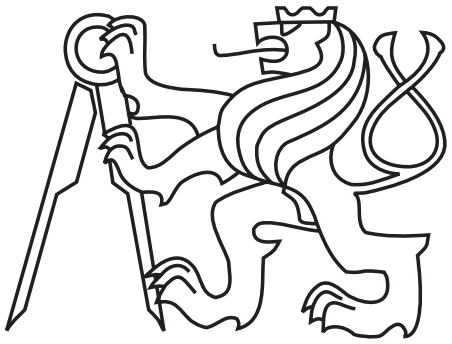
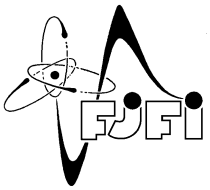
Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering
Department of Physical Electronics



Tight-focusing of short intense laser pulses in PIC simulations of laser-plasma interaction

(Master thesis)

Author: Bc. Petr Valenta
Supervisor: doc. Ing. Ondřej Klimo, Ph.D.
Consultant: Dr. Stefan Weber
Academic year: 2016/2017



ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA JADERNÁ A FYZIKÁLNĚ INŽENÝRSKÁ

Katedra fyzikální elektroniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	**Bc. Petr V a l e n t a**
Obor:	**Informatická fyzika**
Školní rok:	**2015/2016**
Název práce:	**Zaostření krátkého intenzivního laserového impulsu do velmi malého ohniska v PIC simulacích interakce s plazmatem** **Tight-focusing of short intense laser pulses in PIC simulations of laser-plasma interaction**
Vedoucí práce:	**doc. Ing. Ondřej Klimo, PhD.**
Konzultant:	**Dr. Stefan Weber**

Cíl práce:

Cílem práce je implementovat novou okrajovou podmínku v PIC simulačním kódu EPOCH (případně vytvořit za tímto účelem zvláštní program), která umožní simulaci krátkého intenzivního laserového impulsu zaostřeného do ohniska menšího, než umožňuje paraxiální aproximace. Tato okrajová podmínka bude otestována a použita v modelových simulacích, kde bude studován vliv zaostření laserového impulsu na průběh laserové interakce s plazmatem.

Pokyny pro vypracování:

1) Seznamte se s fyzikou interakce ultra-intenzivních laserových impulsů s terči a to zejména s relativistickými aspekty této interakce. Vypracujte přehled možností fokusace velmi-intenzivních impulsů.
2) Provádějte 1D simulace interakce silného laserového pulsu s pevným terčem a studujte pro různé hustoty a tloušťky terče absorpci, reflexi a transmisi laserového pulsu

v závislosti na vlivu vlastního vyzařování částic. Kvantifikujte kolik energie je absorbováno elektrony, kolik energie je předáno iontům a kolik energie je vyzářeno elektrony ven z terče.

3) Pro vybrané případy proveďte také 2D simulace pomocí kódu EPOCH a porovnejte s 1D simulacemi.

Literatura:

1) P. Gibbon, *Short Pulse Laser Interactions with Matter*, Imperial College Press, London, 2005.
2) T. D. Arber et al., Contemporary particle-in-cell approach to laser-plasma modelling, Plasma Physics and Controlled Fusion 57, 113001 (2015).
3) A. Macchi, A Superintense Laser-Plasma Interaction Theory Primer, SpringerBriefs in Physics, Springer, Dordrecht (2013).
4) C. K. Birdsall, A. B. Langdon, Plasma Physics via Computer Simulation, Hilger, Bristol (1991).
5) P. Mulser and D: Bauer, *High Power Laser–Matter Interaction* (Springer, Berlin Heidelberg 2010).

Datum zadání: říjen 2016

Datum odevzdání: 5.květen 2017

……. …….

Vedoucí katedry *Děkan*

Declaration

I declare that I carried out this work independently, and only with the cited sources, literature and other professional sources.

In Prague on .....

.....
Bc. Petr Valenta

Název práce:	**Zaostření krátkého intenzivního laserového impulsu do velmi malého ohniska v PIC simulacích interakce s plazmatem**
Autor:	Bc. Petr Valenta
Druh práce:	Diplomová práce
Obor:	Informatická fyzika
Vedoucí práce:	doc. Ing. Ondřej Klimo, Ph.D. Katedra fyzikální elektroniky, Fakulta jaderná a fyzikálně inženýrská, České vysoké učení technické v Praze
Konzultant:	Dr. Stefan Weber Fyzikální ústav Akademie věd České republiky, v. v. i., projekt ELI-Beamlines

Abstrakt

Content...

Keywords:

Title:	**Tight-focusing of short intense laser pulses in PIC simulations of laser-plasma interaction**
Author:	Bc. Petr Valenta
Type of work:	Master thesis
Branch of study:	Computational Physics
Supervisor:	doc. Ing. Ondřej Klimo, Ph.D. Department of Physical Electronics, Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague
Consultant:	Dr. Stefan Weber Institute of Physics, Academy of Sciences of the Czech Republic, v. v. i., Project ELI-Beamlines

Abstract

Content...

Keywords: plasma-based amplification, PIC simulations, parametric instabilities

Contents

Introduction

Chapter 1

The electromagnetic field

Chapter 2

Laser-matter interaction

When a high-power laser pulse is focused onto the surface of a solid target, a high density plasma layer is produced almost immediately. At the higher irradiances, the electric field of the laser light is sufficient to directly ionize the atoms. At the lower irradiances, the process is more complicated but nonetheless a plasma is produced rapidly. The whole region dominated by laser-plasma interactions, which is called corona, expands radially outwards at sonic velocities.

In the inertial fusion experiments, it is essential to deliver the laser energy into the capsule of fusion fuel as much as possible. However, the efficient transport of radiation within the target strongly depends on non-linear processes that take place during the propagation of the laser light in the coronal plasma. Therefore, laser-plasma interactions provide a key gateway to the study of inertial confinement fusion. A brief introduction to this exciting field, which is rich both in physics and in applications is provided in the following chapter.

2.1 Introduction to plasma physics

A plasma, one of the four fundamental states of matter, is a quasi-neutral gas of charged and neutral particles which exhibits collective behavior. It is necessary to explain some terms used in this definition.

By collective behavior one means motions that depend not only on local conditions but on the state of the plasma in remote regions as well. As charged particles move around, they can generate local concentrations of positive or negative charge, which give rise to electric fields. Motion of charges also generates currents, and hence magnetic fields. These fields affect the motion of other charged particles far away. Thus, the plasma gets a wide range of possible motions.

Quasi-neutrality describes the apparent charge neutrality of a plasma over large volumes, while at smaller scales, there can be charge imbalance, which may give rise to electric fields.

This fact can be expressed mathematically as

$$\sum_s q_s n_s \approx 0, \tag{2.1}$$

where q_s and n_s is, respectively, the charge and density of particles of species s. The index of summation is taken over all the particle species in plasma.

One of the most important parameters, which allows us to more accurately predict the behavior of plasmas, is the degree of its ionization. For a gas containing only single atomic species in thermodynamic equilibrium, the ionization can be clearly recognized from the Saha-Langmuir equation, which can be written in the following form,

$$\frac{n_{k+1}}{n_k} = \frac{2}{n_e h^3} (2\pi m_e k_B T)^{\frac{3}{2}} \frac{g_{k+1}}{g_k} \exp\left(-\frac{\varepsilon_{k+1} - \varepsilon_k}{k_B T}\right). \tag{2.2}$$

Here n_k is the density of atoms in the k-th state of ionization, n_e is the electron density, m_e stands for the mass of electron, k_B is Boltzmann's constant, T is the gas temperature, h is Planck's constant, g_k is the degeneracy of the energy level for ions in the k-th state, ε_k is the ionization energy of the k-th level. It can be clearly seen from equation (2.2) that the fully ionized plasmas exist only at high temperatures. That is the reason why plasmas do not occur naturally on Earth (with a few exceptions).

A fundamental characteristic of the behavior of a plasma is its ability to shield out electric potentials that are applied to it. Therefore, another important quantity λ_{Ds} which is called the Debye length of species s is established,

$$\lambda_{Ds} = \sqrt{\frac{\varepsilon_0 k_B T_s}{q_s^2 n_s}}. \tag{2.3}$$

The physical constant ε_0 is commonly called the permittivity of free space and T_s denotes the temperature of particles of species s. It often happens that the different species of particles in plasma have separate distributions with different temperatures, however each species can be in its own thermal equilibrium. The Debye length is a measure of the shielding distance or thickness of the sheath.

In plasma, each particle tries to gather its own shielding cloud. The previously mentioned concept of Debye shielding is valid only if there are enough particles in that cloud. Therefore, another important dimensionless number N_{Ds}, which is called plasma parameter of species s, is established. Definition of this parameter is given by the average number of particles of species s in a plasma contained within a sphere of radius of the Debye length, thus

$$N_{Ds} = \frac{4}{3}\pi n_s \lambda_{Ds}^3. \tag{2.4}$$

Consider an electrically neutral plasma in equilibrium. Suppose an amount of electrons

is displaced with respect to the ions, for example by intense laser pulse, and then allowed to move freely. An electric field will be set up, causing the electrons to be pulled back toward ions. Thus, the net result is a harmonic oscillation. The frequency of the oscillation is called the electron plasma frequency ω_{pe},

$$\omega_{pe} = \sqrt{\frac{e^2\, n_e}{\varepsilon_0 m_e}}. \tag{2.5}$$

By analogy with the electron plasma frequency (2.5) one could define the ion plasma frequency ω_{pi} for a general ion species. However, the ions are much heavier than electrons, so they do not response to high frequency oscillation of electromagnetic field. It is often possible to treat the massive ions as an immobile, uniform, neutralizing background. However, if the frequency of external radiation source or the waves induced in plasmas is close to this frequency, the ion motion must also be included. An example may be stimulated Brillouin scattering, which derivation can be found at the end of this chapter.

A typical charged particle in a plasma simultaneously undergo Coulomb collisions with all of the other particles in the plasma. The importance of collisions is contained in an expression called the collision frequency ν_c, which is defined as the inverse of the mean time that it takes for a particle to suffer a collision. Relatively accurate calculation of electron-ion collision frequency ν_{ei} can be obtained from the following relation,

$$\nu_{ei} = \frac{Ze^4 n_e}{4\pi\varepsilon_0^2 m_e^2 v^3} \ln\Lambda, \qquad \Lambda = \frac{\lambda_D}{b_0}. \tag{2.6}$$

The coefficient Z denotes the charge number, v is relative velocity of colliding particles and $\ln\Lambda$ is the so-called Coulomb logarithm. It is ratio of the Debye to Landau length. Landau length b_0 is the impact parameter at which the scattering angle in the center of mass frame is $90°$. For many plasmas of interest Coulomb logarithm takes on values between $5-15$. In a plasma a Coulomb collision rarely results in a large deflection. The cumulative effect of the many random small angle collisions that it suffers, however, is often larger than the effect of the few large angle collisions.

In a constant and uniform magnetic field one can find that a charged particle spirals in a helix about the line of force. This helix, however, defines a fundamental time unit and distance scale,

$$\omega_{cs} = \frac{|q_s|\|\mathbf{B}\|}{m_s}, \qquad r_{Ls} = \frac{v_\perp}{\omega_{cs}}. \tag{2.7}$$

These are called the cyclotron frequency ω_{cs} and the Larmor radius r_{Ls} of species s. Here $\mathbf{B}$ is a magnetic field and $v_\perp$ is a positive constant denoting the speed in the plane perpendicular to $\mathbf{B}$. Symbol $\|.\|$ stands for the Euclidean norm.

2.2 Description of plasma

There are basically three different approaches to plasma physics: the hydrodynamic theory, the kinetic theory and the particle theory. Each approach has some advantages and limitations which stems from simplified assumptions appropriate only for certain phenomena and time scales.

The plasma kinetic theory takes into account the motion of all of the particles. This can be done in an exact way, using Klimontovich equation. However, one is not usually interested in the exact motion of the particles, but rather in certain average characteristics. Thus, this equation can be good starting point for the derivation of approximate equations.

The kinetic theory is based on a set of equations for the distribution functions $f_s(\mathbf{x}, \mathbf{v}, t)$ of each plasma particle species s, together with Maxwell equations. Here $\mathbf{x}$ is the vector of coordinates for all the degrees of freedom, $\mathbf{v}$ is the corresponding vector of velocities and t is time. The distribution function is a statistical description of a very large number of interacting particles. If collisions can be neglected (for example in hot plasmas), the evolution of such a system can be described by the collisionless Vlasov equation,

$$\frac{\partial f_s}{\partial t} + \mathbf{v} \cdot \nabla f_s + \frac{q_s}{m_s} (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \frac{\partial f_s}{\partial \mathbf{v}} = 0. \tag{2.8}$$

Here $\mathbf{E}$ and $\mathbf{B}$ are macroscopic electric and magnetic fields acting on the particles.

The equation (2.8) is obtained only by making the assumption that the particle density is conserved, such that the rate of change in a phase-space volume is equal to the flux of particles into that volume. Because of its comparative simplicity, this equation is most commonly used in kinetic theory. However, the assumption to neglect collisions in a plasma is not generally valid. If it is necessary to take them into account, the collision term can be approximated under certain conditions.

The second approach is hydrodynamic theory. In this model, the conservation laws of mass, momentum and energy are coupled to Maxwell equations. The fluid theory is the simplest description of a plasma, however this approximation is sufficiently accurate to describe the majority of observed phenomena. The velocity distribution of each species is assumed to be Maxwellian everywhere, so the dependent variables are functions of only space coordinates and time. The fluid equations are simply the first three moments of the Vlasov equation. These yield the following fluid equations for the density, the momentum and the energy,

$$\frac{\partial n_s}{\partial t} + \nabla \cdot (n_s \mathbf{u}_s) = 0, \tag{2.9}$$

$$m_s n_s \left[\frac{\partial \mathbf{u}_s}{\partial t} + (\mathbf{u}_s \cdot \nabla) \mathbf{u}_s \right] + \nabla \cdot \mathbb{P}_s = q_s n_s (\mathbf{E} + \mathbf{u}_s \times \mathbf{B}), \tag{2.10}$$

$$\frac{\partial}{\partial t} \left(\frac{1}{2} n_s m_s u_s^2 + e_s \right) + \nabla \cdot \left(\frac{1}{2} n_s m_s u_s^2 \mathbf{u}_s + e_s \mathbf{u}_s + \mathbb{P}_s \mathbf{u}_s + \mathbf{Q}_s \right) = q_s n_s \mathbf{u}_s \cdot \mathbf{E}. \tag{2.11}$$

The zeroth-order moment (2.9) gives the continuity equation, where $\mathbf{u}_s(\mathbf{x}, t)$ is the velocity of the fluid of species s. This equation essentially states that the total number of particles is conserved. The first-order moment (2.10) leads to a momentum equation. Here $\mathbb{P}_s(\mathbf{x}, t)$ is the pressure tensor. This comes about by separating the particle velocity into the fluid and a thermal component of velocity. The thermal velocity then leads to the pressure term. Finally, the second-order moment (2.11) corresponds to the energy equation, where e_s is the density of the internal energy and $\mathbf{Q}_s$ describes the heat flux density.

The moment equations are an infinite set of equations and a truncation is required in order to solve these equations. For this equation system to be complete it has to be supplemented by an equation of state, which describes the relation between pressure and density in the plasma. However, the equations of state are well defined only in local thermodynamic equilibrium. Otherwise, the system cannot be described by fluid equations.

