國立台灣大學理學院物理學研究所

碩士論文

Department of Physics
College of Science
National Taiwan University

Master Thesis

以三維粒子模擬研究相對論性雷射電漿交互作用之非線性光學效應

Three Dimensional Particle-In-Cell Simulations of Relativistic Laser-Plasma Interactions

研究生　謝宗翰　撰
Zong-Han Xie

指導教授：汪治平 博士
Advisor: Dr. Jypyng Wang

中華民國九十八年七月


國立臺灣大學
人格品學・愛國愛人

國立臺灣大學碩士學位論文
口試委員會審定書

以電漿粒子模擬研究相對論性雷射電漿交互作用中的非線性光學現象

Three dimensional particle-in-cell simulations of relativistic laser-plasma interactions

本論文係謝宗翰 君（R95222028）在國立臺灣大學物理學所完成之碩士學位論文，於民國 98 年 07 月 09 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

汪治平 （簽名）

（指導教授）

林宏仁	陳仕泰
曹孝鎧	吳茂昆


國立臺灣大學
人格品學・愛國愛人

誌謝

在物理所的三年當中，我學習到非常多的專業知識，不論是進行研究所需要的工具，或是基本的學理。除了知識以外，也觀察並且學習到很多做事的態度與大型組織運作的方法。隨著論文的付梓，即將畢業的我也準備將這些學識與態度用到社會上。

首先感謝我的指導教授汪老師，他對於模擬設施的進度的要求，讓我可以順利利用三維模擬來完成我的論文。因為有他的要求，所以我可以成為國內第一個完成三維模擬的人。並且，利用三維模擬取得與實驗相對應的成果。此外，他在物理原理方面也給我很多方向上的指導，他和師母曹景懿老師的期刊論文，讓我更易於了解其他雷射電漿交互作用的理論期刊文章。

我還要感謝在中央大學的陳仕宏老師，他在模擬技術方面助我良多。因為他和模擬組在基本模擬技術方面的鑽研，讓我更深入去了解一般電磁學和電漿粒子的模擬方法。這些基本知識也幫助我去了解 VORPAL 軟體的操作，並且對於我建立模擬模型有極大的助益。

感謝原分所的電腦技師林許成先生，他不僅在電腦知識上面教我很多，同時也幫忙照顧模擬的電腦機器，讓我的計算工作可以順利完成。

感謝中央大學的呂凌霄老師和林留玉人老師在物理原理上面給我不少的幫助，他們增廣了我對於模擬方法與基本物理的見識。同時也感謝陳守鍊學長在軟體方面幫我找尋到可以使用的套件，讓我的分析程式可以寫的更順利。

其他我還要感謝實驗室的白值豪學長，他的幫助讓我了解一些實驗上的結果和實驗所用的參數，這對於拿我的模擬和實驗比較有很大的助益。在我的進行與實驗相關的模擬的同時也感謝哈立忠同學所帶來的實驗方面對於其中物理現象的解釋，這增長了我在這個領域上面的見識，也幫助了我去找論文來更進一步了解其中物理現象。此外，也感謝其他實驗室同學在人際關係上面的支持，讓我可以做完這三年的工作。還有以前成大流舞社的同學們，你們在我苦悶的時刻讓我感到紓解和感動。

最後，感謝家裡的人對於我三年經濟上的支持，我可以感覺到你們栽培的用心。因為有你們的支持，我才能完成這個學業。

台大物理研究所碩士班
謝宗翰 先生
獻上誠摯的感謝
2009 年 07 月 25 日


國立臺灣大學
人格品學・愛國愛人

Contents

Abstract

With the developement of the chirped pulse amplification techniques, high-power laser systems with terawatt or petawatt output have been constructed around the world. With these powerful lasers, physics of laser-plasma interactions in the relativistic regime has become the most advanced scientific subject. However, the plasma is a very complicated system, it's hard to derive a single analytical theory which can be applied to all phenomena in laser-plasma interactions. The best method to explore the physical mechanisms in such a complicated system is through computer simulations. Among all the plasma simulation methods, particle-in-cell simulation is the most powerful and straight forward simulation method.

In this thesis, three dimensional PIC simulations of relativistic nonlinear optics have been done by VORPAL code [1]. An introduction of phenomena in relativistic laser-plasma interaction are written in Chapter 1. The algorithms used in PIC simulation as well as the methods used in the post-analyses of simulation data are described in Chapter 2. Computer platforms, software tools and performance benchmarks are described in Chapter 3. Simulations of relativistic nonlinear optics in plasmas according to analytical results derived by Prof. Wang and Prof. Tsaur [2, 3] are shown in Chapter 4. The simulation of generating a strong mid-infrared pulse in a plasma bubble by photon acceleration and self-modulation is written in Chapter 5. The simulation results in Chapter 5 can be successfully compared with experiment results. Conclusions and future perspectives of this work is in Chapter 6.

Chapter 1

Introduction

1.1 Introduction to phenomena in laser-plasma interactions

In recent decades, with the development of the chirped pulse amplification technology, laser systems with terawatt or petawatt power have been built around the world [4, 5]. A high power laser pulse which is tightly focused to a spot of radius on the order of micrometers can produce extremely high peak intensity. The target irradiated by such a laser pulse would be ionized by the front portion of the pulse. When the main pulse arrives, it will interact with the plasma. Velocities of plasma electrons driven by such an intense laser pulse would reach the relativistic region. Therefore, the interaction of intense laser pulses with matter becomes relativistic laser-plasma interaction and it can serve as a variety of light sources or particle accelerators.

Many phenomena in the relativistic laser-plasma interactions have been discovered and studied in recent decades. For example, relativistic self-focusing and self-channeling, electron acceleration, the Raman scattering and the Raman amplification, terahertz and mid-infrared pulse generation, and high harmonic generations, are the most well known subjects in this field.

Relativistic self-focusing and self-channeling are caused by the intensity distribution of the intense laser pulse. The non-uniform laser intensity distribution affects the refractive index through the Lorentz factor of plasma electrons. The refractive index of the plasmas is

$$\eta = \sqrt{1 - \frac{\omega_p^2}{\gamma\omega^2}}. \tag{1.1}$$

The ω_p is the plasma wave frequency. It is

$$\omega_p = \sqrt{\frac{n_e e^2}{\epsilon_0 m_e}}, \tag{1.2}$$

where γ is the Lorentz factor. γ can be expressed as $\sqrt{1 + a^2/2}$ and a is the magnitude of transverse field of the laser pulse. a is written in normalized vector potential which is normalized by $\mathbf{a} \equiv e\mathbf{A}/(m_e c^2)$ and n_e is the plasma density which is a function of coordinates. Therefore, the refractive index at the center of the laser pulse is higher than it at the edge of the laser pulse and the laser pulse would be focused under this condition.

The ponderomotive force of a laser pulse is actually the radiation pressure which acts on charged particles. In the nonlinear regime, it is expressed as Eq. 1.3 [6]

$$\mathbf{F_p} = \frac{-m_e c^2}{2\gamma} \nabla a_\perp^2, \tag{1.3}$$

where $a_\perp$ is the transverse laser field in normalized vector potential. The vector potential is normalized by $\mathbf{a} \equiv e\mathbf{A}/(m_e c^2)$. The ponderomotive force of the laser pulse on the radial direction would expel electrons from the their original positions. A radial density channel is created, this plasma channel would enhance self-focusing. This is called ponderomotive self-channeling [7].

A preformed plasma channel can also serve as a waveguide for the laser pulse. In that case, a parabolic plasma density profile on the transverse direction relative to the laser propagation direction would be produced before the main laser beam arrived. The plasma density distribution would creat refractive index distribution similar to the one described above. Therefore, a laser pulse can be guided inside.

When a tightly focused, ultrashort laser pulse propagates through a plasma slab, the ponderomotive force would expel most of the plasma electrons from their original positions on the center of the pulse and a positive charged plasma bubble is formed [8]. This plasma bubble is a large amplitude plasma wave and its phase velocity is equal to the group velocity of the pump laser pulse which is approaching to light speed. When the electrostaic field inside the bubble is larger than the relativistic wave breaking field, the plasma wave breaking happens. Electrons can be trapped inside the bubble when the trapping threshold is reached. This plasma bubble can sustain a high acceleration gradient. The electric field for accelerating electrons in LWFA can be as high as 100 GV/m under 10^{18} cm^{-3} plasma density. Therefore, high energy, monoenergetic, ultrashort electron bunches can be produced by this method. Various simulations and experiments have reported observations of accelerated electron bunches with energies on the order of 100 MeV [29, 41, 43, 45] or GeV [51, 56].

The Raman instability is a basic phenomenon in all laser-plasma interactions [12, 47, 48]. It is an optical parametric process caused by the interactions of the laser pulse with the plasma wave. The frequency of the scattered light is the difference of the plasma wave frequency ω_p and the laser frequency ω_0. The Raman amplification takes advantage of this parametric process. When a seed laser pulse counter-propagates with a pump laser pulse in a plasma slab and the difference of the frequencies of these two laser pulses matches the plasma frequency, the amplitude of the seed pulse would grow fast and become a ultrashort, high intensity laser pulse [58]. This could be the next generation of the laser amplification technology.

High harmonic generation has always been the most important subject in laser-plasma interactions. The purpose of generating high order harmonics is to produce coherent, short wavelength light sources of wavelength on the order of nanometers. There are several ways to produce such light sources. First, it can be produced with the coherent Thomson scattering of a laser pulse propagating through a plasma slab [2, 14]. Experiments of high order harmonic generations in underdense plasmas through the coherent Thomson scattering have been done [25, 26, 27, 28]. However, the energy conversion efficiency is low, and it is hard to generate higher order harmonic signals. The second way is through the laser-solid interactions [49]. A flying and

oscillating surface with overcritical plasma density would upshift the central frequency of the incident laser pulse by the Doppler shift. The advantage of this method is that it can produce higher order harmonics, but it requires high energy laser pulses and a laser system with good contrast.

Terahertz radiation can be generated through the relativistic optical rectification in plasmas. It can also be produced by the transient plasma wave current in a plasma slab [21, 22, 23] or in the plasma density ramp [20]. Mid-infrared pulses can be generated with free electron lasers or the nonlinear evolution of the pump pulse in a plasma bubble. The MIR pulse generating in a plasma bubble through nonlinear propagation of the laser pulse will be studied in Chapter 5.

1.2 Introduction to the simulations of laser-plasma interactions

In order to study the dynamic process of a complex system, numerical simulation is the best tool to provide semi-theoretical solutions of complex systems. In a plasma slab, the system is governed by the Lorentz force equation, the Maxwell equations, the continuity equation, the Vlasov equation (or the Boltzman equation) and the fluid equation of motion. Among them, the Lorentz force equation reads

$$\frac{d\mathbf{P}}{dt} = q\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right). \tag{1.4}$$

The Vlasov equation reads

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla f \frac{q}{m}\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right) \frac{\partial f}{\partial \mathbf{P}} = 0 \tag{1.5}$$

where f is the distribution function of charged particles, $\mathbf{P}$ is the momentum of charged particles and it can be expressed as $\mathbf{P} = \gamma m \mathbf{v}$. m os the mass of the charged particles. $\mathbf{v} = \frac{\mathbf{P}/m}{\sqrt{1+P^2/(m^2c^2)}}$ is the velocity of the charged particle. q is the charge of a charged particle and m is the mass of the charged particles. The fluid equation of motion reads

$$mn\left[\frac{\partial \mathbf{u_f}}{\partial t} + (\mathbf{u_f} \cdot \nabla)\mathbf{u_f}\right] = qn\left(\mathbf{E} + \mathbf{u_f} \times \mathbf{B}\right) \tag{1.6}$$

where $\mathbf{u_f}$ is the velocity of a fluid element, m is the mass of charged particles, n is the number density of charged particles and q is the charge of a charged particle. The course of simulations of plasmas is to self-consistently solve the Maxwell equations and Eqs. 1.4 – 1.6.

There are three major methods to simulate plasmas. First, the fluid method uses the fluid equation of motion, continuity equation, and the Maxwell equations to self-consistently solve the electromagnetic field and the fluid field. However, this method can not simulate the conditions of plasma wave breaking. The Vlasov code uses the Vlasov equation, the continuity equation and the Maxwell equations to self-consistently solve the electromagnetic field and the plasma distributions. In order to solve the plasma distribution, one needs to discretize the entire phase space since the distribution in the Vlasov equation is a function of the velocity and position. This would dramatically increase the computational complexity. Generally, more than 10^{12} cells are needed to perform Vlasov simulations. Therefore, it is impossible to perform such simulations due to the restrictions of current computational resources.

The particle-in-cell simulation (PIC simulation for short) of plasmas can simulate the dynamic processes of all phenomena described in section 1.1. For example, the plasma bubble structure in Fig. 1.1 is hard to be detected in experiments due to technical restrictions.

This method has been developed for several decades and widely applied to many laser-plasma related researches. It is straight forward, easy to implement, and widely suitable on many relativistic laser-plasma interactions. A variety of PIC codes have been written [1, 15, 16, 17, 18] and used on studying the plasma physics. So far, PIC simulations have been intensively performed by many groups and produce a lot of significant results in the study of laser-wakefield electron accelerator [29, 41, 42, 43, 45], laser-solid interactions [15, 49] and nonlinear optics [20, 21, 47, 48, 50].

Vorpal code [1] is a simulation framework written with C++. It aims at versatility with no loss of performance. It can perform multidimensional PIC simulations, therefore, there is no need for users to write another code to perform 2D or 1D simulations which are good at acquiring qualitatively

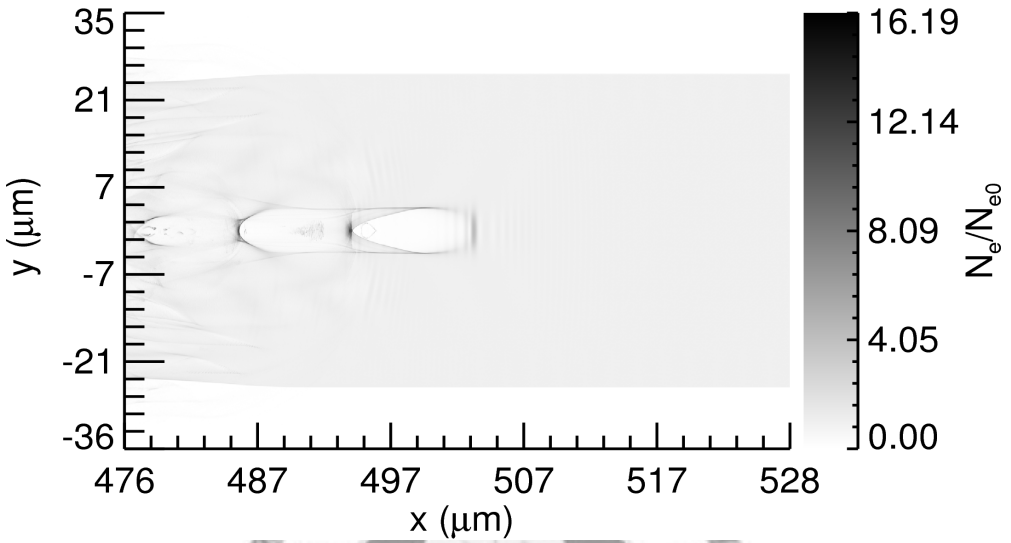


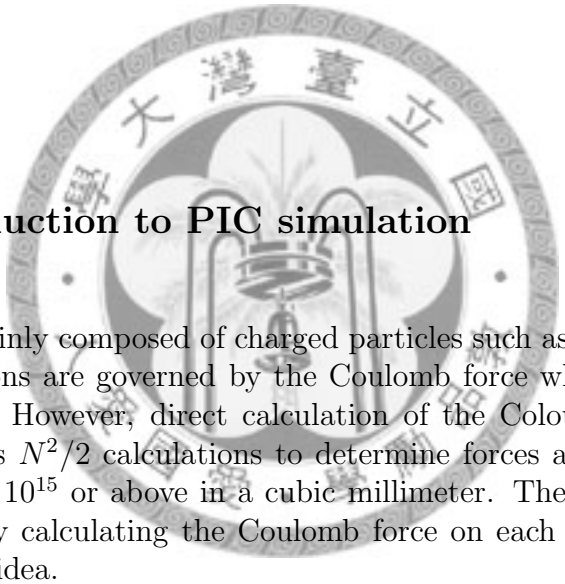
Figure 1.1: A demonstration of the plasma bubble in a 2D PIC simulation done with Vorpal code. The ambient plasma density is $n_0 = 3.4 \times 10^{19}$ cm^{-3}. The wavelength of the incident laser pulse is 810 nm and a_0 of the incident laser pulse is 2.0.

matched results with lower computational complexity. It can also import functions written with Python to produce a complex plasma density or dielectric structures.

This code have been validated with experiments [41] and it has been used on the study of LWFA [41, 44], ion acceleration [46] and nonlinear propagation of intense laser pulse in a plasma channel.

Chapter 2

Methods of PIC simulation and signal analysis



2.1 Introduction to PIC simulation

Plasmas are mainly composed of charged particles such as ions and electrons and their motions are governed by the Coulomb force when we neglect the magnetic field. However, direct calculation of the Coloumb force on each particle requires $N^2/2$ calculations to determine forces and N is, typically, on the order of 10^{15} or above in a cubic millimeter. The attempt to model such systems by calculating the Coulomb force on each particle directly is not a practical idea.

In PIC simulation, the electromagnetic field as well as the density and current fields of charged particles are put on grid points and solved self-consistently with the Maxwell equations by a finite-difference time-domain (FDTD) method. The particles now interact with field components on the grid points while each particle carries their own position and velocity moving through the continuous phase space. This can dramatically reduce the computational complexity. The Lorentz force equation is generally used to push particles in PIC simulation.

Another important feature of PIC simulation is to use "finite-size particle" or "charged cloud" representing roughly tens of thousands of real particles. This feature would reduce short-range forces between charged particles. Therefore, only the collective behavior of plasmas is considered in PIC simulation, the collisional effect is reduced. To deal with particle-mesh interactions, an interpolation method is needed to handle the quantities on the continuous phase space and the quantities on grid points.

A sketch of a typical time step in a PIC code is presented in Fig. 2.1. Firstly, the positions and velocities of charged particles are initially loaded to the program as an initial condition. In this thesis, plasmas are initially cold and quasineutral. And then the charges and velocities of charged particles are interpolated from particles to grid points according to their positions to get the charge density and current fields of the plasma. After the charge density and current fields on grid points are obtained, the electromagnetic field can be updated. After new values of the electromagnetic field is acquired, they are interpolated from grid points to the positions of particles. Acceleration on each macroparticle is solved with the Lorentz force equation and the position and velocity of charged particles can now be updated. With new particle positions and velocities, one can acquire new density and current fields of charged particles. Therefore, the above procedures except the initial condition part can be restarted again.

2.2 Integration of the field equations

Electromagnetic simulation self-consistently solves the Faraday's and Ampere's laws. An explicit, time-centered, finite-difference time-domain method developed by Kane S. Yee [9] has been widely applied in electromagnetic simulation programs including the Vorpal code [1]. In the Yee algorithm, positions of electromagnetic field components are arranged like Fig. 2.2 to adopt the central difference method.

