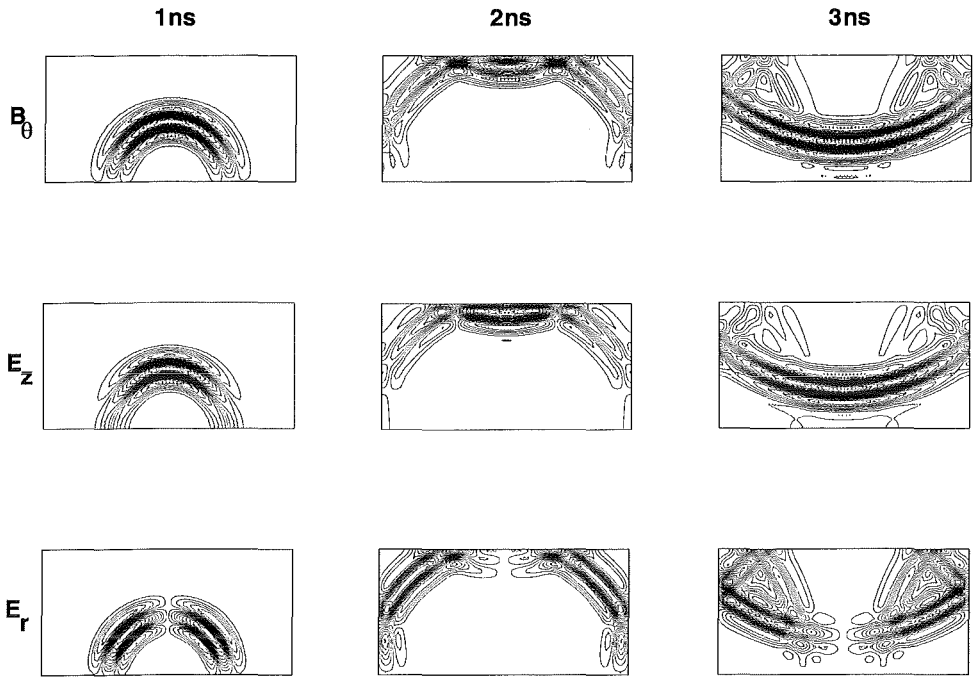

Figure 3.6: Structure of the dipole fields at three different times calculated with the second-order accurate Maxwell solver.

and

$$\begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix} = \frac{1}{c} \begin{pmatrix} \dfrac{-y}{x^2+y^2} \\ \dfrac{x}{x^2+y^2} \\ 0 \end{pmatrix} \sin(kz - \omega t) \,, \tag{3.6b}$$

where the wave number k is related to the pulsation ω according to $\omega = kc$. For the numerical simulation we choose $L = 0.5$ m, $R_1 = 0.5$ m, $R_2 = 0.1$ m and $\omega = \dfrac{c\pi}{L}$ s^{-1}, and initialize the fields according to their analytical values at $t = 0$. The numerical experiments are performed on two different grids composed by 6960 and 54960 tetrahedra, respectively. Applying the CFL restriction on the time step size, an oscillation period is devided into 152 time steps for the coarse and 308 for the fine grid. For both cases, the simulation has been carried out over 2.25 periods. A comparison between the numerically obtained and corresponding exact values of the E_x and E_y field components is presented in the Figures 3.7 and 3.8, respectively. For both computational grids as well as for the first and second-order accurate scheme, we observe a very good agreement between the computed and analytical solution. The numerical result obtained for the E_z field is seen in

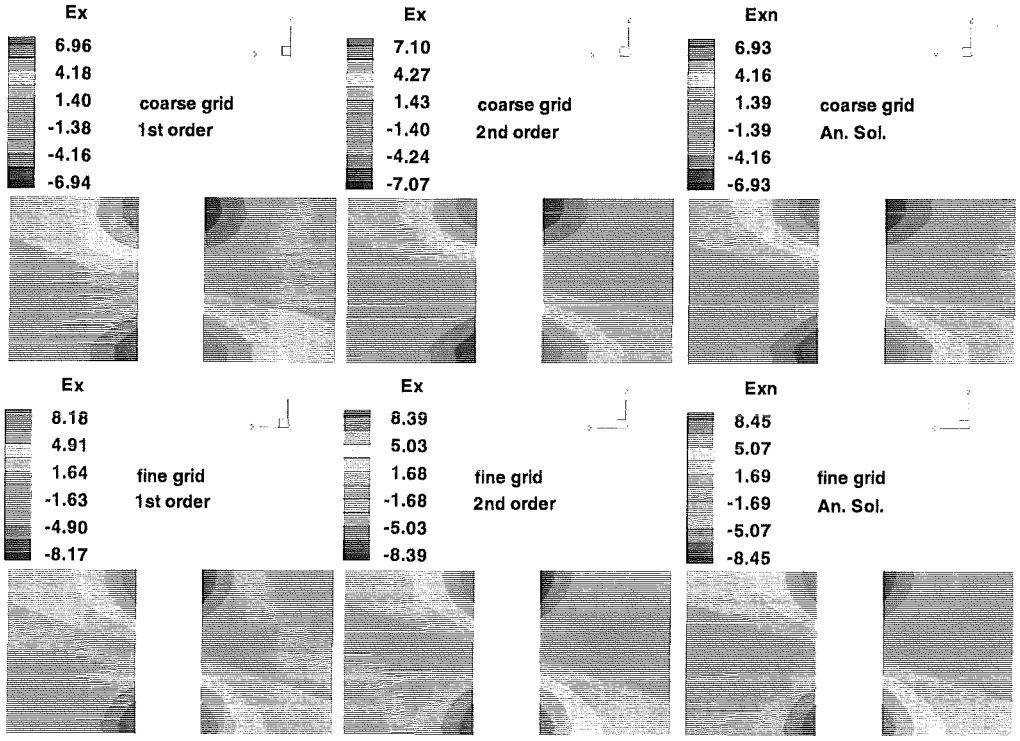

Figure 3.7: E_x field of the coaxial waveguide in the sectional plane $y = 0$.

Figure 3.9. We notice that the biggest deviations from the exact result ($E_z = 0$) are located around the inner circle with the radius R_2. However, this observation can be explained by the fact that the region of strong curvature around the inner circle is not well approximated by the mesh under consideration and, especially there, a refined computational grid seems to be necessary. Furthermore, the discrete L^2-error computed according to

$$\frac{||u^n_{num} - u^n_{ana}||_{L^2}}{||u^n_{ana}||_{L^2}} = \sqrt{\frac{\sum_{i=1}^{N}\left[\sum_{j=1}^{3}\left(\left(u^n_{j,i} - E^n_{j,i}\right)^2 + c^2\left(u^n_{j,i} - B^n_{j,i}\right)^2\right)\right]V_i}{\sum_{i=1}^{N}\left[\sum_{j=1}^{3}\left(\left(E^n_{j,i}\right)^2 + c^2\left(B^n_{j,i}\right)^2\right)\right]V_i}}, \quad (3.7)$$

is depicted in Figure 3.9. In this formula, $E^n_{j,i}$ and $B^n_{j,i}$ are the values of the fields given by (3.6a) and (3.6b) at the barycenter of the cell C_i and at $t = t^n$. This error is approximately reduced by a factor of two when switching from the coarse to the fine grid for the first-order, and by a factor of three for the second-order accuracy, which indicates that the second-order scheme has in fact an effective

order of convergence (EOC) of about 1.6.

For the graphical presentation of the numerical results given in the Figures 3.7 - 3.9 we used the software-tool "TECPLOT". This tool only allows to plot values at the summits of the tetrahedra. However, the numerical solution is defined to be the vector of the average values of the fields over the tetrahedra. Hence, the results seen in the Figures 3.7 - 3.9 are node values, computed as the average of the values of the fields in the cells surrounding the nodes.

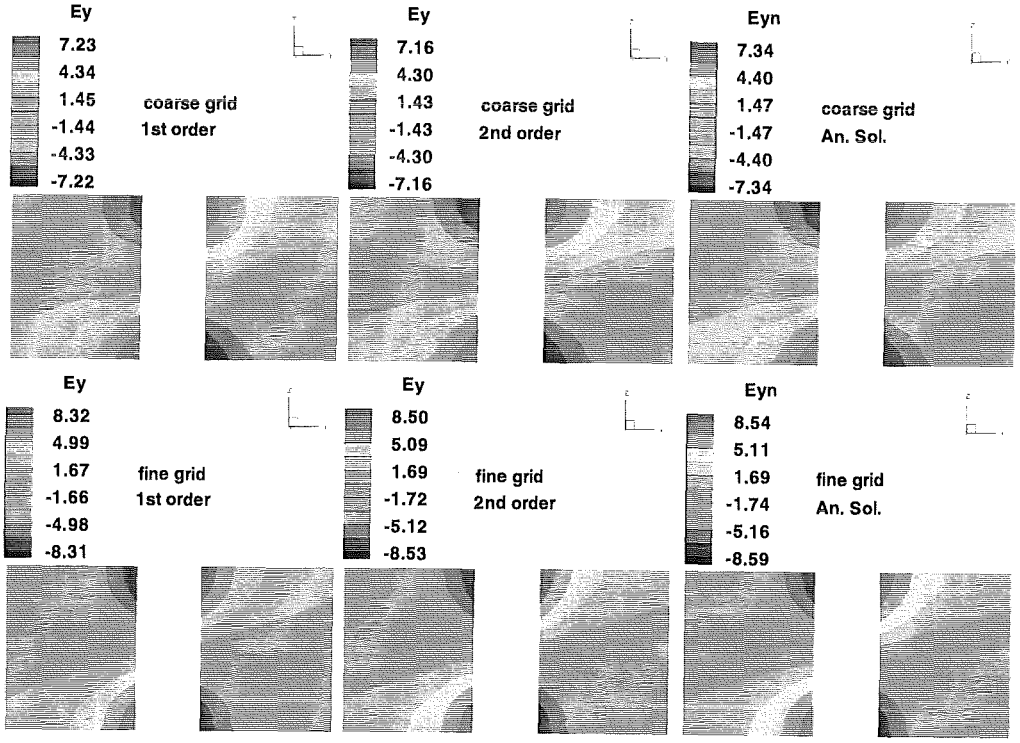

Figure 3.8: E_y field of the coaxial waveguide in the sectional plane $y = 0$.

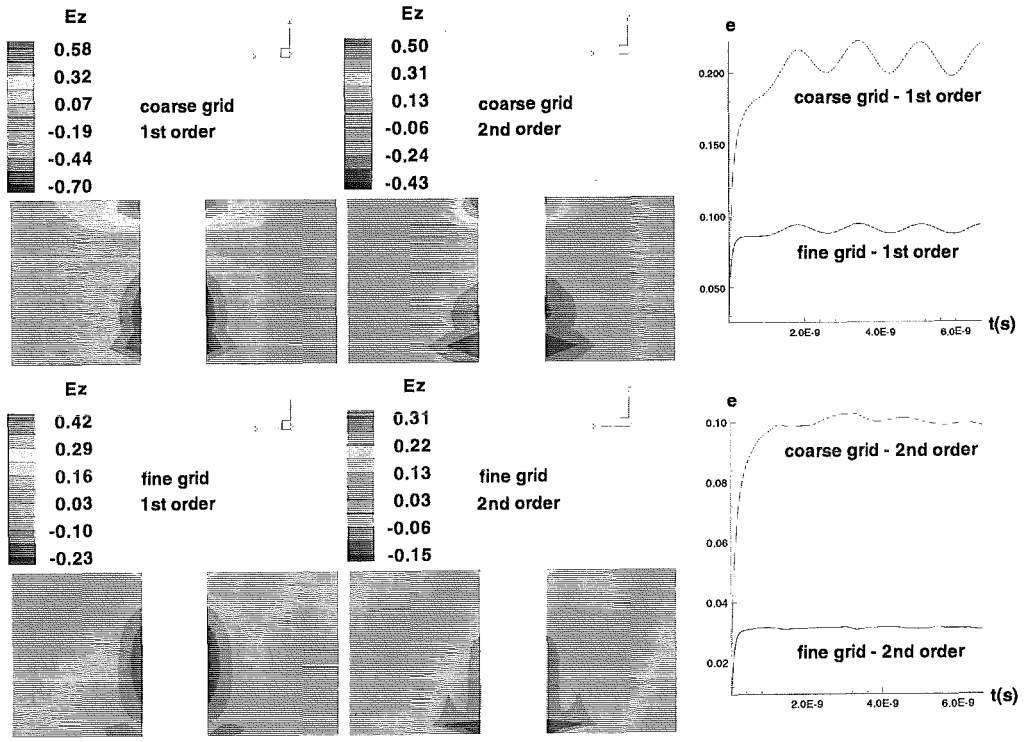

Figure 3.9: E_z field component of the coaxial waveguide and discrete L^2-error defined according to equation (3.7).