The last possible description is the particle theory approach. The plasma is described by electrons and ions moving under the influence of the external (e.g. laser) fields and electromagnetic fields due to their own charge. The basic equation of motion for a charged particle in an electromagnetic field is given by the Newton equations of motion with the Lorentz force,

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v}, \qquad \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \frac{q_s}{m_s}\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right). \tag{2.12}$$

However, plasmas typically consist of an extremely large number of particles that interact in self-consistent fields, so the analysis can be applied only with the help of powerful computing infrastructure and particle simulation codes.

2.3 Electromagnetic waves in plasmas

In this section, some general properties of electromagnetic waves propagation in magnetized plasmas are described. Particularly, consider in some detail the waves traveling parallel to and perpendicular to magnetic field.

First, the dispersion relation is derived from the hydrodynamic plasma equations. Since we assume plasma response to a high frequency field, the ions are treated as stationary, neutralizing background. Thermal motion of particles is also ignored, thus the pressure term can be neglected. The following set of equations has to be solved,

$$\frac{\partial n_e}{\partial t} + \nabla \cdot (n_e \mathbf{u}_e) = 0, \tag{2.13}$$

$$m_e n_e \frac{\partial \mathbf{u}_e}{\partial t} + m_e n_e \left(\mathbf{u}_e \cdot \nabla\right) \mathbf{u}_e = -e n_e \left(\mathbf{E} + \mathbf{u}_e \times \mathbf{B}\right). \tag{2.14}$$

The symbol e denotes the elementary charge. To develop wave equations for the oscillating

electric and magnetic field, Faraday's law and Ampere's law are needed,

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}, \qquad \frac{\partial \mathbf{E}}{\partial t} = \frac{1}{\varepsilon_0 \mu_0} \nabla \times \mathbf{B} + \frac{e n_e \mathbf{u}_e}{\varepsilon_0}. \tag{2.15}$$

where μ_0 is permeability of free space.

Next, the system of equations will be linearized by using the methods of perturbation theory. Consider a small perturbations,

$$n_e = n_{e0} + \delta n_e, \quad \mathbf{u}_e = \delta \mathbf{u}_e, \quad \mathbf{B} = \mathbf{B}_0 + \delta \mathbf{B}, \quad \mathbf{E} = \delta \mathbf{E}. \tag{2.16}$$

After substituting perturbations into initial system of equations and performing Fourier transform one obtains

$$\delta n_e = \mathrm{i} \frac{n_{e0}}{\omega} \mathbf{k} \cdot \delta \mathbf{u}_e, \tag{2.17}$$

$$\delta \mathbf{u}_e = -\mathrm{i} \frac{e}{m_e \omega} \delta \mathbf{E} - \mathrm{i} \frac{e}{m_e \omega} \delta \mathbf{u}_e \times \mathbf{B}_0, \tag{2.18}$$

$$\delta \mathbf{B} = \frac{1}{\omega} \mathbf{k} \times \delta \mathbf{E}, \tag{2.19}$$

$$\delta \mathbf{E} = -\frac{1}{\varepsilon_0 \mu_0 \omega} \mathbf{k} \times \delta \mathbf{B} + \mathrm{i} \frac{e n_0}{\varepsilon_0 \omega} \delta \mathbf{u}_e, \tag{2.20}$$

where i denotes imaginary unit, $\mathbf{k}$ is wave vector and ω is angular frequency.

Equation for density perturbation can be ignored. Eliminating $\delta \mathbf{B}$ and $\delta \mathbf{u}_e$ from the equation (2.20) one gets the equation for perturbation of electric field,

$$\begin{aligned} \left(\omega^2 - \omega_{pe}^2 - c^2 k^2\right) \delta \mathbf{E} + \mathrm{i} \frac{\omega_{ce}}{\omega} \left(\omega^2 - c^2 k^2\right) \delta \mathbf{E} \times \mathbf{e}_B + \\ + c^2 \left(\mathbf{k} \cdot \delta \mathbf{E}\right) \mathbf{k} + \mathrm{i} \frac{\omega_{ce}}{\omega} c^2 \left(\mathbf{k} \cdot \delta \mathbf{E}\right) \mathbf{k} \times \mathbf{e}_B = 0. \end{aligned} \tag{2.21}$$

Here c is the speed of light in free space and $\mathbf{e}_B$ is a unit vector in the direction of the magnetic field,

$$c^2 = \frac{1}{\varepsilon_0 \mu_0}, \qquad \mathbf{e}_B = \frac{\mathbf{B}_0}{B_0}. \tag{2.22}$$

If one choose the coordinate system, where $\mathbf{B}_0 = (0, 0, B_0)$ and $\mathbf{k} = (k \sin\alpha, 0, k \cos\alpha)$, one obtain equation $\mathbb{M} \cdot \delta \mathbf{E} = 0$, where

$$\mathbb{M} = \begin{pmatrix} \omega^2 - \omega_{pe}^2 - c^2 k^2 \cos^2\alpha & \mathrm{i} \frac{\omega_{ce}}{\omega} \left(\omega^2 - c^2 k^2\right) & c^2 k^2 \cos\alpha \sin\alpha \\ -\mathrm{i} \frac{\omega_{ce}}{\omega} \left(\omega^2 - c^2 k^2 \cos^2\alpha\right) & \omega^2 - \omega_{pe}^2 - c^2 k^2 & -\mathrm{i} \frac{\omega_{ce}}{\omega} c^2 k^2 \cos\alpha \sin\alpha \\ c^2 k^2 \cos\alpha \sin\alpha & 0 & \omega^2 - \omega_{pe}^2 - c^2 k^2 \sin^2\alpha \end{pmatrix}. \tag{2.23}$$

The system of equations has non-trivial solution if and only if $\det \mathbb{M} = 0$. This condition leads to desired dispersion relation for an arbitrary angle α,

$$\left[\left(\omega^2 - \omega_{pe}^2 - c^2k^2\cos^2\alpha\right)\left(\omega^2 - \omega_{pe}^2 - c^2k^2\alpha\right) - \left(\frac{\omega_{ce}}{\omega}\right)^2\left(\omega^2 - c^2k^2\right)\left(\omega^2 - c^2k^2\cos^2\alpha\right)\right]$$
$$\left(\omega^2 - \omega_{pe}^2 - c^2k^2\sin^2\alpha\right) - c^4k^4\cos^2\alpha\sin^2\alpha\left[\left(\omega^2 - \omega_{pe}^2 - c^2k^2\right) - \left(\frac{\omega_{ce}}{\omega}\right)^2\left(\omega^2 - c^2k^2\right)\right] = 0. \tag{2.24}$$

Now, it is necessary to find a dispersion relations for the two simplest cases, propagation along and perpendicular to magnetic field. For the waves propagating along $\mathbf{B_0}$ is $\alpha = 0$ and dispersion relation (2.24) gets relatively simple form,

$$\left(\omega^2 - \omega_{pe}^2\right)\left[\left(\omega^2 - \omega_{pe}^2 - c^2k^2\right)^2 - \left(\frac{\omega_{ce}}{\omega}\right)^2\left(\omega^2 - c^2k^2\right)^2\right] = 0. \tag{2.25}$$

The equation (2.25) has three solutions. The first describes plasma oscillations at frequency $\omega = \omega_{pe}$. The second and third solutions give right-handed (R) and left-handed (L) circularly polarized waves,

$$N_R^2 = 1 - \frac{\left(\omega_{pe}/\omega\right)^2}{1 - \omega_{ce}/\omega}, \qquad N_L^2 = 1 - \frac{\left(\omega_{pe}/\omega\right)^2}{1 + \omega_{ce}/\omega}. \tag{2.26}$$

The symbol N stands for the index of refraction, which is more useful for describing how waves propagate through medium.

In a similar manner, for the waves propagating perpendicular to $\mathbf{B_0}$ is $\alpha = \pi/2$ and the dispersion relation (2.24) has the following form,

$$\left(\omega^2 - \omega_{pe}^2 - c^2k^2\right)\left[\left(\omega^2 - \omega_{pe}^2\right)\left(\omega^2 - \omega_{pe}^2 - c^2k^2\right) - \omega_{ce}^2\left(\omega^2 - c^2k^2\right)\right] = 0. \tag{2.27}$$

The equation (2.27) has two solutions, which give ordinary (O) and extraordinary (X) waves,

$$N_O^2 = 1 - \left(\frac{\omega_{pe}}{\omega}\right)^2, \qquad N_X^2 = 1 - \left(\frac{\omega_{pe}}{\omega}\right)^2 \frac{1 - \left(\omega_{pe}/\omega\right)^2}{1 - \left(\omega_{pe}/\omega\right)^2 - \left(\omega_{ce}/\omega\right)^2} \tag{2.28}$$

The ordinary wave corresponds to a linearly polarized wave with electric field lying along the magnetic field direction, so that the motion remains unaffected. The extraordinary wave have electric fields that are perpendicular to magnetic field, but with components both perpendicular and parallel to the wave vector.

The important properties of these waves are distinguished by their cut-offs ($N \to 0$) and resonances ($N \to \infty$). In the vicinity of the resonance there is a total absorption, at a cut-off frequency there is a total reflection of incident waves. All of the cut-offs and resonances of waves are listed in table (1).

Wave	Cut-offs	Resonances
R	$\omega_R = \frac{1}{2}\omega_{ce} + \frac{1}{2}\sqrt{\omega_{ce}^2 + 4\omega_{pe}^2}$	$\omega_{ce} = \frac{eB_0}{m_e}$
L	$\omega_L = -\frac{1}{2}\omega_{ce} + \frac{1}{2}\sqrt{\omega_{ce}^2 + 4\omega_{pe}^2}$	$\omega_{ci} = \frac{ZeB_0}{m_i}$
O	$\omega = \sqrt{\omega_{pe}^2 + \omega_{pi}^2}$	$\omega_{uh} = \sqrt{\omega_{pe}^2 + \omega_{ce}^2}$
X	$\omega = \omega_R, \omega_L$	$\omega_{lh} = \sqrt{\omega_{ce}\,\omega_{ci}}$

Table 1: Summary of cut-offs and resonances for the principal waves

2.4 Laser absorption processes

Absorption of laser energy in laser-plasma interactions is an important issue, which has been closely related to the applications including the inertial fusion research ever since the invention of laser. Intense laser pulse can be absorbed in plasma by different non-linear mechanisms. In the following, the three main mechanisms of absorption of laser radiation are briefly described.

2.4.1 Collisional absorption

One of the most important mechanism of laser light absorption in plasma is collisional absorption, also called inverse bremsstrahlung. It is a process in which an electron absorbs a photon while colliding with an ion or with another electron, and it leads to the heating of all particles in the interaction region. This way electromagnetic energy of the laser wave is transferred into the kinetic energy of plasma.

The change in laser intensity I, passing through plasma in the direction of x-axis, is given by

$$\frac{\mathrm{d}I}{\mathrm{d}x} = -\kappa I, \tag{2.29}$$

where κ is the spatial damping rate of the laser energy caused by collisional absorption. For a slab of plasma of length L, the absorption coefficient α_{abs} is given by

$$\alpha_{\mathrm{abs}} = 1 - \frac{I_{\mathrm{out}}}{I_{\mathrm{in}}} = 1 - \exp\left(-\int_0^L \kappa\,\mathrm{d}x\right). \tag{2.30}$$

Here I_{out} and I_{in} are the outgoing and the incoming laser intensities, respectively. The absorption coefficient for a linear and exponential density profiles is given by solving the integral (2.30) [3],

$$\alpha_{\text{abs}} = 1 - \exp\left(-\frac{32}{15}\frac{\nu_{ei}(n_c)L}{c}\right) \qquad \text{for} \qquad n_e = n_c\left(1 - \frac{x}{L}\right), \tag{2.31}$$

$$\alpha_{\text{abs}} = 1 - \exp\left(-\frac{8}{3}\frac{\nu_{ei}(n_c)L}{c}\right) \qquad \text{for} \qquad n_e = n_c \exp\left(-\frac{x}{L}\right). \tag{2.32}$$

where $\nu_{ei}(n_c)$ is the collision frequency evaluated at the critical density n_c, which is given by formula

$$n_c = \frac{\varepsilon_0 m_e \omega}{e^2}. \tag{2.33}$$

Notice that a significant fraction of the collisional absorption is from the region near the critical density. Collisional absorption is the preferred absorption mechanism for driving matter ablatively with laser beams. In the case of long laser pulse duration with relatively low intensity, collisional absorption can be very efficient.

2.4.2 Resonance absorption

Laser light in the plasma can be also absorbed by resonance absorption. It is a linear process in which an incident laser wave is partially absorbed by conversion into an electron wave at the critical density of plasma.

Resonance absorption takes place when a p-polarized laser pulse is obliquely incident on a plasma with an inhomogeneous density profile. A component of the laser wave electric field perpendicular to the target surface then resonantly excites an electron plasma wave also along the plasma density gradient, thus a part of the laser wave energy is transferred into the electrostatic energy of the electron plasma wave. This wave propagates into the underdense plasma and it is damped either by collision or collisionless damping mechanisms. Consequently, energy is further converted into thermal energy which heats the plasma.

In contrast to collisional absorption, resonance absorption is the main absorption process for high laser intensities and long wavelengths. The efficiency of resonance absorption can also be higher for hot plasma, low critical density, or short plasma scale-length.

2.4.3 Landau damping

Landau damping is a very efficient damping mechanism of longitudinal plasma waves, which may occur even in the absence of collisions. Consider ions as a stationary neutralizing background. Using a linearized version of the Vlasov equation (2.8), one can find dispersion

relation for electron plasma waves,

$$\omega^2 \approx \omega_{pe}^2 + 3\frac{\omega_{pe}^2}{\omega^2} v_{th}^2 k^2 + \mathrm{i}\pi \frac{\omega_{pe}^2 \omega^2}{k^2} \frac{\mathrm{d}\hat{f}_0}{\mathrm{d}v_x}\bigg|_{v_x = v_{ph}}, \tag{2.34}$$

where v_{th} and v_{ph} are, respectively, electron thermal velocity and phase velocity of the wave, and $\hat{f}_0(v_x)$ is the one-dimensional normalized distribution function in the zeroth-order approximation. The imaginary part in (2.34) expresses Landau damping.

Only if the velocity distribution is not constant, the collisionless energy exchange between the wave and the particles occurs. The particles with a velocity almost equal to the phase velocity of the wave v_{ph} can become resonant with the field and therefore can be either slowed down or accelerated.

Let $\hat{f}_0(v_x)$ be Maxwellian, then one can calculate the Landau damping rate γ_L, which is given by

$$\gamma_\mathrm{L} \approx -\sqrt{\frac{\pi}{2}} \frac{\omega_{pe}}{(k\lambda_D)^3} \exp\left(-\frac{1}{2\,(k\lambda_D)^2}\right). \tag{2.35}$$

Since γ_L is negative, there is without doubt a collisionless damping of plasma waves. As is evident from (2.35) this damping becomes important when the wavelength is comparable to the Debye length.

However, the main feature of collisionless heating mechanisms is that only a minority fraction of the plasma electrons acquires most of the absorbed energy. This means that, as a side effect, a population of hot electrons is created, which significantly preheats the plasma.