From Faraday's and Ampere's laws, we have following equations.

$$-\frac{\partial B_x}{\partial t} = \frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z}, \tag{2.1}$$

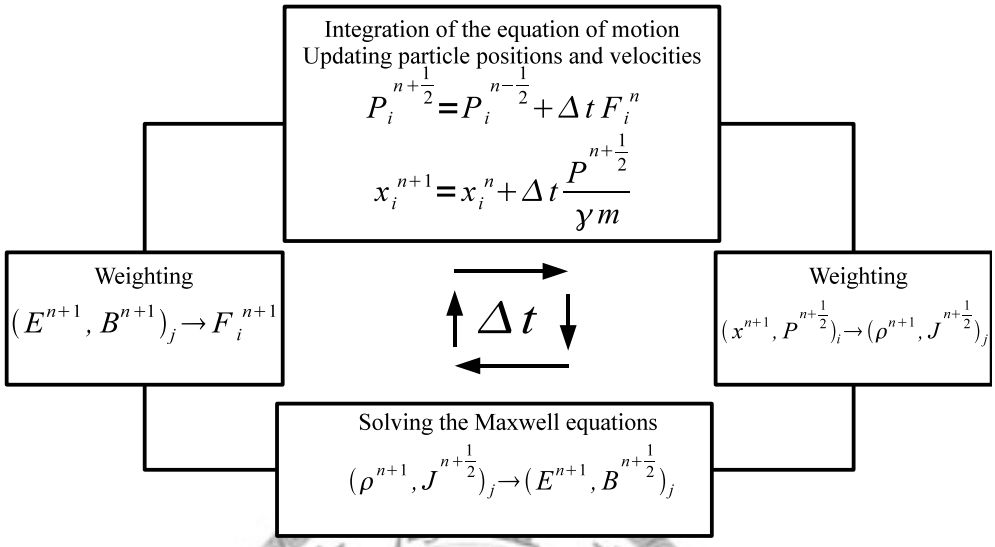


Figure 2.1: A sketch of a typical time step of a PIC simulation program [61]. The particles are numbered as i; the grid indices are labeled as j and the time step is labeled by n.

$$-\frac{\partial B_y}{\partial t} = \frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x}, \tag{2.2}$$

$$\frac{\partial B_z}{\partial t} = \frac{\partial E_x}{\partial y} - \frac{\partial E_y}{\partial x}, \tag{2.3}$$

$$\frac{\partial D_x}{\partial t} = \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} - J_x, \tag{2.4}$$

$$\frac{\partial D_y}{\partial t} = \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} - J_y, \tag{2.5}$$

$$\frac{\partial D_z}{\partial t} = \frac{\partial H_x}{\partial y} - \frac{\partial H_y}{\partial x} - J_z. \tag{2.6}$$

In the Yee algorithm, the electromagnetic field components are separated by a half time step, combined with their alternative spatial positions, Yee has

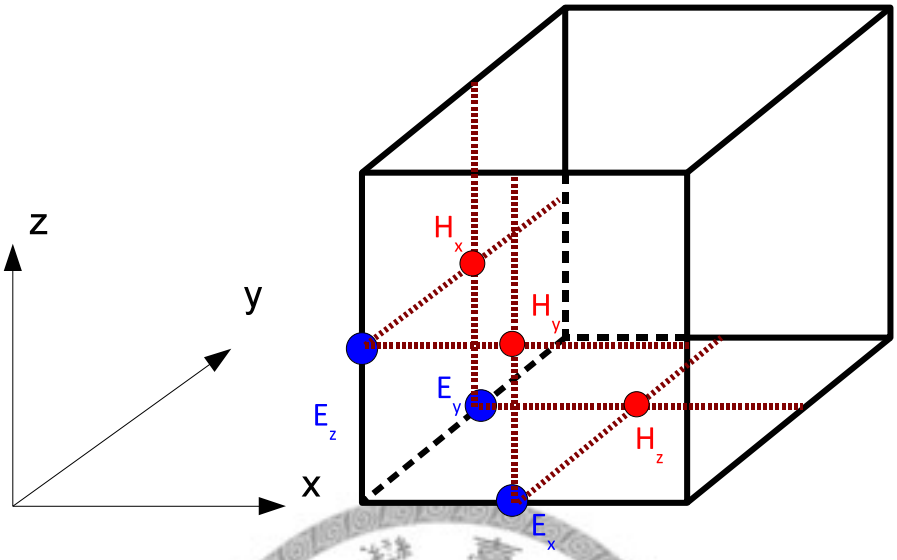


Figure 2.2: Positions of all six electromagnetic field components in the Yee algorithm. The electric field components are in the middle of the edges and the magnetic field components are in the center of the faces.

constructed a set of equations which are adopted with the central difference method in both space and time. From Eq. 2.1 to Eq. 2.6, we have

$$\begin{aligned} & \frac{B_x^{n+\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2}) - B_x^{n-\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2})}{\Delta t} \\ = & \frac{E_y^n(i,j+\frac{1}{2},k+1) - E_y^n(i,j+\frac{1}{2},k)}{\Delta z} \\ - & \frac{E_z^n(i,j+1,k+\frac{1}{2}) - E_z^n(i,j,k+\frac{1}{2})}{\Delta y}. \end{aligned} \tag{2.7}$$

$$\begin{aligned} & \frac{B_y^{n+\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2}) - B_y^{n-\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2})}{\Delta t} \\ = & \frac{E_z^n(i+1,j,k+\frac{1}{2}) - E_z^n(i,j,k+\frac{1}{2})}{\Delta x} \end{aligned}$$

$$- \quad \frac{E_x^n(i+\frac{1}{2},j,k+1) - E_x^n(i+\frac{1}{2},j,k)}{\Delta z}. \tag{2.8}$$

$$\begin{aligned} & \frac{B_z^{n+\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k) - B_z^{n-\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k)}{\Delta t} \\ = \quad & \frac{E_x^n(i+\frac{1}{2},j+1,k) - E_x^n(i+\frac{1}{2},j,k)}{\Delta y} \\ - \quad & \frac{E_y^n(i+1,j+\frac{1}{2},k) - E_y^n(i,j+\frac{1}{2},k)}{\Delta x}. \end{aligned} \tag{2.9}$$

$$\begin{aligned} & \frac{D_x^n(i+\frac{1}{2},j,k) - D_x^{n-1}(i+\frac{1}{2},j,k)}{\Delta t} \\ = \quad & \frac{H_z^{n-\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k+1) - H_z^{n-\frac{1}{2}}(i+\frac{1}{2},j-\frac{1}{2},k+1)}{\Delta y} \\ - \quad & \frac{H_y^{n-\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2}) - H_y^{n-\frac{1}{2}}(i+\frac{1}{2},j,k-\frac{1}{2})}{\Delta z} \\ + \quad & J_x^{n-\frac{1}{2}}(i+\frac{1}{2},j,k). \end{aligned} \tag{2.10}$$

$$\begin{aligned} & \frac{D_y^n(i,j+\frac{1}{2},k) - D_y^{n-1}(i,j+\frac{1}{2},k)}{\Delta t} \\ = \quad & \frac{H_x^{n-\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2}) - H_x^{n-\frac{1}{2}}(i,j-\frac{1}{2},k-\frac{1}{2})}{\Delta z} \\ - \quad & \frac{H_z^{n-\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k) - H_z^{n-\frac{1}{2}}(i-\frac{1}{2},j+\frac{1}{2},k)}{\Delta x} \\ + \quad & J_y^{n-\frac{1}{2}}(i,j+\frac{1}{2},k). \end{aligned} \tag{2.11}$$

$$\begin{aligned} & \frac{D_z^n(i,j,k+\frac{1}{2}) - D_z^{n-1}(i,j,k+\frac{1}{2})}{\Delta t} \\ = \quad & \frac{H_x^{n-\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2}) - H_x^{n-\frac{1}{2}}(i,j-\frac{1}{2},k+\frac{1}{2})}{\Delta y} \end{aligned}$$

$$- \quad \frac{H_y^{n-\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2}) - H_y^{n-\frac{1}{2}}(i-\frac{1}{2},j,k+\frac{1}{2})}{\Delta x}$$
$$+ \quad J_z^{n-\frac{1}{2}}(i,j,k+\frac{1}{2}). \tag{2.12}$$

In vacuum, $\mathbf{D} = \epsilon_0 \mathbf{E}$ and $\mathbf{B} = \mu_0 \mathbf{H}$

The stability of this FDTD scheme requires the Courant-Levy stability criterion to be satisfied. It is

$$\Delta t \leq \frac{\Delta L}{c}, \tag{2.13}$$

where Δt is the time forwarding for a time step and c is the light speed in vacuum. ΔL is defined as following.

$$\Delta L = \Delta x \tag{2.14}$$

where Δx is the grid spacing. Two dimensional and three dimensional forms of Eq. 2.14 are shown in Eqs. 2.15 – 2.16.

$$\frac{1}{\Delta L} = \sqrt{\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right)}, \tag{2.15}$$

$$\frac{1}{\Delta L} = \sqrt{\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}\right)}, \tag{2.16}$$

where Δy and Δz are the grid spacings along the y and z directions.

2.3 Integration of the Equation of Motion

In PIC simulation, particle motions are governed by the Lorentz force equation. An explicit scheme developed by Boris [61] to advance particles with the Lorentz force equation is discussed in this section.

The Lorentz force equation to be integrated is

$$m\frac{d\mathbf{u}}{dt} = q\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right), \tag{2.17}$$

where $\mathbf{u} \equiv \gamma\mathbf{v}$ and γ is the Lorentz factor of a particle. Eq. 2.17 can be written as

$$\frac{\mathbf{u}^{n+1/2} - \mathbf{u}^{n-1/2}}{\Delta t} = \frac{q}{m}\left(\mathbf{E}^n + \frac{\mathbf{u}^{n+1/2} + \mathbf{u}^{n-1/2}}{2\gamma^n} \times \mathbf{B}^n\right). \tag{2.18}$$

The electric field accelerates charged particles and the magnetic field rotates them. The acceleration by the electric field is carried out with

$$\begin{aligned} \mathbf{u}^{n-1/2} &= \mathbf{u}^- - \frac{q\mathbf{E}^n\Delta t}{2m}, \\ \mathbf{u}^{n+1/2} &= \mathbf{u}^+ + \frac{q\mathbf{E}^n\Delta t}{2m}. \end{aligned} \tag{2.19}$$

Substituting Eq. 2.19 into Eq. 2.18, one can obtain

$$\frac{\mathbf{u}^+ - \mathbf{u}^-}{\Delta t} = \frac{q}{2\gamma^n m}\left(\mathbf{u}^+ + \mathbf{u}^-\right) \times \mathbf{B}^n. \tag{2.20}$$

The updates from $\mathbf{u}^-$ to $\mathbf{u}^+$ is done by following procedures.

$$\begin{aligned} \mathbf{u}' &= \mathbf{u}^- + \mathbf{u}^- \times \mathbf{t}^n, \\ \mathbf{u}^+ &= \mathbf{u}^- + \mathbf{u}' \times \mathbf{s}^n, \end{aligned} \tag{2.21}$$

where $\mathbf{t}^n$ and $\mathbf{s}^n$ are defined as following.

$$\begin{aligned} \mathbf{t}^n &\equiv \frac{q\mathbf{B}^n\Delta t}{2\gamma^n m}, \\ \mathbf{s}^n &\equiv \frac{2\mathbf{t^n}}{(1 + t^{n2})}. \end{aligned}$$

The rotation part of the Boris scheme is illustrated in Fig. 2.3. It must be noted that $|\mathbf{u}^+| = |\mathbf{u}^-|$ and the derivation of it can be found in Ref. [62]. Therefore, the Boris scheme conserves the momentum of a particle during the rotation and $\gamma^{n2} = 1 + (\mathbf{u}^-/c)^2 = 1 + (\mathbf{u}^+/c)^2$. The angle of the rotation of a particle carried out by the Boris scheme can be found in section 4-3 of Ref. [61] and it can be expressed as

$$\theta = 2\arctan\frac{qB}{m}\frac{\Delta_t}{2} = \omega_c\Delta_t\left(1 - \frac{(\omega_c\Delta_t)^2}{12} + \cdots\right), \tag{2.22}$$

where $\omega_c = \frac{qB}{m}$ is the cyclotron frequency of a particle driven by the magnetic field B. When $\omega_c \Delta_t$ is less than 0.35, the relative error of the rotation angle of the Boris scheme can be less than 1%. The overall momentum integration is accurate to the second order.

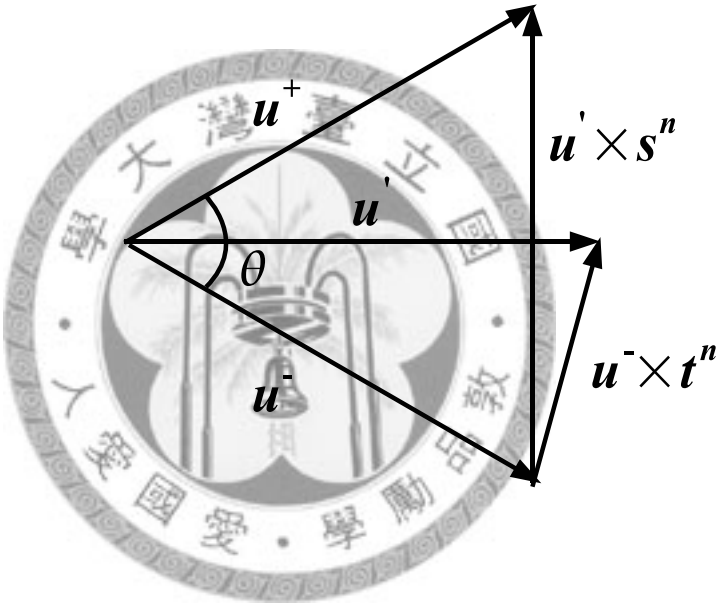


Figure 2.3: An illustration of the rotation part in the Boris scheme. θ represents the rotation angle of a particle in a time step.

2.4 Particle and field weighting

In PIC simulation, the current and density fields of charged particles are put on grid points but the particle position as well as the particle velocity are on continuous space coordinates. Solving field components requires the current density field from position and velocity of charged particles; updating particle positions need to calculate the force acting on particles from the electromagnetic field located on grid points. Therefore, one needs a weighting scheme to deposit the particle position and velocity to the density and current fields on grid points. Inversely, the electromagnetic field on grid points is interpolated to the positions of particles in order to calculate the forces acting on charged particles.

In most cases, we use an uniform cloud-shape particle [61] model, it's called the "cloud-in-cell" model and the corresponding interpolation method is a simple linear interpolation. Fig. 2.4 shows a demonstration to the uniform charged cloud centered at x_i. If we have a macroparticle at x_i, the charge distribute to the point X_j is

$$q_j = q_i \frac{X_{j+1} - x_i}{X_{j+1} - X_j}. \tag{2.23}$$

The particle velocity is interpolated to grid points with the same way, therefore we have charge and current fields on grid points. Another weighting scheme is to calculate the electromagnetic field at the particles from the electromagnetic field components on grid points. Two weighting methods must be the same to avoid the self-force on particles. If the charge and current fields are determined by Eq. 2.23, the electric field components on particles from the electric field on grid points should be calculated as follows.

$$E_i = E_j \frac{X_{j+1} - x_i}{X_{j+1} - X_j} + E_{j+1} \frac{x_i - X_j}{X_{j+1} - X_j} \tag{2.24}$$

Fig. 2.5 explicitly shows the charge and current contributions from charged particles to the grid points. In 2D simulations, this linear interpolation scheme would be expanded to the area weighting method [61] which is shown in Fig. 2.6. A charged particle carries charge q located at (x_1, x_2). The charge

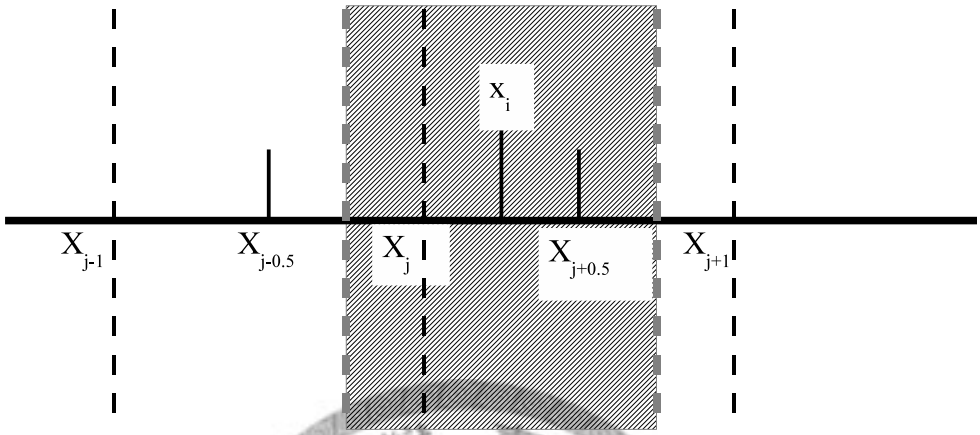


Figure 2.4: The charged particle cloud of linear weighting scheme. The charged particle is centered at x_i. [61]

at 4 nearest grid points (i, j), $(i, j+1)$, $(i+1, j)$ and $(i+1, j+1)$ are decided as Eq. 2.26

$$
\begin{aligned}
q_{i,j} &= q\frac{\Delta x_1 - w_1}{\Delta x_1}\frac{\Delta x_2 - w_2}{\Delta x_2} \\
q_{i+1,j} &= q\frac{w_1}{\Delta x_1}\frac{\Delta x_2 - w_2}{\Delta x_2} \\
q_{i,j+1} &= q\frac{\Delta x_1 - w_1}{\Delta x_1}\frac{w_2}{\Delta x_2} \\
q_{i+1,j+1} &= q\frac{w_1}{\Delta x_1}\frac{w_2}{\Delta x_2}
\end{aligned}
\tag{2.25}
$$

This method can be similarly expanded to a 3D version.

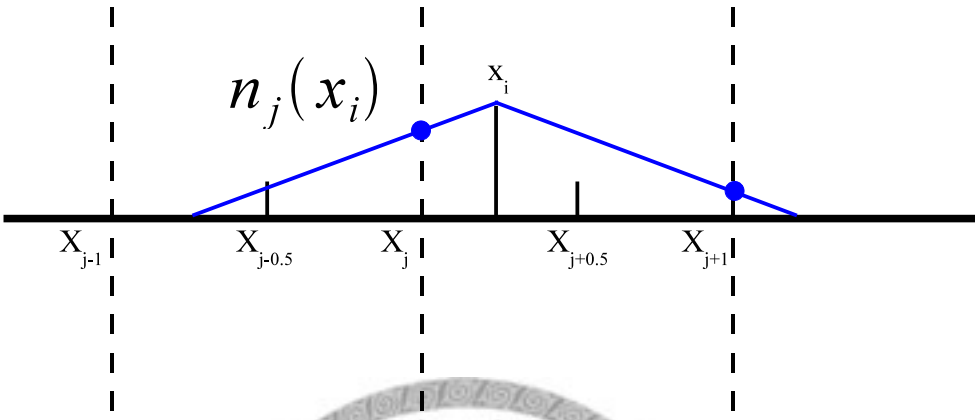


Figure 2.5: The charged particle shape of cloud-in-cell model. A particle centered at x_i has charge density and current contributions to X_j and X_{j+1}. Density and current fields on grid points from position and velocity of charged particles are determined by a linear interpolation. [61]

Some higher order particles involving smoother shapes and more grid points such as "quadratic" or "cubic spline" can be found in Ref. [30] and Ref. [61]. Some works on improving conservation of charge by modifying the weighting scheme has been published such as Ref. [31] and that model has been adopted in the Vorpal code [1].

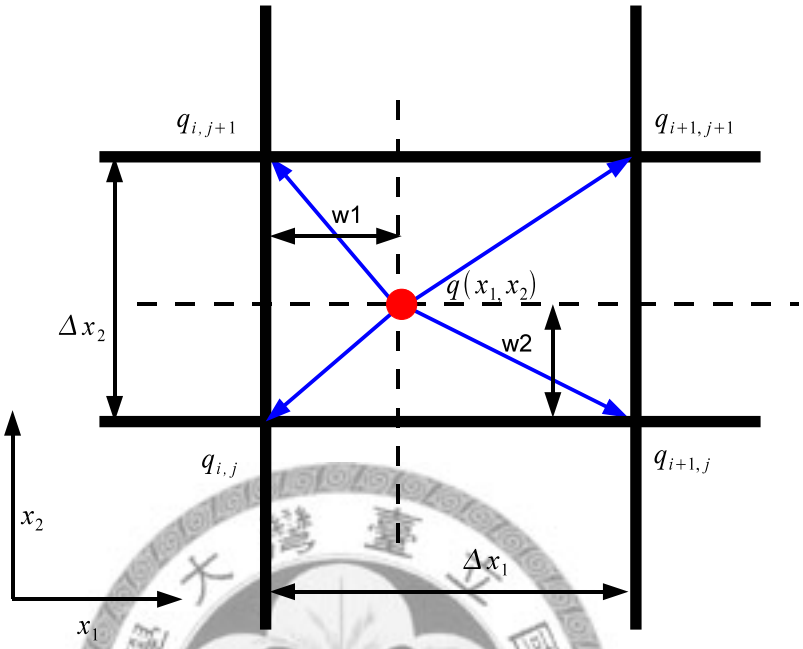


Figure 2.6: A demonstration of the linear interpolation scheme in a two dimensional space. A charged particle (red circle) carries charge q located at (x_1, x_2). The charges deposited to the 4 nearest grid points are $q_{i,j}$, $q_{i+1,j}$, $q_{i,j+1}$, $q_{i+1,j+1}$.

2.5 Post-processing of simulation signals

2.5.1 Introduction to discrete fourier transform

The discrete Fourier transform (DFT) is a major tool in signal analysis and it was frequently used throughout this thesis. In this section, basic properties of signal processing with the DFT are introduced.

Assuming we have a set of discrete data $h_k = h(t_k)$ which is sampled from a continuos function $h(t)$ and $t_k = k \times \Delta$, where Δ is the sampling interval

and k is an integer. The discrete Fourier transform of these data is Ref.[63],

$$H_n = \sum_{k=0}^{N-1} h_k e^{2\pi ikn/N}, \tag{2.26}$$

where N is total sampling numbers, n is the frequency bin, k is the index of each sampled data point and $f_n = n/(\Delta \times N)$.

A special frequency called the "Nyquist critical frequency" is defined as $f_c = 1/(2\Delta)$ at a sampling interval Δ. This value represents the highest frequency component which can be resolved at a sampling interval Δ. Alternatively, it means that "Critical sampling of a sine wave is two points per cycle" [63]. A remarkable fact known as the sampling theorem states "if a continuos function $h(t)$, sampled at an interval Δ, happens to be bandwidth limited to frequencies smaller than f_c in magnitude, then the function $h(t)$ can be completely determined by its samples h_n" [63].