3.2 Two and Three-Dimensional Benchmark Problems for Particle Treatment

3.2.1 Localization and Assignment

In the following, we first focus our attention to the localization and assignment procedures in two and three dimensions described in Section 2.3.2 and 2.3.3, respectively. The computational domain consists of a unit square for the two (see Figure 2.2) and a unit cube for the three-dimensional case. The discretization of these domains by an unstructured computational grid is established by 130 and 200 nodes, resulting in 222 triangle and 744 tetrahedron elements, respectively. Within the computational domain the particles are uniformly distributed, which means for instance in two dimensions, that 80 particles are situated per row and column if $N_p = 80 \times 80 = 6400$. Analogous, if 64000 particles are considered in the unit cube, we have a spatial particle discretization of $40 \times 40 \times 40$. Furthermore, each particle carries a charge of $Q_k = \frac{1}{N_p}$ (measured in Coulomb), so that we

obtain for the charge density

$$\rho_0 = \frac{1}{\mathcal{V}} \sum_{k=1}^{N_p} Q_k = 1 \frac{\mathrm{C}}{\mathrm{m}^3}$$

in the unit volume $\mathcal{V}$.

The performances of the numerical experiments for the two and three-dimensional situations are quite similar: In the first step of the calculation, we localize the particles within the computational mesh according to the Löhner or Assous approach (cf. Section 2.3.2). Then, we compute the shape-functions (2.28a) of each particle in order to perform the assignment process and, finally, determine the charge density in the nodes i of the grid (cf. Section 2.3.3) and at the barycenter B_i of the elements C_i, given by the average

$$\rho(B_i, t) = \frac{1}{\sigma_i} \sum_{\alpha=1}^{\sigma_i} \rho(\mathcal{M}_{i,\alpha}, t) \,, \tag{3.8}$$

where $\rho(\mathcal{M}_{i,\alpha}, t)$ is the charge density at the vertices (nodes) $\mathcal{M}_{i,\alpha}$ of C_i. A first

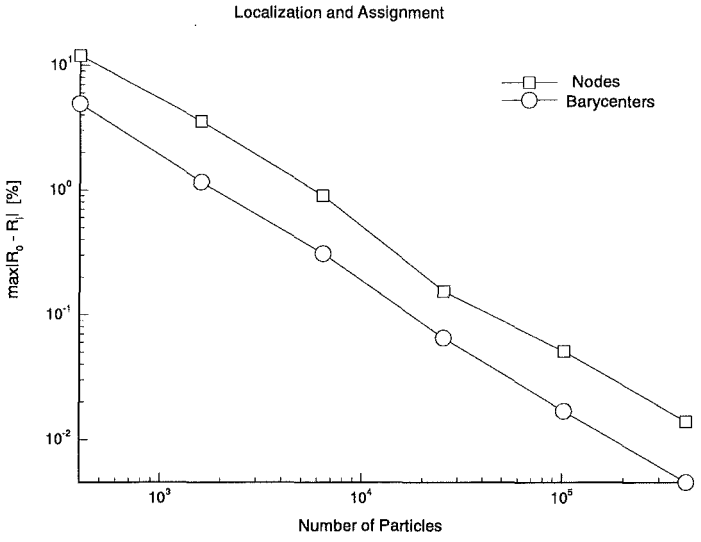

Figure 3.10: Maximum deviation (in per cent) in the nodes (squares) and barycenters (circles) from the charge density $\rho_0 = 1$ versus the number of particles in the computational domain.

result of the two-dimensional localization-assignment procedure is presented in Figure 3.10. There, the maximum of the absolute value of the difference $\rho_0 - \rho_i$

$$\max_{\forall i}\{|\rho_0 - \rho_i|\} = \max_{\forall i}\{|1 - \rho_i|\}$$

is plotted versus the number of particles placed inside the domain ($N_p = 400, 1600, 6400, \ldots, 409600$), where i runs, respectively, over all nodes (indicated by squares in the plot) and barycenter of the elements (indicated by circles). Clearly, the maximum deviation drops below one per cent if more than 6400 particles are inside the domain. Furthermore, we observe that the deviation from the reference value at the barycenters is less pronounced than that at the nodes, being the consequence of the additional averaging given by (3.8). Further re-

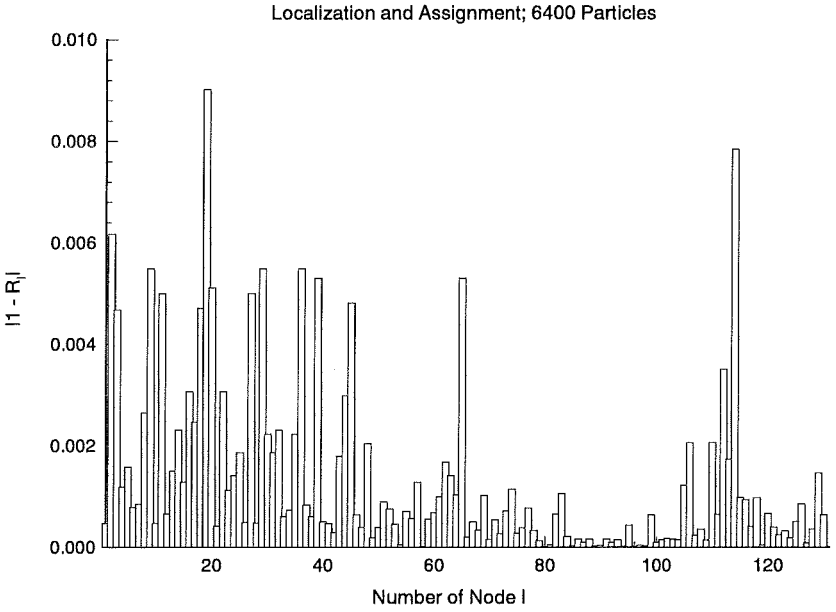

Figure 3.11: Deviation from $\rho_0 = 1$ at the 130 nodes of the computational grid for 6400 particles in two space dimensions.

sults of the localization-assignment part of the particle treatment for the two as well as three-dimensional cases are depicted in the Figures 3.11-3.12 and Figures 3.13-3.14, where 6400 and 64000 particles are distributed in the computational domain, respectively. There, the deviation $\rho_0 - \rho_i$ at the nodes (Figures 3.11 and 3.13) and at the barycenters of the elements (Figures 3.12 and 3.14) are plotted for the two and three-dimensional situations. The plots show that the results for the two-dimensional case are very acceptable: the deviation is always less than one percent. The results for the three-dimensional test case are also satisfactory, but reveal that the discretization of the unit cube with the mesh we use is too coarse.

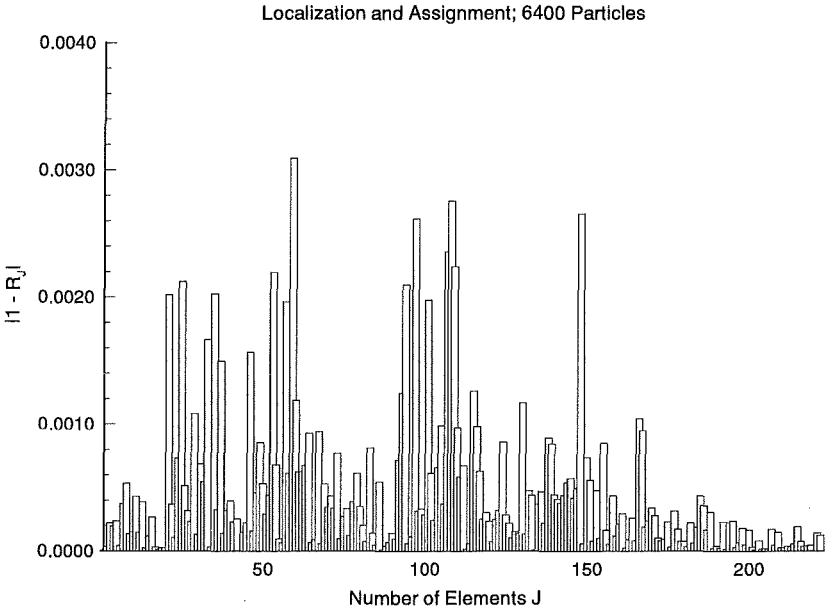

Figure 3.12: Deviation from $\rho_0 = 1$ in the barycenter of the 222 elements of the computational grid for 6400 particles in two space dimensions.

3.2.2 Localization and Interpolation

Now, we consider the localization-interpolation building block of the two and three-dimensional particle treatment. For that purpose, 100 (1000) particles are distributed uniformely in the unit square (cube) in the same manner as already mentioned above. At each particle position $\boldsymbol{x}_k$ the actual value of the externally applied force

$$F(\boldsymbol{x}) = F_0 \sin(\pi \frac{x}{x_0}) \sin(\pi \frac{y}{y_0}) \tag{3.9}$$

is computed, where $x_0 = y_0 = 1\text{m}$ and F_0 is fixed to one Newton. This direct force calculation yields the reference values for the comparison made later on and is denoted by $F_{dir}(\boldsymbol{x}_k)$. Afterwards, the relevant localization procedures for the two or three-dimensional cases are used in order to find the particles with respect to the cells of the computational mesh. Assuming that the kth particle is located at $\boldsymbol{x}_k$ in the mesh zone C_i, then we calculate the shape-function $\mathcal{S}_{i,\alpha}(\boldsymbol{x}_k)$ (cf. (2.28a)) and determine the force (3.9) at the local nodes $\boldsymbol{a}_{i,\alpha}$ of C_i. By applying formula (2.35), the force $F_{lip}(\boldsymbol{x}_k)$ acting at the particle position $\boldsymbol{x}_k$ is obtained. The results of the numerical simulation is seen in Figure 3.15 for the two-dimensional and in Figure 3.16 for the three-dimensional calculations, respectively. There, the directly computed force $F_{dir}(\boldsymbol{x}_k)$ (solid line) and the force determined from the

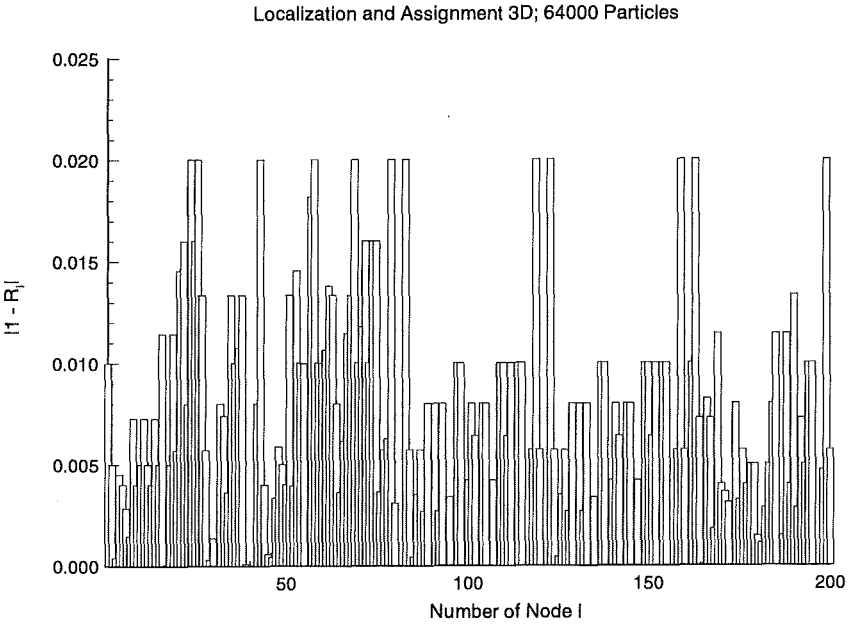

Figure 3.13: Deviation from $\rho_0 = 1$ in the 200 nodes of the computational grid for 64000 particles in three space dimensions.

localization-interpolation process $F_{lip}(\boldsymbol{x}_k)$ (open circles) are depicted as functions of the particle number. The comparison indicates a very good agreement between direct and indirect calculated force values at the particle positions $\boldsymbol{x}_k$.