2.5 Non-linear processes and instabilities

Plasma is strongly non-linear medium. This means that it may be accompanied by very strong electric fields and its non-linearities can be excited easily. When laser pulse with intensity above certain threshold irradiates the plasma, a number of collective non-linear processes may occur which can either enhance or reduce the energy absorption. As discussed before, the knowledge of the penetration depth and absorbed fraction of the incident energy is of vital importance for inertial confinement fusion research.

The non-linear interaction is conveniently described in terms of light pressure and the ponderomotive force, which is introduced first. In the next section, the parametric instabilities are discussed as a consequence of non-linear coupling of electromagnetic laser waves to the plasma. The last section describes the laser beam filamentation and self-focusing.

2.5.1 Ponderomotive force

The ponderomotive force is a most important quantity in the interaction of high intensity laser pulses with plasma. It leads to a wide range of non-linear phenomena. For normal laser

light incidence on plasma, the ponderomotive force $\mathbf{f}_p$ per unit volume is given by

$$\mathbf{f}_p = -\frac{\omega_{pe}^2}{\omega^2}\nabla\frac{\varepsilon_0\left\langle E^2\right\rangle}{2}, \tag{2.36}$$

where the $\langle\,\rangle$ symbol denotes the time average over one laser oscillating period. Notice that in a homogeneous field this time-averaged force vanishes.

The ponderomotive force represents the gradient of the laser electric field acting in a way to push charged particles into regions of lower field amplitude. It is the result of the Lorentz force that works on a charged particles in the electromagnetic wave.

Since the mass of the ions is much higher than electrons, the ponderomotive force acting on the ions is negligible. However, the ponderomotive force exerted on the electrons is transmitted to the ions by the electric field, which is created due to charge separation in the plasma.

2.5.2 Parametric instabilities

A parametric instability is defined in general by the non-linear phenomenon where a periodic variation in the medium supports the excitation of waves at a different frequency. In the following section we describe these wave excitation processes in more detail.

The resonance conditions in the three-wave interactions, under which parametric instabilities may grow, can be written in the form of the energy and momentum conservation laws,

$$\omega_0 = \omega_1 + \omega_2, \qquad \mathbf{k}_0 = \mathbf{k}_1 + \mathbf{k}_2, \tag{2.37}$$

where index 0 denotes parameters of the incident so-called pumping laser wave, which provides the driving force to excite other wave modes in the plasma. Indices $1, 2$ stand for the parameters of the two daughter waves.

These conditions imply that parametric instabilities take place only if a certain relations between the incident pump wave and the plasma frequency are fulfilled. A pumping laser wave can induce the following parametric instabilities that directly correspond to a plasma domain, because the plasma frequency is determined by the density of the plasma. Depending on which types of waves are excited, either the reflectivity or the absorption can be enhanced:

1. A laser pump with frequency $\omega_0 \approx \omega_{pe}$ may decay into an electron wave and an ion wave, leading to laser absorption. This phenomena occurs near the critical density n_c of plasma and is known as parametric decay instability.
2. A laser pump with frequency $\omega_0 > \omega_{pe}$ may decay into an ion wave and another electromagnetic wave, leading to laser scattering, including backscattering. This phenomena occurs in the whole domain of underdense plasma and is known as stimulated Brillouin scattering.
3. A laser pump with frequency $\omega_0 = 2\omega_{pe}$ may decay into two electron waves, leading to

laser absorption. This phenomena occurs at $n_c/4$ and is known as two-plasmon decay instability.

4. A laser pump with frequency $\omega_0 \geqslant 2\omega_{pe}$ may decay into an electron wave and another electromagnetic wave, leading to laser scattering, including backscattering. This phenomena occurs in a plasma with density lower than $n_c/4$ and is known as stimulated Raman scattering.

Even if these conditions are fulfilled, however, these instabilities do not need to be present in a plasma. The intensity of the incoming laser wave has to exceed a certain threshold in order for the parametric instability to occur, because all waves in plasma are damped either by collision and collisionless processes. If the laser intensity is higher than this limit, the amplitude of the parametric decay mode increases.

Chapter 3

The particle-in-cell method

The collective behavior of particles and fields in laser-plasma interactions represent a complex and strongly non-linear problem. The investigation of such systems cannot be carried out only through the application of two traditional techniques, namely, theoretical and experimental work. Since there is a large number of a very complex simultaneous interactions with many degrees of freedom in plasma, analytical modeling seems to be impractical. On the other hand, many of the significant details of laser-plasma interaction are extremely difficult or even impossible to obtain experimentally. Therefore, for the further understanding in this field of research other tools and techniques are required [9].

Numerical simulation is now an integrated part of science and technology. With the advent of powerful computational systems, numerical simulations now play an important role in physics as an essential tool in developing theoretical models and understanding experimental results. Numerical simulation is now rightfully considered as a separate discipline from theory and experiment [11].

Numerical simulations help researchers to develop models covering a wide range of physical scenarios and to investigate their properties. For example, numerical schemes for Newton's equation can be implemented in the study of the molecular dynamic, the techniques used to solve hydrodynamic equations are needed in weather prediction and algorithms for solving the diffusion equation can be applied to air pollution control problems. Only one numerical code can solve a variety of physical problems by modification of the initial and boundary conditions. These so-called computer experiments are often faster and much cheaper than a single real experiment in laboratory [7].

Nowadays, it is clear that a detailed understanding of the physical mechanisms in laser-plasma interaction can only be achieved through the combination of theory, experiment and simulation. Development of parallel algorithms that lead to a stable and sufficiently exact solution, however, belong to the most challenging fields of modern science.

The most part of this chapter is focused on the description of the particle-in-cell method, which is the very popular numerical algorithm used for plasma simulations. There can be found the mathematical background of this method, description of the steps of the simulation

loop and stability conditions. The last section provides a brief overview of the particle-in-cell code EPOCH, which has been used for simulations within this work.

3.1 Particle-in-cell method

The particle-in-cell (PIC) method refers to a technique used to solve a certain class of partial differential equations. The method was proposed in the mid-fifties and it gained a great popularity in plasma physics applications early. It is based on the particle description approach, thus the evolution of the system is conducted in principle by following the trajectory of each particle.

However, the real systems are often extremely large in terms of the number of particles they contain. In order to make simulations efficient or at all possible, so-called macro-particles are used. A macro-particle is a finite-sized computational particle that represents a group of physical particles that are near each other in the phase space. It is allowed to rescale the number of particles, because the Lorentz force depends only on the charge to mass ratio, which is invariant to this transformation. Thus, a macro-particle will follow the same trajectory as the corresponding real particles would [8].

Although this approach significantly reduces the number of computational particles, the binary interactions for every pair of a system cannot be taken into account. The cost would scales quadratically, as the number of particles increases, which makes the computational effort unmanageable in the case of larger systems. Many of the phenomena occur in high-temperature plasmas where collisional effects are very weak, thus one can neglect them. Otherwise, one may use other techniques to include collisional effects [10].

Here, the procedure for deriving the PIC method is considered. The phase space distribution function $f_s(\mathbf{x}, \mathbf{v}, t)$ for a given species s is governed by the Vlasov equation (2.8) in the collisionless plasma. The PIC method can be regarded as a finite element approach but with finite elements that are themselves moving and overlapping. The mathematical formulation of the PIC method is obtained by assuming that the distribution function of each species is given by the sum of distribution functions for macro-particles,

$$f_s(\mathbf{x}, \mathbf{v}, t) = \sum_p f_p(\mathbf{x}, \mathbf{v}, t). \tag{3.1}$$

Index p denotes hereafter the quantities attributable to macro-particles. The distribution function for each macro-particle is further assumed to be

$$f_p(\mathbf{x}, \mathbf{v}, t) = N_p S_x(\mathbf{x} - \mathbf{x}_p(t)) S_v(\mathbf{v} - \mathbf{v}_p(t)), \tag{3.2}$$

where N_p is the number of physical particles that are represented by each macro-particle, and S_x, S_v are the so-called shape functions.

The shape functions cannot be chosen arbitrarily. They have to fulfill a several special

properties. Let S_ξ be the shape function of the phase space coordinate ξ. Then:

1. The support of the shape function is compact, $\exists R > 0$, supp $S_\xi \subset (-R, R)$.
2. Integral of the shape function is unitary, $\int_{-\infty}^{+\infty} S_\xi(\xi)\,\mathrm{d}\xi = 1$.
3. The shape function is symmetrical, $S_\xi(\xi) = S_\xi(-\xi)$.

While these restrictive conditions still offer a wide range of possibilities, the standard PIC method is essentially determined by the choice of the shape function in the velocity direction as a Dirac δ-function and in the spatial direction as a m-th order b-spline basis function b_m,

$$S_v(\mathbf{v} - \mathbf{v}_p(t)) = \delta(\mathbf{v} - \mathbf{v}_p(t)), \qquad S_x(\mathbf{x} - \mathbf{x}_p(t)) = b_m\left(\frac{\mathbf{x} - \mathbf{x}_p(t)}{\Delta p}\right), \tag{3.3}$$

where Δp is the size of the support of the computational particles, typically the same as simulation grid cell. Stability and accuracy of the simulation strongly depend on the choice of the shape functions. The choice of higher-order basis functions results in less numerical noise interpolation of density and field quantities and reduces non-physical phenomena in simulations, obviously at the cost of increased computational time.

The computational cycle of the PIC method is shown in Figure 1. Individual steps are closer described in several following sections. The influence of the choice of the time and spatial step on the stability and accuracy of the PIC method is demonstrated in the last section.

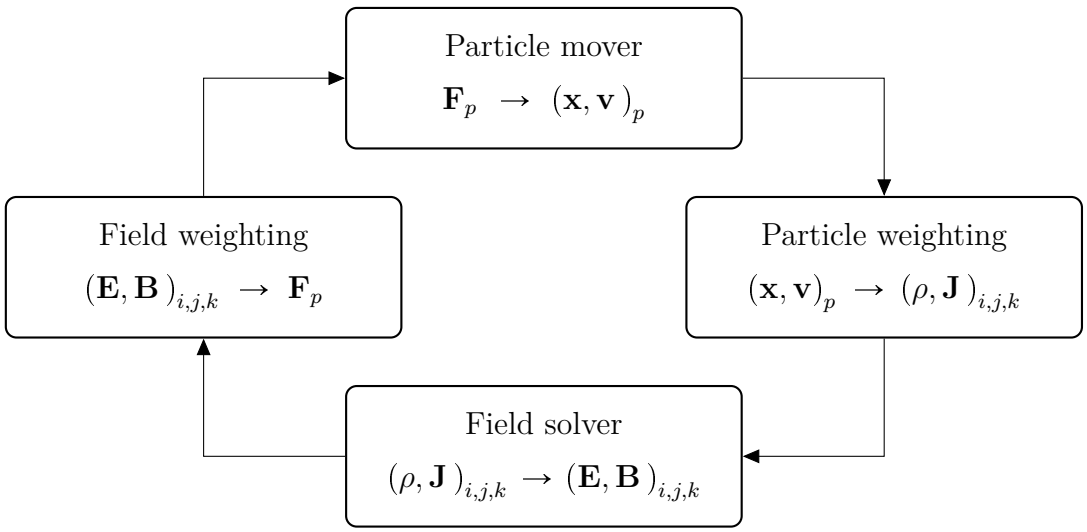


Figure 1: Computational cycle of the particle-in-cell method

3.1.1 Field solver

The behavior of the time varying electromagnetic field in the free space is governed by the microscopic variant of Maxwell equations,

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0}, \qquad \nabla \cdot \mathbf{B} = 0 \tag{3.4}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \qquad \nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \frac{1}{c^2}\frac{\partial \mathbf{E}}{\partial t}, \tag{3.5}$$

where $\rho(\mathbf{x}, t)$ is charge density and $\mathbf{J}(\mathbf{x}, t)$ current density.

To make the simulation possible, PIC method solves field equations only at certain points of the computational domain and time levels. Therefore, it is necessary to perform the discretization of the spatial coordinates $\mathbf{x} \rightarrow x_{i,j,k}$ where $(i, j, k) \in \mathbb{N}^3$ are grid indices and the time coordinate $t \rightarrow t^n$, where $n \in \mathbb{N}$ is the time level index. Solution is outlined here for a three-dimensional equidistant rectangular grid, thus $x_{i,j,k} = (i\Delta x, j\Delta y, k\Delta z)$ where $\Delta x, \Delta y, \Delta z$ are the spatial steps in each direction and $t^n = n\Delta t$ where Δt is the time step.

Typical method to solve Maxwell equations in PIC codes is finite-difference time-domain (FDTD), because it is arguably the simplest technique in terms of the implementation. The vector components of the fields $\mathbf{E}$ and $\mathbf{B}$ are spatially staggered about rectangular unit cells of a Cartesian computational grid,

$$\mathbf{E}(\mathbf{x}, t) \rightarrow \left[(E_x)^n_{i,\, j+1/2,\, k+1/2}, (E_y)^n_{i+1/2,\, j,\, k+1/2}, (E_z)^n_{i+1/2,\, j+1/2,\, k} \right], \tag{3.6}$$

$$\mathbf{B}(\mathbf{x}, t) \rightarrow \left[(B_x)^n_{i+1/2,\, j,\, k}, (B_y)^n_{i,\, j+1/2,\, k}, (B_z)^n_{i,\, j,\, k+1/2} \right]. \tag{3.7}$$

This scheme, which has proven to be very robust, is now known as a Yee lattice [13]. The illustration of a standard Cartesian Yee cell used for FDTD is shown in Figure 2. Components of the current density $\mathbf{J}$ are defined in the same way as the components of $\mathbf{E}$, charge density is defined in the middle of the cell. For marching in time a leap-frog scheme is used, thus discretized Maxwell equations (3.4), (3.5) have the following form,

$$\nabla^+ \cdot \mathbf{E}^n = \frac{\rho^n}{\varepsilon_0}, \quad \nabla^- \cdot \mathbf{B}^{n+1/2} = 0. \tag{3.8}$$

$$\frac{\mathbf{B}^{n+1/2} - \mathbf{B}^{n-1/2}}{\Delta t} = -\nabla^- \times \mathbf{E}^n, \quad \frac{1}{c^2}\frac{\mathbf{E}^{n+1} - \mathbf{E}^n}{\Delta t} = \nabla^+ \times \mathbf{B}^{n+1/2} - \mu_0 \mathbf{J}^{n+1/2}, \tag{3.9}$$

Notice that FDTD method achieve second-order accuracy in both, space and time. Discrete operators (∇^+) and (∇^-) act on a scalar field $f_{i,j,k}$,

$$\nabla^+ f_{i,\, j,\, k} = \left(\frac{f_{i+1,\, j,\, k} - f_{i,\, j,\, k}}{\Delta x}, \frac{f_{i,\, j+1,\, k} - f_{i,\, j,\, k}}{\Delta y}, \frac{f_{i,\, j,\, k+1} - f_{i,\, j,\, k}}{\Delta z} \right), \tag{3.10}$$

$$\nabla^- f_{i,\,j,\,k} = \left(\frac{f_{i,\,j,\,k} - f_{i-1,\,j,\,k}}{\Delta x}, \frac{f_{i,\,j,\,k} - f_{i,\,j-1,\,k}}{\Delta y}, \frac{f_{i,\,j,\,k} - f_{i,\,j,\,k-1}}{\Delta z} \right). \tag{3.11}$$

These operators have the following properties,

$$\nabla^- \cdot \nabla^- \times = \nabla^+ \cdot \nabla^+ \times = 0, \qquad \nabla^- \cdot \nabla^+ = \nabla^+ \cdot \nabla^- = \Delta. \tag{3.12}$$