On the contrary, if a continuous function $h(t)$ is not bandwidth limited to be less than the Nyquist frequency in magnitude and then the "aliasing" effect would occur. It is basically the power outside of the Nyquist frequency f_c is folded into the region with frequencies between f_c and $-f_c$. This effect can be easily realized with Eq. 2.26 by checking $H_{n+N} = H_n$. High frequency components close to f_c are actually the mixture of real spectrum and the power from another spectrum cycle centered at $2f_c$. A simple demonstration of the aliased spectrum is shown in Fig. 2.7.

In IDL, the DFT for a given data set $\{h_k\}$ is defined as

$$H_n = \frac{1}{N} \sum_{k=0}^{N-1} h_k e^{2\pi ikn/N}, \tag{2.27}$$

where N is the total number of the elements in $\{h_k\}$. All power spectrum density (PSD) plots in this thesis were taken from square of the absolute value of the spectrum data which is transformed by IDL "fft" routine. Components in the negative frequency region were abandoned since they were the complex conjugate of components in the positive frequency region, the absolute values of them would be the same. Therefore, according to Eq. 2.27, all the PSD in this thesis for a given N elements data set $\{h_k\}$ are defined as $P(f_n) = |H_n|^2$ where $\{f_n\}$ is ranging from 0 to the Nyquist frequency.

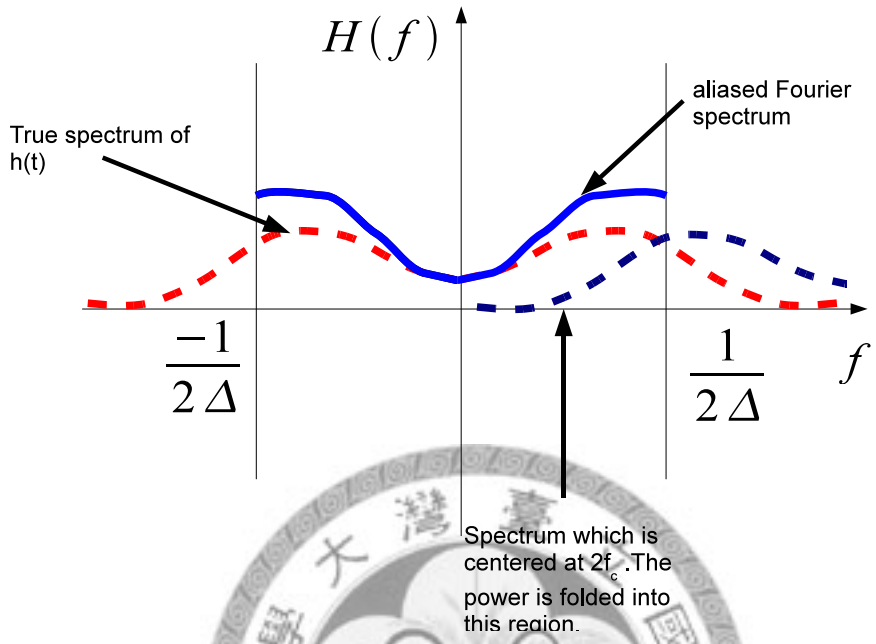


Figure 2.7: Power outside of $f_c = 1/(2\Delta)$ is "aliased" into the region with frequencies between f_c and $-f_c$.

Now we start to discuss the data windowing. When a continuous function $h(t)$ is given and we sample the function to a discrete data set $\{h_k\}$ at an interval Δ. The PSD value at frequency f_n obtained from the discrete Fourier transform does not equal to $P(f_n)$ obtained from the continuous Fourier transform of $h(t)$. There is always certain power leakage from neighborly frequency bins [63]. Data windowing can reduce this effect by multiplying the sampled data with a window function. Various window functions like the "Hann window", "Barlett window", "Welch window" and the square window can be found in Ref. [63]. The function shapes and their characteristics on the power leakage are shown in section 13.4 of Ref. [63]. Clearly, the "Hann window" can reduce most of the power leakage from neighborly frequency bins. Therefore, it is adopted in my spectral analysis programs and was used in parts of the power spectrum analyses in this thesis.

2.5.2 Digital signal filters used in this thesis

Another important application of the Fourier transform is the signal filtering. One can obtain a signal with certain frequency band from a Fourier spectrum by applying a mask function on the spectrum. This mask function can set the power of other frequency components to zero. The inverse Fourier transform of the shaded spectrum is the signal with the desired frequency band. In this thesis, The Gaussian or Super-Gaussian shape mask functions were used on the digital signal filtering. Different power Gaussian functions were shown in Fig. 2.8. Bandwidth of a mask function is defined privately as full width between e^{-1} of it.

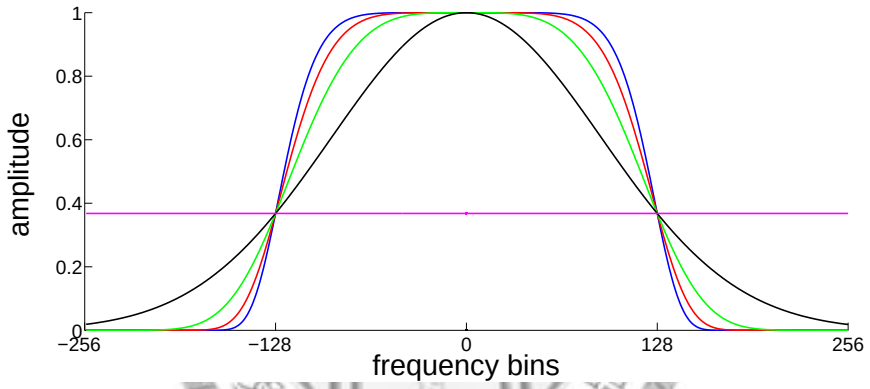


Figure 2.8: The black line is the shape of the e^{-x^2} function; green line is the shape of the e^{-x^4} function; the red line is the shape of the e^{-x^6} function; the blue line is the shape of the e^{-x^8} function. The Magenta line represents the e^{-1} amplitude and the distance between two intersections of the magenta line and the shape functions is the filter bandwidth.

An analytical Gaussian beam in one dimensional form is expressed as the following equation.

$$A(x) = A_0 e^{\frac{x^2}{2L^2}} e^{-ik_0 x} \tag{2.28}$$

The Fourier transform of the above equation is

$$A(k) = \frac{A_0}{\sqrt{1/\tau^2}} e^{\frac{(k-k_0)^2}{2/L^2}}. \tag{2.29}$$

If one define the spatial width of Eq. 2.28 as $\Delta x = 2\sqrt{2}L$ and the width of Eq. 2.29 as $\Delta k = 2\sqrt{2}/L$. It can be found that $\Delta x \Delta k = 8$. This formula can help us to determine the bandwidth of an ideal Gaussian beam.

One dimensional DFT tests have been done with Matlab to show the differences between the Gaussian and the 8th power Super-Gaussian filters. An analytical Gaussian beam with $\lambda = 810$ nm has been used and it was sampled at an interval of 32.4 cells per cycle. The duration of this Gaussian beam is 45 (fs) and it applies that $\Delta k = 3.49 \times 10^5$ m^{-1}. Comparisons between the Gaussian and the 8th power Super-Gaussian filters are shown below in Figs. 2.9 – 2.11. The bandwidths of the filters are taken as 3.49×10^5 m^{-1}, 6.98×10^5 m^{-1} and 1.0×10^6 m^{-1} and they are equal to Δk, $2\Delta k$ and $3\Delta k$. Red lines are the original analytical Gaussian pulse and blue lines are the filtered Gaussian pulse. The results show that the 8th power Super-Gaussian filter is better than the Gaussian filter when the same bandwidth is used.

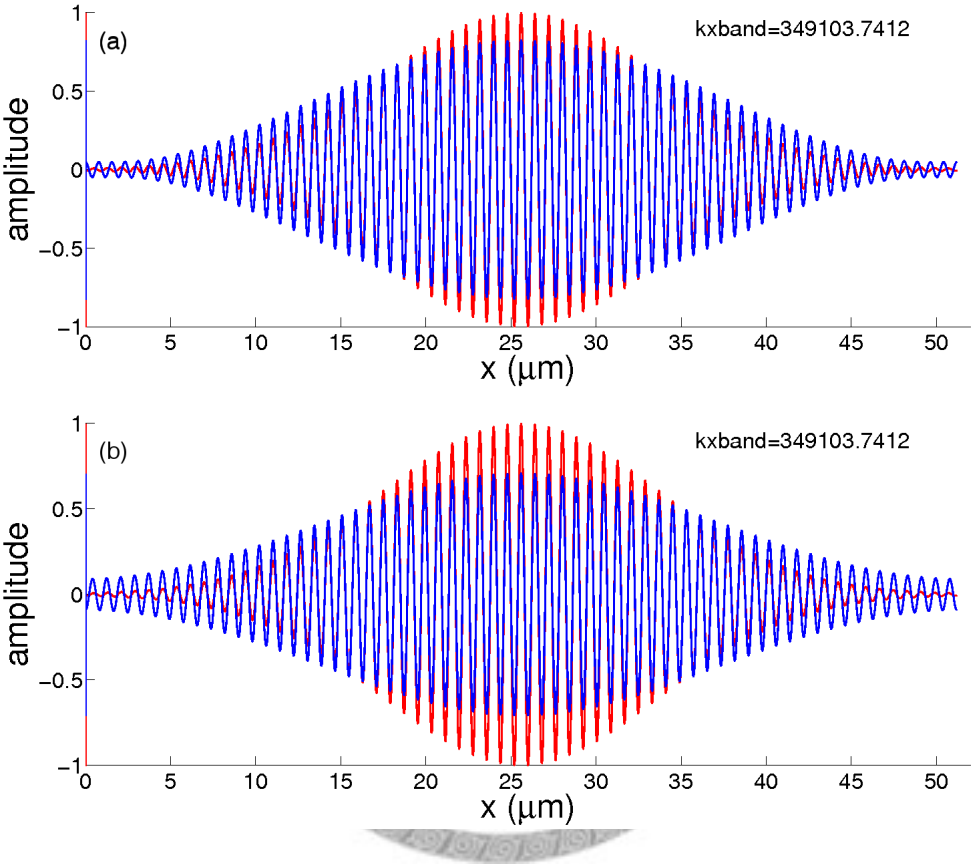


Figure 2.9: The red line is the original Gaussian beam. the filtered Gaussian beam is presented by the blue line. The blue line in the upper plot is the signal filtered by the 8th power Super-Gaussian filter. The blue line in the lower plot is the signal filtered by the Gaussian filter. The filter bandwidth is taken as 3.49×10^5 m^{-1}.

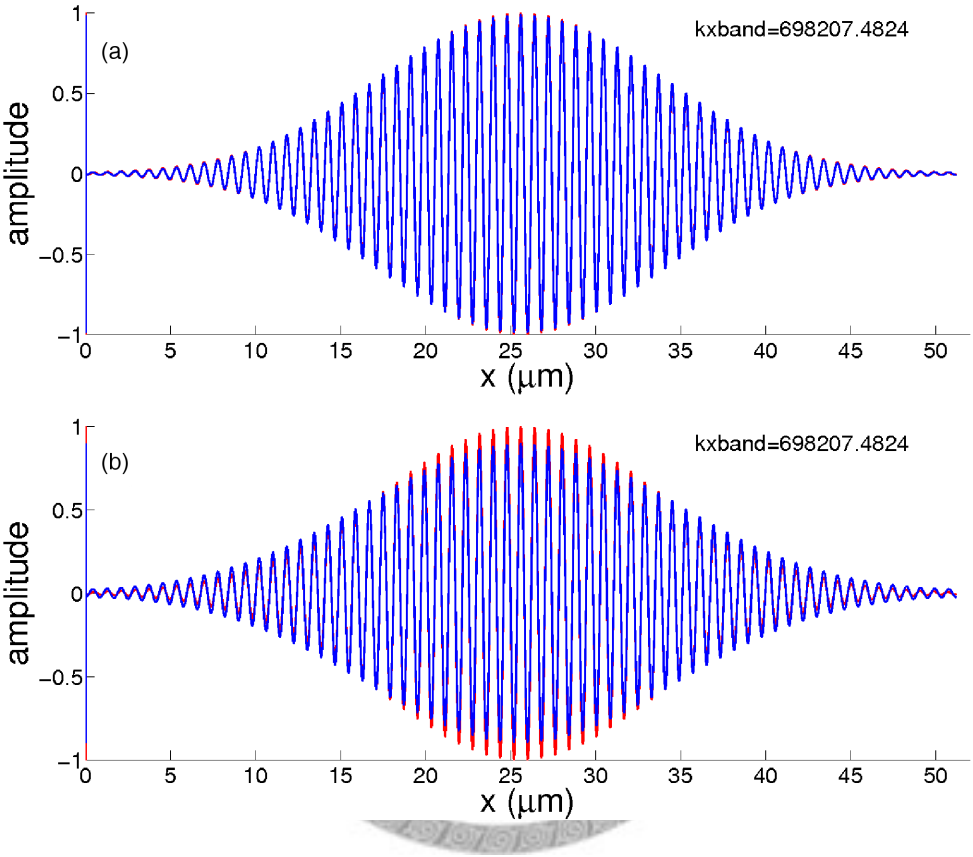


Figure 2.10: The red line is the original Gaussian beam. the filtered Gaussian beam is presented by the blue line. The blue line in the upper plot is the signal filtered by the 8th power Super-Gaussian filter. The blue line in the lower plot is the signal filtered by the Gaussian filter. The filter bandwidth is taken as 6.98×10^5 m^{-1}.

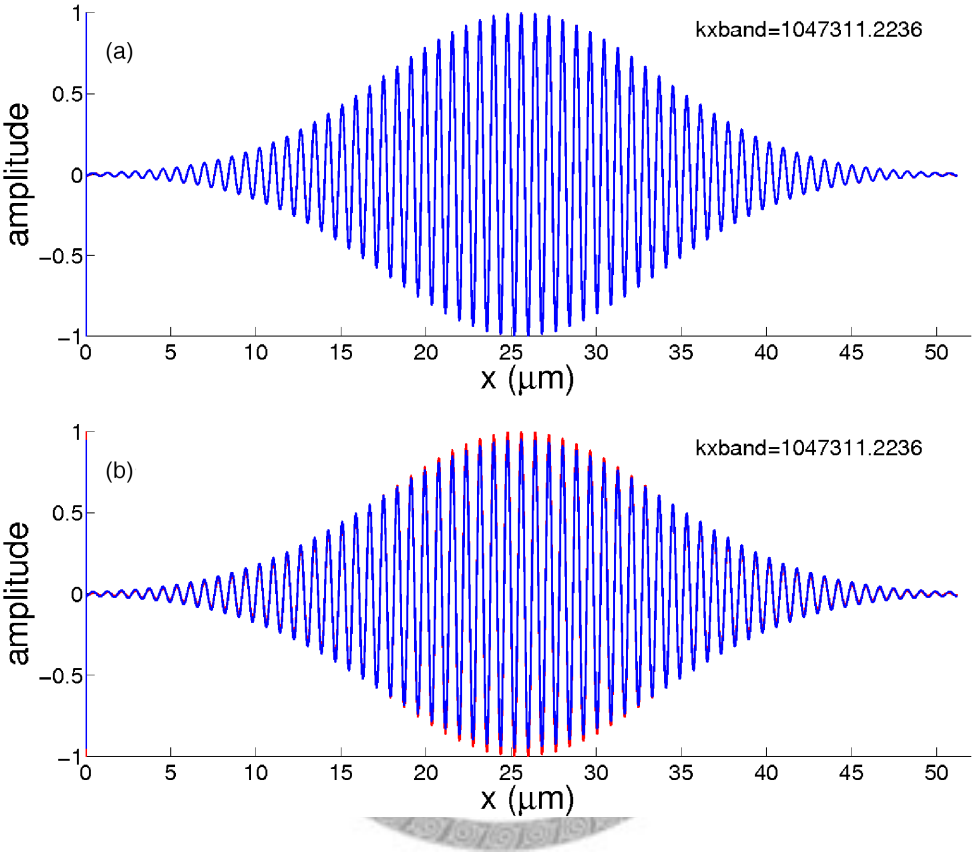


Figure 2.11: The red line is the original Gaussian beam. the filtered Gaussian beam is presented by the blue line. The blue line in the upper plot is the signal filtered by the 8th power Super-Gaussian filter. The blue line in the lower plot is the signal filtered by the Gaussian filter. The filter bandwidth is taken as 1.0×10^6 m^{-1}.


國立臺灣大學
學術品德・愛國愛人

Chapter 3

Computing environment

3.1 Introduction to computation platforms

Within these decades, supercomputers made of personal computers have been intensively developed and widely constructed around the world [39, 40]. This kind of supercomputer is called "PC Cluster". It is cheap compared with the mainframe machine and can be easily assembled. A program written for such platforms is usually using "MPI" standard [33, 34, 35] to accomplish the parallel processing. There are two PC clusters where I performed most of the simulations in this thesis. The first one is b0 cluster in IAMS, another one is IBM Cluster 1350 in NCHC. They are both equipped with two Xeon 5160 woodcrest CPUs, 16 GB FB-DIMM memory modules and an InfiniBand network adapter on each node. We have totally 32 computing nodes which are equal to 128 CPU cores on b0 cluster. The network structure of b0 is shown in Fig. 3.1. We also can access totally 64 computing nodes equivalent to 256 CPU cores on IBM Cluster 1350. These computational resources allow us to perform large-scale 3D PIC simulations especially on LWFA or relativistic plasma nonlinear optics.

A popular choice of the computer operating system is Linux. It's an open-source, free software available freely on the web. The Linux kernel was

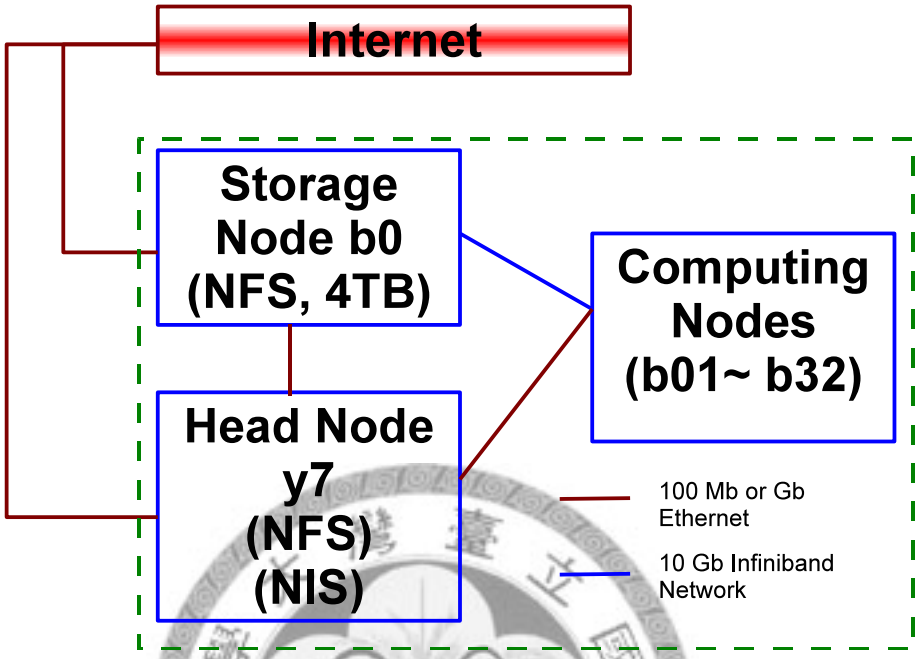


Figure 3.1: A sketch of the network structure of b0 cluster. The InfiniBand network is adopted only among the 32 computing nodes and the storage node. The computers inside the the green dashed square area use internal networks.

initially developed by Linus Torvalds in 1991 and version 1.0 was released in 1994 under the GPL license. A Linux distribution is made by a large collection of softwares such as desktop environments, office suites, media players and server applications build on top of the Linux kernel. These softwares contained inside the the Linux distribution are generally released under GPL-compatible licenses. Debian, Fedora Linux, Red Hat Linux, Ubuntu, Slackware, Gentoo are the most common Linux distribution. In b0 cluster, we chose openSuSE 10.2 as the operating system on each node. "SUSE Linux Enterprise Server 9 (x86_64)" with patch level 3 was chosen as the operating system in IBM cluster 1350.

The SSH protocol is normally used to connect to the cluster head node. Under an unix-like environment, users should type a command like "ssh -Y -p 60 UserAccountName@HostAddress" on their PC terminals to login the workstation. The "-Y" option in the above command is to activate X11 Forwarding which can tunnel the connection of X Window System between the server and client. The "-p 60" option is to connect to port 60 of the cluster head node.

The SSH protocol encrypts the information transferred between the client and server by RSA algorithm. one can use "ssh-keygen -t rsa" to generate a key pair and append the public key in the file "id_rsa.pub" to the "authorized_keys" file on a server. After the public key has been added to the server, one can login to the server without entering the password.

File transferring between machines depends on the sftp or ftp protocols. RSH or Remote Shell protocol is used in the communication between computer nodes.

In order to share the storage disks with all computing nodes, Network File System or NFS should be activated.

It's important to share user information between the head node and the computing nodes so that a user can login to computing nodes and activate the parallel processes. The user information is shared between nodes through NIS server. NIS server should be configured and activated before using the PC cluster.