3.2.3 Interpolation, Particle Pushing and Localization

The following problem is tailored to test the interplay between interpolation, particle pushing and localization. For that, we consider the Lorentz force computed from the externally applied fields

$$E_x(\boldsymbol{x}) = E_{0x} \sin(\pi x) \sin(\pi y) , \tag{3.10a}$$

$$B_z(\boldsymbol{x}) = \frac{E_{0z}}{c} \cos(\pi x) \cos(\pi y) , \tag{3.10b}$$

in two space dimensions, and from

$$E_x(\boldsymbol{x}) = E_{0x} \sin(\pi x) \sin(\pi y) \sin(\pi z) , \tag{3.11a}$$

$$B_z(\boldsymbol{x}) = \frac{E_{0z}}{c} \cos(\pi x) \cos(\pi y) \cos(\pi z) , \tag{3.11b}$$

in the three-dimensional case, where the coordinates x, y and z are normalized to one meter and E_{0z} is fixed equal to 1 V/m. This Lorentz force

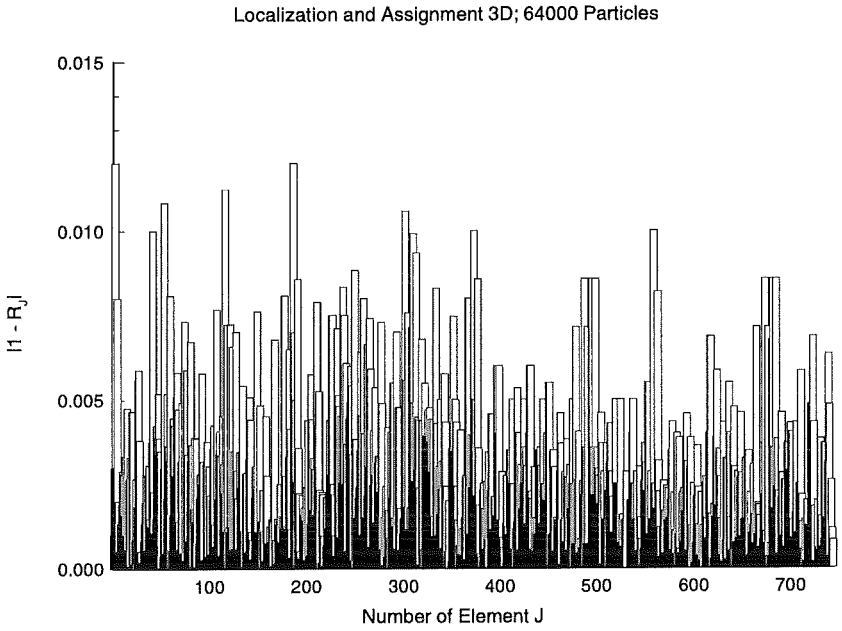

Figure 3.14: Deviation from $\rho_0 = 1$ in the barycenter of the 744 elements of the computational grid for 64000 particles in three space dimensions.

acts on the macro particle (with charge $Q = 1$ Coulomb) initially located at $\boldsymbol{x}(0) = (0.2, 0.5)^T$ in two and $\boldsymbol{x}(0) = (0.5, 0.5, 0.5)^T$ in three dimensions. At these starting points, the intitial velocity (normalized to the speed of light) is chosen to be $\boldsymbol{v}(0) = \left(0.777 \cdot 10^{-2}, 0.202 \cdot 10^{-1}\right)^T$ and $\boldsymbol{v}(0) = \left(0.785 \cdot 10^{-2}, 0.785 \cdot 10^{-2}, 0.395 \cdot 10^{-1}\right)^T$, respectively. To prove the efficiency of the numerical schemes outlined in Section 2.3.1-2.3.3, it is important to notice that for the Lorentz force established by the fields (3.10) or (3.11) an analytical solution of the classical laws of mechanics (1.2) can be found (see, e.g., [15]), yielding the trajectory of the particle in the computational domain.

To perform the numerical simulation, we use the standard discretization of the unit square (130 nodes, 222 elements) and unit cube (200 nodes, 744 elements), and determine the particle position $\boldsymbol{x}(t)$ and velocity $\boldsymbol{v}(t)$ in two and three space dimensions with respect to time. The results of the two-dimensional numerical experiment are presented in Figure 3.17, where 100 temporal iteration cycles with $\Delta t = 0.05$ are performed. This plot demonstrates clearly, that the numerical calculation of the particle position agrees very well with the analytical solution drawn there as solid and dashed lines. The simulation result together with the exact solution for the three-dimensional case are depicted in Figure 3.18. There, we observe discrepancies between the analytical and numerically determined solutions, clearly indicating that the unstructured mesh used is too coarse

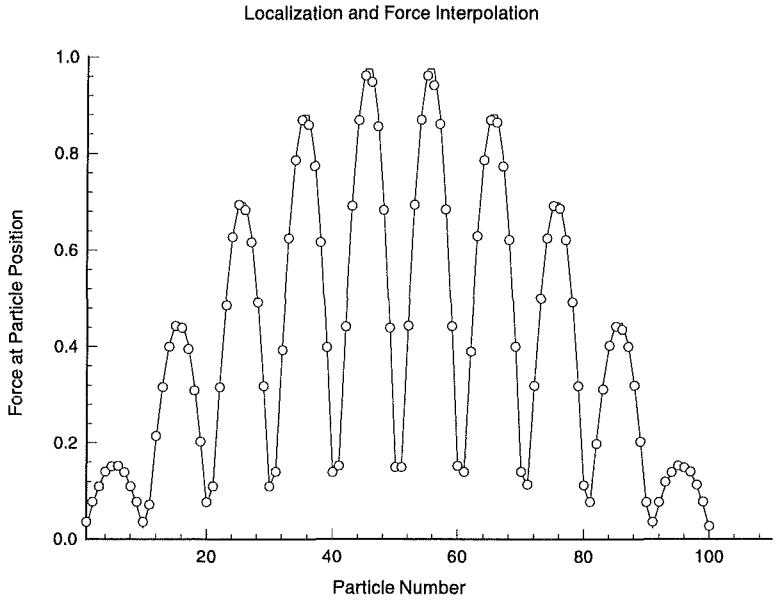

Figure 3.15: Directly calculated $F_{dir}(\boldsymbol{x}_k)$ (solid line) and from the localization-interpolation procedure obtained force $F_{lip}(\boldsymbol{x}_k)$ (circles) as a function of the particle number for the two-dimensional case.

and that a further refinement of the computational grid is necessary for high quality simulations.

In order to get an imagination of the influence of the grid fineness on the discrete solution procedure, we consider a further two-dimensional test problem, solved numerically within the frame of the interpolation, particle pushing and localization building block. This problem deals with the movement of a particle in a central force field given by

$$F_x = \frac{-0.1 r_x}{\left(r_x^2 + r_y^2\right)^{3/2}}, \tag{3.12a}$$

$$F_y = \frac{0.1 r_y}{\left(r_x^2 + r_y^2\right)^{3/2}}, \tag{3.12b}$$

where $r_x = x - 0.5$ and $r_y = y - 0.5$. The dimensionless initial data of the particle are specified according to $x(0) = 0.2$, $y(0) = 0.5$, $v_x(0) = 0$ and $v_y(0) = 1/\sqrt{3}$. For the numerical experiment we use two different computational grids, namely, GRID1 with 130 nodes and 222 elements and GRID2 possessing 701 nodes resulting in 1320 elements. The simulation results for GRID1 and GRID2 together with the exact solution are depicted in Figure 3.19 and 3.20, respectively, where 700 temporal steps are computed. It is clearly visible from Figure 3.19 that

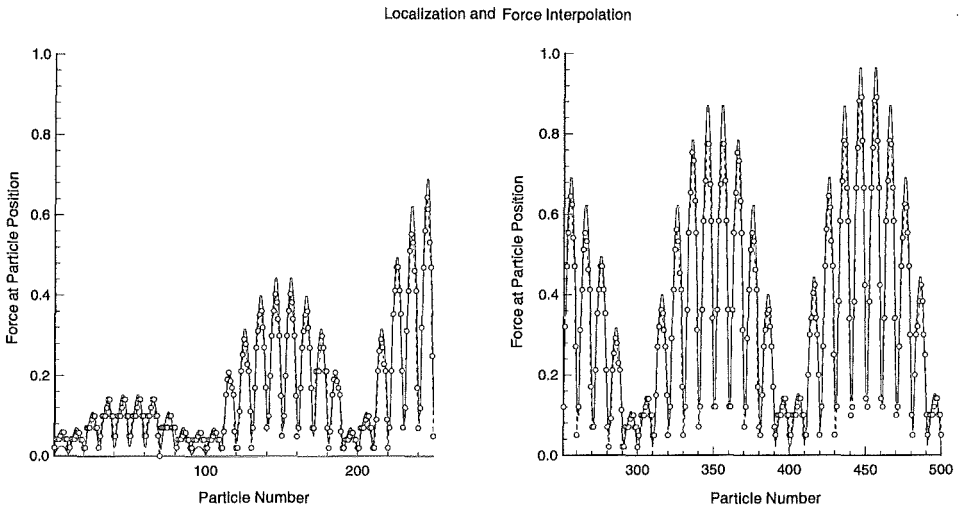

Figure 3.16: Directly calculated $F_{dir}(\boldsymbol{x}_k)$ (solid line) and from the localization-interpolation procedure obtained force $F_{lip}(\boldsymbol{x}_k)$ (circles) as a function of the particle number for the three-dimensional situation.

the trajectory of the particle for the coarse mesh (GRID1) gets unstable due to the inadequate discretization. Moreover, the encircled phase space area oscillate when the particle circles several times around the origin of the central force. This lack is removed if the fine mesh (GRID2) is used for the numerical computation, impressively demonstrated in Figure 3.20, where we recognize a nearly perfect agreement between the numerically obtained and the exact solutions.

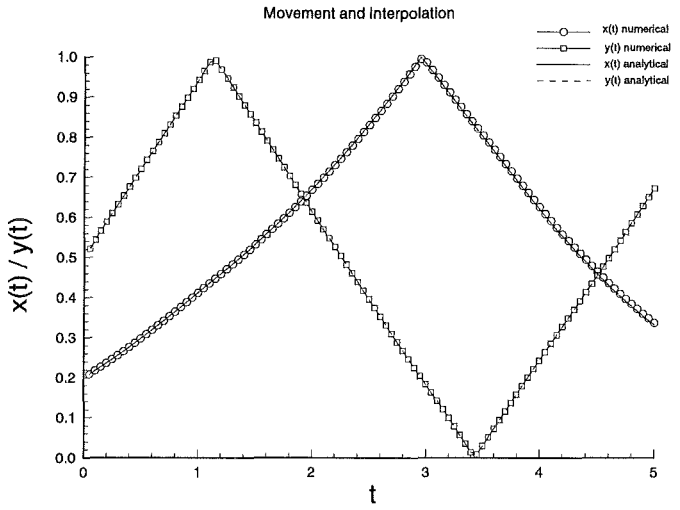

Figure 3.17: Movement of a macro charge in a unit square, where an externally crossed electromagnetic field is applied. The numerical result (open circles and squares) is compared to the analytical solution plotted as solid and dashed lines.

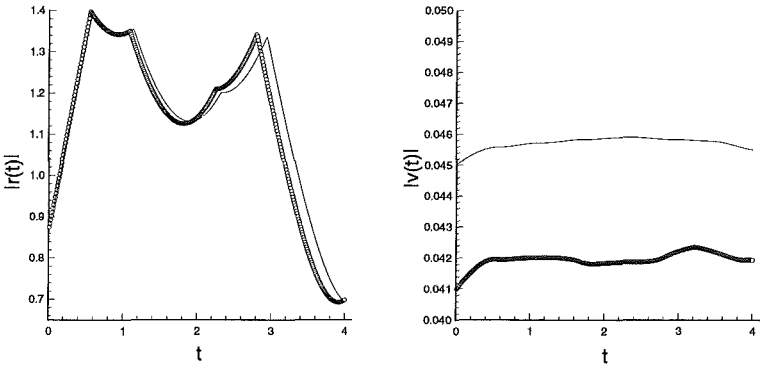

Figure 3.18: Movement of a macro charge in a unit cube, where an externally crossed electromagnetic field is applied. The numerical result (open circles) is compared to the analytical solution plotted as solid lines.

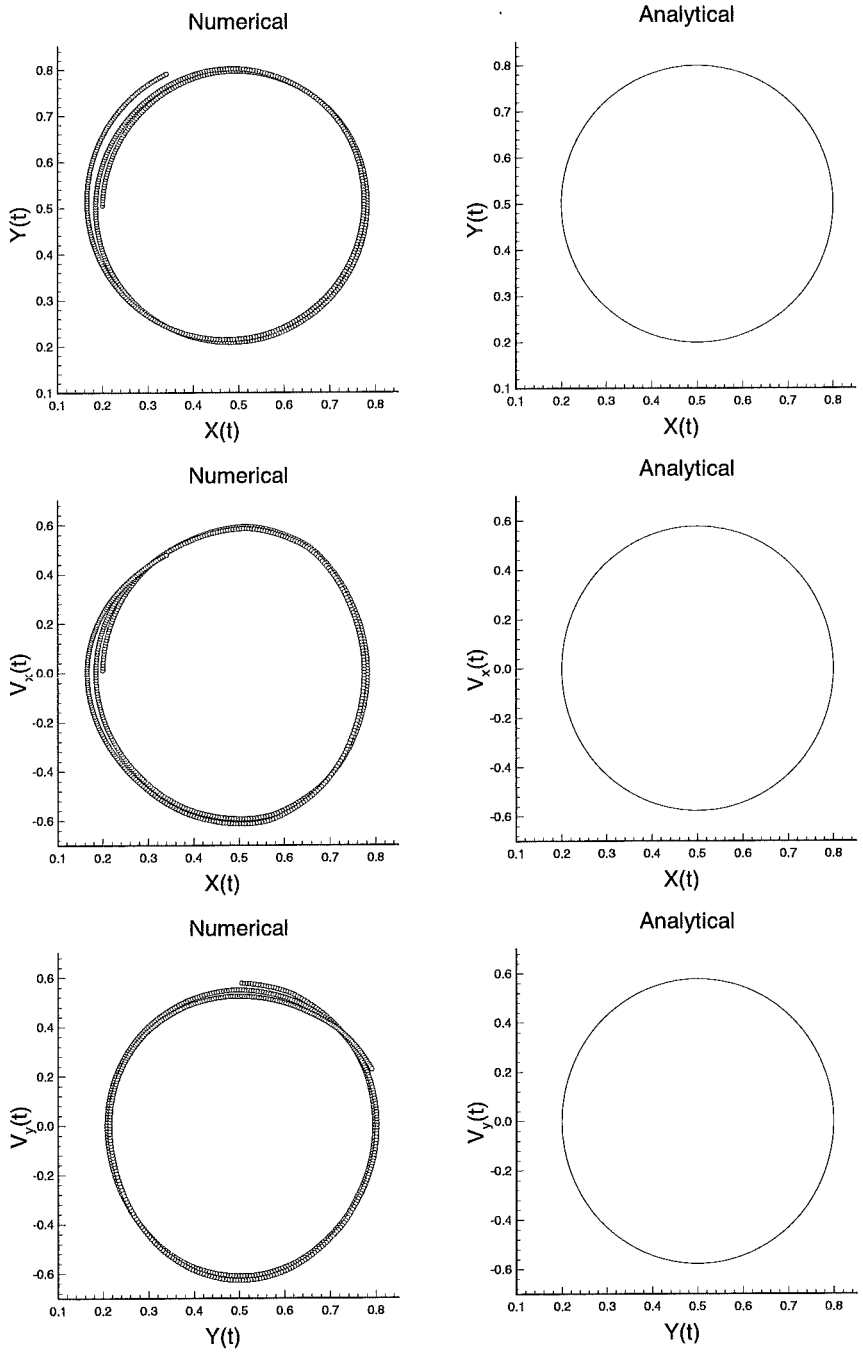

Figure 3.19: Numerically obtained (left) and exact (right) solutions of the central force problem on the coarse computational mesh (GRID1: 130 nodes, 222 elements).