Symbol Δ stands for the discrete Laplace operator in central differences,

$$\Delta f_{i,j,k} = \frac{f_{i-1,j,k} + 2f_{i,j,k} + f_{i+1,j,k}}{\Delta x^2} + \frac{f_{i,j-1,k} + 2f_{i,j,k} + f_{i,j+1,k}}{\Delta y^2} + \frac{f_{i,j,k-1} + 2f_{i,j,k} + f_{i,j,k+1}}{\Delta z^2}. \tag{3.13}$$

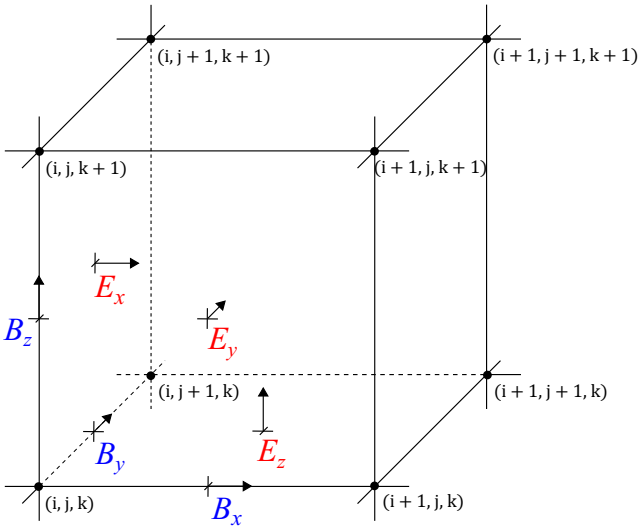


Figure 2: Standard Cartesian Yee cell used for FDTD method

Before trying to find the solution of Maxwell equations, one have to realize that this system of equations is not independent. In the three-dimensional case, there are eight first-order differential equations, but only six unknown vector components. Acting on the first and second of the equations (3.9) by operators $(\nabla^- \cdot)$ and $(\nabla^+ \cdot)$, respectively, one obtains

$$\frac{\nabla^- \cdot \mathbf{B}^{\,n+1/2} - \nabla^- \cdot \mathbf{B}^{\,n-1/2}}{\Delta t} = 0, \tag{3.14}$$

$$\frac{\rho^{\,n+1} - \rho^{\,n}}{\Delta t} + \nabla^+ \cdot \mathbf{J}^{\,n+1/2} = 0. \tag{3.15}$$

It means that it is possible to solve only equations (3.9), while the divergence equations (3.8) can be considered as initial conditions. In this case, the continuity equation in finite differences (3.15) have to be fulfilled.

3.1.2 Particle and field weighting

The particle weighting refers to the part of the code in which the charge and current densities are assigned to the discrete grid points from the continuous particle positions. After the fields are obtained, they are assigned back at the particles to calculate the Lorentz force. This step is called field weighting. It implies some form of interpolation.

First, have a look at the particle weighting. Charge density $\rho(\mathbf{x}, t)$ and current density $\mathbf{J}(\mathbf{x}, t)$ are given by the following integrals over the velocity space,

$$\rho(\mathbf{x}, t) = \sum_s q_s \int f_s(\mathbf{x}, \mathbf{v}, t)\, \mathrm{d}\mathbf{v}, \qquad \mathbf{J}(\mathbf{x}, t) = \sum_s q_s \int f_s(\mathbf{x}, \mathbf{v}, t)\, \mathbf{v}\, \mathrm{d}\mathbf{v}. \tag{3.16}$$

The interpolation function is defined as

$$W_{i,j,k}\left(\mathbf{x}_p^n\right) = \frac{1}{\Delta V} \int_\Omega S_x\left(\mathbf{x} - \mathbf{x}_p^n\right) \mathrm{d}\mathbf{x}, \qquad \Delta V = \Delta x \Delta y \Delta z, \tag{3.17}$$

where $\Omega = \{\mathbf{x} \in \mathbb{R}^3 : |x_1 - x_{i,j,k}| \leqslant \Delta x/2 \wedge |x_2 - x_{i,j,k}| \leqslant \Delta y/2 \wedge |x_3 - x_{i,j,k}| \leqslant \Delta z/2\}$. Then the charge density $\rho_{i,j,k}^n$ in the arbitrary grid point $x_{i,j,k}$ and time level t^n is constructed as

$$\rho_{i,j,k}^n = \sum_p q_p W_{i,j,k}\left(\mathbf{x}_p^n\right), \qquad q_p = q_s N_p, \tag{3.18}$$

where the sum is taken over all computational particles.

By analogy, one could assign the current densities to the grid points using this interpolation function, but the discrete continuity equation for charge (3.15) would not be satisfied exactly. In this case one have to solve Poisson's equation for correction of electric field or use one of the several numerical schemes for computing the current density satisfying the continuity equation, which are called charge conservation methods. In the next paragraph, the charge density decomposition method proposed by Esirkepov [4] is described.

Due to linearity of the continuity equation (3.15), it is sufficient to define the current density associated with the motion of a single computational particle,

$$\begin{aligned}
(J_x)_{i,\,j,\,k}^n &= (J_x)_{i,\,j,\,k}^n - q_p\,(v_x)_p^n\,(\Pi_x)_{i,\,j,\,k}^n\,, \\
(J_y)_{i,\,j+1,\,k}^n &= (J_y)_{i,\,j,\,k}^n - q_p\,(v_y)_p^n\,(\Pi_y)_{i,\,j,\,k}^n\,, \\
(J_z)_{i,\,j,\,k+1}^n &= (J_z)_{i,\,j,\,k}^n - q_p\,(v_z)_p^n\,(\Pi_z)_{i,\,j,\,k}^n\,.
\end{aligned} \tag{3.19}$$

In accordance with continuity equation one can write

$$(\Pi_x)_{i,\,j,\,k}^n + (\Pi_y)_{i,\,j,\,k}^n + (\Pi_z)_{i,\,j,\,k}^n = W_{i,j,k}\left(\mathbf{x}_p^n + \tilde{\mathbf{x}}\right) - W_{i,j,k}\left(\mathbf{x}_p^n\right), \tag{3.20}$$

where $\tilde{\mathbf{x}} \in \mathbb{R}^3$ is the motion induced position shift of the computational particle in one simu-

lation time step. Shift of the computational particle generates eight variants of interpolation functions,

$$\begin{aligned} &W_{i,j,k}\left(\mathbf{x}_p^n\right), \quad W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_1\right), \quad W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_2\right), \quad W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_3\right), \\ &W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_1+\tilde{x}_2\right), \quad W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_1+\tilde{x}_3\right), \quad W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_2+\tilde{x}_3\right), \\ &W_{i,j,k}\left(\mathbf{x}_p^n+\tilde{x}_1+\tilde{x}_2+\tilde{x}_3\right). \end{aligned} \tag{3.21}$$

We assume that the vector $\Pi^n_{i,\,j,\,k}$ is linearly dependent on the functions (3.21). It turned out that only one linear combination exists.

Now, have a look at the field weighting. After calculating Maxwell equations at the grid points, it is necessary to assign electric and magnetic fields back at the particle positions. Recall the definition of interpolation function (3.17). By analogy to the particle weighting, one can assign the components of electric field,

$$\begin{aligned} (E_x)_p^n &= \sum_{i,j,k} (E_x)^n_{i,\,j+1/2,\,k+1/2}\, W_{i,\,j+1/2,\,k+1/2}\left(\mathbf{x}_p^n\right), \\ (E_y)_p^n &= \sum_{i,j,k} (E_y)^n_{i+1/2,\,j,\,k+1/2}\, W_{i+1/2,\,j,\,k+1/2}\left(\mathbf{x}_p^n\right), \\ (E_z)_p^n &= \sum_{i,j,k} (E_z)^n_{i+1/2,\,j+1/2,\,k}\, W_{i+1/2,\,j+1/2,\,k}\left(\mathbf{x}_p^n\right), \end{aligned} \tag{3.22}$$

and the components of magnetic field,

$$\begin{aligned} (B_x)_p^n &= \sum_{i,j,k} (B_x)^n_{i+1/2,\,j,\,k}\, W_{i+1/2,\,j,\,k}\left(\mathbf{x}_p^n\right), \\ (B_y)_p^n &= \sum_{i,j,k} (B_y)^n_{i,\,j+1/2,\,k}\, W_{i,\,j+1/2,\,k}\left(\mathbf{x}_p^n\right), \\ (B_z)_p^n &= \sum_{i,j,k} (B_z)^n_{i,\,j,\,k+1/2}\, W_{i,\,j,\,k+1/2}\left(\mathbf{x}_p^n\right). \end{aligned} \tag{3.23}$$

It is desirable to use the same interpolation function for both, density and force calculations, in order to eliminate a self-force and ensure conservation of momentum [5].

3.1.3 Particle pusher

The behavior of computational particles is controlled by the Newton-Lorentz equations. Since the particles can reach velocities near the velocity of light, it is necessary to perform relativistic generalization,

$$\mathbf{u}_p = \gamma\, \mathbf{v}_p, \qquad \gamma = \sqrt{1+\left(\frac{\mathbf{u}_p}{c}\right)^2}. \tag{3.24}$$

Assume that the electric and magnetic fields are interpolated from the grid points to the particles at the time level t^n. Then the computational particle equations of motion to be integrated are

$$\frac{\mathrm{d}\mathbf{x}_p}{\mathrm{d}t} = \frac{\mathbf{u}_p}{\gamma}, \qquad \frac{\mathrm{d}\mathbf{u}_p}{\mathrm{d}t} = \frac{q_s}{m_s}\left(\mathbf{E}_p + \frac{\mathbf{u}_p}{\gamma} \times \mathbf{B}_p\right). \tag{3.25}$$

To discretize equations of motion (3.25) a time-centered leap-frog scheme is used. One obtains

$$\frac{\mathbf{x}_p^{n+1} - \mathbf{x}_p^n}{\Delta t} = \frac{\mathbf{u}_p^{n+1/2}}{\gamma^{n+1/2}}, \tag{3.26}$$

$$\frac{\mathbf{u}_p^{n+1/2} - \mathbf{u}_p^{n-1/2}}{\Delta t} = \frac{q_s}{m_s}\left(\mathbf{E}_p^n + \frac{\mathbf{u}_p^{n+1/2} + \mathbf{u}_p^{n-1/2}}{2\gamma^n} \times \mathbf{B}_p^n\right). \tag{3.27}$$

Although these equations appear to be very simple, the solution is the most time-consuming part of the simulation, because they have to be solved for every single computational particle. Standard for particle pushing in plasma simulation PIC codes is elegant Boris method, which completely separates the effect of electric and magnetic fields [2]. Substitute

$$\mathbf{u}_p^{n-1/2} = \mathbf{u}_p^- - \frac{q_s\mathbf{E}_p^n}{m_s}\frac{\Delta t}{2}, \qquad \mathbf{u}_p^{n+1/2} = \mathbf{u}_p^+ + \frac{q_s\mathbf{E}_p^n}{m_s}\frac{\Delta t}{2} \tag{3.28}$$

into equation (3.27), then the electric field cancels entirely,

$$\frac{\mathbf{u}_p^+ - \mathbf{u}_p^-}{\Delta t} = \frac{q_s}{2\gamma^n m_s}\left(\mathbf{u}_p^+ + \mathbf{u}_p^-\right) \times \mathbf{B}_p^n. \tag{3.29}$$

Equation (3.29) describes a rotation of vector $\mathbf{u}_p^-$ to $\mathbf{u}_p^+$ in one simulation time step Δt. The angle θ between vector $\mathbf{u}_p^-$ and $\mathbf{u}_p^+$ we expect to be $\theta = \omega_c\Delta t$.

To implement the Boris method, first add half the electric impulse $\mathbf{E}_p^n$ to $\mathbf{u}_p^{n-1/2}$, then perform the full rotation by the angle θ, and finally, add another half the electric impulse $\mathbf{E}_p^n$. From the basic geometry Boris derived following steps to obtain $\mathbf{u}_p^{n+1/2}$. From the first of equations (3.28) express vector $\mathbf{u}_p^-$,

$$\mathbf{u}_p^- = \mathbf{u}_p^{n-1/2} + \frac{q_s\mathbf{E}_p^n}{m_s}\frac{\Delta t}{2} \tag{3.30}$$

and construct an auxiliary vector $\mathbf{u}_p'$, which is simultaneously perpendicular to $\left(\mathbf{u}_p^+ - \mathbf{u}_p^-\right)$ and to $\mathbf{B}_p^n$,

$$\mathbf{u}_p' = \mathbf{u}_p^- + \mathbf{u}_p^- \times \mathbf{t}, \qquad \mathbf{t} = \frac{q_s\mathbf{B}_p^n\Delta t}{2\gamma^n m_s}. \tag{3.31}$$

Vector $\mathbf{t}$ has to be logically parallel to $\mathbf{B}_p^n$ with the length of $\tan(\theta/2) \approx \theta/2$ for small angles.

Next, utilize the fact that the vector $(\mathbf{u}_p' \times \mathbf{B}_p^n)$ is parallel to $\left(\mathbf{u}_p^+ - \mathbf{u}_p^-\right)$ and express $\mathbf{u}_p^+$,

$$\mathbf{u}_p^+ = \mathbf{u}_p^- + \mathbf{u}_p' \times \mathbf{s}, \qquad \mathbf{s} = \frac{2\mathbf{t}}{1+t^2}. \tag{3.32}$$

Here vector $\mathbf{s}$ is parallel to $\mathbf{B}_p^n$ and its length has to fulfill the condition $\|\mathbf{u}_p^+\| = \|\mathbf{u}_p^-\|$. The transition from $\mathbf{u}_p^-$ to $\mathbf{u}_p^+$ can be written more clearly using the matrix,

$$\mathbf{u}_p^+ = \begin{pmatrix} 1 - s_2t_2 - s_3t_3 & s_2t_1 + s_3 & s_3t_1 - s_2 \\ s_1t_2 - s_3 & 1 - s_1t_1 - s_3t_3 & s_3t_2 + s_1 \\ s_1t_3 + s_2 & s_2t_3 - s_1 & 1 - s_1t_1 - s_2t_2 \end{pmatrix} \mathbf{u}_p^-. \tag{3.33}$$

Finally, substitute vector $\mathbf{u}_p^+$ into the second from the equations (3.28),

$$\mathbf{u}_p^{n+1/2} = \mathbf{u}_p^+ + \frac{q_s \mathbf{E}_p^n}{m_s} \frac{\Delta t}{2}. \tag{3.34}$$

The position of computational particle is advanced according to

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \frac{\mathbf{u}_p^{n+1/2}}{\gamma^{n+1/2}} \Delta t, \qquad \gamma^{n+1/2} = \sqrt{1 + \left(\frac{\mathbf{u}_p^{n+1/2}}{c}\right)^2}. \tag{3.35}$$

3.1.4 Stability and accuracy

The stability and accuracy of the standard PIC method is directly dependent on the size of the spatial and temporal simulation steps. In order to find correct parameters one has to know the absolute accuracy and corresponding stability conditions.