Three dimensional PIC simulations can produce a huge amount of data. Therefore, a massive storage device is needed to store and analyze simulation results. Basically, PIC simulation should dump the data of the electromagnetic field, the charge density and current fields and the positions and velocities of particles for a snapshot. We consider a simulation case with total $2560 \times 192 \times 192$ cells and 4 macro-particles per cell for example. The electromagnetic field and the charge density and current fields would occupy 10 double-precision floating-point numbers per cell. Each double-precision floating point number occupies 8 bytes on the storage disk. Therefore, around 7 GB of the storage space is needed for storing a snapshot of the electromag-

netic field and the charge density and current fields. Each macro-particle carries 7 double-precision floating-point numbers, which are the positions, velocities and the relative particle sizes of macro-particles. Therefore, around 16 GB of the storage space is needed for a snapshot for storing particle information. In VORPAL code, the electromagnetic field data in the perfectly matched layer would also be dumped. VORPAL also dumps the nodal field which is the interpolated electromagnetic field. Summing the sizes of all the dumped files, a user may need around 32-GB storage space for a snapshot or 320 GB disk space for 10 snapshots. b0 cluster is equipped with a storage node which has 4TB disk space to store simulation results. It is constructed by a Raid 5 disk raid. The storage disk uses the XFS filesystem which is good for large files commonly seen in PIC simulations.

3.2 Introduction of software tools in VORPAL simulations

3.2.1 Hierarchical data format; HDF5

The HDF5 is a hierarchical binary data format maintained by the HDF Group [36]. It is a portable file format which can preserve a variety of data and metadata. It can store a great deal of data objects with no limits on the number or size in a collection. The HDF5 Group also provides APIs which can be used through high-level languages such as C++, Fortran and Java on the laptop, desktop or a massively parallel system. HDF5 format has been widely used on storing scientific data. For example, the NASA MLS, OMI on-line data access and VORPAL code are using HDF5 data format.

The data objects in an HDF5 file are arranged in a hierarchical order through the HDF5 APIs provided by the HDF5 Group or other softwares which are supporting HDF5 file format such as IDL or MATLAB. A dataset object in an HDF5 file contains a N-dimensional data array. An HDF5 group is a structure containing zero or more HDF5 objects. Attributes are small named datasets attached to primary datasets, groups, or named datatypes. Attributes can be used to store additional information. An attribute has two parts: (1) a name and (2) a value. The value part contains one or more data

values of the same datatype.

The HDF5 Group provides "HDFView" as the official HDF5 viewer. It is a versatile program which can allow users to view data directly through a data table like Fig. 3.2 or create colored contour image of selected data like

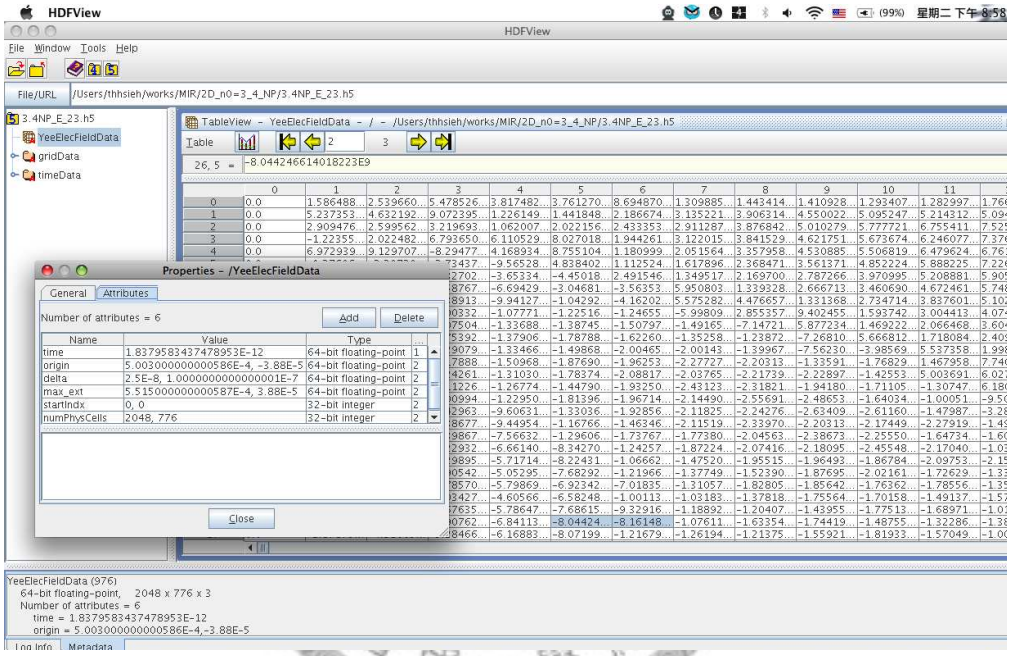


Figure 3.2: A snapshot of HDF view.

Fig. 3.3.

Fig. 3.2 shows the hierarchical structure of a HDF5 file dumped by VORPAL code [1]. The attributes of the dataset contain the information about the simulation domain. The object structure of HDF5 files dumped by VORPAL may change with VORPAL.

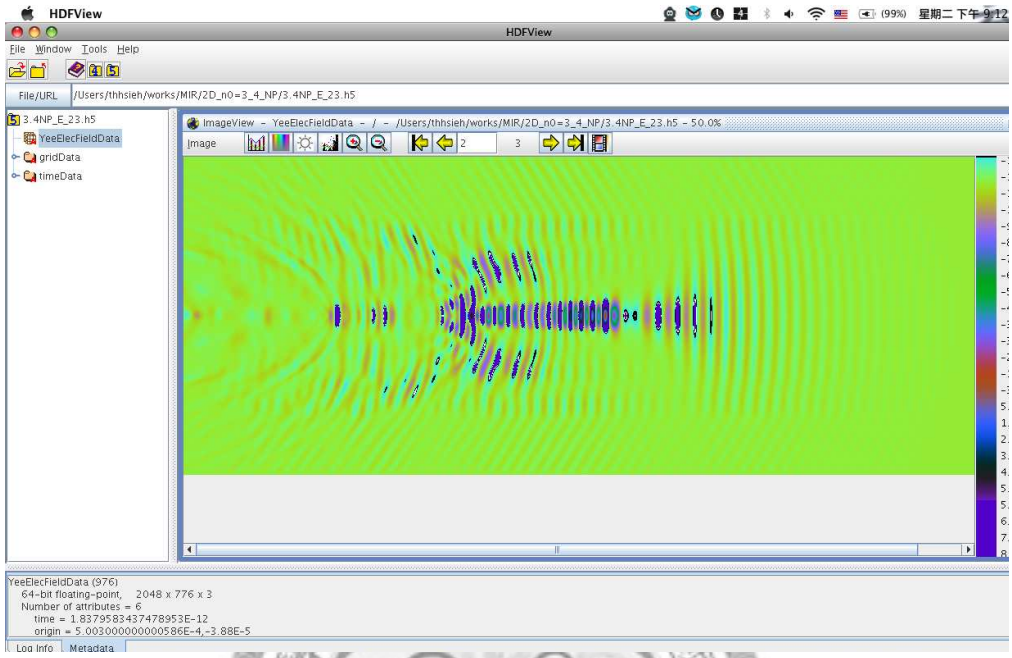


Figure 3.3: A snapshot of HDF view.

3.2.2 Introduction to VORPAL code

VORPAL code [1] is a versatile, cross-platform simulation code which performs electrostatic, electromagnetic and PIC simulation. It's based on other software libraries including "TxPhysics", "TxBase", "hdf5" and "MPICH" or "openMPI". It can import functions written in Python to create complex simulation geometry. The compilation procedure of all above libraries and VORPAL has been appended in the Appendix B.

The VORPAL input file uses XML-like structure and that makes it more organized. TechX Inc. has provided a preprocessor program called "txpp.py" which transforms a preprocessor file (*.pre) into a VORPAL input file. This tool allows users to write a programmable input file. The following pseudo

code is an example of the preprocessor file.

```
$ PreProcessorParameters1= 10
$ PreProcessorParameters2= 20
<Object UserDefinedObjectName>
ObjectParameters=PreProcessorParameters1
<SubObject SubObjectName>
SubParameters=PreProcessorParameters2
</SubObject>
</Object>
```

The following pseudo code transformed from the code above shows what "txpp.py" would accomplish.

```
#$ PreProcessorParameters1= 10
#$ PreProcessorParameters2= 20
<Object UserDefinedObjectName>
ObjectParameters=10
<SubObject SubObjectName>
SubParameters=20
</SubObject>
</Object>
```

An example of the VORPAL preprocessor file which is for 3D PIC simulation is attached in the Appendix A.

VORPAL has built in a great deal of space-time functions which are used on producing complex plasma density profiles or generating various kinds of the laser pulse. The field objects in VORPAL can be defined by users in the input file along with an update sequence. "Updater" objects in VORPAL allow users to update or even manipulate the data values in field objects. Through an user-defined update sequence, users can perform complicated simulation processes without rewriting the simulation code. Therfore, VORPAL has become a powerful simulation framework in the physics of plasmas or electrodynamics.

The performance of VORPAL code has been tested. I compared the running time of VORPAL jobs using the same simulation model running through different network protocols. This simulation model contains total $2048 \times 192 \times 192$ cells and 4 particles in a cell in $2048 \times 128 \times 128$ cells. Each time step forwards 8.265×10^{-17} s in this test case. The results show that the performance can be dramatically improved when VORPAL is compiled and run through "open fabrics" protocol instead of through TCP/IP protocol simulated on an InfiniBand networks. When VORPAL is run through "open fabrics" protocol on an InfiniBand network, the running simulation model described above only takes 100 minutes for 1000 steps. However, when VORPAL is run through TCP/IP protocol simulated on an InfiniBand networks, it takes 1073 minutes to forward 1000 steps. The huge differences shows the importance of InfiniBand networks on large-scale 3D simulations.

VORPAL can write dumped data into several HDF5 files which collect data from all parallel VORPAL processes on different nodes. The storage disks must be shared through NFS or other parallel file systems with computing nodes so that each VORPAL process can write data into the same file. The following error about NFS has been encountered on b0 cluster while VORPAL was run in parallel mode.

```
File locking failed in ADIOI_Set_lock. If the file system is NFS, you
need to use NFS version 3, ensure that the lockd daemon is running on
all the machines, and mount the directory with the 'noac' option (no
attribute caching).
[0] MPI Abort by user Aborting program !
[0] Aborting program!
```

It was found that the "statd" program in the "nfs-utils" package must be manually compiled and activated on each node including the storage node to avoid the above error.

3.2.3 IDL and my analysis programs

Interactive data language (IDL) is a popular script language for data analysis and visualization among scientists and it has been widely applied in space science researches [38]. Followings are important characteristics of IDL:

1. The varaibles are all dynamical type.
2. It passes parameters by reference.
3. It provides a basic form of object-oriented programming.
4. IDL is not case sensitive to the variable names.
5. The variables do not need to be pre-declared.
6. It provides a simple and efficient index slice syntax to extract data from large arrays.

A guide of the basic syntax of IDL can be found in my lecture note [37].

I wrote some VORPAL post-processing procedures under IDL enviroment. The structure of these VORPAL post-processing procedures are shown in Fig. 3.4. These procedures are designed to analyze the simulation data automatically. The procedures surrounded by the blue square area in Fig. 3.4 are the analysis procedures. The analysis procedures export the simulation results in colored contour plots or in curve plots. They are also responsible for performing the Fourier analysis of the simulation results. The Fourier analysis of the simulation data includes the power spectrum density of the simulation data and the Fourier transform filter of the simulation data. The main program is written by users to call the procedures responsible for analyzing the simulation data so that the analysis jobs can be done automatically.

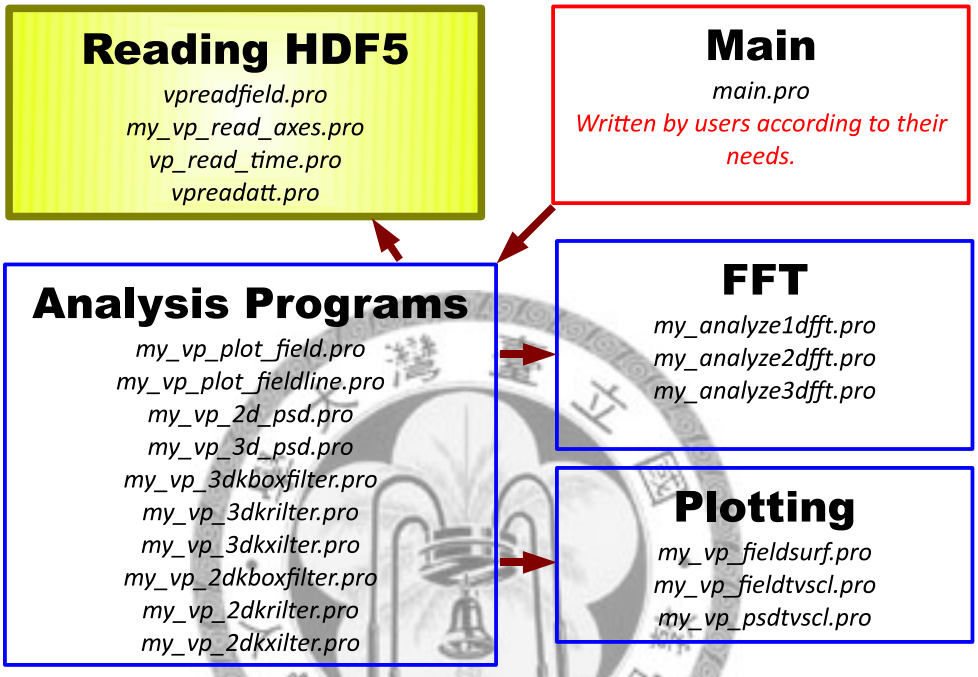


Figure 3.4: The structure of VORPAL post-processing procedures developed under IDL environment. The red arrows indicate the dependences between procedures. Procedures in the yellow area are mostly provided by TechX Inc. to read HDF5 files. The main program contains a loop for generating all required plots. Other procedures are responsible for analyzing the simulation data and create plots.

Chapter 4

Comparison between simulations and analytical results

4.1 Introduction to analytical results

Nonlinear optics of the laser-plasma interaction in the relativistic region has been an attractive subject in the past few decades. Recently, Prof. Tsaur and Prof. Wang has published a three dimensional theory framework using the paraxial approximation to analyze the relativistic collective electron motions in the laser-plasma interactions [2, 3].

Previous work on the coherent Thomson scattering [14] was done with one dimensional analysis. However, it can be found that $\partial a_{1,2}/\partial x_\perp$ are on the first order perturbation in Table. I of Ref. [2]. $\partial a_{1,2}/\partial x_\perp$ were the transverse derivatives of the laser envelope. These terms are ignored in previous one dimensional analysis. Other three dimensional analytical theories were done but they were restricted in the regions of $a \ll 1$ or $a^2 \ll 1$. In the new theory, Prof. Tsaur and Prof. Wang used $a\epsilon$ as the perturbation factor, where a is the laser envelope and $\epsilon = 1/(kw)$. k is the laser center wave number and w is the laser waist size. This framework is not restricted by $a \ll 1$ or $a^2 \ll 1$ where new and interesting phenomena are emerging.

By keeping the transverse derivatives of the laser envelope, the ponderomotive force can be seen in the Eq. 3.17 of Ref. [2]. It can be realized through Eq. 3.24 to Eq. 3.29 of Ref. [2] that the laser ponderomotive force is balanced by the coulomb force from the space charge separation in the long pulse region.

The new theory frame work can be used to analyze harmonic generations, relativistic birefringence and relativistic optical rectification in an underdense plasma. The following two points are predicted by this new theory framework and verified in my simulation works.

1. The 3rd harmonic signal can not be produced when a circularly polarized incident laser pulse is used as the pump beam.

2. Eq. 4.1 is the waveform of the electric field generated by relativistic optical rectification [3]. The comparisons between the theoretical solution and the simulation are shown in the next section.

$$\mathbf{E} \approx -\frac{m_e c^2 a^2}{e(1+a^2/2)^{1/2}} \left(\frac{x}{w_0^2}, \frac{y}{w_0^2}, \frac{\zeta}{L^2} \right), \tag{4.1}$$

   where a is the laser envelope written in normalized vector potential, w_0 is the beam waist size and L is the half-width of the laser pulse at e^{-1} amplitude. m_e and c are the electron mass and light speed. The normalized vector potential is defined as $\mathbf{a} = |e|\mathbf{A}/(m_e c^2)$, where e is the electron charge.

The comparison between the simulations and theoretical solutions are presented in the next section. The coordinate system is chosen to be the same with the analytical theory.

4.2 Comparison between simulations and analytical results

The three dimensional PIC simulations of the nonlinear optics in the relativistic laser-plasma interaction were done with the VORPAL code. Main

beams are linearly or circularly polarized Gaussian beams (λ = 810 nm) propagating through a plasma slab of 3.5×10^{19} cm^{-3} in electron number density along z-direction. A 5 μm-long density ramp was placed at the plasma-vacuum boundary to reduce transient effects. The FWHM of the focal spot size and the duration of the pulse of the Gaussian beam were 8 μm and 45 fs. The peak intensities were 1.022×10^{18} W/cm^2 for linearly polarized Gaussian beam and 5.011×10^{17} W/cm^2 for circularly polarized Gaussian beam. These peak intensities were corresponding to $a_0 = 0.7$. The physical domain of the simulation was 51.2 μm $\times$ 64 μm $\times$ 64 μm discretized with 2048 $\times$ 256 $\times$ 256 cells. The absorbing boundary layer was adopted on the transverse directions relative to the laser axis outside the physical domain of the simulation. The perfectly matched layer [11] was used as the absorbing boundary layer because it has a wide absorbing band. A power-8 Super-Gaussian shape mask function was used to filter the simulation data and obtain the waveforms of signals from relativistic optical rectification.

First, I compare the third harmonic signals between two cases with linearly and circularly polarized Gaussian beam as the pump beams. It can be verified in Fig. 4.1 that there is no third harmonic signal on the power spectrum when the main laser beam is circularly polarized.

The waveforms of filtered low-frequency simulation signals are compared with Eq. 4.1 which describes the signals generated by relativistic optical rectification in Figs. 4.2 – 4.6. The transverse electric field is defined as $E_r = \sqrt{E_x^2 + E_y^2}$. Radius r is defined as $r = \sqrt{x^2 + y^2}$ and the longitudinal coordinate ζ is defined as $\zeta = \eta z - ct$. η is the refractive index of the laser pulse. According to Eq. 4.14 in Ref. [2], η is expressed as $\eta = \sqrt{1 - k_p^2/k^2/(1 + \phi_s)}$, where ϕ_s is the slow varying electric potential, and $k_p = \omega_p/c$.

The transverse electric field is plotted as a function of radius in Fig. 4.2 while ζ is chosen at $\zeta = 0$. It is also plotted as a function of ζ in Fig. 4.3 and r is chosen at $r = 5$ μm. The longitudinal electric field is plotted as a function of radius in Fig. 4.4 while ζ is chosen at $\zeta = 11.5$ μm. It is also plotted as a function of ζ in Fig. 4.5 while r is chosen at $r = 0$. Fig. 4.6 is a grayscale image of the transverse electric field.

These simulation results show that PIC simulations done with VORPAL

code can be verified by the analytical theory. Therefore, we can use this code to simulate more complicated systems.

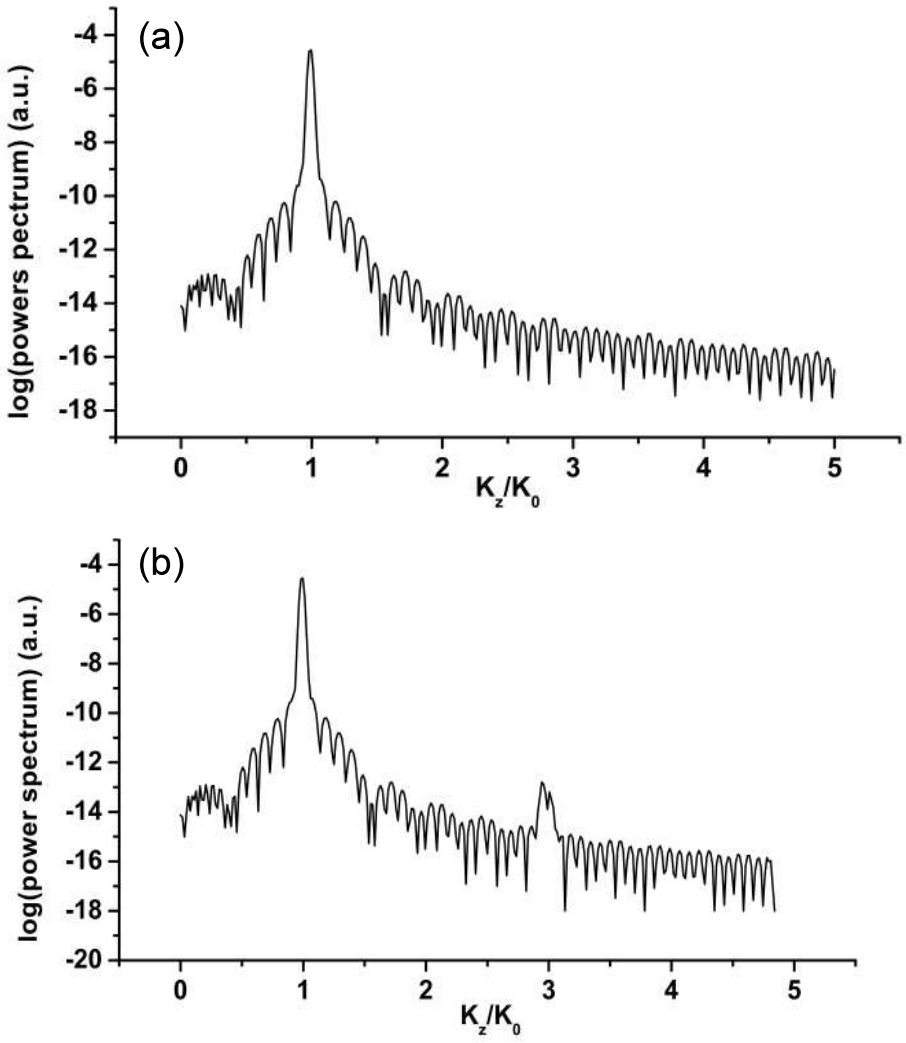


Figure 4.1: Power spectrums of the x-polarized electric field on the mode of $k_x = 0$, $k_y = 0$. The main laser beam propagated along x direction. (a) is the case with circularly polarized laser pulse, and (b) is another case with linearly polarized laser pulse.