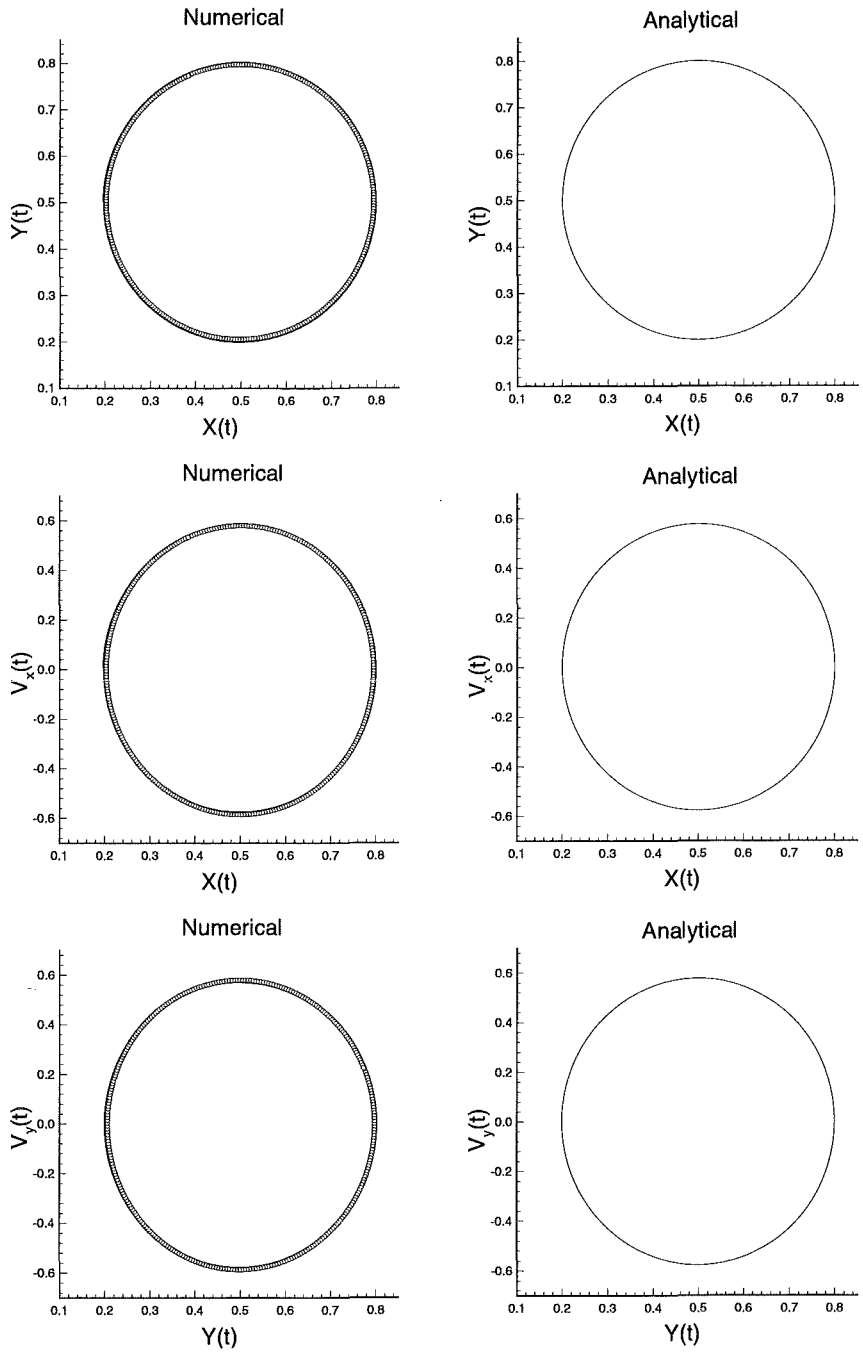

Figure 3.20: Numerically computed (left) and exact (right) solutions of the central force problem on the refined computational grid (GRID2: 701 nodes, 1320 elements).

Chapter 4

Algorithms and Data Structure

Before entering in a more detailed code description of KADI$_3^2$D , we have to mention some more general requirements. Since the code is written in FORTRAN 77, the dimensions of all necessary arrays have to be specified as parameter statements in the main program. Therefore, the following parameter agreements, essential for code running, should be declared in the main program, which subsequently call the subroutines of the Maxwell solver and particle treatment. In the following description, the integer number labeled as **ndim** abbreviates the spatial dimensionality (i.e., two or three) of the problem. In detail, we have to specify the following parameters:

- **melem** : maximum number of elements.
- **mpoin** : maximum number of vertices.
- **mside** : maximum number of sides (faces) if $ndim = 2$ ($ndim = 3$). This value is set equal to **3*melem** ($ndim = 2$) or to **4*melem** ($ndim = 3$).
- **mnelv** : maximum number of elements a given vertex belongs to.
- **mbsi1** : maximum number of boundary sides (faces) of type 1 (being, for instance, a perfect conductor).
- **mbsi2** : maximum number of boundary sides (faces) of type 2.
- **mbsi3** : maximum number of boundary sides (faces) of type 3.
- **mbsi4** : maximum number of boundary sides (faces) of type 4.
- **mptyp** : maximum number of particle species (for instance, $mptyp = 2$ if electrons and ions are considered).
- **mpart** : maximum number of particles within each species.
- **mpato** : maximum total number of particles which is set equal to **mpart * mptyp** .

- **mpodi** : maximum number of mesh points where diagnostic monitoring should be made.

In the **ndim** = 3 case, the considered elements are tetrahedra; in addition to the maximum number of faces and of points already defined, we also need to define maximum values concerning the sides of the elements.

- **maret** : maximum number of sides (if *ndim* = 3).
- **mnela** : maximum number of elements a given side (if *ndim* = 3) belongs to.

The values of these parameters are declared in the file **param.h** and can easily be modified if they are not adequate. However, in this case the code has to be recompiled for further execution.

In the present description of the algorithms, we also define the different arrays used in the code. For that purpose, it is helpful to specify explicitly the dimensions of the arrays according to

- **array**(ndim1,ndim2, ... ,ndimp).

Sometimes it is very convenient to define locally integer numbers representing a given element, a given side, etc.. For that, the following nomenclature is adopted:

- **ielem** : represents the global number of a given element and has a value lying between **1** and **melem**
- **iside** : denotes the global number of a given side and, therefore, is between **1** and **mside**
- **ipoin** : abbreviates the global number of a certain vertex and has a value between **1** and **mpoin**
- **locsi** : is a local number of a side (for *ndim* = 2) or a face (for *ndim* = 3) of a given element. Hence, it ranges from **1** to **3** or from **1** to **4**, respectively.

Moreover, in the following, **nsid (ndim)** denotes the number of sides or faces of the mesh elements, and is equal to three (for *ndim* = 2) or four (for *ndim* = 3).

In the following, we first present some details of the general structure of the particle simulation program KADI$_3^2$D , which is composed of three main parts, namely, the

- Preparatory Step
- Time Loop

- Post-Processing Step.

Afterwards, we discuss in more detail the algorithmical realization of the numerical schemes introduced in Chapter 2. Especially, we go through the different steps of the Maxwell solver, being of central importance for the field computation at each iteration cycle.

4.1 Preparatory Step

In this building block, the following actions are performed:

- Open the files from which the basic information is read (SUBROUTINE **fileop**). These are
  1. the file provided from the mesh-generator: Its name has to be **nopo** and should be located in the current directory.
  2. the command file defined by the user: Its name has to be **data** and should also be located in the current directory. An example of such a command file is given in Appendix E.
- Read these two files with the SUBROUTINE **userpre** and SUBROUTINE **readmsh** .
- Create the files needed for the diagnostic monitoring at different points of the mesh (SUBROUTINE **creadia**).
- Build up the data structure required for the computations on the unstructured grid (SUBROUTINE **indvtoe** and SUBROUTINE **ietsste**).
- Compute all geometric values associated with the grid: volumes of the elements, areas of the faces, lengths of the sides, etc.. This is done in the SUBROUTINE **geomesh** .
- Initialize the field values and the particle coordinates, velocities, localizations and weights within the grid zones (SUBROUTINE **initfld** and SUBROUTINE **intprtc**).

4.2 Time Loop

This is the core block of the Maxwell-Lorentz solver and executes the following actions:

- Carrying out the particle treatment consisting of interpolation, particle pushing, localization and charge assignment (SUBROUTINE **ptreat**).

- Computation of the source terms, depending on the charge and current densities created by the particles, and on possible externally applied sources (SUBROUTINE **compsou**).
- Take into account the contributions of the sources over a half time-step (SUBROUTINE **halfsou**).
- Gradient computation in the case where the second-order accurate scheme is used (SUBROUTINE **inngrad**).
- Computation of the field values at the midpoints of the sides of the elements at time $t = t^{n+1/2}$ in order to solve the Riemann problems. Furthermore, a special treatment of the boundary sides is performed (SUBROUTINE **sideval** and SUBROUTINE **presiva**).
- Determination of the numerical fluxes based on the solution of the Riemann problem (SUBROUTINE **riemann**).
- Computation of the solution of the homogeneous Maxwell equations (SUBROUTINE **intefld**).
- Take into account the contributions of the sources over a further half time-step (SUBROUTINE **halfsou**).
- Generation of the diagnostic files whenever it is desired (SUBROUTINE **creamsh** and SUBROUTINE **diagpoi**).

4.3 Post-Processing

In this last step, the following actions are performed:

- The files are closed (SUBROUTINE **filecl**).
- Whatever for the further post-processing procedure is needed can be achieved subsequently, for instance, writing the results in a suitable plot format.

4.4 Details of the Maxwell Solver

In this section, we describe in greater detail the different computational steps carried out by the Maxwell solver within a temporal iteration cycle.

4.4.1 Computation of the Sources

The sources of the Maxwell equations may be composed by two different contributions: the current density obtained from the charged particle distribution and an externally imposed current density specified by the user. To store the values of these sources we define six arrays according to:

- **sourc**μ (melem) with $\mu = 1, \ldots, 6$: Values of the μth component of the source terms at the (weighted) barycenters of the elements.

Due to the application of a splitting ansatz for the numerical solution of the Maxwell equations (cf. Section 2.2.1), we first have to compute the contribution of the source terms to the electromagnetic fields at the intermediate time $t^{n+1/2} = (n + 1/2)\,\Delta t$.

4.4.2 Computation of the Gradients

Choosing the second-order accurate scheme for the numerical calculation, the piecewise linear reconstruction of the solution requires the computation of the gradients of the fields. For that purpose, the determination of the local gradients in a given cell, based on the values of the fields in the three (or four) neighboring elements of the considered mesh zone is necessary (cf. Section 2.2.4). Hence, we define the following six arrays to store the values of the gradients of the electromagnetic field:

- **xygra**μ (melem,ndim) with $\mu = 1, \ldots, 6$: Values of the $ndim$ u_μ gradient coordinates, where u_μ is the μth component of $u = (E_1, E_2, E_3, B_1, B_2, B_3)^T$.

In order to have access to the values of the fields in the three (or four) neighboring elements, we define the following array:

- **indxee** (melem,nsid(ndim)): Global numbers of the three ($ndim = 2$) or four ($ndim = 3$) neighbors of a given element.

Moreover, the gradients in the inner and boundary cells of the computational domain have to be computed in different ways. To manage this in an efficient way, we introduce

- **intrel** : Number of inner cells
- **gninel** (melem) : Global numbers of these inner cells
- **iselob** : Number of boundary cells.
- **gnbdel** (melem) : Global numbers of these boundary cells.

Finally, the computation of the gradients according to the system (2.18) depends on geometrical factors which remain constant in time. As these factors depend themselves on the locations of the barycenters of the neighboring cells, they are different when we use weighted barycenters ($ndim = 2$ and (z-r) geometry) or when we use non-weighted barycenters ($ndim = 2$ and (x-y) geometry), see Appendix C. In order to make the computation easier, we store these geometrical factors in the array

- **xtabgr** (2,melem,2,2) : if ndim = 2, the first index represents the case of non-weighted barycenters when equal to 1 (x-y geometry), or the case of weighted barycenters when equal to 2 (z-r geometry).
- **xtabgr** (melem,3,3) : if ndim = 3, we only consider non-weighted barycenters.