The effect of the spatial grid is to smooth the interaction forces and to couple plasma perturbations to perturbations at other wavelengths, called aliases. It can lead to non-physical instabilities and numerical heating. To avoid these effect, the spatial step needs to resolve the Debye length. Thus, it is desirable to fulfill the following condition,

$$\Delta x \leqslant \lambda_D. \tag{3.36}$$

In general electromagnetic case, the time step has to satisfy the Courant–Fridrichs–Levy (CFL) condition [9],

$$C = c^2 \Delta t^2 \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2} \right). \tag{3.37}$$

where the dimensionless number $C \leqslant 1$ is called the CFL number. This condition limits

the range of motion of all objects in simulation during one time step. It ensures, that these particles will not cross more than one cell in one time step. When this condition is violated, the growth of non-physical effects can be very rapid.

The leap-frog scheme, used to solve the field equations and equations of motion, is second-order accurate in both, time and space. In addition, this scheme is explicit and time-reversible.

3.2 Code EPOCH

The abbreviation EPOCH refers to an Extendable PIC Open Collaboration project [1]. EPOCH is a multi-dimensional, relativistic, electromagnetic code for plasma physics simulations based on the PIC method. The code, which is developed in University of Warwick, is written in FORTRAN and parallelized using MPI library. EPOCH is explicit and is able to achieve second-order accuracy. The entire core of the code uses SI units.

The main features include dynamic load balancing option for making optimal use of all processors when run in parallel, allowing restart on an arbitrary number of processors. The setup of EPOCH is controlled through a customizable input deck. An input deck is text file which can be used to set simulation parameters for EPOCH without needing to edit or recompile the source code. Most aspects of a simulation can be controlled, such as the number of grid points in the simulation domain, the initial distribution of particles and the initial electromagnetic field configuration. In addition, EPOCH was written to add more modern features and to structure the code in such a way that the future expansion of the code may be made as easily as possible.

By default, EPOCH uses triangular particle shape functions with the peak located at the position of computational particle and a width of two cells, which provides relatively clean and fast solution. However, user can select higher order particle shape functions based on spline interpolation by enabling compile time option in the makefile.

The electromagnetic field solver uses a FDTD scheme with second order of accuracy. The field components are spatially staggered on a standard Cartesian Yee cell. The solver is directly based on the scheme derived by Hartmut Ruhl [6]. The particle pusher is relativistic, Birdsall and Landon type [2] and uses Villasenor and Buneman current weighting [12].

Chapter 4

Tight focusing of laser beams

Chapter 5

Results

5.1 Calculation

5.2 Algorithm

In the following section, the practical implementation of a boundary conditions based on the previously derived solution of Maxwell equations is presented. Assume, that the laser beam propagates in a forward direction along the axis z. In the beginning, it is necessary to prescribe the electric laser field $\mathbf{E}_{0,\perp}(\mathbf{r}_\perp, t)$ in the plane $\mathcal{P}$ at $z = z_0$. Note, that it can be defined by arbitrary function of space and time. The goal is then to find the fields $\mathbf{E}_\mathrm{B}(\mathbf{r}_\perp, t)$ and $\mathbf{B}_\mathrm{B}(\mathbf{r}_\perp, t)$ at the corresponding boundary $z = z_\mathrm{B}$.

Consider that the transverse part of simulation domain is made of equidistant rectangular grid described by x^i, y^j, where $i, j \in \{1, \ldots, N_{x,y}\}$, and the grid steps δx, δy. The simulation time t^n, where $n \in \{1, \ldots, N_t\}$, is also divided into equidistant time steps of size δt.

The algorithm allows to calculate fields $\mathbf{E}_\mathrm{B}^{ij}(t)$ and $\mathbf{B}_\mathrm{B}^{ij}(t)$ for any given time t from the interval $\left[t^1 - \frac{z_\mathrm{B}-z_0}{c}, t^{N_t} - \frac{z_\mathrm{B}-z_0}{c}\right]$. In order to preserve clarity, the algorithm below is given in the exact form as in the original paper.

1. Calculate $\hat{\mathbf{E}}_{0,\perp}^{ijn}$ via discrete Fourier transforms in time:

$$\omega^n = \frac{2\pi}{N_t \delta t}\left(-\frac{N_t}{2} + n\right), \tag{5.1}$$

$$\hat{\mathbf{E}}_{0,\perp}^{ijn} = \frac{\delta t}{2\pi}\sum_{l=1}^{N_t} \mathbf{E}_{0,\perp}^{ijl} \mathrm{e}^{\mathrm{i}\omega^n t^l}, \quad n \in \{1, \ldots, N_t\}. \tag{5.2}$$

2. Calculate $\bar{\mathbf{E}}^{ijn}_{0,\perp}$ via two-dimensional discrete Fourier transforms in transverse space:

$$k_x^i = \frac{2\pi}{N_x \delta x}\left(-\frac{N_x}{2}+i\right), \quad k_y^j = \frac{2\pi}{N_y \delta y}\left(-\frac{N_y}{2}+j\right), \tag{5.3}$$

$$\bar{\mathbf{E}}^{ijn}_{0,\perp} = \frac{\delta x \delta y}{(2\pi)^2}\sum_{l,m=1}^{N_x,N_y} \hat{\mathbf{E}}^{lmn}_{0,\perp} \mathrm{e}^{-\mathrm{i}\left(k_x^i x^l + k_y^j y^m\right)}, \quad i,j \in \{1,\ldots,N_{x,y}\}. \tag{5.4}$$

3. Calculate transverse electric field components at the boundary $z = z_\mathrm{B}$:

$$k_z^{ijn} = \Re\sqrt{\frac{(\omega^n)^2}{c^2} - (k_x^i)^2 - (k_y^j)^2}, \tag{5.5}$$

$$\bar{\mathbf{E}}^{ijn}_{\mathrm{B},\perp} = \begin{cases} \bar{\mathbf{E}}^{ijn}_{0,\perp}\mathrm{e}^{\mathrm{i}k_z^{ijn}(z_\mathrm{B}-z_0)} & \text{for } k_z^{ijn} > 0 \\ 0 & \text{for } k_z^{ijn} = 0 \end{cases}. \tag{5.6}$$

Symbol $\Re$ stands for the real part of a complex number.

4. Calculate longitudinal electric field components at the boundary $z = z_\mathrm{B}$:

$$\bar{E}^{ijn}_{\mathrm{B},z} = \begin{cases} -\frac{k_x^i \bar{E}^{ijn}_{\mathrm{B},x} + k_y^j \bar{E}^{ijn}_{\mathrm{B},y}}{k_z^{ijn}} & \text{for } k_z^{ijn} > 0 \\ 0 & \text{for } k_z^{ijn} = 0 \end{cases}. \tag{5.7}$$

5. Calculate the magnetic field at the boundary $z = z_\mathrm{B}$:

$$\mathbb{R}^{ijn} = \begin{pmatrix} -k_x^i k_y^j & (k_x^i)^2 - (\omega^n)^2/c^2 \\ (\omega^n)^2/c^2 - (k_y^j)^2 & k_x^i k_y^j \\ -k_y^j k_z^{ijn} & k_x^i k_z^{ijn} \end{pmatrix}, \tag{5.8}$$

$$\bar{\mathbf{B}}^{ijn}_{\mathrm{B}} = \begin{cases} (\omega^n k_z^{ijn})^{-1}\mathbb{R}^{ijn}\bar{\mathbf{E}}^{ijn}_{\mathrm{B},\perp} & \text{for } k_z^{ijn} > 0 \\ 0 & \text{for } k_z^{ijn} = 0 \end{cases}. \tag{5.9}$$

6. Calculate $\hat{\mathbf{E}}^{ijn}_\mathrm{B}$, $\hat{\mathbf{B}}^{ijn}_\mathrm{B}$ via two-dimensional inverse discrete Fourier transforms:

$$\hat{\mathbf{E}}^{ijn}_\mathrm{B} = \frac{(2\pi)^2}{N_x N_y \delta x \delta y}\sum_{l,m=1}^{Nx,Ny} \bar{\mathbf{E}}^{lmn}_\mathrm{B} \mathrm{e}^{\mathrm{i}(k_x^l x^i + k_y^m y^j)}, \tag{5.10}$$

$$\hat{\mathbf{B}}_{\mathrm{B}}^{ijn} = \frac{(2\pi)^2}{N_x N_y \delta x \delta y} \sum_{l,m=1}^{Nx,Ny} \bar{\mathbf{B}}_{\mathrm{B}}^{lmn} \mathrm{e}^{\mathrm{i}(k_x^l x^i + k_y^m y^j)}. \tag{5.11}$$

7. Calculate $\mathbf{E}_{\mathrm{B}}^{ij}(t)$, $\mathbf{B}_{\mathrm{B}}^{ij}(t)$ for any given time $t \in [t^1 - \frac{z_{\mathrm{B}}-z_0}{c}, t^{N_t} - \frac{z_{\mathrm{B}}-z_0}{c}]$.

$$\mathbf{E}_{\mathrm{B}}^{ij}(t) = \frac{2\pi}{N_t \delta t} \sum_{n=1}^{N_t} \hat{\mathbf{E}}_{\mathrm{B}}^{ijn} \mathrm{e}^{-\mathrm{i}\omega^n t}, \tag{5.12}$$

$$\mathbf{B}_{\mathrm{B}}^{ij}(t) = \frac{2\pi}{N_t \delta t} \sum_{n=1}^{N_t} \hat{\mathbf{B}}_{\mathrm{B}}^{ijn} \mathrm{e}^{-\mathrm{i}\omega^n t}. \tag{5.13}$$

5.3 Implementation

One of the main goals of this work has been to implement the algorithm mentioned in the previous section, to evaluate its correctness in several test simulations and finally, to exploit resulting implementation for simulations of tightly focused Gaussian beams in laser-matter interaction. The main requirement on implementation has been easy to use with 2D version of particle-in-cell simulation code EPOCH. For this reason, several possible solutions has been taken into account.

The final decision has been to create a static library, which will be able to compute desired quantities and will provide functions for communication with the main simulation code. The essential advantage is that it could be basically linked with any laser-plasma simulation code. Also, since it is necessary to call only two additional functions, the instrumentation will be fast, easy and the main simulation code will not be excessively disturbed. Furthermore, the implementation itself come with the CMake support, which simplify the compilation process using platform and compiler independent configuration files.

The library has been written in C++ language and is object oriented so the algorithm can be easily extended to three dimensional geometry. In order to speedup the whole underlying computation, the algorithm has been parallelized using hybrid techniques. The time domain has been decomposed into the stripes corresponding to individual computational processes, the communication between these processes is ensured by MPI library. Furthermore, the computationally most expensive cycles are parallelized using OpenMP implementation of multi-threading. Later on, the speedup and parallel scaling performance will be briefly discussed.

Fourier transforms form the core of the computational process and their performance is crucial for the overall performance of the code. For this reason, many currently available libraries have been considered. Eventually, the Fourier transforms in the algorithm can be computed using FFTW library, Intel MKL library or it is also possible to directly evaluate the formulas without using any additional library. The user specifies his option before the compilation. Regarding both libraries, a threaded versions of 1D in-place complex fast Fourier

transforms has been used throughout the code. According to several measures, there is no significant difference between the speed of both implementations.

One potential bottleneck could happen during the computation of spatial Fourier transforms since the arrays with spatial data are decomposed into different processes. The cluster versions of functions performing the Fourier transforms has been tested, however they did not bring any significant speedup. The reason is as follows. They require to have the global array in memory and use its own distribution which involves overlapping. Since the size of global arrays is usually not so large and since it is necessary to perform a lot of different Fourier transforms, the majority of computational time is spent rather for communication, mainly if many of computational cores are used.

This issue has been solved by gathering the data on master process, performing the Fourier transforms in space by only one processor and scattering the data back to corresponding processes. This is the reason why the code does not scale well, however, the time to compute all desired quantities is in most cases negligible in comparison with the time required by main simulation cycle. Nevertheless, this issue could be improved in future.

Since it is necessary to compute the whole time evolution of the laser field at boundary for each grid point before the simulation starts, the resulting amount of data can be significantly large and does not have to fit in a computer memory. Thus, it is inevitable to dump the data into a file, which will be then accessed by the main simulation code. Due to the performance purposes, each computational process stores its data into a shared file with corresponding offset and in binary coding. Therefore, the output operations are as fast as possible and save the storage resources. Library then provide a function which allows to seek an arbitrary position in a file. This function is then called each time step of the main simulation loop to fill the laser source arrays with all the relevant data. This way of accessing data does not cause any significant slowdown or memory overhead.

The EPOCH code require only transverse components of laser electric field, all other quantities are computed by the FDTD solver. The implementation of the library allows fully connection with EPOCH. In practice, if the user wants to simulate tight-focusing, he has to enable corresponding flag as a compile-time option and then specifies all required parameters in the input file. The code then automatically computes all necessary data. It works generally regardless the number of lasers in the simulation or boundaries that they are attached to.

The current version of library does not work for obliquely incident laser pulses, because in this case one cannot exploit the advantage of an efficient computation with fast Fourier transforms. However, the code allows to compute the laser fields at boundary by evaluating Fourier integrals directly, so it could be easily extended. Second, it is at the moment possible to simulate only Gaussian laser pulses. However, the user can easily prescribe its own shape and position of the beam in focal plane by modifying corresponding part of the code.

Several most important data structures, functions and methods that form the core of library for tight-focusing can be seen in Appendix.

5.4 Evaluation

To evaluate the correctness of the algorithm presented in the previous section of this chapter as well as to demonstrate the drawbacks of the paraxial approximation, several test simulations in 2D geometry have been performed. In the following text, a two limit cases are presented. The first pair of simulations employs tightly focused Gaussian laser beam, with the size of the focus comparable with the center laser wavelength, whilst the second one shows the case of the Gaussian beam with the size of the focus one order of magnitude larger than the center laser wavelength, where both approaches should return identical results. Note, that all simulations have been computed using 2D version of particle-in-cell code EPOCH instrumented with library for tight focusing.

First, let us have a look at the simulation of a tightly focused Gaussian beam. The p-polarized laser pulse with center wavelength $\lambda = 1\ \mu$m propagates from left hand side boundary to the right. Its duration has been chosen to $\tau = 20$ fs in FWHM and amplitude $E_0 = 1 \cdot 10^{15}\,\mathrm{V/m}$. The beam width in focus $w_0 = 0.7\,\mu$m is shorter than the laser wavelength, which implies that non-negligible parts of $\bar{\mathbf{E}}_{0,\perp}(k_x, \omega)$ are evanescent. The focus is located at a distance $x = 8\,\mu$m from the boundary that the laser is attached to.

The size of the simulation domain is $16\lambda \times 16\lambda$, with 100 cells per laser wavelength in both directions, thus $\Delta x = \Delta y = \lambda/100 = 10\,\mathrm{nm}$. The one simulation time step is according to CFL condition $\Delta t = \sqrt{2}\lambda/100c \approx 0.05$ fs, the whole simulation time is then $t = 150$ fs. The pulse propagates in vacuum in order to get rid of all effects that could be potentially caused by plasma.

In the following paragraph, the results of the first simulation are discussed in a more detail. Fig. 3 shows transverse and longitudinal electric field components at their maximal intensity in the focus for both cases, laser beam propagating under the paraxial approximation (Fig. 3 - a, b) and according to the approach consistent with Maxwell equations (Fig. 3 - c, d). In the case of paraxial approximation, one can clearly see strong distortions and asymmetry in the shape of both electric field components. In addition, the focus location is shifted about 1 μm closer to the left boundary and the corresponding amplitude in focus is less than half the required value. In contrast, the fields produced by the simulation using Maxwell consistent calculation of laser fields at boundary are symmetric with respect to the focus and without any distortions. Furthermore, the focus location as well as the amplitude fulfills the initial requirements precisely.