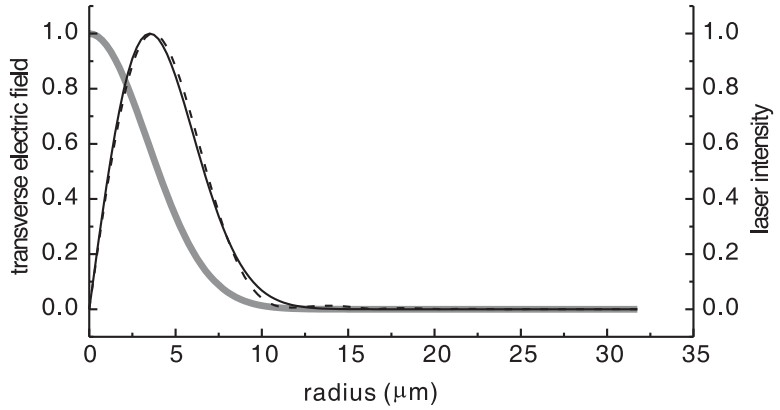


Figure 4.2: Comparison of the transverse electric field E_r as a function of the radius r between the theoretical solution (solid curve) and filtered simulation signals (dashed curve). It is plotted at $\zeta = 0$. The gray curve stands for the spatial intensity profile of the incident laser beam.

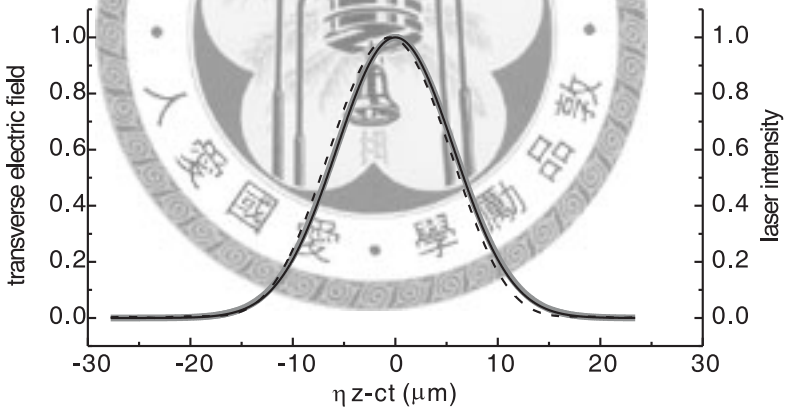


Figure 4.3: Comparison of the transverse electric field E_r as a function of ζ between the theoretical solution (solid curve) and filtered simulation signals (dashed curve). It is plotted at $r = 5$ μm. The gray curve stands for the spatial intensity profile of the incident laser beam.

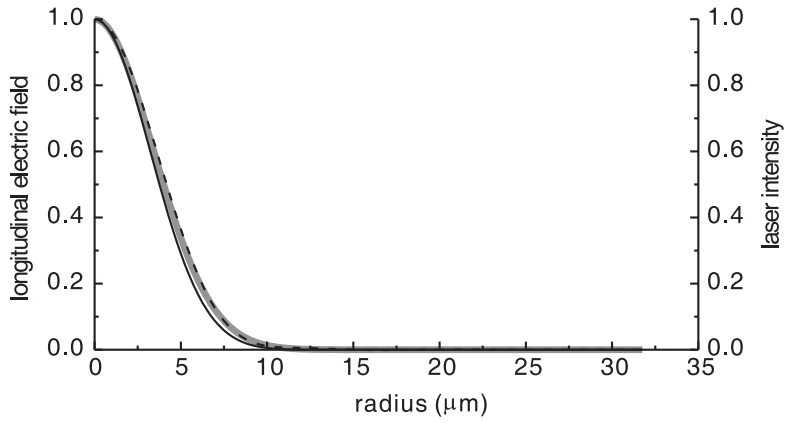


Figure 4.4: Comparison of the longitudinal electric field E_z as a function of r between the theoretical solution (solid curve) and filtered simulation signals (dashed curve). It is plotted at $\zeta = 11.5$ μm. The gray curve stands for the spatial intensity profile of the incident laser beam.

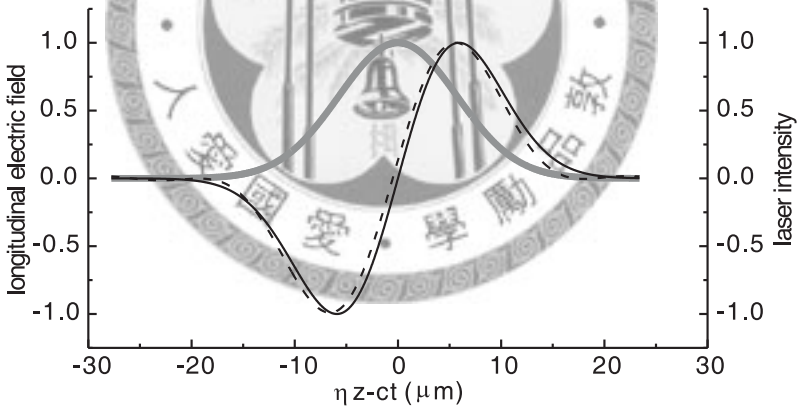


Figure 4.5: Comparison of the longitudinal electric field E_z as a function of ζ between the theoretical solution (solid curve) and filtered simulation signals (dashed curve). It is plotted at $r = 0$. The gray curve stands for the spatial intensity profile of the incident laser beam.

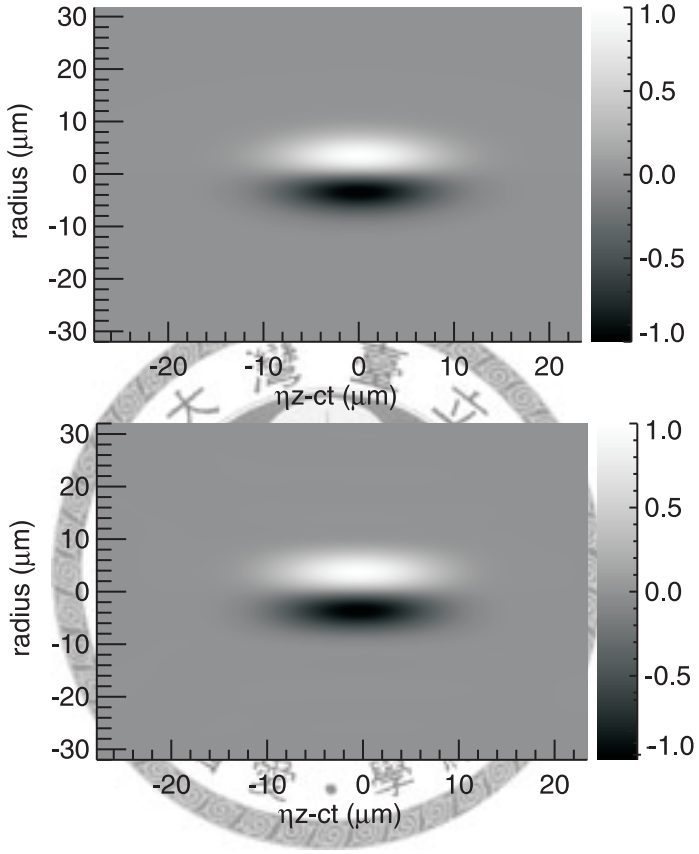


Figure 4.6: Comparison of the waveforms of transverse electric field E_r between the theoretical solution and filtered simulation signals. The upper plot is the theoretical solution and the plot below is the filtered simulation signal.

Chapter 5

Generating the mid-infrared pulse in SM-LWFA

5.1 Introduction to generating the MIR pulse in SM-LWFA

Mid-infrared (MIR) radiation is electromagnetic radiation whose wavelength is between $2-20$ μm. The generation of an MIR pulse is important because of its extensive applications in biology, physics and chemistry. There are several methods to generate MIR pulses. Generally, generating MIR pulses through nonlinear wave mixing in gas or nonlinear crystal can only generate MIR pulses with energies on the order of several microjoules. Free-electron laser can generate strong MIR pulses with energies on the order of hundreds of microjoules. In this work, I observed strong and broadband MIR radiation in the plasma bubble through 3D PIC simulations. The energy of the MIR pulse can be close to minijoules. The laser pulse used in the PIC simulation can be generated from a 10-TW laser system. The simulation results are compared with the experimental results.

In the researches of LWFA, a plasma density cavitation called "plasma bubble" has been observed in various simulations [8, 51]. This structure is created by the strong ponderomotive force of an intense laser pulse propagat-

ing in plasma. This density cavitation creates a low density area surrounded by sharp density gradients. This structure allows electromagnetic radiation with the frequency lower than the cutoff frequency to exist and propagate in the plasma. Therefore, to generate low frequency radiation of the wavelength at the bubble size is possible.

In this work, I observed the generation of low frequency radiation in the plasma bubble in LWFA. The laser pulse parameter used in the simulation satisfies that the plasma wave period $\lambda_p = 2\pi c/(\omega_p)$ is shorter than L, where L is defined as the temporal half-width of e^{-1} amplitude of a Gaussian beam. In this simulation work, λ_p is 5.215 μm and L is 10.69 μm. The parameters of the laser pulse and the plasma density indicate that these parameters are located at self-modulated laser wake field accelerator (SM-LWFA) regime. The down-shifted laser frequency in the plasma can be caused by photon acceleration and self-modulation which is enhanced by self-focusing.

Photon acceleration occurs when there is a gradient of the phase velocity on the longitudinal direction of the laser pulse [52]. The gradient of the phase velocity is caused by the the plasma density distribution and the γ factor in Eq. 1.1, where $\gamma = \sqrt{1 + a^2/2}$ and a is the laser envelope in normalized vector potential normalized by $\mathbf{a} = |e|\mathbf{A}/(m_e c^2)$. It indicates that the laser intensity distribution can cause different refractive index and result in different laser phase velocities at different positions. When the laser pulse encounters a descending slope of the plasma wave, the refractive index distribution would down-shift the laser central frequency. The spectrum broadening toward the long-wavelength side has been observed in Ref. [51, 54, 55]

Self-modulation is caused by the interaction of the laser pulse with the plasma wave. The laser pulse would break up into a pulse train. This effect is like a 2D version of the Raman scattering. It generates Stokes and anti-Stokes satellites. The increase of P/P_c could enhance self-modulaional instability [53], where P/P_c is the ratio of the laser power and the critical power of relativistic self-focusing.

W. Lu *et. al* has observed down-shifted laser frequency in Ref. [51]. He used laser pulses with $a_0 = 1$, $a_0 = 4$, $a_0 = 10$ and the plasma density of 1.5×10^{18} cm^{-3} in electron number density. The plasma wave period

$\lambda_p = 2\pi c/(\omega_p)$ is longer than the pulse duration. Therefore, the down-shifted laser frequency shown in Fig. 2 of Ref. [51] may be caused by photon acceleration only. Self-modulational instability is weak under this parameter.

D. F. Gordon *et. al* and C. D. Murphy *et. al* have observed asymmetric self-phase modulation in simulations and experiments. The results show that the laser frequencies are mainly downshift. Photon-in-cell simulations done by C. D. Murphy *et. al* gives the detailed dynamic process of interactions of photons with the plasma. The down-shifted laser frequency is clearly identified in the Fig. 2 of Ref. [54] and Fig. 3 of Ref. [55]. However, they both have used laser pulses with lower intensities. Thus, the downshift on the laser frequency in laser-plasma interaction when higher laser intensities and higher plasma densities were both used haven't been observed.

In our simulation, the down-shifted laser frequency when higher main beam intensities and higher plasma densities were both used has been observed in the power spectrum taken when the laser pulse propagated into the vacuum region. The simulation was done with VORPAL code. The MIR pulse profiles were obtained by the Fourier transform filter descried in Chapter 2. The simulation results proved that plasma bubble can provide a low density area which allows the MIR pulse to exist and propagate through the plasma. The energy conversion efficiency can be higher than 1% and MIR pulses with energy on the order of millijoules has been observed in the simulation. This simulation results has been compared experiment results in the next section [57].

5.2 3D PIC simulation of generating the MIR pulse in SM-LWFA

This work contains a three dimensional particle-in-cell simulation done with VORPAL code [1]. The main laser beam propagated along the x-axis. The incident laser pulse was linearly z-polarized Gaussian beam with 810-nm wavelength and 42-fs pulse duration in FWHM. The intensities of pump laser beams were 6.243×10^{18} W/cm^2, which is corresponding to $a_0 = 1.73$. Moving window technique was used to reduce computational complexity.

The physical domain of simulation was 64 μm×72 μm×72 μm with 2560 × 180 × 180 cells. The plasma slab was at the center of the physical domain and a 4-μm wide vacuum region were placed near the boundary of physical domain at transverse direction to avoid direct contact of the plasma slab with the absorptive layer. The absorptive layer was adopted outside the domain of simulation, and 25-μm long density ramps were placed at the plasma-vacuum interfaces to reduce transient effects. The perfectly matched layer was chosen as our absorptive layer because of its wide absorption band [11]. Conductive boundary conditions of the electric field were applied to each transverse boundaries.

The nonlinear evolution of the laser pulse is shown in Figs. 5.1 – 5.5. The corresponding wavelength of the cutoff frequency is 5.215 μm which is equal to λ_p. Therefore, the propagation of electromagnetic radiation whose wavelength is longer than that is theoretically forbidden. However, electromagnetic radiation with 6-μm to 10-μm wavelength has been discovered in the filtered simulation signals. It can be clearly identified that radiation whose wavelength is longer than 5.215 μm can exist inside the plasma bubble and propagates through the plasma slab in Figs. 5.1 – 5.4. The total field plots show strong self-focusing and self-modulation of the laser pulse. These effects could enhance the self-modulational instabilities [7, 53] of the laser pulse. It can also be identified that the spectrum is mainly broadened toward long-wavelength side. Therefore, the laser pulse energy was transfered to the radiation of long wavelength. These phenomena could be responsible for the generation of the MIR pulse.

The length of the first bucket is changing with time. The lengths of the first bucket are mentioned in the captions of Figs. 5.1 – 5.3. The peak on the spectrums at different simulation time are changing accordingly. This could indicate that the longest wavelength which the broadened spectrum can reach is at the bubble size. The wavelength of radiation longer than the bubble size can not exist since they are longer than the corresponding wavelength of the cutoff frequency. This indicates that a larger bubble could contain radiation of longer wavelength. Therefore, it is possible to generate radiation of longer wavelength by using higher laser intensity which can generate a larger bubble.

The comparison of the spectrums between the simulation and the exper-

iment is shown in Fig. 5.6. The spectrum from simulation signals of $k_y = 0$ and $k_z = 0$ mode in (a) is similar to the "pump only" spectrum in (b) of Fig. 5.6.

The MIR-pulse durations are shown in Fig. 5.7. These pulse durations are for the MIR pulses whose wavelength are at $6 - 10$ μm. The MIR-pulse duration is 39 fs at the plasma-vacuum interface and 46 fs when the laser pulse propagated into the vacuum region. The MIR-pulse duration is roughly close to the duration of laser pulse.

The growth of the MIR-pulse energy calculated from filtered signals are shown in Fig. 5.8. The energy of the MIR pulse in vacuum with $4 - 12$ μm can be as high as 1.7 mJ which is close to 3 mJ observed in the experiments. The energy of $6 - 10$ μm radiation is roughly 820 μJ when the laser pulse is in the vacuum region. By comparing the growth of MIR signals filtered from simulation data with Figs. 5.1 – 5.5, it can be observed that the growth of the MIR-pulse energy happened roughly at the same time when the electrons were injected into the bubble. This is also observed in the experiment [57].

In conclusion, this simulation work has demonstrated the generation of a high energy MIR pulse in SM-LWFA. The energy conversion efficiency can be higher than 1%. The MIR-pulse energy is larger than the MIR pulse from the free-electron laser and three orders higher than MIR pulse generated with the conventional wave mixing. This simulation work also reveals possible mechanisms responsible for the generation of the MIR pulse and the potential of using SM-LWFA as a tool of generating long wavelength radiation by highly nonlinear optics in the relativistic regime.

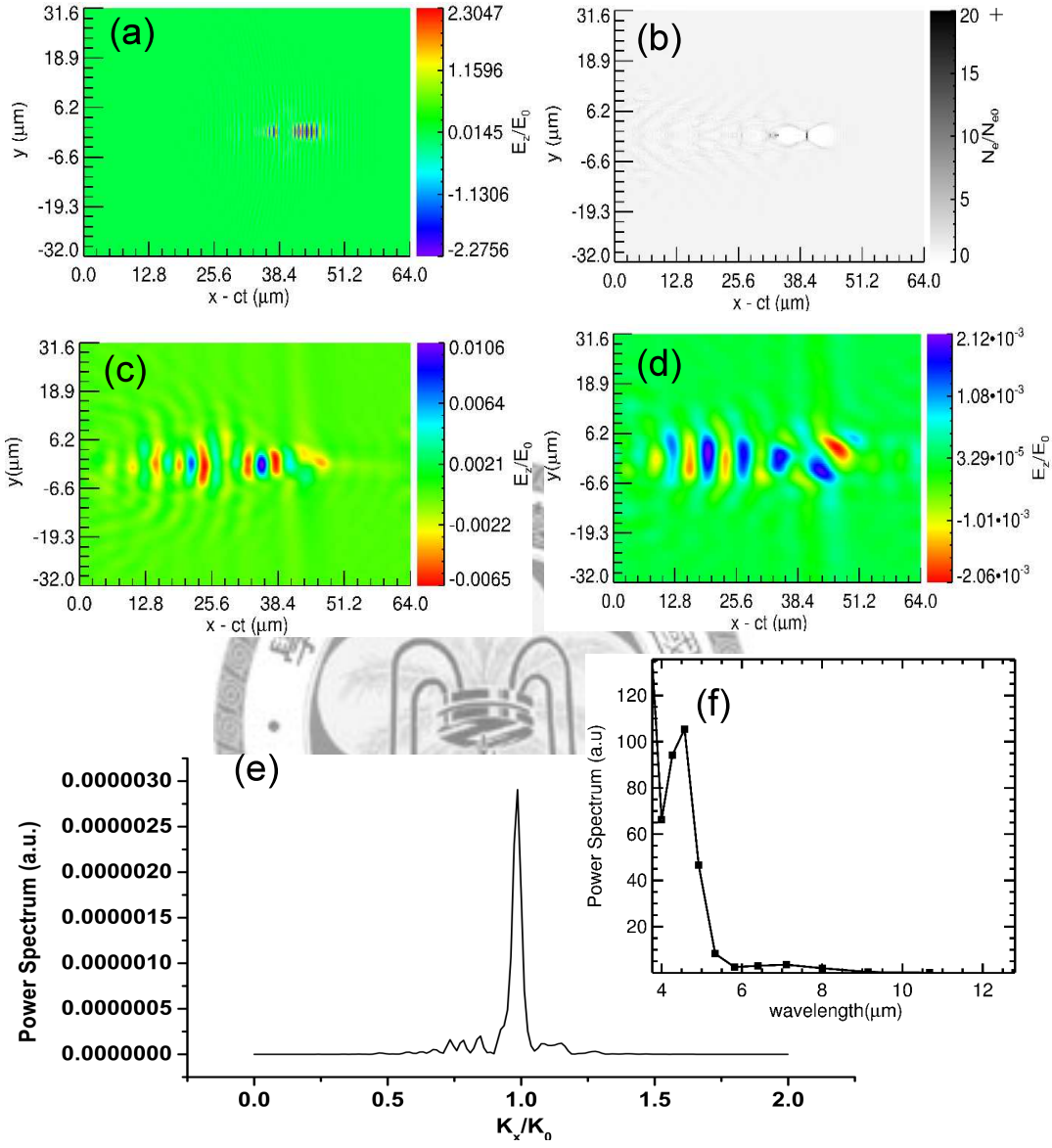


Figure 5.1: A snapshot of simulation signals taken at $T = 0.943$ ps. (a) represents a slice of the total z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (b) is a slice of the total charge density field normalized to 4.1×10^{19} cm^{-3}. (c) contains a slice of the filtered $4 - 12$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (d) is a slice of the filtered $6 - 10$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (e) and (f) is the power spectrum of the z-polarized electric field on the mode of $k_y = 0$, $k_z = 0$. The length of the first bucket shown in (b) is roughly 5.0 μm.