4.4.3 Field Values at a Common Border of Two Elements

The flux calculation through the sides (faces) of a triangle (tetrahedron) requires the solution of a Riemann problem at adjacent sides (faces) and, therefore, the knowledge of the field values at the midpoint of the common side (face) of two elements (cf. Section 2.2.2). To take this fact into account, we define the following arrays:

- **e1pm** (mside,2) : Left and right E_1 values at the common side (face) of two adjacent elements.
- **e2pm**(mside,2) : Left and right E_2 values at the common side (face) of two adjacent elements.
- **e3pm**(mside,2) : Left and right E_3 values at the common side (face) of two adjacent elements.
- **b1pm**(mside,2) : Left and right B_1 values at the common side (face) of two adjacent elements.
- **b2pm**(mside,2) : Left and right B_2 values at the common side (face) of two adjacent elements.
- **b3pm**(mside,2) : Left and right B_3 values at the common side (face) of two adjacent elements.

To compute these values, it is important to know the two global numbers of the adjacent elements. These numbers are stored in the array

- **indxse** (mside,2) : Global numbers of the first and second element having a common side (face).

In the case of the first-order accurate scheme, the values of the fields on each side (face) of the elements are simply set equal to the values of the fields in each neighboring element. For the second-order scheme it is necessary to compute these values from the field values given at the barycenters, the values of the gradients determined according to the procedure discussed previously, and the coordinates of the vectors joining the barycenters (or the weighted barycenters) of the elements and the midpoints of the sides (or faces). These coordinates are computed once and stored in an array :

- **xybace** (2,mside,ndim,2) : For $ndim = 2$, this array contains the two coordinates of the vectors joining the midpoint of a given side and the non-weighted (resp. weighted) barycenters of the two neighboring elements if the first index is equal to 1 (resp. 2).

- **xybace** (mside,ndim,2) : If ndim = 3, this array gives the 3 coordinates of the vectors joining the midpoint of a given side and the barycenters of the two neighboring elements.

4.4.4 Riemann Problem

The fluxes through the sides (faces) of the mesh zones are computed from the knowledge of the field values of two neighboring elements and the unit normal to the side (cf. Section 2.2.3). By convention, this unit normal is oriented for the global side *iside* from indxse(iside,1) to indxse(iside,2). The *ndim* components of this vector are computed only once and stored in the following array:

- **xynors** (mside,ndim): The *ndim* components of the unit normal to the sides (faces).

Afterwards, the six components of the numerical flux are calculated and stored in the following arrays:

- **flx**μ (mside) with $\mu = 1, \ldots, 6$: Value of the μth component of the numerical flux influencing the quantity u_μ, where $u = (E_1, E_2, E_3, B_1, B_2, B_3)^T$.

4.4.5 Homogeneous Equations

The values of the electromagnetic fields are updated according to the numerical scheme presented previously in Section 2.2.1. For each element and for each of its sides (faces), one must know the global number of this side (face) for which the numerical flux has been computed and stored. These global numbers are stored in the array

- **indxes** (melem,nsid(ndim)) : Global numbers of the three ($ndim = 2$) sides or four ($ndim = 3$) faces of each element.

Furthermore, we calculate the product of the length (area) of this side (face) with the time-step size and divide this product by the area (volume) of the element. If the unit normal at the side (face) under consideration is oriented outwards of the element, then this coefficient is multiplied by 1, otherwise, if the normal is oriented inwards, the coefficient is multiplied by -1. This information is held in the array

- **tlnsvo** (melem,nsid(ndim)) : Value computed for the coefficient given by the time-step size times the length (area) divided by the area (volume) times the orientation (1 or -1) if $ndim = 2$ ($ndim = 3$).

4.4.6 Final Values of the Fields

The second half-contribution of the source terms is taken into account, finally, yielding the solution of the Maxwell equations at the new time level, which concludes the chain of approximations realized by the Maxwell solver.

4.5 Grid Informations and Structure of the Data

Before starting the time loop and the further computations, the program must first read a mesh file provided by the mesh-generator, in order to access to the following information:

- **npoin** : total number of mesh vertices which should be smaller than the maximum number **mpoin** .
- **nelem** : total number of mesh elements which should not exceed the permitted maximum number **melem** .
- **coord** (mpoin,ndim) : table of the coordinates of each vertex given by the two ($ndim = 2$) or three ($ndim = 3$) real numbers.
- **indxev** (melem,nver(ndim)) : table specifying the global numbers of the three ($ndim = 2$) or four ($ndim = 3$) vertices of each element.
- **nrefsi** (melem,nsid(ndim)) : table indicating whether a local side ($ndim = 2$) or face ($ndim = 3$) is :

an inner side (face)	nrefsi(ielem,locsi) = 0
on a conducting boundary	nrefsi(ielem,locsi) = 1
on an open boundary	nrefsi(ielem,locsi) = 2
on a loading boundary	nrefsi(ielem,locsi) = 3
on a symmetry axis	nrefsi(ielem,locsi) = 4.

The reading of this basic information is carried out by the SUBROUTINE **readmsh** .

The following conventions are imposed by the Modulef mesh-generator and should be called to mind if other grid generation modules are used:

1. For a two-dimensional mesh:
   - each triangle is positively oriented.
   - local side 1 is between local vertices 1 and 2.
   - local side 2 is between local vertices 2 and 3.
   - local side 3 is between local vertices 3 and 1.
2. For a three-dimensional mesh:

- each tetrahedron is positively oriented.
- local face 1 is established by the local vertices 1, 2 and 3.
- local face 2 is established by the local vertices 1, 3 and 4.
- local face 3 is established by the local vertices 1, 2 and 4.
- local face 4 is established by the local vertices 2, 3 and 4.

In order to obtain the entire needed information about the mesh structure, the following strategy is adopted: First, two auxiliary arrays are determined, playing also an important role in the context of particle treatment:

1. **nelv** (mpoin) : Number of elements a given mesh vertex belongs to. This number should be smaller than the **mnelv** parameter.
2. **indxve** (mpoin,mnelv) : List of the global element numbers a considered vertex belongs to.

The algorithm to extract these informations is very simple :

- loop1 over the global numbers of the elements (current element: *ielem*).
  - loop2 over the three (four) local vertices of *ielem*:
    * access to the global number (ipoin1) of this local vertex thanks to the table **indxev** .
    * add 1 to **nelv** (ipoin1).
    * ielem is the next element of the **indxve** (ipoin1,*) list : **indxve** (ipoin1,nelv(ipoin1)) = ielem.
  - end of loop2.
- end of loop1.

Secondly, we evaluate at the same time the arrays **indxes** , **indxee** and **indxse** according to the following procedure:

- Loop on the three (four) local sides (faces) of each element.
- If this side (face) has not been taken into account as a global side yet then: add 1 to the number of sides (faces).
  - If this side (face) is an inner side then :
    * Thanks to the mesh-generator conventions and to **indxev** , find out the global numbers of the two (three) local vertices of this local side (face).
    * The intersection of the two (three) lists of global elements these two (three) vertices belong to, is composed of two elements: the current element and its neighbour through the current global side.

    * Then we know for this side (face) the two adjacent elements (table **indxse**),
    * and for these two elements the global number of their common local side (face) (table **indxes**)
    * and the global number of their neighbors through this side (face) (table **indxee**).
- If this side belongs to a boundary of a certain type p then:
    * add 1 to the number of sides (faces) of this boundary type.
    * Then we know the global number of this boundary side (face) (table **indxbs**).
    * this boundary side (face) is associated with the current global element (table **indxbe**).

Chapter 5

Conclusional Remarks and Outlook

In the present report, we describe the basic concepts and approximation techniques to come in useful for the numerical solution of the time-dependent Maxwell-Lorentz equations in two and three space dimensions on unstructured mesh arrangements.

The solution strategy is based on the PIC method: The charged particles are advanced in the continuous computational domain resulting in a changed macro particle distribution. Quantities computed from these redistributed charges are coupled to the grid-based electromagnetic fields according to interpolation and localization procedures for unstructured mesh zones spanning triangles in two and tetrahedra in three space dimensions. The spatial and temporal evolution of the electromagnetic fields on the computational grid is determined by solving numerically the Maxwell equations with a very robust high-resolution FV scheme. The coupling of this FV Maxwell solver with the PIC method for unstructured grids is a new way of approximation in the context of self-consistent particle simulation in electromagnetic fields.

Standard test calculations performed with the Maxwell and particle treatment solvers separately, clearly indicate that the implementation of the resulting algorithms is done properly. Furthermore, the agreement between the simulation results and exact reference solutions is very satisfactory, encouraging and stimulating our computational endeavor.

At the moment only a little experience is available running the entire FV-PIC Maxwell-Lorentz simulation program $\mathsf{KADI}_3^2\mathsf{D}$ in two and three space dimensions on unstructured computational meshes. However, this situation will be changed in the course of this year: Especially, for relevant two-dimensional cylinder symmetrical benchmark problems, a simulation campaign is planned in order to verify

and assess the new KADI$_3^2$D code to the last detail.

It is a well-know fact [3], that the different steps of particle treatment introduce numerical errors and, consequently, charge conservation is not guaranteed on this discrete level of approximation. One way to get out of this numerically caused lack is to use sophisticated charge and current assignment techniques [8, 28] to enforce Gauß's law on the particle level. In this field, we recently proposed a new particle handling approach based on finite-size particle approximations. The description of the resulting numerical schemes as well as the results obtained with the evaluated algorithms will be the item of a forthcoming paper [9].

The other way to enforce the divergence condition starts from a constrained form of the Maxwell equations [1] and approximate this hyperbolic-elliptic system with different kinds of numerical methods [5, 20, 16]. For our activities in this context, we replace the hyperbolic-elliptic by a strictly hyperbolic problem and construct an efficient and very fast high-resolution FV scheme [23], which is successfully applied in the standard KADI2D code. Hence, it is desireable to adapt this hyperbolic correction method also for the simulation program KADI$_3^2$D designed to run on unstructured grid arrangements. One essential advantage of this hyperbolic approach is that the charge correction is nested in an explicit FV scheme which is inherently parallel in nature. This is an important fact, fitting in an excellent manner in our parallelization endeavor of the KADI$_3^2$D program system in order to get a highly efficient production code in near future.

Acknowlegement

It is a pleasure to thank Ursula Voß (Centre de Mathematiques Appliquées, Ecole Polytechnique, Palaiseau) and Eric Sonnendrücker (Université Henri Poincaré Nancy I, Institut Elie Cartan, Nancy) for numerous stimulating and very fruitful discussions and for advice concerning technical details in grid generation with Modulef and testing the Maxwell solver.

Bibliography

[1] F. Assous, P. Degond, E. Heintze, P. Raviart, and J. Segré, *On a finite-element method for solving the three-dimensional Maxwell equations*, J. Comput. Phys., 109 (1993), pp. 222–237.

[2] F. Assous, P. Degond, and J. Segré, *A particle-tracking method for 3D electromagnetic PIC codes on unstructured meshes*, Comput. Phys. Commun., 72 (1992), pp. 105–114.

[3] C. Birdsall and A. Langdon, *Plasma Physics via Computer Simulation*, McGraw-Hill, New York, 1985.

[4] B. Bodin, *Résolution des équations de Maxwell par des méthodes hyperboliques*, Rapport de Stage, Ecole Polytechnique, Palaiseau, (1991).

[5] J. Boris, *Relativistic plasma simulations - Optimization of a hybrid code*, in Proc. 4th Conf. on Num. Sim. of Plasmas, NRL Washington, Washington DC, 1970, pp. 3–67.

[6] J. Cioni, L. Fezoui, and D. Issautier, *High order upwind schemes for solving time-domain Maxwell equations*, La Recherche Aérospatiale, 5 (1994), pp. 319–328.

[7] L. Durlofsky, B. Engquist, and S. Osher, *Triangle based adaptive stencils for the solution of hyperbolic conservation laws*, J. Comput. Phys., 98 (1992), pp. 64–73.

[8] J. Eastwood, *The virtual particle electromagnetic particle-mesh method*, Comp. Phys. Commun., 64 (1991), pp. 252–266.

[9] M. Fedoruk, C.-D. Munz, P. Omnes, D. Romanov, and R. Schneider, *Aspects of charge conservation in electromagnetic particle simulation using a two-dimensional finite-size particle scheme on unstructured grid arrangements*, in Preparation, (1998).

[10] P. L. George, *Modulef: Construction et modification de maillages*, tech. rep., INRIA, Paris, 1990.

[11] E. HALTER, M. KRAUSS, C.-D. MUNZ, R. SCHNEIDER, E. STEIN, U. VOSS, AND T. WESTERMANN, *A concept for the numerical solution of the Maxwell-Vlasov system*, Forschungszentrum Karlsruhe – Technik und Umwelt, **FZKA 5654**, (1995).

[12] F. HERMELINE, *Two Coupled Particle-Finite Volume Methods Using Delaunay-Voronoi Meshes for the Approximation of Vlasov-Poisson and Vlasov-Maxwell equations*, J. Comput. Phys., 106 (1993), pp. 1–18.

[13] R. HOCKNEY AND J. EASTWOOD, *Computer Simulation using Particles*, McGraw-Hill, New York, 1981.

[14] R. HOLLAND, *Finite-difference solution of Maxwell's equations in generalized nonorthogonal coordinates*, IEEE Trans. Nucl. Sci., 30 (1983), p. 4589.

[15] J. JACKSON, *Classical Electrodynamics*, Wiley, New York, 1975.