Fig. 4 shows transverse and longitudinal slices of transverse electric field component in focus for the case of laser beam propagating under the paraxial approximation (Fig. 4 - a, b) as well as for the case where the beam propagation has been resolved within the Maxwell consistent approach (Fig. 4 - c, d). For the case of paraxial approximation, one can clearly see the asymmetry of the field shape in the longitudinal slice (Fig. 4 - b), which consequently leads to a decrease of the amplitude in focus and to the strong side-wings in the spatial beam profile, as might be seen from the transverse slice (Fig. 4 - a). On the other hand, Maxwell consistent approach calculates fields of perfect symmetry with respect to the focus (Fig. 4 -

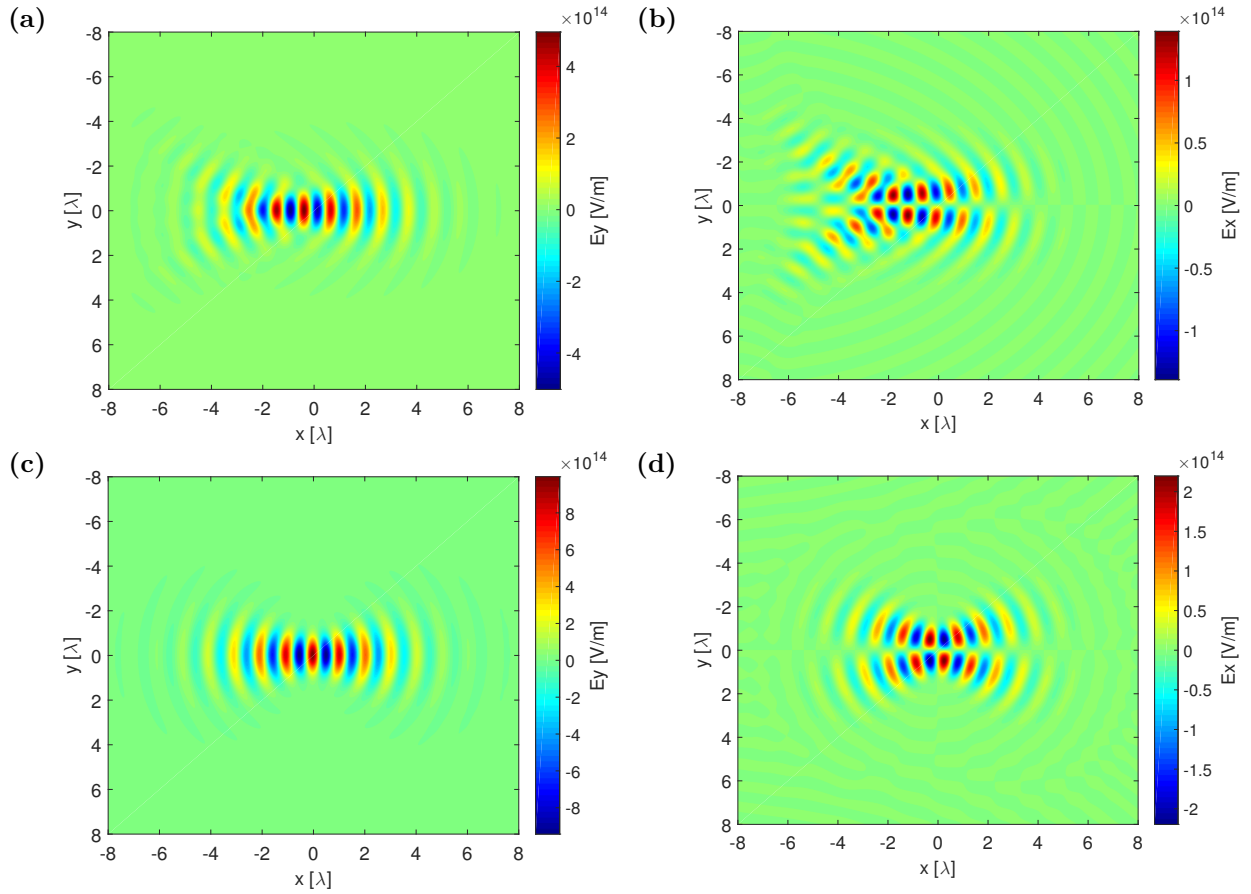


Figure 3: Transverse (E_y) and longitudinal (E_x) electric laser field components captured at the time step of their maximal intensity in the focal spot. The cases **(a)**, **(b)** correspond to the laser pulse propagating under the paraxial approximation, whilst **(c)**, **(d)** come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. In the case of paraxial approximation, both components reveal strong distortions and asymmetry, their focal spot is located about 1μm closer to the left boundary than specified and the corresponding amplitude is significantly lower. The laser has been attached to the left hand side boundary.

c) and no side-wings or distortions are present (Fig. 4 - d).

In Fig. 5 one can examine the time evolution of transverse (Fig. 5 - a) and longitudinal (Fig. 5 - b) electric field components at boundary as computed using Maxwell consistent approach. Note, that one have to chose carefully the transverse size of the domain, since the beam width at boundary may be much larger than in focus because of a diffraction.

To evaluate a correctness of the beam propagation using Maxwell consistent approach, several criteria has been defined. The correctness of the amplitude and beam width at focus as well as the right focus location has already been verified. Additional criteria has been set on a beam symmetry. In Fig. 6, one can find a comparison of the transverse electric laser

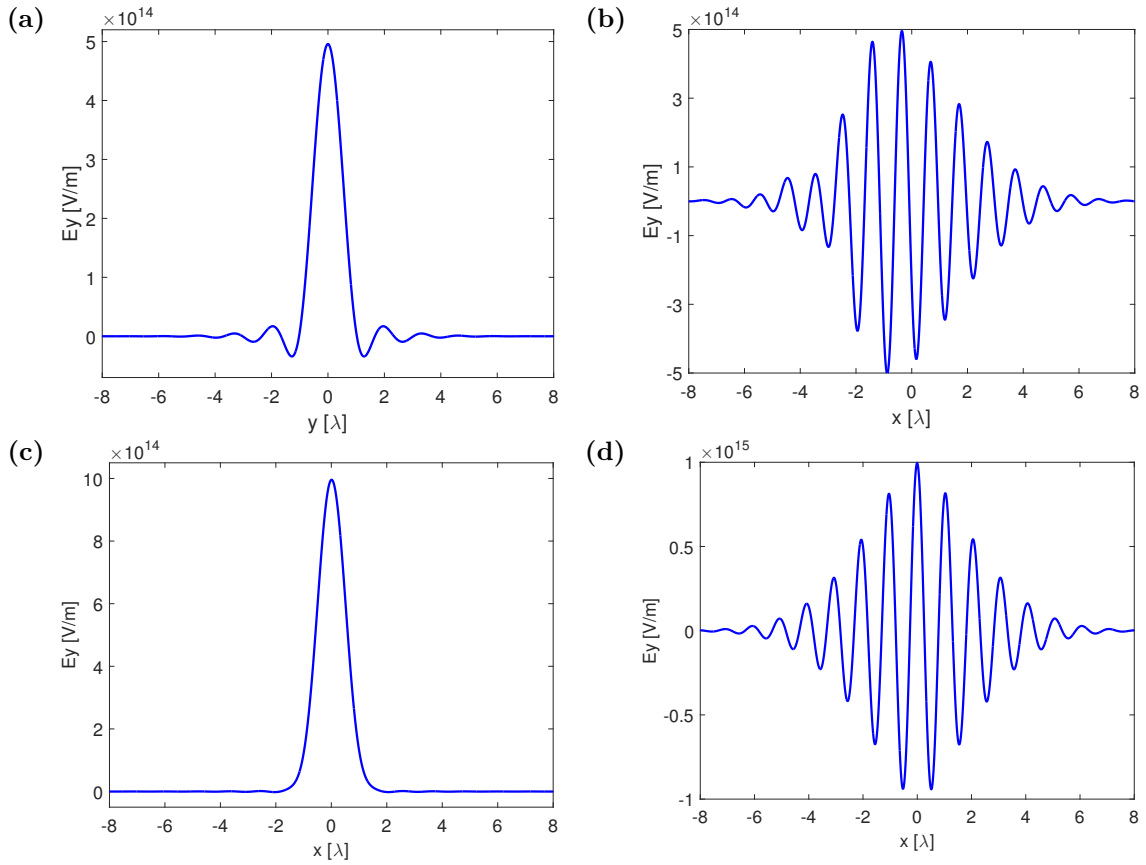


Figure 4: Transverse **(a)**, **(c)** and longitudinal **(b)**, **(d)** slices of the transverse electric laser field (E_y) at the time step when it reaches maximal intensity in the focal spot. The cases **(a)**, **(b)** correspond to the laser pulse propagating under the paraxial approximation, whilst **(c)**, **(d)** come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. In the case of paraxial approximation, one can clearly see strong side-wings in the spatial beam profile **(a)** as well as the asymmetry of the field in the longitudinal line-out **(b)**.

field component when it achieves its maximal intensity at front and rear boundary. One can clearly see the exact match between the field shapes at different time steps of the simulation in transverse (Fig. 6 - a) and longitudinal (Fig. 6 - b) slices. Moreover, all the aforementioned criteria has been fulfilled also in other simulations with different input parameters that are not presented here. Consequently, these observations prove the correctness of the propagation at least of the tightly focused Gaussian laser beams.

For the second simulation, all the input parameters remained the same except the beam width in focus. Now, the parameter $w_0 = 5\ \mu$m, which is about the limit case for the beam propagating under the paraxial approximation. Thus, one would expect that the simulation results will be almost identical.

Similarly as in Fig. 3, Fig. 7 shows again transverse and longitudinal electric field com-

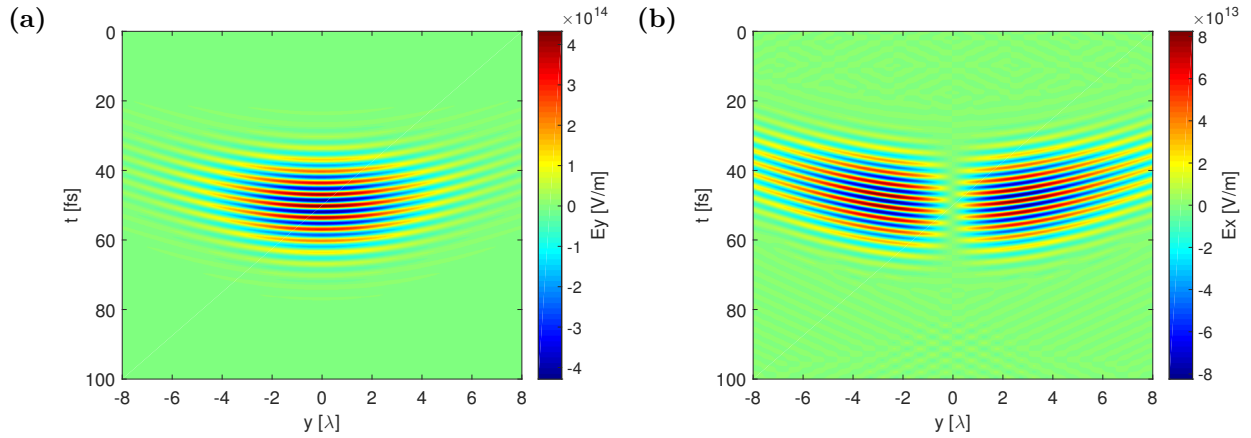


Figure 5: The time evolution of transverse (E_y) **(a)** and longitudinal (E_x) **(b)** electric laser field components at the boundary that the laser is attached to. Both components has been calculated according to the Maxwell consistent approach.

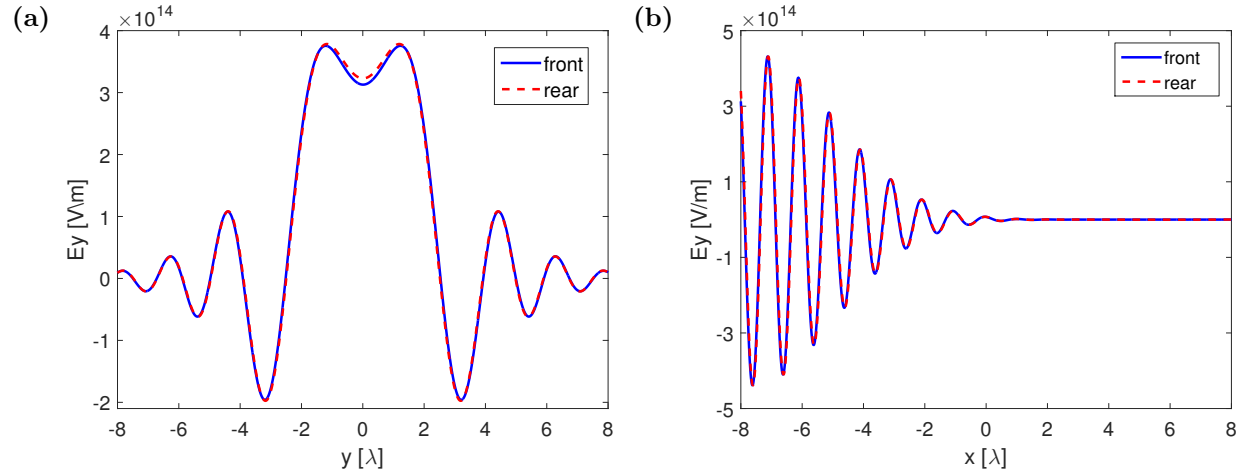


Figure 6: Transverse **(a)** and longitudinal **(b)** slice of the transverse electric laser field (E_y) when it reaches its maximal intensity at the front (blue) and rear (red) boundary. The results come from the simulation where the Maxwell consistent approach for laser propagation has been used. For better comparison, the field at the rear boundary in **(b)** has been horizontally flipped. The exact match between the field shapes at a different time steps of simulation proves the correctness of the laser beam propagation.

ponents at their maximal intensity in the focus for both cases, laser beam propagating under the paraxial approximation (Fig. 7 - a, b) and according to the approach consistent with Maxwell equations (Fig. 7 - c, d). Here, one cannot register any difference between the results corresponding to both approaches.