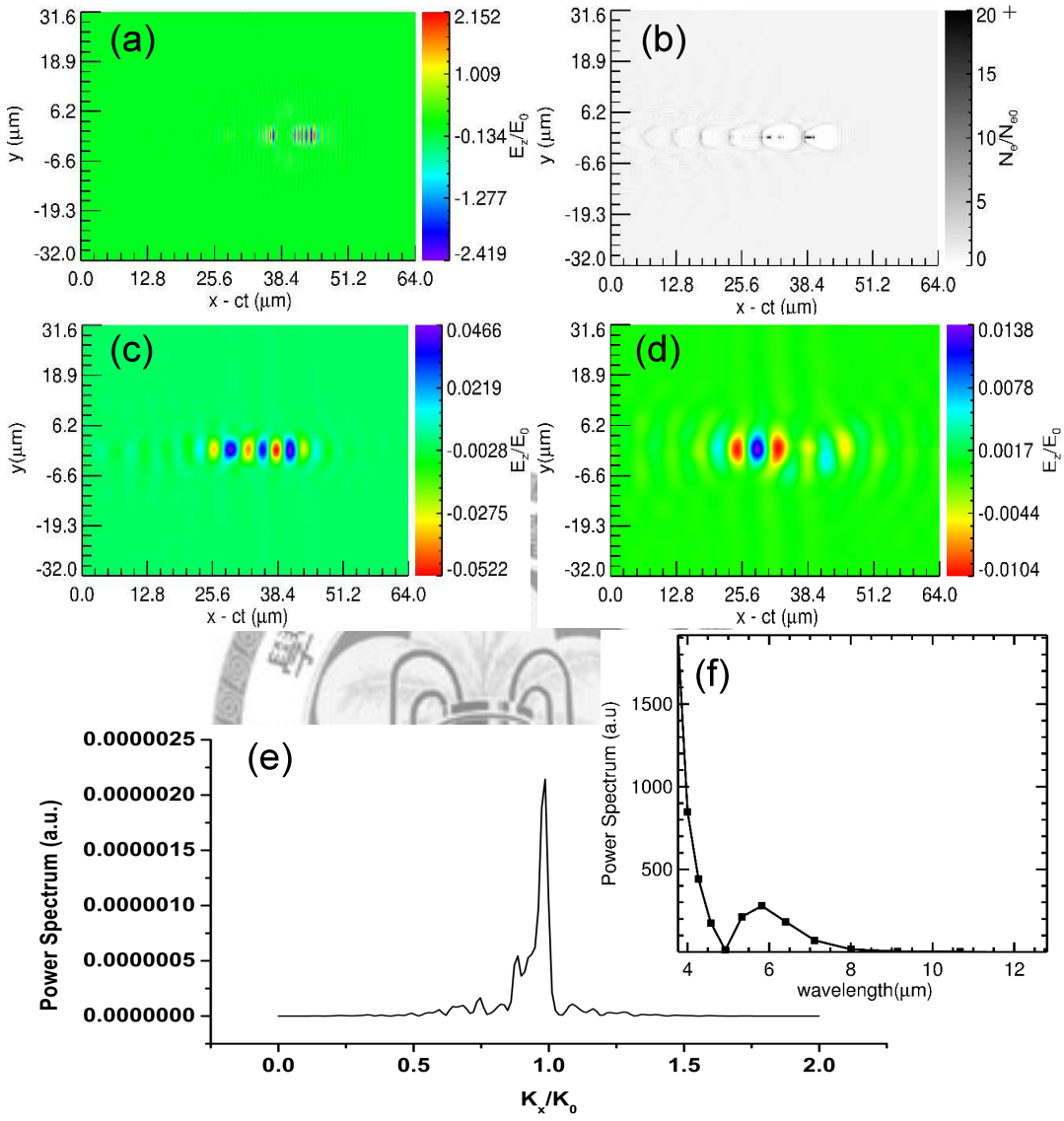


Figure 5.2: A snapshot of simulation signals taken at $T = 1.1$ ps. (a) represents a slice of the total z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (b) is a slice of the total charge density field normalized to 4.1×10^{19} cm^{-3}. (c) contains a slice of the filtered $4 - 12$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (d) is a slice of the filtered $6 - 10$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (e) and (f) is the power spectrum of the z-polarized electric field on the mode of $k_y = 0$, $k_z = 0$. The length of the first bucket shown in (b) is roughly 6.6 μm.

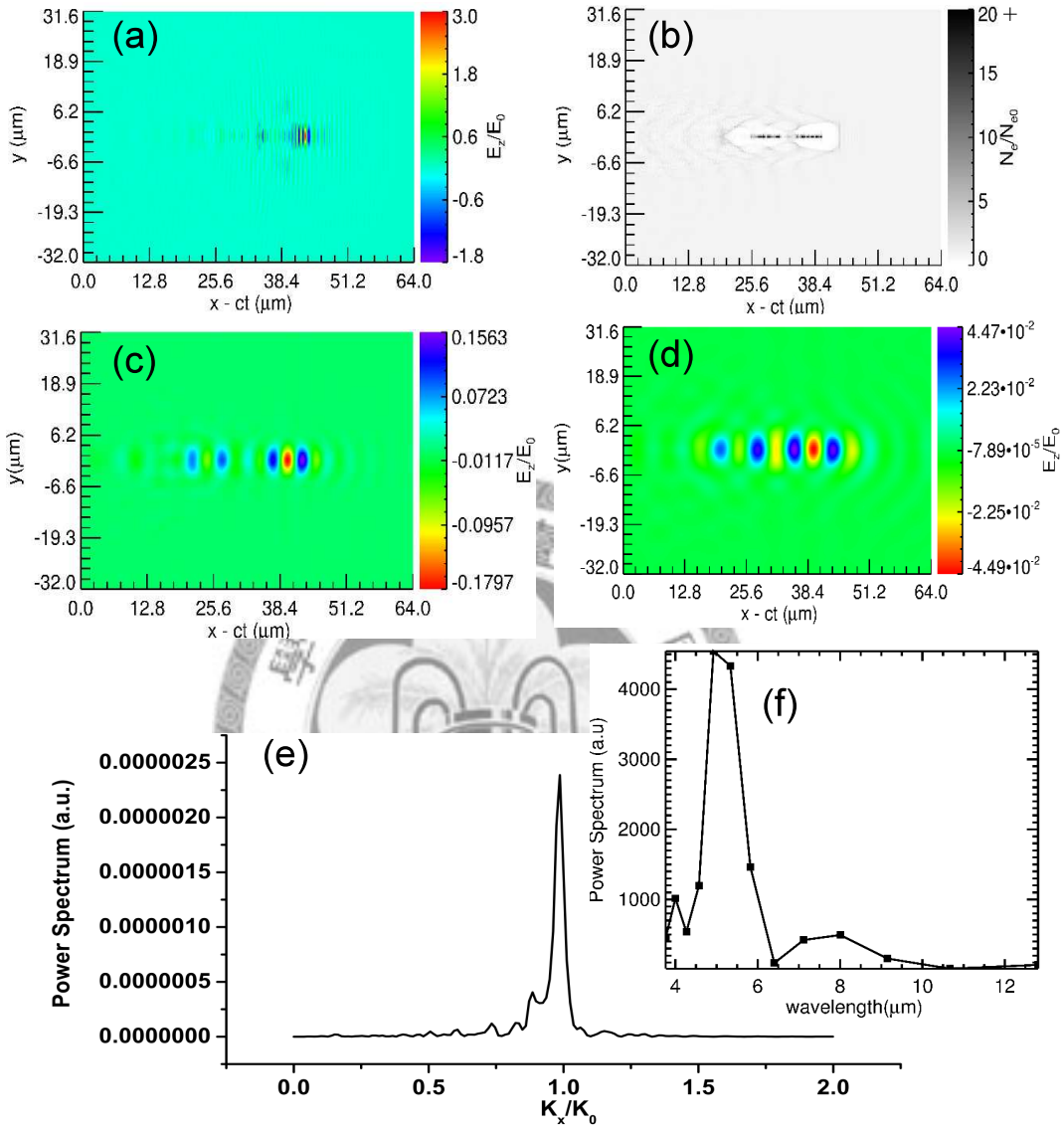


Figure 5.3: A snapshot of simulation signals taken at $T = 1.258$ ps. (a) represents a slice of the total z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (b) is a slice of the total charge density field normalized to 4.1×10^{19} cm^{-3}. (c) contains a slice of the filtered $4 - 12$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (d) is a slice of the filtered $6 - 10$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (e) and (f) is the power spectrum of the z-polarized electric field on the mode of $k_y = 0$, $k_z = 0$. The length of the first bucket shown in (b) is roughly 8.46 μm.

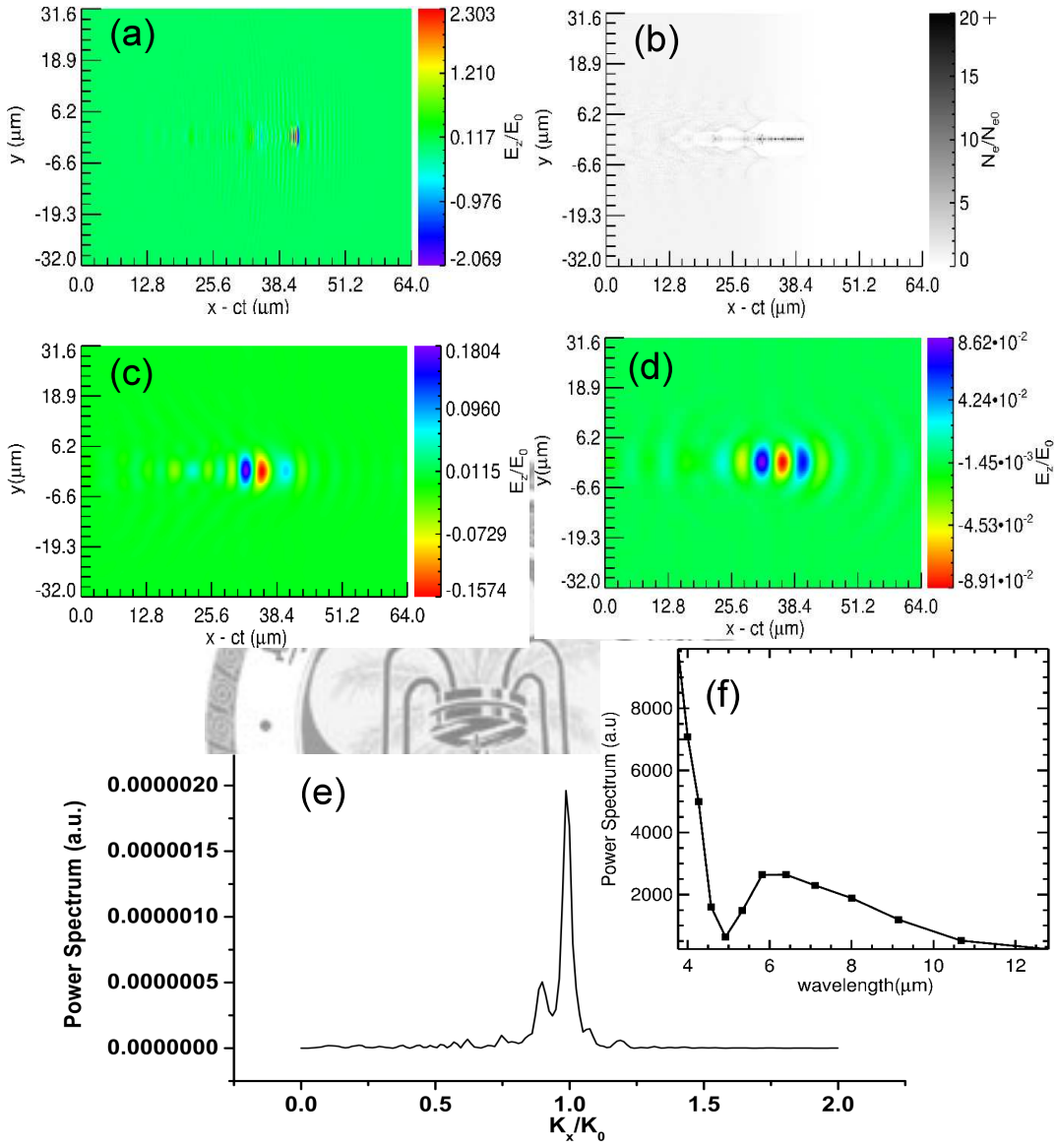


Figure 5.4: A snapshot of simulation signals taken at $T = 1.415$ ps. (a) represents a slice of the total z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (b) is a slice of the total charge density field normalized to 4.1×10^{19} cm^{-3}. (c) contains a slice of the filtered $4 - 12$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (d) is a slice of the filtered $6 - 10$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (e) and (f) is the power spectrum of the z-polarized electric field on the mode of $k_y = 0$, $k_z = 0$.

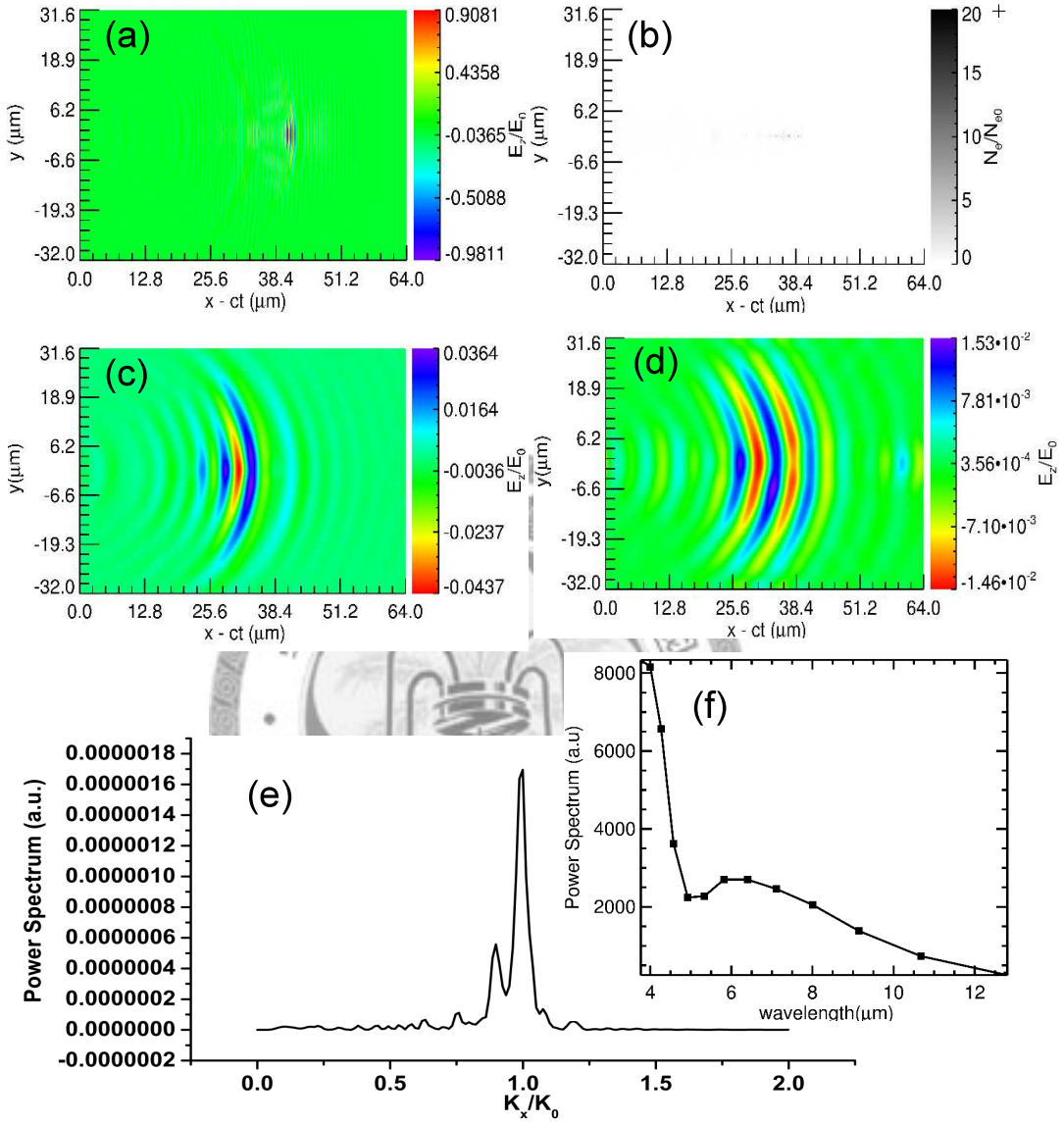


Figure 5.5: A snapshot of simulation signals taken when the main laser pulse propagated into the vacuum region after plasma. (a) represents a slice of the total z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (b) is a slice of the total charge density field normalized to 4.1×10^{19} cm^{-3}. (c) contains a slice of the filtered $4 - 12$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (d) is a slice of the filtered $6 - 10$ μm z-polarized electric field normalized to $E_0 = 6.86 \times 10^{12}$ V/m. (e) and (f) is the power spectrum of the z-polarized electric field on the mode of $k_y = 0$, $k_z = 0$. The divergence angle of the filtered signals in (c) is roughly 10^o. The divergence angle of the filtered signals in (d) is roughly 16^o.

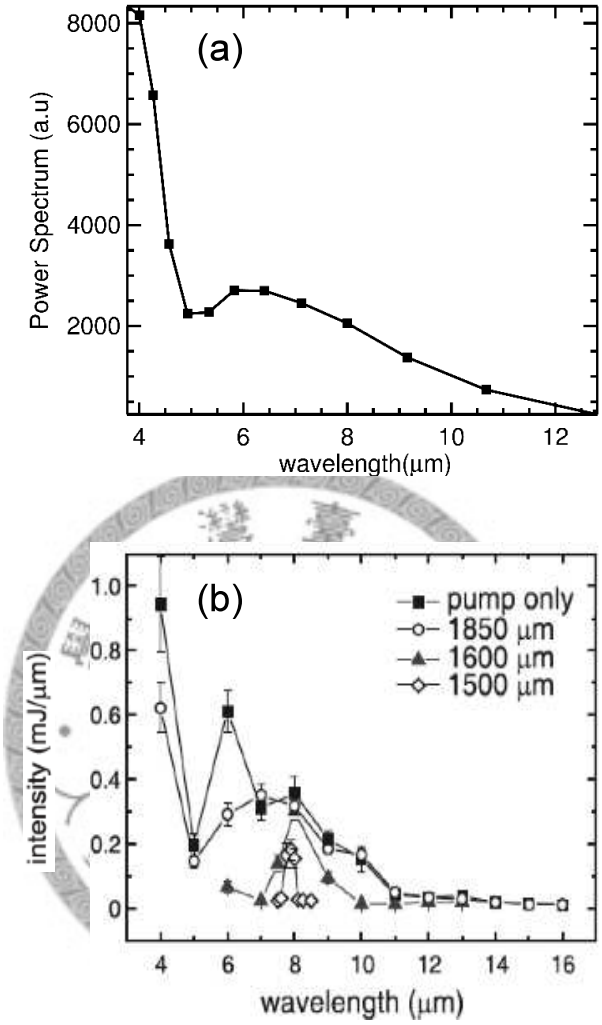


Figure 5.6: The comparison of the MIR-pulse spectrums. (a) is the spectrum of $k_y = 0$ and $k_z = 0$ mode taken from the simulation data when the laser pulse has propagated into the vacuum region. (b) is the spectrum from experiment results [57].