[16] A. LANGDON, *On enforcing Gauss' law in electromagnetic particle-in-cell codes*, Comput. Phys. Commun., 70 (1992), pp. 447–450.

[17] R. LEVEQUE, *Numerical Methods for Conservation Laws*, Birkhäuser, Basel, 1990.

[18] LÖHNER, *Robust, vectorized search algorithms for interpolation on unstructured grids*, J. Comput. Phys., 118 (1995), pp. 380–387.

[19] R. LÖHNER AND AMBROSIANO, *A vectorized particle tracer for unstructured grids*, J. Comput. Phys., 91 (1990), pp. 22–31.

[20] B. MARDER, *A method incorporating Gauß' law into electromagnetic PIC codes*, J. Comput. Phys., 68 (1987), pp. 48–55.

[21] C.-D. MUNZ, *Theorie und Numerik nichtlinearer hyperbolischer Differentialgleichungen - I. Die skalare Erhaltungsgleichung*, Kernforschungszentrum Karlsruhe GmbH, **KfK 4805**, (1990).

[22] C.-D. MUNZ, R. SCHNEIDER, E. SONNENDRÜCKER, E. STEIN, U. VOSS, AND T. WESTERMANN, *A Finite-Volume Particle-in-Cell Method for the Numerical Treatment of the Maxwell-Lorentz Equations on Boundary-Fitted Meshes*, Int. J. Numer. Meth. Engng., accepted for Publication, (1998).

[23] C.-D. MUNZ, R. SCHNEIDER, AND U. VOSS, *A finite-volume method for Maxwell equations in time domain*, appears in SIAM J. Sci. Comput., (1997).

[24] ——, *A Finite-Volume Particle-in-Cell Method for the Numerical Simulation of Devices in Pulsed Power Technology*, Surveys in Mathematics for Industry; Special Issue: Scientific Computing in Electrical Engineering, accepted for publication, (1998).

[25] A. PETERSON, *Absorbing boundary conditions for the vector wave equation*, Microwave and Opt. Techn. Lett., 1 (1988), pp. 62–64.

[26] J. QUINTENZ, D. SEIDEL, M. KIEFER, T. POINTON, R. COATS, S. ROSENTHAL, T. MEHLHORN, M. DESJARLAIS, AND N. KRALL, *Simulation codes for light-ion diode modeling*, Laser and Particle Beams, 12 (1994), pp. 283–324.

[27] J. VERBONCOEUR, A. LANGDON, AND N. GLADD, *An object-oriented electromagnetic PIC code*, Comput. Phys. Commun., 67 (1995), pp. 199–211.

[28] J. VILLASENOR AND O. BUNEMAN, *Rigorous charge conservation for local electromagnetic field solvers*, Comput. Phys. Commun., 69 (1992), pp. 306–316.

[29] U. VOSS, *Finite-Volumen-Verfahren für die instationären Maxwellgleichungen*, Forschungszentrum Karlsruhe – Technik und Umwelt, **FZKA 5884**, (1997).

[30] T. WESTERMANN, *Teilchenfortbewegung in elektro-magnetischen Feldern*, Kernforschungszentrum Karlsruhe GmbH, **KfK 4325**, (1988).

[31] ——, *Localization schemes in 2D boundary-fitted grids*, J. Comput. Phys., 101 (1992), p. 307.

[32] K. YEE, *Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media*, IEEE Trans. Ant.& Prop., 14 (1966), p. 302.

Appendix A

Miscellaneous Properties of the Matrix used for the FV Approach

In the following we summarize the essential properties of the matrix $\mathcal{A} \in \mathbb{R}^{6\times 6}$ (cf. equation (2.8b)), given by the linear combination

$$\mathcal{A} = \sum_{j=1}^{3} n_j \mathcal{K}_j \ , \tag{1.1a}$$

where the components of the unit normal $n = (n_1, n_2, n_3)^T$, with $n_j \neq 0 \ \forall j$, at the face $S_{i,\alpha}$ fulfil the relation $\sum_{j=1}^{3} n_j^2 = 1$. Using the block-structured matrices $\mathcal{K}_j$ defined by (2.3b), $\mathcal{A}$ is explicitly given by

$$\mathcal{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & n_3c^2 & -n_2c^2 \\ 0 & 0 & 0 & -n_3c^2 & 0 & n_1c^2 \\ 0 & 0 & 0 & n_2c^2 & -n_1c^2 & 0 \\ 0 & -n_3 & n_2 & 0 & 0 & 0 \\ n_3 & 0 & -n_1 & 0 & 0 & 0 \\ -n_2 & n_1 & 0 & 0 & 0 & 0 \end{pmatrix} , \tag{1.1b}$$

where c denotes the velocity of light. The eigenvalues of $\mathcal{A}$ are

$$\Lambda = \mathrm{diag}\,(\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6) = \mathrm{diag}\,(-c, -c, 0, 0, c, c) \ , \tag{1.2a}$$

where Λ is a 6×6 diagonal matrix. Since two eigenvalues coincide, respectively, three classes of waves with different propagating velocities $\lambda_1 = \lambda_2 = -c$, $\lambda_3 = \lambda_4 = 0$ and $\lambda_5 = \lambda_6 = c$ occur. The right and left eigenvectors of $\mathcal{A}$ are the columns and rows of the matrices

$$\mathcal{R} = \begin{pmatrix} 0 & 1 & 0 & \frac{n_1}{n_2} & 1 & 0 \\ \frac{-n_3c}{n_1n_2} & \frac{-(n_1^2+n_3^2)}{n_1n_2} & 0 & 1 & \frac{-(n_1^2+n_3^2)}{n_1n_2} & \frac{n_3c}{n_1n_2} \\ \frac{c}{n_1} & \frac{n_3}{n_1} & 0 & \frac{n_3}{n_2} & \frac{n_3}{n_1} & \frac{-c}{n_1} \\ \frac{-(n_2^2+n_3^2)}{n_1n_2} & \frac{-n_3}{n_1n_2c} & \frac{n_1}{n_3} & 0 & \frac{n_3}{n_1n_2c} & \frac{-(n_2^2+n_3^2)}{n_1n_2} \\ 1 & 0 & \frac{n_2}{n_3} & 0 & 0 & 1 \\ \frac{n_3}{n_2} & \frac{1}{n_2c} & 1 & 0 & \frac{-1}{n_2c} & \frac{n_3}{n_2} \end{pmatrix} , \tag{1.2b}$$

$$\mathcal{R}^{-1} = \begin{pmatrix} \frac{-n_3}{2c} & 0 & \frac{n_1}{2c} & \frac{-n_1n_2}{2} & \frac{n_1^2+n_3^2}{2} & \frac{-n_2n_3}{2} \\ \frac{n_2^2+n_3^2}{2} & \frac{-n_1n_2}{2} & \frac{-n_1n_3}{2} & 0 & \frac{-n_3c}{2} & \frac{n_2c}{2} \\ 0 & 0 & 0 & n_1n_3 & n_2n_3 & n_3^2 \\ n_1n_2 & n_2^2 & n_2n_3 & 0 & 0 & 0 \\ \frac{n_2^2+n_3^2}{2} & \frac{-n_1n_2}{2} & \frac{-n_1n_3}{2} & 0 & \frac{n_3c}{2} & \frac{-n_2c}{2} \\ \frac{n_3}{2c} & 0 & \frac{-n_1}{2c} & \frac{-n_1n_2}{2} & \frac{n_1^2+n_3^2}{2} & \frac{-n_2n_3}{2} \end{pmatrix} , \tag{1.2c}$$

respectively. Since the eigenvalues of the matrix $\mathcal{A}$ are real numbers and the right eigenvectors are linearly independent, the Maxwell equation (2.1) is strictly hyperbolic. With (2b), (2c) and

$$|\Lambda| = \mathrm{diag}\,(c, c, 0, 0, c, c) \ , \tag{1.3a}$$

we obtain

$$|\mathcal{A}| = \mathcal{R}|\Lambda|\mathcal{R}^{-1} = \begin{pmatrix} \mathcal{D} & 0 \\ 0 & \mathcal{D} \end{pmatrix} \tag{1.3b}$$

with

$$\mathcal{D} = \begin{pmatrix} (n_2^2 + n_3^2)c & -n_1 n_2 c & -n_1 n_3 c \\ -n_1 n_2 c & (n_1^2 + n_3^2)c & -n_2 n_3 c \\ -n_1 n_3 c & -n_2 n_3 c & (n_1^2 + n_2^2)c \end{pmatrix} . \tag{1.3c}$$

Finally from (1.1b) and (1.3b) the matrices

$$\mathcal{A}^{\pm} = \frac{1}{2}\left(\mathcal{A} \pm |\mathcal{A}|\right) \tag{1.4a}$$

can be calculated which are given in explicit form according to

$$\mathcal{A}^{\pm} = \frac{1}{2} \begin{pmatrix} \mathcal{D}^{\pm} & c^2\mathcal{E} \\ -\mathcal{E} & \mathcal{D}^{\pm} \end{pmatrix} , \tag{1.4b}$$

where the abbreviations $\mathcal{D}^{\pm} = \pm\mathcal{D}$ and

$$\mathcal{E} = \begin{pmatrix} 0 & n_3 & -n_2 \\ -n_3 & 0 & n_1 \\ n_2 & -n_1 & 0 \end{pmatrix} \tag{1.4c}$$

are used.

Appendix B

Formulas based on the Solution of the Riemann Problem

Applying the right and left eigenvector matrices $\mathcal{R}$ and $\mathcal{R}^{-1}$ of $\mathcal{A}$ given in Appendix A to (2.15), the solution of the Riemann problem (RP) may be written explicitly in the form

$$\begin{aligned} u_{0_1} &= \frac{1}{2}\Big[(n_2^2+n_3^2)(u_1^+ + u_1^-) - n_1n_2(u_2^+ + u_2^-) - n_1n_3(u_3^+ + u_3^-) \\ &\quad +2n_1(n_1\overline{u_1} + n_2\overline{u_2} + n_3\overline{u_3}) \\ &\quad +cn_3(u_5^+ - u_5^-) - cn_2(u_6^+ - u_6^-)\Big], \end{aligned} \tag{2.1a}$$

$$\begin{aligned} u_{0_2} &= \frac{1}{2}\Big[-n_1n_2(u_1^+ + u_1^-) + (n_1^2+n_3^2)(u_2^+ + u_2^-) - n_2n_3(u_3^+ + u_3^-) \\ &\quad +2n_2(n_1\overline{u_1} + n_2\overline{u_2} + n_3\overline{u_3}) \\ &\quad -cn_3(u_4^+ - u_4^-) + cn_1(u_6^+ - u_6^-)\Big], \end{aligned} \tag{2.1b}$$

$$\begin{aligned} u_{0_3} &= \frac{1}{2}\Big[-n_1n_3(u_1^+ + u_1^-) - n_2n_3(u_2^+ + u_2^-) + (n_1^2+n_2^2)(u_3^+ + u_3^-) \\ &\quad +2n_3(n_1\overline{u_1} + n_2\overline{u_2} + n_3\overline{u_3}) \\ &\quad +cn_2(u_4^+ - u_4^-) - cn_1(u_5^+ - u_5^-)\Big], \end{aligned} \tag{2.1c}$$

$$\begin{aligned} u_{0_4} &= \frac{1}{2c}\Big[-n_3(u_2^+ - u_2^-) + n_2(u_3^+ - u_3^-) \\ &\quad +2cn_1(n_1\overline{u_4} + n_2\overline{u_5} + n_3\overline{u_6}) \\ &\quad +c(n_2^2+n_3^2)(u_4^+ + u_4^-) - cn_1n_2(u_5^+ + u_5^-) - cn_1n_3(u_6^+ + u_6^-)\Big], \end{aligned} \tag{2.1d}$$

$$\begin{aligned} u_{0_5} &= \frac{1}{2c}\Big[n_3(u_1^+ - u_1^-) - n_1(u_3^+ - u_3^-) \\ &\quad +2cn_2(n_1\overline{u_4} + n_2\overline{u_5} + n_3\overline{u_6}) \\ &\quad -cn_1n_2(u_4^+ + u_4^-) + c(n_1^2+n_3^2)(u_5^+ + u_5^-) - cn_2n_3(u_6^+ + u_6^-)\Big], \end{aligned} \tag{2.1e}$$

$$\begin{aligned} u_{0_6} &= \frac{1}{2c}\Big[-n_2(u_1^+ - u_1^-) + n_1(u_2^+ - u_2^-) \\ &\quad +2cn_3(n_1\overline{u_4} + n_2\overline{u_5} + n_3\overline{u_6}) \\ &\quad -cn_1n_3(u_4^+ + u_4^-) - cn_2n_3(u_5^+ + u_5^-) + c(n_1^2+n_2^2)(u_6^+ + u_6^-)\Big]. \end{aligned} \tag{2.1f}$$