Also, the transverse slice of the transverse electric laser field component in focus (Fig. 8

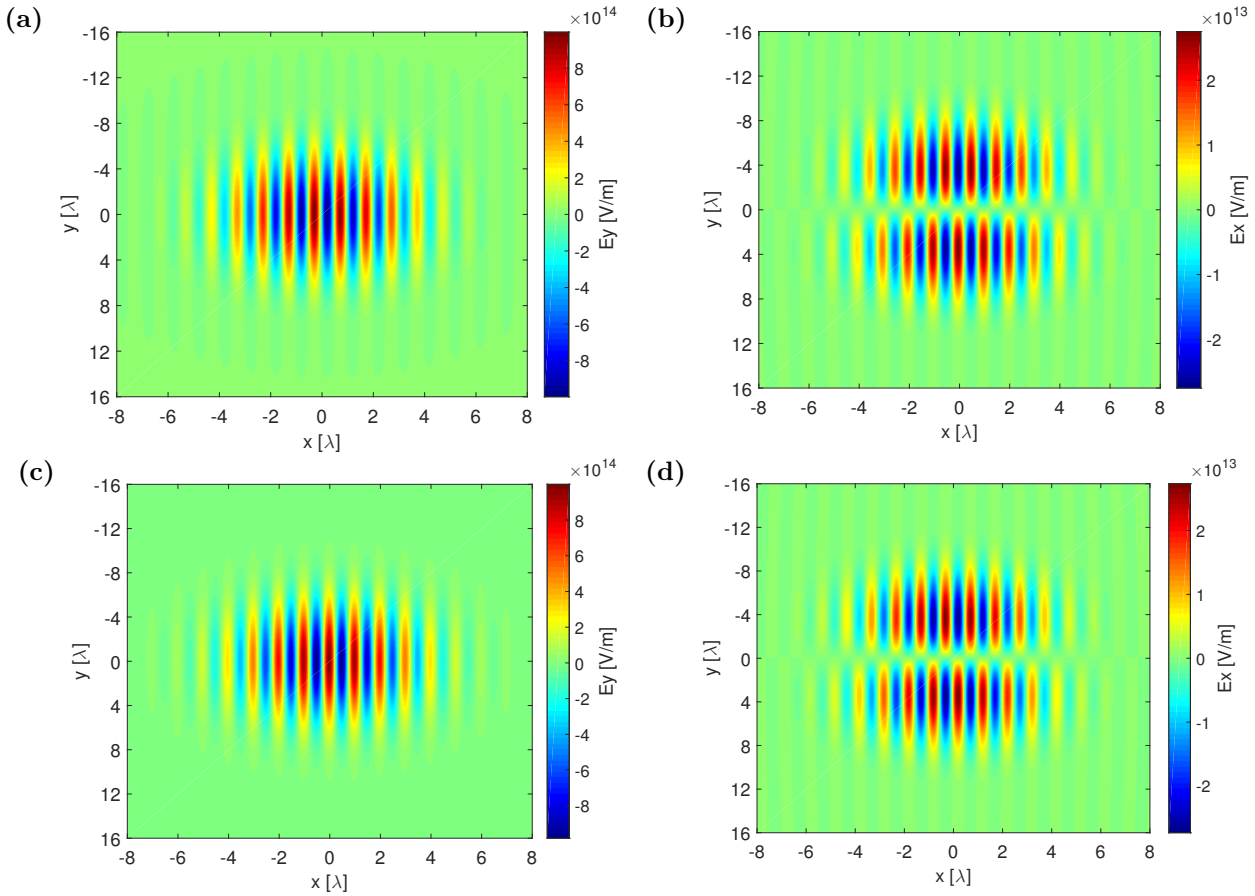


Figure 7: Transverse (E_y) and longitudinal (E_x) electric laser field components captured at the time step of their maximal intensity in the focal spot. The cases **(a)**, **(b)** correspond to the laser pulse propagating under the paraxial approximation, whilst **(c)**, **(d)** come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. The size of the focus has been chosen to be one order of the magnitude larger than the center laser wavelength. One can clearly see, that there is no significant difference between the shapes of the electric field components.

- a) shows the correct shape and amplitude for both cases. The longitudinal slice (Fig. 8 - b), however, points out the fact that the location of the focus is still little bit shifted closer to the left hand side boundary. Nevertheless, this difference could be in practice neglected. At the end of the day, for the Gaussian beams propagating under the paraxial approximation, the beam diameter at focus should be at least one order of magnitude larger than the center laser wavelength.

In conclusion, one should be aware that propagation of tightly focused laser pulses cannot be described by paraxial approximation. Above, it has been shown that for the beams focused to a spot with the size comparable to a center laser wavelength, paraxial approximation leads

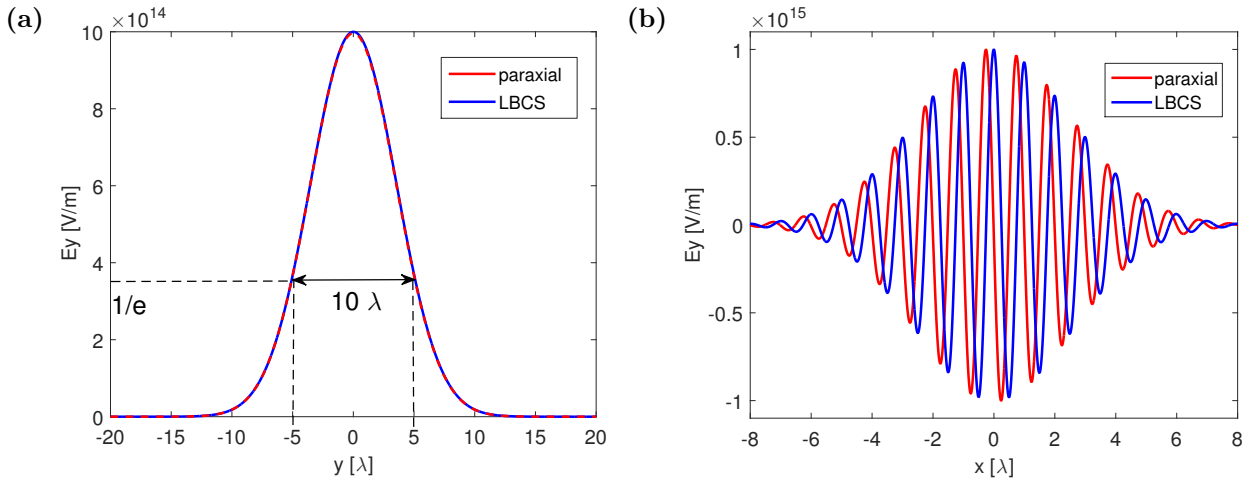


Figure 8: Transverse **(a)** and longitudinal **(b)** slices of the transverse electric laser field (E_y) at the time step when it reaches maximal intensity in the focal spot. Red lines correspond to the laser pulse propagating under the paraxial approximation, whilst blue lines come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. The size of the focus has been chosen to be one order of magnitude larger than the center laser wavelength. In the case of paraxial approximation, the focus is slightly shifted closer to the left boundary **(b)**, otherwise the size of the focus as well as the amplitude is correct for both cases **(a)**.

to a shifted location of the focus, asymmetric laser field profiles with distortions and lower amplitude. These deviations are far from negligible and have without any doubt strong impact on the laser-matter interaction results. On the other hand, the propagation of tightly focused Gaussian laser beams prescribed at boundaries according to the Maxwell consistent approach has been proven to be correct.

E and **B** fields in terms of scalar potential Φ and vector potential **A**:

$$\mathbf{E} = -\nabla\Phi - \frac{\partial \mathbf{A}}{\partial t}, \qquad \mathbf{B} = \nabla \times \mathbf{A}$$

Φ and **A** in the form of plane waves propagating along z-axis:

$$\mathbf{A}\left(x,y,z,t\right) = \mathbf{A_0}\left(x,y,z\right)\mathrm{e}^{i(k_z z-\omega t)}, \qquad \Phi\left(x,y,z,t\right) = \Phi_0\left(x,y,z\right)\mathrm{e}^{i(k_z z-\omega t)}$$

Lorentz gauge condition:

$$\nabla \cdot \mathbf{A} + \frac{1}{c^2}\frac{\partial \Phi}{\partial t} = 0$$

scalar potential in terms of vector potential:

$$\frac{\partial \Phi}{\partial t} = -\mathrm{i}\omega\Phi \qquad \longrightarrow \qquad \Phi = -\mathrm{i}\frac{c}{k_z}\nabla \cdot \mathbf{A}$$

E and **B** fields in terms of vector potential **A**:

$$\mathbf{E} = \mathrm{i}\frac{c}{k_z}\nabla\left(\nabla \cdot \mathbf{A}\right) + \mathrm{i}\omega\mathbf{A}, \qquad \mathbf{B} = \nabla \times \mathbf{A}$$

wave equation for vector potential **A**:

$$\Delta\mathbf{A} - \frac{1}{c^2}\frac{\partial^2 \mathbf{A}}{\partial t^2} = 0$$

Helmholtz equation:

$$\Delta\mathbf{A_0} + 2\mathrm{i}k_z\frac{\partial \mathbf{A_0}}{\partial z} = 0$$

paraxial (slowly varying envelope) approximation:

$$\left\|\frac{\partial^2 \mathbf{A_0}}{\partial z^2}\right\| \leqslant \left\|2k_z\frac{\partial \mathbf{A_0}}{\partial z}\right\| \longrightarrow \frac{\partial^2 \mathbf{A_0}}{\partial x^2} + \frac{\partial^2 \mathbf{A_0}}{\partial y^2} + \frac{\partial^2 \mathbf{A_0}}{\partial z^2} + 2\mathrm{i}k_z\frac{\partial \mathbf{A_0}}{\partial z} = 0$$

Solution - Gaussian beam:

$$\mathbf{E}\left(x,y,z\right)=\mathbf{E_0}\frac{w_0}{w(z)}\exp\left(-\frac{x^2+y^2}{w(z)^2}\right)\cos\left(\omega t-k_z\left(z+\frac{x^2+y^2}{2R(z)}\right)+\phi_G(z)\right)$$

where

$w(z)=w_0\sqrt{1+\left(\frac{z}{z_R}\right)^2}\ldots$ evolving beam width

$z_r=\frac{\pi w_0^2}{\lambda}\ldots$ Rayleigh range

$R(z)=z\left(1+\left(\frac{z_R}{z}\right)^2\right)\ldots$ evolving radius of curvature

$\phi(z)=\tan^{-1}\left(\frac{z}{z_R}\right)\ldots$ Guoy phase

$\theta=\tan^{-1}\left(\frac{w(z)}{z}\right)\simeq\frac{\lambda}{\pi w_0}\ldots$ beam divergence angle

Features:

- 2D version of algorithm, Ey, Bx, Bz omitted (identically equal to 0)
- Code written in C++, object oriented to be easily extended to 3D, compiled to static library
- Linked into EPOCH as a static library (in order not to disturb the code, for this reason also added support for CMake – machine independent)
- Parallelized using hybrid techniques (OpenMP + MPI – computation time in most cases negligible in comparison with the main simulation)
- Fourier transforms can be computed using Intel MKL library, FFTW library or without any library (compile time option)
- Computed fields dumped into shared files using binary coding (speed up output, save disk storage)
- Only transverse component of electric field (Ex) passed to the EPOCH at each time step (no significant slowdown or memory overhead), other fields computed by EPOCH
- All new parameters needed for tight-focusing (w0, focal length, etc.) may be specified via input file

- Implementation works generally regardless the number of lasers in the simulation or boundaries that they are attached to

Listing 5.1: Function performing forward fast Fourier transform using MKL library

```
std::vector<std::complex<double>> fft::mkl_fft_forward(std::vector<std::
    complex<double>> in) {
DFTI_DESCRIPTOR_HANDLE desc;
MKL_LONG status;
DftiCreateDescriptor(&desc, DFTI_DOUBLE, DFTI_COMPLEX, 1, static_cast<MKL_LONG
    >(in.size()));
DftiCommitDescriptor(desc);
status = DftiComputeForward(desc, in.data());
if(status != 0) {
std::cerr << DftiErrorMessage(status) << std::endl;
abort();
}
DftiFreeDescriptor(&desc);
return in;
}
```

Listing 5.2: Function performing backward fast Fourier transform using MKL library

```
std::vector<std::complex<double>> fft::mkl_fft_backward(std::vector<std::
    complex<double>> in) {
DFTI_DESCRIPTOR_HANDLE desc;
MKL_LONG status;
DftiCreateDescriptor(&desc, DFTI_DOUBLE, DFTI_COMPLEX, 1, static_cast<MKL_LONG
    >(in.size()));
DftiCommitDescriptor(desc);
status = DftiComputeBackward(desc, in.data());
if(status != 0) {
std::cerr << DftiErrorMessage(status) << std::endl;
abort();
}
DftiFreeDescriptor(&desc);
return in;
}
```

Listing 5.3: Function performing forward fast Fourier transform using FFTW library

```
std::vector<std::complex<double>> fft::fftw_fft_forward(std::vector<std::
    complex<double>> in) {
fftw_plan p = fftw_plan_dft_1d(in.size(), reinterpret_cast<fftw_complex*>(in.
    data()), reinterpret_cast<fftw_complex*>(in.data()), FFTW_FORWARD,
    FFTW_ESTIMATE);
fftw_execute(p);
fftw_destroy_plan(p);
return in;
```

```
}
```

Listing 5.4: Function performing backward fast Fourier transform using FFTW library

```
std::vector<std::complex<double>> fft::fftw_fft_backward(std::vector<std::
    complex<double>> in) {
fftw_plan p = fftw_plan_dft_1d(in.size(), reinterpret_cast<fftw_complex*>(in.
    data()), reinterpret_cast<fftw_complex*>(in.data()), FFTW_BACKWARD,
    FFTW_ESTIMATE);
fftw_execute(p);
fftw_destroy_plan(p);
return in;
}
```

Listing 5.5: Function performing forward discrete Fourier transform without using any library

```
std::vector<std::complex<double>> fft::fft_forward(std::vector<std::complex<
    double>> in) {
std::vector<std::complex<double>> out(in.size());
for(auto j = 0; j < out.size(); j++) {
for(auto l = 0; l < out.size(); l++) {
out.at(j) += in.at(l) * exp(-2.0 * constants::pi * I * l * j / in.size());
}
}
return out;
}
```

Listing 5.6: Function performing backward discrete Fourier transform without using any library

```
std::vector<std::complex<double>> fft::fft_backward(std::vector<std::complex<
    double>> in) {
std::vector<std::complex<double>> out(in.size());
for(auto j = 0; j < out.size(); j++) {
for(auto l = 0; l < out.size(); l++) {
out.at(j) += in.at(l) * exp(+2.0 * constants::pi * I * l * j / in.size());
}
}
return out;
}
```

Listing 5.7: Method for performing discrete Fourier transform in time

```
void laser_bcs::dft_time(field_2d<std::complex<double>>& field) const {
#ifdef OPENMP
#pragma omp parallel for schedule(static)
#endif
for(auto j = 0; j < this->domain->Nx; j++) {
#ifdef USE_MKL
field.add_col(fft::mkl_fft_backward(field.get_col(j)), j);
```

```
#elif USE_FFTW
field.add_col(fft::fftw_fft_backward(field.get_col(j)), j);
#else
field.add_col(fft::fft_backward(field.get_col(j)), j);
#endif
}
field.multiply(this->domain->dt / (2.0 * constants::pi));
return;
}
```

Listing 5.8: Method for performing inverse discrete Fourier transform in time

```
void laser_bcs::idft_time(field_2d<std::complex<double>>& field) const {
#ifdef OPENMP
#pragma omp parallel for schedule(static)
#endif
for(auto j = 0; j < this->domain->Nx; j++) {
#ifdef USE_MKL
field.add_col(fft::mkl_fft_forward(field.get_col(j)), j);
#elif USE_FFTW
field.add_col(fft::fftw_fft_forward(field.get_col(j)), j);
#else
field.add_col(fft::fft_forward(field.get_col(j)), j);
#endif
}
field.multiply(2.0 * (2.0 * constants::pi) / (this->domain->Nt * this->domain
    ->dt));
return;
}
```