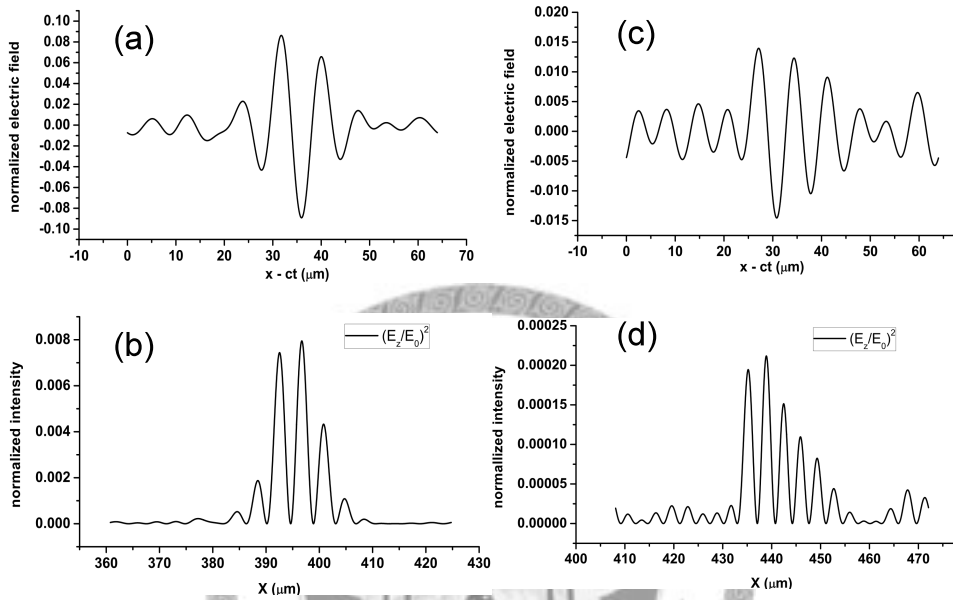


Figure 5.7: These are the on-axis pulse shapes and the wavelength of the pulse is at $6-10$ μm. (a) is taken at $T = 1.415$ ps and (b) is the square of the field in (a). It can be seen that the MIR-pulse duration when the laser pulse in the plasma is about 39.13 fs. (c) is taken when the laser pulse propagated into the vacuum region. and (d) is the square of the field in (c). It can be seen that the MIR-pulse duration when the laser pulse in the vacuum region is about 46.97 fs.

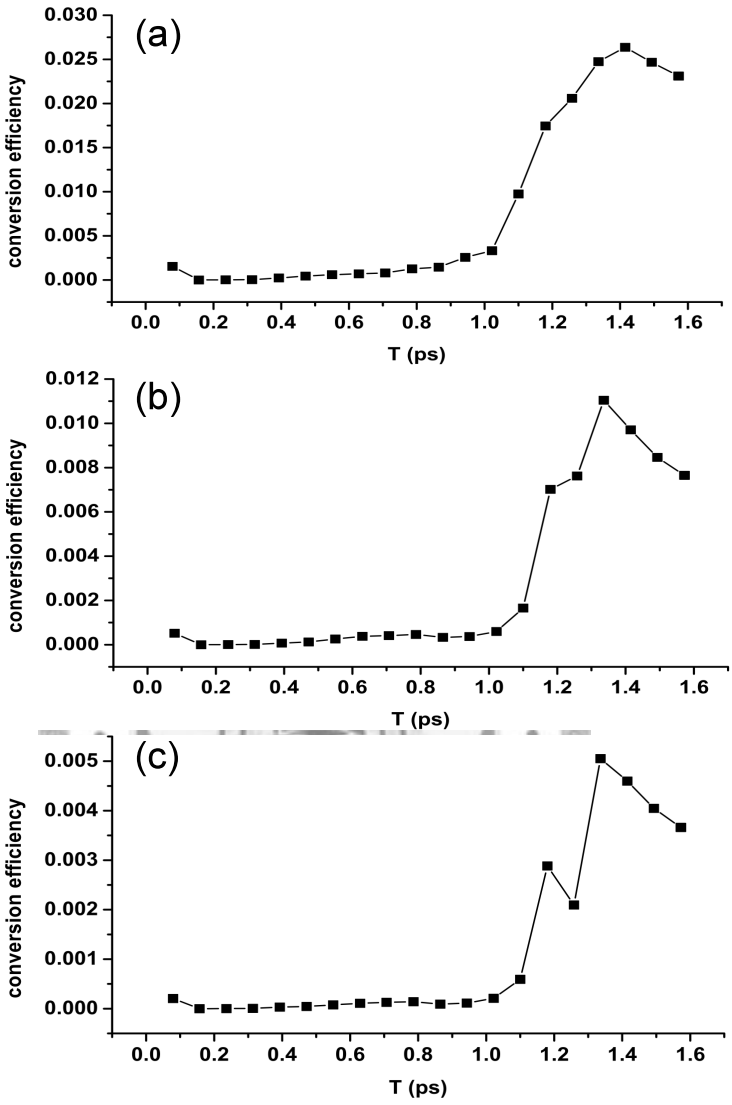


Figure 5.8: The evolution of the filtered electromagnetic field energy. The conversion efficiency is the energy of the filtered electromagnetic field divided by the main beam energy which is 205 mJ. (a) is the evolution of the energy of the filtered electromagnetic field and the bandwidth of the filter was taken as $2 - 20$ μm. (b) is the evolution of the energy of the filtered electromagnetic field and the bandwidth of the filter was taken as $4 - 12$ μm. (c) is the evolution of the energy of the filtered electromagnetic field and the bandwidth of the filter was taken as $6 - 10$ μm.

Chapter 6

Conclusions and perspectives

The particle-in-cell simulation method in Chapter 2 is the most basic method. Unphysical heating due to the rough macro-particle shape has been discovered in Ref. [30]. Higher order shapes of the macro-particle can inhibit such effects. Charge conservation is another important issue in PIC simulation. A correction method, due to Boris, can be found in section 15.6 of Ref. [61], and it has been widely used in many PIC simulation codes. A charge conservation scheme of weighting with arbitrary form-factor has been published in Ref. [31] and this model has been applied in VORPAL code. In order to accurately model experimental conditions, including ionization and collision models in PIC simulation is necessary. These features have also been applied in VORPAL code.

I have introduced the computational capability of our computation facilities and software tools in Chapter 3. The simulation methods with high order weighting scheme, ionization and collision models would increase the computational complexity. It is important to build a larger cluster to perform simulations adopted with more sophisticated models or of higher resolutions. So that we can acquire more accurate simulation data. With the development of the 10 Gigabit Ethernet and the advance on the computational capability of CPUs, it becomes easier to build a computation facility which

is more powerful than current one. With my structured IDL procedures, to massively analyze the simulation results can be done quickly. However, these procedures are written in IDL and they can be redeveloped under open source environments such as Python to avoid using commercial softwares in the future.

In Chapter 4, the theoretical formulas in Ref. [2] have been verified by the PIC simulation done with VORPAL code. These results prove that PIC simulations can be used to verify theoretical analyses such as relativistic birefringence or terahertz generation through difference frequency of laser pulses.

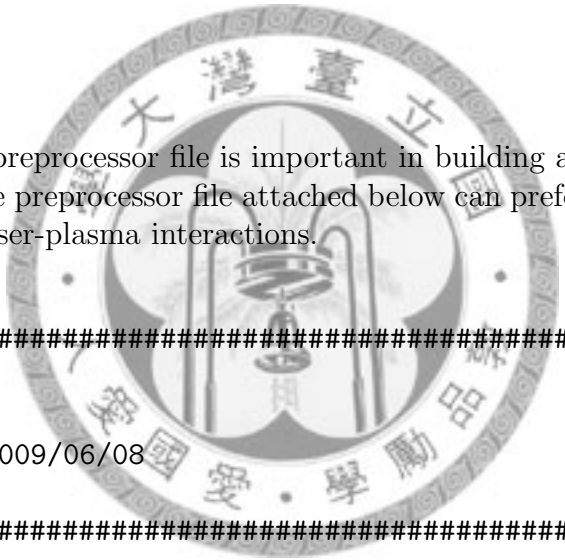
The generation of an MIR pulse in SM-LWFA has been simulated and the results are shown in Chapter 5. These simulation results has been successfully compared with experiment results on the power spectrum. The evolution of the laser pulse in the SM-LWFA region is highly nonlinear. The laser pulse undergoes self-modulational instability and photon acceleration. Strong self-focusing of the laser pulse is observed in Chapter 5 and this effect would enhance the sidebands generated by self-modulational instability. Since there is no accurate theories can provide detailed dynamics under the highly nonlinear conditions created by a tightly focused short-pulse laser, the simulation is the only way to explore the dynamic processes in these experiments. By increasing the laser intensities, A longer bubble can be generated. Therefore, it is possible to extend the broadened spectrum toward longer wavelength and the generation of an MIR pulse whose wavelength is longer than 6 μm can be done. The scaling laws of the MIR-pulse energies and wavelength with the laser intensity and the plasma densities still need to be investigated. Only free-electron laser can produce a MIR pulse with energy close to the MIR pulse generated in LWFA and it is very expensive to maintain a free-electron laser facility. Therefore, it is imperative to investigate the scalings by means of simulations so that a cheap, tabletop, high energy MIR source can be created.

In conclusion, PIC simulation is a crucial tool in exploring detailed physics in the relativistic laser-plasma interactions. By constructing larger computation facilities and performing simulations with more sophisticated simulation models, new methods of generating various kinds of light sources whose fre-

quencies are ranging from terahertz to hard x-ray radiation can be discovered and the dynamic processes responsible for generating these light sources can be revealed.

Appendix A

Example of the VORPAL preprocessor file



The VORPAL preprocessor file is important in building a VORPAL simulation model. The preprocessor file attached below can preform the most basic simulation of laser-plasma interactions.

```
########################################################################
# File: mir3d
#
# Z.-H. Xie 2009/06/08
#
########################################################################

# Useful constants

$ PI = math.pi
$ PIO2 = 0.5*PI
$ LIGHTSPEED = 2.99792458e+08
$ ELECMASS = 9.1093826e-31
$ FUNDCHARGE = 1.60217653e-19
$ EPSILON0 = 8.854187817620390e-12
$ ELECCHARGE = -FUNDCHARGE
```

```
# Grid parameters

$ NX = 2560 # cell numbers of physical domain along x direction
$ NY = 180 # cell numbers of physical domain along y direction
$ NZ = 180 # cell numbers of physical domain along z direction

$ DX = (800.e-9)/32. # grid spacing along x direction
$ DY = 16. * DX # grid spacing along y direction
$ DZ = 16. * DX # grid spacing along z direction

$ LX = DX * NX # length of physical domain of simulations along x direction
$ LY = DY * NY # length of physical domain of simulations along y direction
$ LZ = DZ * NZ # length of physical domain of simulations along z direction

$ NX1 = NX + 1
$ NY1 = NY + 1
$ NZ1 = NZ + 1

# Specify the size of the PML region around the simulation domain

$ NY_PML_L = 7 # Absorptive layer cell numbers
$ NY_PML_U = 7 # Absorptive layer cell numbers

$ NZ_PML_L = 7 # Absorptive layer cell numbers
$ NZ_PML_U = 7 # Absorptive layer cell numbers

$ LY_PML_L = NY_PML_L * DY # Length of the absorptive layer
$ LY_PML_U = NY_PML_U * DY # Length of the absorptive layer

$ LZ_PML_L = NZ_PML_L * DZ # Length of the absorptive layer
$ LZ_PML_U = NZ_PML_U * DZ # Length of the absorptive layer

# total simulation size -- physical domain plus PML's

$ NY_TOT = NY + NY_PML_L + NY_PML_U # Total cell numbers along y direction
$ NY_TOT1 = NY_TOT + 1
```

```
$ LY_TOT = NY_TOT * DY # Total length along y direction

$ NZ_TOT = NZ + NZ_PML_L + NZ_PML_U # Total cell numbers along z direction
$ NZ_TOT1 = NZ_TOT + 1
$ LZ_TOT = NZ_TOT * DZ # Total length along z direction

# indices for specifying the physical domain, excluding the PML's

$ NY_BEGIN = NY_PML_L #staring point of physical domain
$ NY_BEGIN1 = NY_BEGIN + 1
$ NY_END = NY_BEGIN + NY #ending point of physical domain
$ NY_END1 = NY_END + 1
$ NY_ENDm1 = NY_END - 1

$ NZ_BEGIN = NZ_PML_L #staring point of physical domain
$ NZ_BEGIN1 = NZ_BEGIN + 1
$ NZ_END = NZ_BEGIN + NZ #ending point of physical domain
$ NZ_END1 = NZ_END + 1
$ NZ_ENDm1 = NZ_END - 1

# specify in physical units [m] the limits of the full domain

$ XSTART = 0.
$ XEND = XSTART + LX

$ YSTART = -0.5 * LY_TOT
$ YEND = YSTART + LY_TOT

$ ZSTART = -0.5 * LZ_TOT
$ ZEND = ZSTART + LZ_TOT

# physical coordinates of the PML interfaces with the physical domain

$ Y_PML_L = YSTART + LY_PML_L
$ Y_PML_U = YSTART + LY_PML_L + LY

$ Z_PML_L = ZSTART + LZ_PML_L
```

```
$ Z_PML_U = ZSTART + LZ_PML_L + LZ

# Courant condition (3d)
$ DL = 1. / math.sqrt(1./DX/DX + 1./DY/DY + 1./DZ/DZ)
$ DT = 0.995 * DL / LIGHTSPEED

# Courant condition (2d)
# $ DL = 1. / math.sqrt(1./DX/DX + 1./DY/DY)
# $ DT = 0.995 * DL / LIGHTSPEED

# Shift position - Time to start moving window
#                  when the pulse propagates this distance
$ DSHFTPOS = 0.99 * XEND


# PML parameters
# sigma_max (divided by epsilon_0 or mu_0 for electric
#            or magentic conductivity)

$ SIGMA_MAX = LIGHTSPEED / DY
$ PML_EXP = 3.  #power-law exponent of sigma functional form

# Load parameters
$ PPCX = 1 #number of particles along x per cell
$ PPCY = 2 #number of particles along y per cell
$ PPCZ = 2 #number of particles along z per cell
$ PPC = PPCX * PPCY * PPCZ #number of particles per cell

# density profile

$ SlabStartCell = 0.51*NX #start to deploy plasma
$ STARTRAMP = SlabStartCell*DX #start to deploy plasma
$ RAMPLEN = 25.e-6 # length of a density ramp
$ RAMPNX = int(RAMPLEN/DX) #cells of a density ramp
$ STARTFLAT = STARTRAMP + RAMPLEN #start density flattop
# Distance of laser propagating in the flattop density area.
$ InteractionLeng = 382.5e-6 - STARTFLAT # cut plasma at 382.5 (um)
```

```
$ ENDFLATTOP = STARTFLAT + InteractionLeng # ending point of flattop density
$ ENDPOSITION = ENDFLATTOP + RAMPLEN # stop deploing plasma
$ NY_Empty = 10 #vacuum region between plasma and PML
$ NZ_Empty = 10 #vacuum region between plasma and PML

# transverse coordinates of plasma area.
$ Y_START_LOAD = Y_PML_L + NY_Empty*DY
$ Z_START_LOAD = Z_PML_L + NZ_Empty*DZ
$ Y_END_LOAD = Y_PML_U - NY_Empty*DY
$ Z_END_LOAD = Z_PML_U - NZ_Empty*DZ
$ ambientDensity = 4.1e+25  #nominal e- density [m^-3]

# Laser parameters

$ WAVELENGTH = 0.81e-6
$ FREQUENCY = LIGHTSPEED / WAVELENGTH
$ OMEGA = 2. * PI * FREQUENCY
$ KAY = 2. * PI / WAVELENGTH

# Conversion from a0 to E
$ EOVERA = ELECMASS * OMEGA * LIGHTSPEED / FUNDCHARGE

# Pump parameters

$ TAU_PULSE = 42.e-15 # temporal FWHM
$ L_FWHM = TAU_PULSE * LIGHTSPEED # temporal FWHM in meter.
$ W0_PULSE = (8.e-6) /math.sqrt(2.*log(2.)) #waist size of Gaussian pulse
# logitudinal half length of the pulse
$ WXPulse = 2.0 * TAU_PULSE *LIGHTSPEED /math.sqrt(2.*log(2.))
# input parameters for w0 (notice the sqrt(2) factor)
$ WYPulse = W0_PULSE * math.sqrt(2.)
$ WZPulse = W0_PULSE * math.sqrt(2.)

# laser pulse amplitude
$ C0 = 0. # z-pol
$ A0 = 1.73
$ AYPUMP = A0*C0
```

```
$ AZPUMP = A0*math.sqrt(1.0-C0*C0)
$ EYPUMP = AYPUMP * EOVERA
$ EZPUMP = AZPUMP * EOVERA
$ XSTARTPulse = XSTART - WXPulse

# Simulation time - pump gets to end

$ SIMLENGTH = ENDFLATTOP + RAMPLEN + 1.2*LX
# should be increased for production runs

$ SIMTIME = SIMLENGTH / LIGHTSPEED
$ NSTEPS = int( SIMTIME / DT )


# Number of time steps between data dumps

$ DUMP_PER = int(NSTEPS/20)

# ------------------------------------------------------------------
# ------------------------------------------------------------------

# The following two variables determine the basic object
dimension = 3
floattype = double
sortPtcls = false
# The grid

<Grid globalGrid>
  numPhysCells = [NX     NY_TOT  NZ_TOT]
  lengths = [LX     LY_TOT  LZ_TOT]
  startPositions = [XSTART YSTART  ZSTART]
</Grid>

# time step, number of steps, etc.

dt = DT
nsteps = NSTEPS
```

```
dumpPeriodicity = DUMP_PER

# The decomposition

<Decomp decomp>
  decompType = regular
</Decomp>

# Moving window

downShiftDir = 0
downShiftPos = DSHFTPOS


# ------------------------------------------------------------------
# Here we set up the multifields in all their glorious complexity!
# ------------------------------------------------------------------

<EmField my_YeeEmField>

  kind = emMultiField

# ------------------------------------------------
# First, we define all the fields that we'll need --

# The Yee electric field

  <Field YeeElecField>
    dumpPeriod = 20
    numComponents = 3
    kind = xtnd
    offset = edge

# Here, we launch the laser pulse

    <BoundaryCondition pumpLauncherEy>
      lowerBounds = [0  NY_BEGIN1  NZ_BEGIN1]
```

```
      upperBounds = [1  NY_ENDm1   NZ_ENDm1 ]
      minDim = 1
      kind = variable
      indices = [1]  #use this for y-polarization
      k = [KAY  0.  0.]
      vg = LIGHTSPEED
      omega = OMEGA
      amplitudes = [EYPUMP]
      phases = [0.]
      function = gaussianPulse
      L_fwhm = L_FWHM
# distance from the orgin to foucused point
      waistDisplacement = $STARTFLAT - XSTARTPulse + WAVELENGTH/4.$
      widths = [WXPulse     WYPulse   WZPulse]
# pulse origin
      origin = [$XSTARTPulse-WAVELENGTH/4.$  0.0      0.0   ]
    </BoundaryCondition>

    <BoundaryCondition pumpLauncherEz>
      lowerBounds = [0  NY_BEGIN1  NZ_BEGIN1]
      upperBounds = [1  NY_ENDm1   NZ_ENDm1 ]
      minDim = 1
      kind = variable
      indices = [2]  #use this for z-polarization
      k = [KAY  0.  0.]
      vg = LIGHTSPEED
      omega = OMEGA
      amplitudes = [EZPUMP]
      phases = [0.]
      function = gaussianPulse
      L_fwhm = L_FWHM
# distance from the orgin to foucused point
      waistDisplacement = $STARTFLAT- XSTARTPulse$
      widths = [WXPulse      WYPulse   WZPulse]
# pulse origin
      origin = [XSTARTPulse  0.0      0.0   ]
    </BoundaryCondition>
```

```
<BoundaryCondition leftConductor1>
  lowerBounds = [0         -1         -1]
  upperBounds = [1  NY_BEGIN1  NZ_BEGIN1]
  minDim = 1
  kind = constant
  indices = [1  2 ]  #Ey and Ez  (along PML region)
  amplitudes = [0. 0.]
</BoundaryCondition>

<BoundaryCondition leftConductor2>
  lowerBounds = [0  NY_ENDm1  NZ_ENDm1]
  upperBounds = [1  NY_TOT1   NZ_TOT1 ]
  minDim = 1
  kind = constant
  indices = [1  2 ]  #Ey and Ez  (along PML region)
  amplitudes = [0. 0.]
</BoundaryCondition>

<BoundaryCondition rightConductor>
  lowerBounds = [NX        -1         -1]
  upperBounds = [NX1  NY_TOT1  NZ_TOT1]
  minDim = 1
  kind = constant
  indices = [1  2 ] #Ey and Ez
  amplitudes = [0. 0.]
</BoundaryCondition>

<BoundaryCondition backConductor>
  lowerBounds = [ -1  0       -1]
  upperBounds = [NX1  1  NZ_TOT1]
  minDim = 2
  kind = constant
  indices = [0  2 ]  #Ex and Ez
  amplitudes = [0. 0.]
</BoundaryCondition>
```

```
    <BoundaryCondition frontConductor>
      lowerBounds = [ -1   NY_TOT       -1]
      upperBounds = [NX1  NY_TOT1  NZ_TOT1]
      minDim = 2
      kind = constant
      indices = [0  2 ]  #Ex and Ez
      amplitudes = [0. 0.]
    </BoundaryCondition>

    <BoundaryCondition bottomConductor>
      lowerBounds = [ -1       -1  0]
      upperBounds = [NX1  NY_TOT1  1]
      minDim = 3
      kind = constant
      indices = [0  1 ]  #Ex and Ey
      amplitudes = [0. 0.]
    </BoundaryCondition>

    <BoundaryCondition topConductor>
      lowerBounds = [ -1       -1  NZ_TOT]
      upperBounds = [NX1  NY_TOT1  NZ_TOT1]
      minDim = 3
      kind = constant
      indices = [0  1 ]  #Ex and Ey
      amplitudes = [0. 0.]
    </BoundaryCondition>

  </Field>

# The Yee magnetic field

  <Field YeeMagField>
    dumpPeriod = 20
    numComponents = 3
    kind = xtnd
    offset = face
  </Field>
```

```
# The nodal electric field

  <Field nodalE>
    numComponents = 3
    overlap = [1 2]
    offset = none
  </Field>

# The nodal magnetic field

  <Field nodalB>
    numComponents = 3
    overlap = [1 2]
    offset = none
  </Field>

# It has 4 components, the 0th one is the charge on the nodes

  externalFields = [SumRhoJ]


# ------------------------------------------------
# Next, we define all the updaters to be used --

# standard Yee "Ampere" update of the electric field, including currents

  <FieldMultiUpdater yeeAmpere>
    contractFromBottomInNonComponentDir = 1
    kind = yeeAmpereUpdater
    components = [ 0  1  2]
    readFields = [YeeMagField  SumRhoJ]
    writeFields = [YeeElecField]

    lowerBounds = [ 0   NY_BEGIN   NZ_BEGIN]
    upperBounds = [NX   NY_END     NZ_END  ]
  </FieldMultiUpdater>
```

```
# standard Yee "Faraday" update of the magnetic field

  <FieldMultiUpdater yeeFaraday>
    expandToTopInComponentDir = 1
    kind = yeeFaradayUpdater
    components = [ 0  1  2]
    readFields = [YeeElecField]
    writeFields = [YeeMagField]

    lowerBounds = [ 0   NY_BEGIN   NZ_BEGIN]
    upperBounds = [NX   NY_END     NZ_END  ]
  </FieldMultiUpdater>

# Move B-field components from faces to nodes for particle push

  <FieldUpdater nodalBupdate>
    kind = faceToNodeVec
    readFields = [YeeMagField]
    writeFields = [nodalB]

    lowerBounds = [ 0   NY_BEGIN   NZ_BEGIN]
    upperBounds = [NX   NY_END     NZ_END  ]
  </FieldUpdater>

# Move E-field components from edges to nodes for particle push

  <FieldUpdater nodalEupdate>
    kind = edgeToNodeVec
    readFields = [YeeElecField]
    writeFields = [nodalE]

    lowerBounds = [ 0   NY_BEGIN   NZ_BEGIN]
    upperBounds = [NX   NY_END     NZ_END  ]
  </FieldUpdater>

# PML
```

```
  <PmlRegion Pml>
    energyWritePeriod = 0
    eFieldName = YeeElecField
    bFieldName = YeeMagField
    ampereUpdaterName = yeeAmpere
    faradayUpdaterName = yeeFaraday
    <Region inner>
      lowerBounds = [ 0   NY_BEGIN   NZ_BEGIN]
      upperBounds = [NX   NY_END     NZ_END  ]
    </Region>
    <Region outer>
      lowerBounds = [ 0      0      0]
      upperBounds = [NX NY_TOT NZ_TOT]
    </Region>
    sigmaForm = SIGMA_MAX * abs(w)^PML_EXP

  </PmlRegion>


# ------------------------------------------
# Then, we specify the field initializations --

# Initialize nodal fields and their updates

  <InitialUpdateStep initStep1a>
    alsoAfterRestore = true
    messageFields = [nodalE     ]
    updaters = [nodalEupdate]
  </InitialUpdateStep>

# Initialize nodal fields and their updates

  <InitialUpdateStep initStep1b>
    alsoAfterRestore = true
    messageFields = [nodalB      ]
    updaters = [nodalBupdate]
  </InitialUpdateStep>
```

```
# ------------------------------------------
# Finally, we specify the sequence of updates --
# ------------------------------------------

# Step 0:  advance the magnetic field by a half time step

  <UpdateStep step0>
    toDtFrac = 0.5
    messageFields = [YeeMagField]
    updaters = [yeeFaraday]
  </UpdateStep>

#
#  The particle push and current deposition happens here.
#

# Step 2:  advance the electric field by a full time step

  <UpdateStep step2>
    toDtFrac = 1.0
    messageFields = [YeeElecField]
    updaters = [yeeAmpere]
  </UpdateStep>

# Step 3:  advance magnetic field by a 2nd half-time-step, so that it's
#          now known at the same time as the electric field

  <UpdateStep step3>
    toDtFrac = 1.0
    messageFields = [YeeMagField]
    updaters = [yeeFaraday]
  </UpdateStep>

# Step 4:  move the Yee fields to the nodes for the particle push
```

```
  <UpdateStep step4a>
    toDtFrac = 1.
    messageFields = [nodalE      ]
    updaters = [nodalEupdate]
  </UpdateStep>

  <UpdateStep step4b>
    toDtFrac = 1.
    messageFields = [nodalB      ]
    updaters = [nodalBupdate]
  </UpdateStep>

</EmField>


##########
#
# Cold, quiet PIC electrons
#
##########
<Species electrons>
  dumpPeriod = 0
  kind = relBorisVW
  charge = ELECCHARGE
  mass = ELECMASS
  emField = my_YeeEmField
  # The following 2 parameters decide how many
    e- in a macro-particles.
  nominalDensity = ambientDensity
  nomPtclsPerCell = PPC

  <ParticleSource channelSrc>
    kind = gridDenSrcVW # quite, no random
    numPerDir=[PPCX PPCY PPCZ] # particles per cell in each direction
    density = ambientDensity # ambient plasma density
    lowerBounds = [0.  Y_START_LOAD  Z_START_LOAD]
    upperBounds = [1.    Y_END_LOAD    Z_END_LOAD]
```

```
    doShiftLoad = 1

    <STFunc weightFunc>
      kind = cosineFlattop
      direction = [1. 0. 0.]
      startPosition = STARTRAMP
      startFlattop = STARTFLAT
      endFlattop = ENDFLATTOP
      endPosition = ENDPOSITION
      startAmplitude = 0.
      endAmplitude = 0.
      flattopAmplitude = 1.
    </STFunc>
  </ParticleSource>
  <ParticleSink leftAbsorber>
    kind = absorber
    minDim = 1
    lowerBounds = [-1        -1          -1]
    upperBounds = [ 0   NY_TOT1    NZ_TOT1]
  </ParticleSink>
  <ParticleSink rightAbsorber>
    kind = absorber
    minDim = 1
    lowerBounds = [NX        -1          -1]
    upperBounds = [NX1  NY_TOT1   NZ_TOT1]
  </ParticleSink>
  <ParticleSink frontAbsorber>
    kind = absorber
    minDim = 2
    lowerBounds = [ -1   -1          -1]
    upperBounds = [NX1    0    NZ_TOT1]
  </ParticleSink>
  <ParticleSink backAbsorber>
    kind = absorber
    minDim = 2
    lowerBounds = [ -1  NY_TOT         -1]
    upperBounds = [NX1  NY_TOT1  NZ_TOT1]
```