Another convenient formulation of the solution of the RP at $\xi = 0$ is given by [11, 29]

$$u(0,t) = \frac{1}{2}\left(u^+ + \sum_{j\in J^-}\alpha_j r_j + u^- - \sum_{j\in J^+}\alpha_j r_j\right), \tag{2.2a}$$

where the coefficients $\alpha = (\alpha_1, \ldots, \alpha_6)^T$ are computed from the decomposition of the jump in the initial data into the basis of the right eigenvectors of $\mathcal{A}$:

$$u^- - u^+ = \sum_{j=1}^{6} r_j \alpha_j = \mathcal{R}\alpha \,. \tag{2.2b}$$

Furthermore, $J^{\pm}$ are index sets defined as $J^- = \{j|\lambda_j < 0\}$ and $J^+ = \{j|\lambda_j > 0\}$, respectively. Especially, the numerical flux (2.9) is then calculated from

$$\mathcal{A}\,u(0,t) = \frac{1}{2}\mathcal{A}\left(u^- + u^+\right) + \sum_{j\in J^-} \lambda_j \alpha_j r_j - \sum_{j\in J^+} \lambda_j \alpha_j r_j \,, \tag{2.3a}$$

where the definition of the eigenvalue problem $\mathcal{A}r_j = \lambda_j r_j$ is used. The last equation can be recast in the form

$$\begin{aligned} \mathcal{A}\,u(0,t) &= \frac{1}{2}\left(u^- + u^+\right) - \frac{1}{2}\sum_{j=1}^{6} |\lambda_j| \alpha_j r_j \\ &= \frac{1}{2}\left(u^- + u^+\right) - \frac{1}{2}\mathcal{R}|\Lambda|\mathcal{R}^{-1}\left(u^- - u^+\right) \,, \end{aligned} \tag{2.3b}$$

where the definition (3a) of Appendix A and equation (1b) is applied. With (3c) and (4a) from Appendix A and some rearrangements we finally obtain

$$\mathcal{A}\,u(0,t) = \mathcal{A}^+\,u^+ + \mathcal{A}^-\,u^- \,. \tag{2.3c}$$

Appendix C

Numerical Scheme for the Maxwell Equations in Cylinder Symmetrical Geometry

Assuming that the electromagnetic fields are independent of the variable θ, the homogeneous Maxwell equations in cylinder coordinates may be written as

$$\frac{\partial}{\partial t}(e_z) - c^2 \frac{1}{r}\frac{\partial}{\partial r}(r b_\theta) = 0 \,, \tag{3.1a}$$

$$\frac{\partial}{\partial t}(e_r) + c^2 \frac{\partial}{\partial z}(b_\theta) = 0 \,, \tag{3.1b}$$

$$\frac{\partial}{\partial t}(b_\theta) + \frac{\partial}{\partial z}(e_r) - \frac{\partial}{\partial r}(e_z) = 0 \,, \tag{3.1c}$$

where we restrict ourselves to the TM system (e_z, e_r, b_θ). Introducing new variables according to

$$Q = (d_z, d_r, b_\theta)^T = (r e_z, r e_r, b_\theta)^T \,, \tag{3.2}$$

a new set of equations is obtained:

$$\frac{\partial}{\partial t}(d_z) - c^2 \frac{\partial}{\partial r}(r b_\theta) = 0 \,, \tag{3.3a}$$

$$\frac{\partial}{\partial t}(d_r) + c^2 \frac{\partial}{\partial z}(r b_\theta) = 0 \,, \tag{3.3b}$$

$$\frac{\partial}{\partial t}(b_\theta) + \frac{\partial}{\partial z}\left(\frac{1}{r} d_r\right) - \frac{\partial}{\partial r}\left(\frac{1}{r} d_z\right) = 0 \,. \tag{3.3c}$$

This system can be recast in conservation form

$$\frac{\partial}{\partial t}(Q) + \frac{\partial}{\partial z}(AQ) + \frac{\partial}{\partial r}(BQ) = 0 \,, \tag{3.4a}$$

where the matrices $A, B \in \mathbb{R}^{3\times 3}$ are given by

$$A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & c^2 r \\ 0 & \frac{1}{r} & 0 \end{pmatrix} \,, \quad B = \begin{pmatrix} 0 & 0 & -c^2 r \\ 0 & 0 & 0 \\ -\frac{1}{r} & 0 & 0 \end{pmatrix} \,. \tag{3.4b}$$

The integration of (3.4a) over a grid cell T_i whose sides are denoted by $S_{i,\alpha}$, leads to the following exact equation:

$$\int_{T_i} \frac{\partial}{\partial t}(Q)\, dS + \sum_\alpha \int_{S_{i,\alpha}} C_{i,\alpha} Q \, dl = 0 \,, \tag{3.5}$$

where no approximations are made. Here, the matrix $C_{i,\alpha} \in \mathbb{R}^{3\times 3}$ is defined by

$$C_{i,\alpha} = \begin{pmatrix} 0 & 0 & -c^2 r n_r \\ 0 & 0 & c^2 r n_z \\ -\dfrac{n_r}{r} & \dfrac{n_z}{r} & 0 \end{pmatrix} , \tag{3.6}$$

and n_z, n_r are the components of the outwards directed normal vector $n_{i,\alpha} = (n_z, n_r)^T$ at $S_{i,\alpha}$. Integrating (3.5) over the time interval $[n\Delta t, (n+1)\Delta t]$, applying the midpoint rule and denoting the cell average by

$$Q_i^n = \frac{1}{|T_i|} \int_{T_i} Q \, dS ,$$

we obtain the explicit scheme

$$Q_i^{n+1} \approx Q_i^n - \frac{\Delta t}{|T_i|} \sum_{\alpha} L_{i,\alpha} \, C_{i,\alpha} \, Q\left(M_{i,\alpha}, t^{n+1/2}\right) , \tag{3.7}$$

where $t^{n+1/2} = \left(n + \frac{1}{2}\right)\Delta t$, $M_{i,\alpha}$ is the midpoint of side $S_{i,\alpha}$ and $|T_i|$ is the area of the cell T_i. Assuming that Q is constant during the time step size Δt, we approximate $Q\left(M_{i,\alpha}, t^{n+1/2}\right)$ at the midpoint by the solution of the Riemann problem (RP) at $\xi = 0$:

$$\begin{aligned} &\frac{\partial}{\partial t}(Q) + \frac{\partial}{\partial \xi}(G_{i,\alpha} Q) = 0 , \\ &Q(\xi, 0) = \begin{cases} Q^+ & \text{if } \xi < 0 \\ Q^- & \text{if } \xi > 0 \end{cases} , \end{aligned} \tag{3.8a}$$

where Q^+ and Q^- are approximated values of $Q\left(M_{i,\alpha}, t^{n+1/2}\right)$ on each neighboring element to $S_{i,\alpha}$ and calculated according to

$$Q^+ = \lim_{\xi \to 0^-} Q\left(M_{i,\alpha} + \xi n_{i,\alpha}, \, t^{n+1/2}\right) , \tag{3.8b}$$

$$Q^- = \lim_{\xi \to 0^+} Q\left(M_{i,\alpha} + \xi n_{i,\alpha}, \, t^{n+1/2}\right) . \tag{3.8c}$$

Furthermore, the matrix $G_{i,\alpha} \in \mathbb{R}^{3\times 3}$ is similar to the one defined by (3.6), but now evaluated at $r_{0i,\alpha}$, the r-coordinate of the midpoint $M_{i,\alpha}$

$$G_{i,\alpha} = \begin{pmatrix} 0 & 0 & -c^2 r_{0i,\alpha} n_r \\ 0 & 0 & c^2 r_{0i,\alpha} n_z \\ -\dfrac{n_r}{r_{0i,\alpha}} & \dfrac{n_z}{r_{0i,\alpha}} & 0 \end{pmatrix} . \tag{3.8d}$$

Since $G_{i,\alpha}$ is a constant matrix, the solution of the Riemann problem (3.8a) is easy to find and given by

$$\begin{pmatrix} d_{Rz} \\ d_{Rr} \\ b_{R\theta} \end{pmatrix} = \begin{pmatrix} cn_r r_{0i,\alpha}(p_2 - p_3) \\ -cn_z r_{0i,\alpha}(p_2 - p_3) \\ p_2 + p_3 \end{pmatrix} \tag{3.9a}$$

where

$$\begin{pmatrix} p_2 \\ \\ p_3 \end{pmatrix} = \begin{pmatrix} \dfrac{d_z^-}{r_{0i,\alpha}}\dfrac{n_r}{2c} - \dfrac{d_r^-}{r_{0i,\alpha}}\dfrac{n_z}{2c} + \dfrac{b_\theta^-}{2} \\ \\ -\dfrac{d_z^+}{r_{0i,\alpha}}\dfrac{n_r}{2c} + \dfrac{d_r^+}{r_{0i,\alpha}}\dfrac{n_z}{2c} + \dfrac{b_\theta^+}{2} \end{pmatrix}, \tag{3.9b}$$

and where we have omitted the dependencies with respect to $\overline{Q} = Q(0,0)$, because, as before, the final result does not depend on $\overline{Q}$. With that result, the numerical scheme (3.7) can be written in the following way

$$\left(\frac{d_{zi}}{r_i}\right)^{n+1} = \left(\frac{d_{zi}}{r_i}\right)^{n} - \frac{\Delta t}{r_i|T_i|}\sum_\alpha L_{i,\alpha}[-c^2 n_r r_{0i,\alpha}(p_2 + p_3)] , \tag{3.10a}$$

$$\left(\frac{d_{ri}}{r_i}\right)^{n+1} = \left(\frac{d_{ri}}{r_i}\right)^{n} - \frac{\Delta t}{r_i|T_i|}\sum_\alpha L_{i,\alpha}[c^2 n_z r_{0i,\alpha}(p_2 + p_3)] , \tag{3.10b}$$

$$b_{\theta i}^{n+1} = b_{\theta i}^{n} - \frac{\Delta t}{|T_i|}\sum_\alpha L_{i,\alpha}[-c(p_2 - p_3)] , \tag{3.10c}$$

where r_i is the r-coordinate of the barycenter of the cell T_i. Replacing in (3.9b)

$$\frac{d_z^-}{r_{0i,\alpha}} \text{ by } e_z^- , \quad \frac{d_z^+}{r_{0i,\alpha}} \text{ by } e_z^+ ,$$

$$\frac{d_r^-}{r_{0i,\alpha}} \text{ by } e_r^- , \quad \frac{d_r^+}{r_{0i,\alpha}} \text{ by } e_r^+ ,$$

and renaming in (3.10a)-(3.10c) $\dfrac{d_{zi}}{r_i}$ by $\tilde{e}_{zi}$ and $\dfrac{d_{ri}}{r_i}$ by $\tilde{e}_{ri}$, the numerical scheme (3.7) can finally be brought into the form

$$\begin{pmatrix} \tilde{e}_{zi}^{n+1} \\ \tilde{e}_{ri}^{n+1} \\ b_{\theta i}^{n+1} \end{pmatrix} = \begin{pmatrix} \tilde{e}_{zi}^{n} \\ \tilde{e}_{ri}^{n} \\ b_{\theta i}^{n} \end{pmatrix} - \frac{\Delta t}{|T_i|}\sum_\alpha L_{i,\alpha}\left[\mathcal{M}_{i,\alpha}^+ \begin{pmatrix} e_{zi,\alpha}^+ \\ e_{ri,\alpha}^+ \\ b_{\theta i,\alpha}^+ \end{pmatrix} + \mathcal{M}_{i,\alpha}^- \begin{pmatrix} e_{zi,\alpha}^- \\ e_{ri,\alpha}^- \\ b_{\theta i,\alpha}^- \end{pmatrix}\right] , \tag{3.11a}$$

with

$$\mathcal{M}^{\pm}_{i,\alpha} = \frac{1}{2}\begin{pmatrix} \pm c\frac{r_{0i,\alpha}}{r_i}n_r^2 & \mp c\frac{r_{0i,\alpha}}{r_i}n_z n_r & -c^2\frac{r_{0i,\alpha}}{r_i}n_r \\ \mp c\frac{r_{0i,\alpha}}{r_i}n_z n_r & \pm c\frac{r_{0i,\alpha}}{r_i}n_z^2 & c^2\frac{r_{0i,\alpha}}{r_i}n_z \\ -n_r & n_z & \pm c \end{pmatrix} . \tag{3.11b}$$

All what remains to do now, is to specify adequate first (resp. second) order approximations of $e^{\pm}_{zi,\alpha}$, $e^{\pm}_{ri,\alpha}$ and $b^{\pm}_{\theta i,\alpha}$ in order to complete the formulation of our first (resp. second) order accurate numerical scheme. However, for that purpose, we will first prove the following lemma:

Lemma 1 *Let β be in $\{0;1\}$, then for any regular function $u(z,r)$ we have :*

$$\frac{\int_{T_i} r^{\beta} u(z,r)dzdr}{r_i^{\beta}|T_i|} = u(z_{A_{i\beta}}, r_{A_{i\beta}}) + O(\delta^2) ,$$

where $A_{i\beta}$ is the point given by the coordinates $(z_{A_{i\beta}}, r_{A_{i\beta}})$ defined according to:

$$z_{A_{i\beta}} = \frac{\int_{T_i} r^{\beta} z dzdr}{r_i^{\beta}|T_i|} , \; r_{A_{i\beta}} = \frac{\int_{T_i} r^{1+\beta} dzdr}{r_i^{\beta}|T_i|} ,$$

and where δ is a measure of the space increment of the grid.