Listing 5.9: Method for performing discrete Fourier transform in space

```
void laser_bcs::dft_space(field_2d<std::complex<double>>& field) const {
std::vector<std::complex<double>> row_global(this->domain->Nx_global);
std::vector<std::complex<double>> row_local;
for(auto j = 0; j < this->domain->Nt; j++) {
row_local = field.get_row(j);
MPI_Gatherv(row_local.data(), this->domain->Nx, MPI_CXX_DOUBLE_COMPLEX,
    row_global.data(), this->domain->counts.data(), this->domain->displs.data
    (), MPI_CXX_DOUBLE_COMPLEX, 0, MPI_COMM_WORLD);
if(this->domain->rank == 0) {
#ifdef USE_MKL
row_global = fft::mkl_fft_forward(row_global);
#elif USE_FFTW
row_global = fft::fftw_fft_forward(row_global);
#else
row_global = fft::fft_forward(row_global);
#endif
}
```

```
16 MPI_Scatterv(row_global.data(), this->domain->counts.data(), this->domain->
       displs.data(),    MPI_CXX_DOUBLE_COMPLEX, row_local.data(), this->domain->
       Nx, MPI_CXX_DOUBLE_COMPLEX, 0, MPI_COMM_WORLD);
17 field.add_row(row_local, j);
18 }
19 field.multiply(this->domain->dx / (2.0 * constants::pi));
20 return;
21 }
```

Listing 5.10: Method for performing inverse discrete Fourier transform in space

```
1  void laser_bcs::idft_space(field_2d<std::complex<double>>& field) const {
2  std::vector<std::complex<double>> row_global(this->domain->Nx_global);
3  std::vector<std::complex<double>> row_local;
4  for(auto j = 0; j < this->domain->Nt; j++) {
5  row_local = field.get_row(j);
6  MPI_Gatherv(row_local.data(), this->domain->Nx, MPI_CXX_DOUBLE_COMPLEX,
       row_global.data(), this->domain->counts.data(), this->domain->displs.data
       (), MPI_CXX_DOUBLE_COMPLEX, 0, MPI_COMM_WORLD);
7  if(this->domain->rank == 0) {
8  #ifdef USE_MKL
9  row_global = fft::mkl_fft_backward(row_global);
10 #elif USE_FFTW
11 row_global = fft::fftw_fft_backward(row_global);
12 #else
13 row_global = fft::fft_backward(row_global);
14 #endif
15 }
16 MPI_Scatterv(row_global.data(), this->domain->counts.data(), this->domain->
       displs.data(), MPI_CXX_DOUBLE_COMPLEX, row_local.data(), this->domain->Nx,
        MPI_CXX_DOUBLE_COMPLEX, 0, MPI_COMM_WORLD);
17 field.add_row(row_local, j);
18 }
19 field.multiply((2.0 * constants::pi) / (this->domain->Nx_global * this->domain
       ->dx));
20 return;
21 }
```

Listing 5.11: Method for dumping data into shared file

```
1  template <typename T>
2  void field_2d<T>::dump_to_shared_file(std::string name, int row_first, int
       row_last, int row_size_local, int row_size_global, int col_start) const {
3  MPI_File file;
4  MPI_Offset offset = 0;
5  MPI_Status status;
6  MPI_Datatype local_array;
7  int col_size = row_last - row_first;
8  const int ndims = 2;
```

```
std::array<int, ndims> size_global = {col_size, row_size_global};
std::array<int, ndims> size_local = {col_size, row_size_local};
std::array<int, ndims> start_coords = {0, col_start};
MPI_Type_create_subarray(2, size_global.data(), size_local.data(),
    start_coords.data(), MPI_ORDER_C, MPI_DOUBLE, &local_array);
MPI_Type_commit(&local_array);
std::vector<double> real_part(col_size * row_size_local);
for(auto i = std::make_pair(row_first, 0); i.first < row_last; i.first++, i.
    second++) {
for(auto j = 0; j < row_size_local; j++) {
real_part[i.second * row_size_local + j] = std::real(this->data[i.first *
    row_size_local + j]);
}
}
MPI_File_open(MPI_COMM_WORLD, name.data(), MPI_MODE_CREATE|MPI_MODE_WRONLY,
    MPI_INFO_NULL, &file);
MPI_File_set_view(file, offset, MPI_DOUBLE, local_array, "native",
    MPI_INFO_NULL);
MPI_File_write_all(file, real_part.data(), col_size * row_size_local,
    MPI_DOUBLE, &status);
MPI_File_close(&file);
MPI_Type_free(&local_array);
return;
}
```

Listing 5.12: Extern C++ function to fill Fortran arrays with laser fields dumped in binary file

```
void populate_laser_field_on_boundary(double* field, int* id, const char*
    data_dir, const char* name, int* timestep, int* size_global, int* first,
    int* last) {
double num = 0.0;
std::string laser_id = std::to_string(*id);
std::string output_path(data_dir);
std::string filename(name);
std::ifstream in;
in.open(output_path + "/" + filename + laser_id + ".dat", std::ios::binary);
if(in.is_open()) {
in.seekg(((*timestep) * (*size_global) + (*first) - 1) * sizeof(num));
for(auto i = 0; i < *last - *first + 1; i++) {
in.read(reinterpret_cast<char*>(&num), sizeof(num));
field[i] = num;
}
in.close();
} else {
std::cout << "error: cannot read file " << output_path + "/" + filename +
    laser_id + ".dat" << std::endl;
}
return;
}
```

Listing 5.13: Fortran interfaces for C++ library functions

```
INTERFACE

SUBROUTINE compute_laser_fields_on_boundary(rank, nproc, laser_start,
    laser_end, fwhm_time, t_0, omega, pos, amp, w_0, id, L_min, L_max, L_focus
    , T_min, T_max, T_ncells, cpml_thickness, t_end, T_cell_size, L_cell_size,
     dt, output_path) bind(c)
USE, INTRINSIC :: iso_c_binding
IMPLICIT NONE
INTEGER(c_int), INTENT(IN) :: rank, nproc, id, T_ncells, cpml_thickness
CHARACTER(kind=c_char), DIMENSION(*), INTENT(IN) :: output_path
REAL(c_double), INTENT(IN) :: laser_start, laser_end, fwhm_time, t_0, omega,
    pos,    &
amp, w_0, L_min, L_max, L_focus, T_min, T_max, t_end, T_cell_size, L_cell_size
    , dt
END SUBROUTINE compute_laser_fields_on_boundary

SUBROUTINE populate_laser_field_on_boundary(field, laser_id, output_path,
    field_name, timestep, size_global, first, last) bind(c)
USE, INTRINSIC :: iso_c_binding
IMPLICIT NONE
INTEGER(c_int), INTENT(IN) :: laser_id, timestep, size_global, first, last
CHARACTER(kind=c_char), DIMENSION(*), INTENT(IN) :: output_path, field_name
REAL(c_double), DIMENSION(*), INTENT(OUT) :: field
END SUBROUTINE populate_laser_field_on_boundary

END INTERFACE
```

Listing 5.14: Fortran subroutines for Maxwell consistent computation of laser fields on boundaries

```
SUBROUTINE Maxwell_consistent_computation_of_EM_fields

TYPE(laser_block), POINTER :: current

current => laser_x_min
DO WHILE(ASSOCIATED(current))
CALL compute_laser_fields_on_boundary(rank, nproc, current%t_start, current%
    t_end, current%fwhm_time, current%t_0, current%omega, current%pos, current
    %amp, current%w_0, current%id, x_min, x_max, current%focus, y_min, y_max,
    ny_global, cpml_thickness, t_end, dy, dx, dt, TRIM(data_dir)//C_NULL_CHAR)
current => current%next
ENDDO

current => laser_x_max
DO WHILE(ASSOCIATED(current))
CALL compute_laser_fields_on_boundary(rank, nproc, current%t_start, current%
    t_end, current%fwhm_time, current%t_0, current%omega, current%pos, current
    %amp, current%w_0, current%id, x_min, x_max, current%focus, y_min, y_max,
    ny_global, cpml_thickness, t_end, dy, dx, dt, TRIM(data_dir)//C_NULL_CHAR)
```

```
14 current => current%next
15 ENDDO
16
17 current => laser_y_min
18 DO WHILE(ASSOCIATED(current))
19 CALL compute_laser_fields_on_boundary(rank, nproc, current%t_start, current%
      t_end, current%fwhm_time, current%t_0, current%omega, current%pos, current
      %amp, current%w_0, current%id, y_min, y_max, current%focus, x_min, x_max,
      nx_global, cpml_thickness, t_end, dx, dy, dt, TRIM(data_dir)//C_NULL_CHAR)
20 current => current%next
21 ENDDO
22
23 current => laser_y_max
24 DO WHILE(ASSOCIATED(current))
25 CALL compute_laser_fields_on_boundary(rank, nproc, current%t_start, current%
      t_end, current%fwhm_time, current%t_0, current%omega, current%pos, current
      %amp, current%w_0, current%id, y_min, y_max, current%focus, x_min, x_max,
      nx_global, cpml_thickness, t_end, dx, dy, dt, TRIM(data_dir)//C_NULL_CHAR)
26 current => current%next
27 ENDDO
28
29 END SUBROUTINE Maxwell_consistent_computation_of_EM_fields
```

Listing 5.15: Fortran subroutines for populating laser sources on boundaries

```
1  SUBROUTINE get_source_x_boundary(source1, source2, laser_id)
2  REAL(num), DIMENSION(:), INTENT(INOUT) :: source1, source2
3  REAL(num), DIMENSION(ny) :: laser_ex, laser_ey
4  INTEGER, INTENT(IN) :: laser_id
5  INTEGER :: i
6  CALL populate_laser_field_on_boundary(laser_ex, laser_id, TRIM(data_dir)//
      C_NULL_CHAR, "e_x"//C_NULL_CHAR, step, ny_global, ny_global_min,
      ny_global_max)
7  laser_ey = 0.0_num
8  DO i = 1, ny
9  source1(i) = source1(i) + laser_ex(i)
10 source2(i) = source2(i) + laser_ey(i)
11 ENDDO
12 END SUBROUTINE get_source_x_boundary
13
14 SUBROUTINE get_source_y_boundary(source1, source2, laser_id)
15 REAL(num), DIMENSION(:), INTENT(INOUT) :: source1, source2
16 REAL(num), DIMENSION(nx) :: laser_ex, laser_ey
17 INTEGER, INTENT(IN) :: laser_id
18 INTEGER :: i
19 CALL populate_laser_field_on_boundary(laser_ex, laser_id, TRIM(data_dir)//
      C_NULL_CHAR, "e_x"//C_NULL_CHAR, step, nx_global, nx_global_min,
      nx_global_max)
20 laser_ey = 0.0_num
```

```
21 DO i = 1, nx
22 source1(i) = source1(i) + laser_ey(i)
23 source2(i) = source2(i) + laser_ex(i)
24 ENDDO
25 END SUBROUTINE get_source_y_boundary
```

Conclusion

The work briefly presents the introduction to the inertial fusion research as well as basic physical processes which take place during the interaction of intense laser pulse with plasma. Particularly, for better interpretation and understanding of ongoing experiments that study the possibilities of nuclear fusion ignition by shock wave, the conditions of interaction have been set accordingly to them.

The main benefits of this work are successful implementation of boundary conditions for the effective absorption of hot electrons in the two-dimensional version of the computational code EPOCH. Its correct functionality has been later verified by plenty of numerical tests. Afterwards, two large scale simulations of laser system PALS in two-dimensional geometry on its fundamental wavelength 1,315 μm with intensity $1 \cdot 10^{20}\,\mathrm{W/m^2}$ and with initial electron temperatures $\mathrm{T_e} = 0.5\,\mathrm{keV}$ and $\mathrm{T_e} = 2.5\,\mathrm{keV}$ have been performed. Both simulations capture the time period of 20 ps. Simulations have been performed using the particle-in-cell code EPOCH [1]. Initial profiles of plasma density and temperature have been approximated from hydrodynamic simulations, which have been performed previously.

The total absorption of incident laser energy in plasma for the case of the simulation with $\mathrm{T_e} = 0.5\,\mathrm{keV}$ was estimated to 42.4 %, for the case of the simulation with $\mathrm{T_e} = 2.5\,\mathrm{keV}$, the total absorption was significantly lower, about 33.1 %. The temperature of the hot electrons in the case of the simulation with $\mathrm{T_e} = 0.5\,\mathrm{keV}$ was estimated to 36 keV, in the case of the simulation with $\mathrm{T_e} = 2.5\,\mathrm{keV}$ the temperature was about 25 keV. In both cases, the number of hot electrons is relatively low and their temperatures are not too high to prevent the fuel target compression in the later phase. However, it is necessary to further investigate their effect performing more accurate simulations.

Acknowledgments

I wish express my gratitude to both, my supervisor doc. Ing. Ondřej Klimo, Ph.D. and consultant Dr. Stefan Weber for constant support and guidance, as well as for providing invaluable advice and direction.

Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Infrastructure for Research, Development, and Innovations" (LM2010005), is greatly appreciated.

Access to the CERIT-SC computing and storage facilities provided under the programme Center CERIT Scientific Cloud, part of the Operational Program Research and Development for Innovations, reg. no. CZ. 1.05/3.2.00/08.0144, is greatly appreciated.

The development of the EPOCH code was funded in part by the UK EPSRC grants EP/G054950/1, EP/G056803/1, EP/G055165/1 and EP/ M022463/1.

Bc. Petr Valenta

Bibliography

[1] T D Arber, K Bennett, C S Brady, A Lawrence-Douglas, M G Ramsay, N J Sircombe, P Gillies, R G Evans, H Schmitz, A R Bell, and C P Ridgers. Contemporary particle-in-cell approach to laser-plasma modelling. *Plasma Physics and Controlled Fusion*, 57(11):113001, 2015.

[2] C. K. Birdsall and A. B. Langdon. *Plasma Physics via Computer Simulation*. Series in Plasma Physics. CRC Press, 2004.

[3] S. Eliezer. *The Interaction of High-Power Lasers with Plasmas*. Series in Plasma Physics. CRC Press, 2010.

[4] T. Zh. Esirkepov. Exact charge conservation scheme for particle-in-cell simulation with an arbitrary form-factor. *Computer Physics Communications*, 135(2), 2001.

[5] H. Fehske, R. Schneider, and A. Weiße. *Computational Many-Particle Physics*. Springer, 2008.

[6] Kai Germaschewski, William Fox, Narges Ahmadi, Liang Wang, Stephen Abbott, Hartmut Ruhl, and Amitava Bhattacharjee. The plasma simulation code: A modern particle-in-cell code with load-balancing and gpu support. *arXiv preprint arXiv:1310.7866*, 2013.

[7] H. Gould, J. Tobochnik, and W. Christian. *An introduction to computer simulation methods: applications to physical systems*. Addison–Wesley series in physics. Addison–Wesley, 3 edition, 2007.

[8] R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles*. CRC Press, 2010.

[9] D. A. Jaroszynski, R. Bingham, and R. A. Cairns. *Laser-Plasma Interactions*. Scottish Graduate Series. CRC Press, 2009.

[10] G. Lapenta. Particle in cell methods with application to simulations in space weather. Lecture notes.

[11] T. Pang. *An introduction to computational physics*. Cambridge University Press, 2 edition, 2006.

[12] J. Villasenor and O. Buneman. Rigorous charge conservation for local electromagnetic field solvers. *Computer Physics Communications*, 69(2-3), 1992.

[13] K. Yee. Numerical solution of inital boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14(3), 1966.

Appendices

Appendix A

Code listings

Appendix B

Input files

B.1 Simulace

B.2 Simulace

Appendix C

CD Content

directory/file	specification