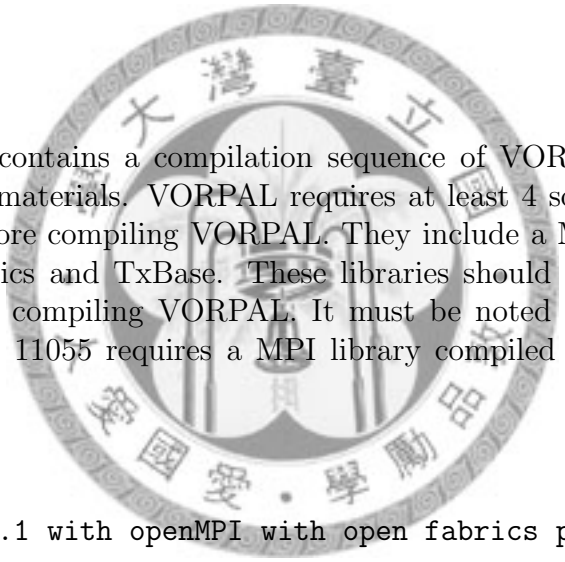
```
  </ParticleSink>
  <ParticleSink bottomAbsorber>
    kind = absorber
    minDim = 3
    lowerBounds = [ -1        -1   -1]
    upperBounds = [NX1   NY_TOT1    0]
  </ParticleSink>
  <ParticleSink topAbsorber>
    kind = absorber
    minDim = 3
    lowerBounds = [ -1        -1     NZ_TOT]
    upperBounds = [NX1   NY_TOT1    NZ_TOT1]
  </ParticleSink>

</Species>
```


國立臺灣大學
學術倫理・愛國愛人

Appendix B

Compilation sequence of VORPAL code



This appendix contains a compilation sequence of VORPAL code and related reference materials. VORPAL requires at least 4 software libraries to be installed before compiling VORPAL. They include a MPI library, HDF5 library, TxPhysics and TxBase. These libraries should be compiled in sequence prior to compiling VORPAL. It must be noted that VORPAL 3.1 with subversion 11055 requires a MPI library compiled with a Fortran 90 compiler.

```
2009/04/22
Build Vorpal3.1 with openMPI with open fabrics protocol on b0.
Refs.:
https://collaborate.txcorp.com/collaborate/swinstall/OpenMpiInstall.html/
https://collaborate.txcorp.com/collaborate/swinstall/Hdf5Install.html/
http://www.open-mpi.org/faq/?category=building
http://www.open-mpi.org/faq/?category=building#build-rte-tm
http://www.open-mpi.org/faq/?category=openfabrics
System gcc:
b0:~>gcc -v
Using built-in specs.
Target: x86_64-suse-linux
```

```
Configured with: ../configure --enable-threads=posix
--prefix=/usr --with-local-prefix=/usr/local --infodir=/usr/share/info
--mandir=/usr/share/man --libdir=/usr/lib64 --libexecdir=/usr/lib64
--enable-languages=c,c++,objc,fortran,obj-c++,java,ada
--enable-checking=release --with-gxx-include-dir=/usr/include/c++/4.1.2
--enable-ssp --disable-libssp --disable-libgcj --with-slibdir=/lib64
--with-system-zlib --enable-shared --enable-__cxa_atexit
--enable-libstdcxx-allocator=new
--program-suffix=-4.1 --enable-version-specific-runtime-libs
--without-system-libunwind
--with-cpu=generic --host=x86_64-suse-linux
Thread model: posix
gcc version 4.1.2 20061115 (prerelease) (SUSE Linux)
OS:
OpenSUSE 10.2
Kernel:
b0:~>uname -a
Linux b0 2.6.25.4b0bc.b-52ff #6 SMP Sat Jun 7
13:42:27 CST 2008 x86_64 x86_64 x86_64 GNU/Linux
IB: ibverbs 1.1.1
*********************************************************
Build Openmpi
*********************************************************
1. Download the openmpi 1.1.3 source code and extract it.
2. Make object directory <objdir>
3. cd <objdir>
4. <confifure path> --prefix=<desired destination directory>\
--with-openib=<ofed path> --disable-shared --enable-static
--disable-mpi-f90
note: --disable-f90 can be removed when there is
      Fortran 90 compiler installed on the system.
5. make
6. make install
NOTE: If one wants to compile with a Fortran 90 compiler,
          he/she needs to set an environment variable 'FC' with
          the path of Fortran 90 compiler and remove the "
          --disable-mpi-f90" parameter.
```

```
**********************************************************
Build HDF5-1.6.5 with openMPI
**********************************************************
1. Define the environment variable "CC" to specify the path of ‘‘mpicc",
which should be located inside the Open MPI directory.
Use "setenv CC <path to mpicc in openmpi>" in c shell or
use "export CC=<path to mpicc in openmpi>" in bash shell.
2. Download the HDF-1.6.5 source code and extract it.
3. Make an object directory <objdir>
4. cd <objdir>
5. <condifure path> --prefix=<desired destination directory>\
--enable-static --disable-shared --enable-parallel
6. make
7. make install
8. Use the following command to add path of HDF5 libraries
into system variable
"LD_LIBRARY PATH" before compiling VORPAL.
   "setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:<hdf5 path>/lib"
   is used under C shell environment.
   "export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:<hdf5 path>/lib"
   is used under bash environment.
**********************************************************
Build txbase
**********************************************************
1. Set the compiler as the system compiler instead of the MPI compiler.
Use "setenv CC gcc" in C shell or use "export CC=gcc" in bash shell
   Use "setenv CXX g++" in C shell or use "export CXX=g++" in bash shell
3. Extract the source code
4. Make object directory <objdir>
5. cd <objdir>
6. <condifure path> --prefix=<desired destination directory>
7. make
8. make install
**********************************************************
Build txphysics
**********************************************************
1. Set the compiler as the system compiler instead of the MPI compiler.
```

```
Use "setenv CC gcc" in c shell or use "export CC=gcc" in bash shell,
use "setenv CXX g++" in c shell or use "export CXX=g++" in bash shell.
2. Extract the source code
3. Make object directory <objdir>
4. cd <objdir>
5. <condifure path> --prefix=<desired destination directory>
6. make
7. make install
**********************************************************
Build VORPAL-3.1
**********************************************************
1. Extract the VORPAL source code
2. Make object directory <objdir>
3. cd <objdir>
4. Set following environment variables.(use "export" in bash shell)
setenv CC <openmpi path>/bin/mpicc
setenv CXX <openmpi path>/bin/mpicx
setenv PATH <openmpi path>/bin:${PATH}
setenv LD_LIBRARY_PATH <openmpi path>/lib64:${LD_LIBRARY_PATH}
setenv LD_RUN_PATH <openmpi path>/lib64
setenv MANPATH <openmpi path>/share/man
5. ../configure --enable-fulloptimize --with-txbase-dir=<txbase path>\
--with-txphysics-dir=<txphysics path> --with-hdf5-dir=<hdf5-1.6.5 path>\
--disable-trilinos --disable-python --with-CC=<openmpi path>/bin/mpicc\
--with-CXX=<openmpi path>/bin/mpicxx --with-MPICC=<openmpi path>/bin/mpicc\
--with-MPICXX=<openmpi path>/bin/mpicxx --prefix=<desired destination path>
6. make
7. make install
```

In some smaller clusters, the InfiniBand networks may not be adopted. Users can choose to use MPICH library to compile VORPAL. If users try to use MPICH instead of Open MPI, all users need to do is to compile a MPICH binary and replace all "openmpi path" with the path of compiled MPICH executables and libraries. The following content is the compiling sequence of mpich-1.2.7p1.

```
1. Extract the mpich-1.2.7p1 source code and enter the source directory.
2. Set following environmental variables.(use "export" in bash shell)
   setenv CC /usr/bin/gcc
   setenv CPP /usr/bin/cpp
   setenv CXX /usr/bin/g++
   setenv FC <Fortran compiler path> (optional)\
3. ./configure --prefix=<desired destination path>
-rsh=ssh\
-cflags="-D_LARGEFILE_SOURCE\
-D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64"
"-rsh=ssh" is optional for the machines communicating
with ssh between nodes.
4. make
5. make install
```

.

Bibliography

[1] Chet Nieter, and John R. Cary, Journal of Computational Physics, **196**, 448-473 (2004)

[2] Gin-yih Tsaur, and Jyhpyng Wang. Phys. Rev. A **76**, 063815 (2007)

[3] Gin-yih Tsaur, and Jyhpyng Wang. accepted by Phys. Rev. A (2009)

[4] H.-H Chu, et.al. Applied Physics B **79**, 193 (2004)

[5] http://www.luli.polytechnique.fr/

[6] Eric Esarey *et. al.* IEEE TRANSACTIONS ON PLASMA SCIENCE **24**, 2 (1997).

[7] Eric Esarey *et. al.* IEEE J. Quantum Electron. **33**, 11 (1997).

[8] A. Pukhov *et al.*, Applied Physics B **74**, 355-361 (2002)

[9] Kane S. Yee. IEEE transaction in antenna and propagation 14(3), 302 (1966)

[10] Gerard A. Mourou *et. al.* Review of Modern Physics **78**, 309(2006)

[11] Jean-Pierre Berenger. Hournal of computational physics **127**, 363 (1995)

[12] William. L. Kruer, The Physics of Laser Plasma Interaction, Addison-Wesley

[13] John M. Dawson. Review of Modern Physics, **55(2)**, 403-447 (1983)

[14] P. Sprangle, E. Esarey, and A. Ting, Phys. Rev. Lett **64**, 2011 (1990)

[15] K. J. Bowers *et. al.* Phys. of. Plasmas **15**, 055703(2008)

[16] Jianguo Wang *et. al.* Phys. of. Plasmas **16**, 033108 (2009)

[17] R. E. W. Pfund, R. Lichters *et. al.* AIP Conf. Proc. 426, New York 1998, p. 141

[18] J. P. Verboncoeur, A. B. Langdon and N. T. Gladd, Computer Physics Communications, **87(1-2)**, 199 (1995)

[19] A. Pukhov. Journal of Plasma Physics, **61**, 425 (1999)

[20] Hui-Chun Wu, Zheng-Ming Sheng *et. al.* Phys. Rev. E **77**, 046405 (2008)

[21] Li-Hua Gao, Han Xu *et. al.* Phys. Rev. E **70**, 046408 (2004)

[22] H. Hamster, A. Sullivan *et. al.* Phys. Rev. Lett, **71**, 17 (1993)

[23] H. Hamster, A. Sullivan *et. al.* Phys. Rev. E **49**, 1 (1993)

[24] Szu-yuan Chen *et. al.* Nature, 396, pp. 653-655 (1998)

[25] S.-Y Chen *et. al.* Phys. Rev. Lett, **84**, 24 (2000)

[26] Eiji Takahashi, *et. al.* Phys. Rev. E **65**, 016402 (2001)

[27] S. Banerjee, *et. al.* J. Opt. Soc. Am. B **20**, 1 (2002)

[28] C.-C Kuo, *et. al.* Phys. Rev. Lett, **98**, 033901 (2007)

[29] X. Davoine *et. al. et. al.* Phys. of. Plasmas **15**, 113102 (2008)

[30] Estelle Cormier-Michel *et. al.* Phys. Rev. E **78**, 016404 (2008)

[31] T. Zh. Esirkepov *et. al.* Computer Physics Communications 135(2), 144-153 (2001)

[32] http://en.wikipedia.org/wiki/InfiniBand

[33] http://www.open-mpi.org

[34] http://www.mcs.anl.gov/research/projects/mpi

[35] http://www.mcs.anl.gov/research/projects/mpi/mpich1

[36] http://www.hdfgroup.org/HDF5

[37] http://ltl.iams.sinica.edu.tw/document/
training_lectures/2008/TH_Hsieh/IDL_Visualization.pdf

[38] http://en.wikipedia.org/wiki/
IDL_(programming_language)

[39] http://www.nchc.org.tw/tw/services/
supercomputing/supercomputing_1/ibm_cluster_1350.php

[40] http://en.wikipedia.org/wiki/IBM_Roadrunner

[41] C. G. R. Geddes *et. al.* Nature, 431, pp. 538-541 (2004)

[42] S. P. D. Mangles *et. al.* Nature, 431, pp. 536-538 (2004)

[43] J. Faure *et. al.* Nature, 431, pp. 541-544 (2004)

[44] Karoly Nemeth *et. al.* Phys. Rev. Lett, **100**, 095002 (2008)

[45] C. G. R. Geddes *et. al.* Phys. Rev. Lett, **100**, 215004 (2008)

[46] Liangliang Ji *et. al.* Phys. Rev. Lett, **101**, 164802 (2008)

[47] L. Yin. Journal *et. al.* Phys. of. Plasmas **13**, 072701 (2006)

[48] L. Yin *et. al.* Phys. Rev. Lett, **99**, 265004 (2007)

[49] U. Teubner *et. al.* REVIEWS OF MODERN PHYSICS, **81**, 445-479 (2007)

[50] J. E. Ralph *et. al.* Phys. Rev. Lett, **102**, 175003 (2008)

[51] W. Lu, M. Tzoufras, and C. Joshi *et. al.* PHYSICAL REVIEW SPECIAL TOPICS-ACCELERATORS AND BEAMS, **10**, 061301 (2007)

[52] W. B. Mori, IEEE J. Quantum Electron. **33**, 1942 (1997)

[53] J. R. Peñano *et. al.* Phys. Rev. E **66**, 036402 (2002)

[54] D. F. Gordon *et. al.* Phys. Rev. Lett, **90**, 21 (2003)

[55] C. D. Murphy *et. al.* Phys. of. Plasmas **13**, 033108(2006)

[56] F. S. Tsung *et. al.* Phys. Rev. Lett, **93**, 18 (2004)

[57] C.-H. Pai *et. al.* Phys. Rev. Lett, submitted

[58] M. Dreher. Experimental Demonstration of Superradiant Amplification of Ultra-Short Laser Pulses in a plasma, PhD thesis (2004).

[59] Allen Taflove, Computational Electrodynamics: The Finite-Difference Time-Domain Method, Artech House. (1995)

[60] Dennis M. Sullivan. Electromagnetic Simulation Using FDTD Method, IEEE Press. (2000)

[61] C. K Birdsall, A. B. Langdon, Plasma Physics via Computer Simulation, Taylor and Francis (2004)

[62] http://www.ss.ncu.edu.tw/∼lyu/Laser/st_20071221.pdf

[63] William H. Press et.al. Numerical Recipes, The Art of Scientific Computing Third Edition, Cambridge (2007)