Proof : Since $u(z,r)$ is regular, we can expand u in a Taylor series around $A_{i\beta}$ yielding

$$\begin{aligned} &\int_{T_i} r^{\beta} u(z,r)dzdr = \\ &\left[\int_{T_i} r^{\beta} dzdr\right] u(z_{A_{i\beta}}, r_{A_{i\beta}}) + \left[\int_{T_i} r^{\beta}(z - z_{A_{i\beta}})dzdr\right] \frac{\partial u}{\partial z}(z_{A_{i\beta}}, r_{A_{i\beta}}) + \\ &\qquad \left[\int_{T_i} r^{\beta}(r - r_{A_{i\beta}})dzdr\right] \frac{\partial u}{\partial r}(z_{A_{i\beta}}, r_{A_{i\beta}}) + \int_{T_i} r^{\beta} O(\delta^2) dzdr . \end{aligned}$$

Because r^β is a linear function, (we recall that $\beta \in \{0;1\}$) we have : $\displaystyle\int_{T_i} r^\beta dzdr = r_i^\beta |T_i|$, and thus we get:

$$\frac{\displaystyle\int_{T_i} r^\beta u(z,r)dzdr}{r_i^\beta |T_i|} =$$

$$u(z_{A_{i\beta}}, r_{A_{i\beta}}) + \left(\frac{\displaystyle\int_{T_i} r^\beta z dzdr}{r_i^\beta |T_i|} - z_{A_{i\beta}} \right) \frac{\partial u}{\partial z}(z_{A_{i\beta}}, r_{A_{i\beta}}) +$$

$$\left(\frac{\displaystyle\int_{T_i} r^{1+\beta} dzdr}{r_i^\beta |T_i|} - r_{A_{i\beta}} \right) \frac{\partial u}{\partial r}(z_{A_{i\beta}}, r_{A_{i\beta}}) + O(\delta^2) ,$$

which proves the result.

From this lemma, we first conclude that

$$\tilde{e}_{zi}^n = e_z(A_{i1}, t^n) + O(\delta^2) , \ \tilde{e}_{ri}^n = e_r(A_{i1}, t^n) + O(\delta^2) , \tag{3.3a}$$

and

$$b_{\theta i}^n = b_\theta(A_{i0}, t^n) + O(\delta^2) . \tag{3.3b}$$

Moreover, we approximate the field values at the midpoint $M_{i,\alpha}$ of the side $S_{i,\alpha}$ at time $t = t^{n+1/2}$ according to

$$\begin{aligned} e_z(M_{i,\alpha}, t^{n+1/2}) = e_z(A_{i1}, t^n) + \overrightarrow{A_{i1}M_{i,\alpha}} \cdot \nabla e_z(A_{i1}, t^n) + \\ \frac{\Delta t}{2} \frac{\partial e_z}{\partial t}(A_{i1}, t^n) + O(\Delta t^2) + O(\delta^2) + O(\delta \Delta t) , \end{aligned} \tag{3.4a}$$

$$\begin{aligned} e_r(M_{i,\alpha}, t^{n+1/2}) = e_r(A_{i1}, t^n) + \overrightarrow{A_{i1}M_{i,\alpha}} \cdot \nabla e_r(A_{i1}, t^n) + \\ \frac{\Delta t}{2} \frac{\partial e_r}{\partial t}(A_{i1}, t^n) + O(\Delta t^2) + O(\delta^2) + O(\delta \Delta t) , \end{aligned} \tag{3.4b}$$

$$\begin{aligned} b_\theta(M_{i,\alpha}, t^{n+1/2}) = b_\theta(A_{i0}, t^n) + \overrightarrow{A_{i0}M_{i,\alpha}} \cdot \nabla b_\theta(A_{i0}, t^n) + \\ \frac{\Delta t}{2} \frac{\partial b_\theta}{\partial t}(A_{i0}, t^n) + O(\Delta t^2) + O(\delta^2) + O(\delta \Delta t) , \end{aligned} \tag{3.4c}$$

In order to complete the numerical scheme, we have to propose appropriate approximations of the three gradients and derivatives with respect to time appearing

in (3.4a) - (3.4c). Similar to section 2.2.4, we consider an inner cell T_i of the computational domain and its three neighbor cells T_{i_α} with $\alpha \in \{1;2;3\}$. For any regular function u, we define the vector $w \in \mathbb{R}^2$ by:

$$\begin{cases} u(A_{i_1\beta}) = u(A_{i_2\beta}) + \overrightarrow{A_{i_2\beta}A_{i_1\beta}} \cdot w \\ u(A_{i_1\beta}) = u(A_{i_3\beta}) + \overrightarrow{A_{i_3\beta}A_{i_1\beta}} \cdot w \end{cases} , \tag{3.5}$$

being a first-order accurate approximation of the gradient of the function u: $w = \widetilde{\nabla} u + O(\delta)$. Finally, the time derivatives occuring in (3.4a) - (3.4c) are simply shifted to derivatives of the fields with respect to space by using the three original equations (3.1a) - (3.1c). Now, the chain of approximations is closed and the values $e^{\pm}_{zi,\alpha}$, $e^{\pm}_{ri,\alpha}$ and $b^{\pm}_{\theta i,\alpha}$ determined by

$$\begin{array}{rclrcl} e^{+}_{zi,\alpha} & = & \tilde{e}^{n}_{zi} , & e^{-}_{zi,\alpha} & = & \tilde{e}^{n}_{zi_\alpha} , \\ e^{+}_{ri,\alpha} & = & \tilde{e}^{n}_{ri} , & e^{-}_{ri,\alpha} & = & \tilde{e}^{n}_{ri_\alpha} , \\ b^{+}_{\theta i,\alpha} & = & b^{n}_{\theta i} , & b^{-}_{\theta i,\alpha} & = & b^{n}_{\theta i_\alpha} , \end{array} \tag{3.6}$$

in case of a first order accurate scheme, and computed from

$$\begin{aligned} e^{+}_{zi,\alpha} &= \tilde{e}^{n}_{zi} + \overrightarrow{A_{i1}M_{i,\alpha}} \cdot \widetilde{\nabla} e^{n}_{zi} + \frac{\Delta t}{2}\left[c^2\left(\frac{b^{n}_{\theta i}}{r_i} + \left(\widetilde{\nabla} b^{n}_{\theta i}\right)_r\right)\right] , \\ e^{+}_{ri,\alpha} &= \tilde{e}^{n}_{ri} + \overrightarrow{A_{i1}M_{i,\alpha}} \cdot \widetilde{\nabla} e^{n}_{ri} + \frac{\Delta t}{2}\left[-c^2\left(\widetilde{\nabla} b^{n}_{\theta i}\right)_z\right] , \\ b^{+}_{\theta i,\alpha} &= b^{n}_{\theta i} + \overrightarrow{A_{i0}M_{i,\alpha}} \cdot \widetilde{\nabla} b^{n}_{\theta i} + \frac{\Delta t}{2}\left[-\left(\widetilde{\nabla} e^{n}_{ri}\right)_z + \left(\widetilde{\nabla} e^{n}_{zi}\right)_r\right] , \end{aligned} \tag{3.7a}$$

$$\begin{aligned} e^{-}_{zi,\alpha} &= \tilde{e}^{n}_{zi_\alpha} + \overrightarrow{A_{i_\alpha 1}M_{i,\alpha}} \cdot \widetilde{\nabla} e^{n}_{zi_\alpha} + \frac{\Delta t}{2}\left[c^2\left(\frac{b^{n}_{\theta i_\alpha}}{r_{i_\alpha}} + \left(\widetilde{\nabla} b^{n}_{\theta i_\alpha}\right)_r\right)\right] , \\ e^{-}_{ri,\alpha} &= \tilde{e}^{n}_{ri_\alpha} + \overrightarrow{A_{i_\alpha 1}M_{i,\alpha}} \cdot \widetilde{\nabla} e^{n}_{ri_\alpha} + \frac{\Delta t}{2}\left[-c^2\left(\widetilde{\nabla} b^{n}_{\theta i_\alpha}\right)_z\right] , \\ b^{-}_{\theta i,\alpha} &= b^{n}_{\theta i_\alpha} + \overrightarrow{A_{i_\alpha 0}M_{i,\alpha}} \cdot \widetilde{\nabla} b^{n}_{\theta i_\alpha} + \frac{\Delta t}{2}\left[-\left(\widetilde{\nabla} e^{n}_{ri_\alpha}\right)_z + \left(\widetilde{\nabla} e^{n}_{zi_\alpha}\right)_r\right] . \end{aligned} \tag{3.7b}$$

in case of an explicit FV scheme of second-order accuracy.

Appendix D

Two-dimensional Particle Localization with the Assous Approach

For the sake of completeness, we summarize briefly the basic features of the two-dimensional localization algorithm, the plane geometry analogue of the three-dimensional case described in Section 2.3.2. The situation on hand is seen in Figure D.1, where the macro particle $\mathcal{P}^n$ with the coordinates $\boldsymbol{x}^n$ is situated in the triangle C_i at time $t = t^n$. In order to decide whether the particle is inside the

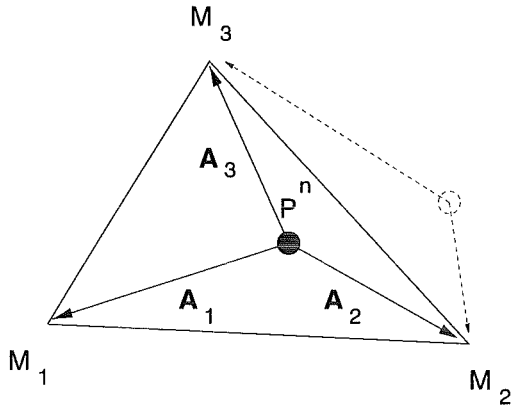

Figure D.1: Particle localization in two dimensions according to the scheme proposed by Assous et al. [2].

element C_i (possessing the vertices $\mathcal{M}_{i,\alpha}$ with the coordinates $\boldsymbol{a}_{i,\alpha} = (a_\alpha, b_\alpha, 0)^T$ where $1 \leq \alpha \leq \sigma_i = 3$), we have to calculate the following three determinants:

$$\Delta_{i,1} = (\boldsymbol{A}_2 \times \boldsymbol{A}_3) \cdot \boldsymbol{e}_3 \,, \tag{4.1a}$$

$$\Delta_{i,2} = (\boldsymbol{A}_3 \times \boldsymbol{A}_1) \cdot \boldsymbol{e}_3 \,, \tag{4.1b}$$

$$\Delta_{i,3} = (\boldsymbol{A}_1 \times \boldsymbol{A}_2) \cdot \boldsymbol{e}_3 \,, \tag{4.1c}$$

where $\boldsymbol{e}_3 = (0, 0, 1)^T$ is the unit vector orientated with the x_3-direction and $\boldsymbol{A}_\alpha$ is the abbreviation for $\boldsymbol{A}_\alpha = \overrightarrow{\mathcal{P}^n \mathcal{M}_{i,\alpha}} = \boldsymbol{a}_{i,\alpha} - \boldsymbol{x}^n$. With the machine zero $\epsilon > 0$, the following alternatives have to be considered:
If

$$\Delta_{i,\alpha} \geq -\epsilon \,; \quad \forall \alpha \,, \quad 1 \leq \alpha \leq 3 \,,$$

then the particle is situated in the triangle C_i and the corresponding shape-function (cf. 2.28a) for this particle may be computed.
Otherwise, if one of the

$$\Delta_{i,\alpha} < -\epsilon \,; \quad 1 \leq \alpha \leq 3 \,,$$

(in Figure D.1 $\Delta_{i,1} < -\epsilon$) then a more careful particle handling is necessary. Especially, it is important to include more temporal information of the particle movement in order to find out through which side of the element C_i the particle passed within the time step $\Delta t = t^{n+1} - t^n$.

Appendix E

Example of a Command File

In order to run the code, the user has to provide the system with a command file named **data** , where the possible course of actions are declared and the essential information for the code running is specified. The structure of the command file **data** is arranged as follows:

- **fname** : name of the diagnostic files generated during the computation. Type: **character*7**.
- **itmax** : maximum number of temporal iteration cycles. Type: **integer**.
- **itdia** : frequency of diagnostic file generation during the simulation. Type: **integer**.
- **icomp** : desired diagnostic information:
  0 : E_1, E_2 and B_3 components.
  1 : B_1, B_2 and E_3 components. Type: **integer**.
  2 : E_1, E_2, E_3, B_1, B_2 and B_3.
- **npodi** : Number of mesh points where the diagnostic information is recorded. This number should be smaller than **mpodi** . Type: **integer**.
- **xypodi (i,1), xypodi(i,2)** (and **xypodi(i,3)** if $ndim = 3$) :Coordinates of the **npodi** points where the diagnostic information should be monitored. Type: **real*8**.
- **idibeg , idiend** : The printing of diagnostics starts with the iteration **idibeg** and ends with the iteration **idiend**. Type: **integer**.
- **iorder** : Order of the numerical scheme (1 or 2). Type: **integer**.
- **igeome** : Type of geometry ($ndim = 2$) :
  1 for the Cartesian (x, y) geometry,
  2 for the cylinder symmetrical (z, r) geometry.
- **dt** : Desired value of the time step size before an additional correction according to the CFL condition is performed. Type: **real*8**.
- **nptyp** : Number of particle species used for the simulation. This number must be smaller than **mptyp** . Type: **integer**.
- The following values have to be specified *nptyp* times in the following order :
  - **npart (i)** : number of initial particles of species **i**. Type: **integer**. Must be smaller than **mpart** .
  - **qtypp (i)** : electrical charge of the particles of species **i**. Type: **real*8**.
  - **mtypp (i)** : mass of the particles of species **i**. Type: **real*8**.