Simulation kollektiver Phänomene in unterdichten Plasmen

Diplomarbeit

Axel Farr

Technische Universität Darmstadt
Fachbereich Physik

August 1999

Zusammenfassung

Durch die Fortschritte der letzten Jahre im Bereich der Höchstleistungslaser sind Feldstärken erreicht worden, bei denen Elektronen innerhalb eines Viertels einer Schwingungsdauer des Lasers auf relativistische Geschwindigkeiten beschleunigt werden können.

Für viele der Wechselwirkungen zwischen Licht und Materie bei solch extremen Intensitäten sind kollektive Phänomene wichtig, die erst durch die gemeinsame Wechselwirkung vieler Teilchen mit dem elektrischen und magnetischen Feld sowohl des Lasers als auch des Plasmas selbst entstehen.

Ein exemplarisches Phänomen der Anregung kollektiver Schwingungen wurde herausgegriffen und wird in dieser Arbeit zunächst einmal analytisch präsentiert. Im Anschluß werden dann Simulation verschiedener numerischer Plasmen gezeigt.

Zu diesem Zweck wurde ein 2-dimensionaler PIC-Code geschrieben, der die Bewegungsgleichung von Teilchenensembles und gleichzeitig die Maxwellgleichungen auf mikroskopischer Ebene löst. Dieser Code liegt in einer parallelisierten Version vor. Dies geschah in Zusammenarbeit mit Mitarbeitern des Förderprojektes SILASI der Europäischen Union, das die Wechselwirkungen zwischen Hochenergielasern und Materie erforschen soll.

Wichtig ist das Wissen um die Vorgänge bei der Laser-Materie-Wechselwirkung insbesondere bei dem Ziel, mit Hilfe von Lasern in Wasserstoffplasmen Dichten sowie Druck- und Temperaturverhältnisse zu erreichen, die auf das Ziel der kontrollierten Kernfusion hinführen.

Dies ist die gekürzte Fassung meiner Diplomarbeit, die ohne den Teil der Ergebnisse der numerischen Simulation auskommen muß. Dafür ist diese PDF-Datei um zwei Drittel kleiner.

Inhaltsverzeichnis

Kapitel 1

Einführung

Diese Arbeit beschäftigt sich mit dem Verhalten von geladenen Teilchen in starken Laserfeldern. „Stark“ bedeutet in diesem Zusammenhang: Die Laserfelder sind so intensiv, daß Elektronen innerhalb eines Viertels einer Schwingungsdauer auf relativistische Geschwindigkeiten gebracht werden.

Eine wie auch immer geartete Betrachtung muß also die spezielle Relativitätstheorie berücksichtigen, und zwar sowohl bei den Bewegungsgleichungen für ein einzelnes Teilchen als auch bei der Berechnung der Temperatur eines Teilchenensembles.

1.1 Konventionen

Im folgenden werden für die physikalischen Formeln stets Größen in den SI-Einheiten zugrunde gelegt. Ensprechend wurden Formelangaben im System der SI-Einheiten gemacht.

Auch im Kapitel über das im Verlauf dieser Arbeit geschriebene Computerprogramm wurden die Berechnungsgleichungen in diesem System angegeben; im Programm selbst wurde dann mit angepaßten Einheiten gerechnet, um nicht unnötig Rechenleistung durch das Mitrechnen von Konstanten aufzubrauchen. In jedem Fall sind aber die vollständigen Gleichungen im Quelltext als Kommentarzeile angegeben.

1.1.1 physikalische Größen

Die Formeln dieser Arbeit verwenden die folgenden Symbole für häufig wiederkehrende physikalische Größen:

t	Laborzeit
$\mathbf{x}$	Ort des Teilchens
$\mathbf{p}$	Impuls des Teilchens
ϵ_0	Vakuum-Dielektizitätskonstante
μ_0	Vakuum-Magnetisierbarkeit
c	Lichtgeschwindigkeit im Vakuum
$\mathbf{E}$	elektrischer Feldstärkevektor
$\mathbf{B}$	magnetischer Feldstärkevektor
$\mathbf{A}$	elektromagnetisches Vektorpotential
$\mathbf{j}$	elektrische Stromdichte
ϱ	elektrische Ladungsdichte
e	elektrische Elementarladung
m_e	Elektronenmasse
γ	relativistischer Gammafaktor

Für eine vektorielle Größen wird die **fette** Schreibweise verwendet, während für eine skalare Größe die *normale* Schreibweise der Variablen in TEX verwendet wird.

Im dieser Arbeit wird meist eine Orts- und/oder Zeitabhängigkeit einer Größe A nicht explizit in der Form $\mathbf{A}(\mathbf{x}, t)$ angegeben, sondern es wird zu Anfang eines Kapitels beschrieben, wie Orts- und Zeitabhängigkeiten gegeben sind und danach einfach die Form $\mathbf{A}$ verwendet.

Für die Beiträge der Materie in den elektrischen Grundgleichungen wird die mikroskopische Formulierung verwendet, d. h., die Materieeigenschaften werden durch Ladungs- und Stromdichten in die Gleichungen eingebracht.

Zahlenangaben erfolgen in dieser Arbeit in der internationalen Form, d.h., es wird ein Punkt (.) anstelle des im deutschsprachigen Raum üblichen Kommas (,) verwendet. Der Grund ist der, daß eine Auflistung verschiedener Dezimalzahlen mit dem Komma als Dezimaltrennzeichen in dem Augenblick schwierig zu lesen ist, in dem mehrere Zahlen durch ein Komma getrennt in einer Zeile stehen.

Kapitel 2

Wellenphänomene im unterdichten Plasma

Wenn ein kurzer Laserpuls relativistischer Intensität durch ein unterdichtes Plasma läuft, hinterläßt er ein „Wake“, eine Störung im Plasma, die an das Kielwasser eines Schiffes erinnert und daher diesen Namen bekam.

Der Grund für dieses Verhalten liegt in der Verdrängungswirkung des Laserpulses: durch das ponderomotorische Potential des Pulses werden die Elektronen aus dem Bereich hoher elektromagnetischer Feldstärke verdrängt, wodurch sich eine Dichteschwankung ergibt, die sich mit der Gruppengeschwindigkeit des Laserpulses durch das Plasma bewegt. Hinter dem Laserpuls versuchen die Elektronen den Bereich geringerer Dichte wieder aufzufüllen, wobei es dann zu Oszillationen der Elektronendichte kommt.

Im folgenden soll eine Theorie ausgearbeitet werden, die einige der Aspekte dieser Wakefields analytisch ausarbeitet. Die Beschreibung der Phänomene erfolgt mit Hilfe von Fluidgleichungen, im wesentlichen werden dabei Störungsrechnungen anhand des Einflüssigkeitsmodells benutzt.

Die Darstellung orientiert sich dabei an der Beschreibung elektrischer und magnetischer Wakefields in [4]. Im Unterschied dazu wird hier jedoch davon ausgegangen, daß der Laser sich in positive x-Richtung ausbreitet[1].

2.1 Lichtausbreitung im Fluidmodell

Die verwendete Beschreibung geht von der Modellierung des Plasmas als ein geladenes Fluid aus. Die Elektronen bilden dieses Fluid, werden aber im Fall des neutralen Plasmas von ortsfesten Ionen neutralisiert. Dies ist in der schnellen Zeitskala der Lichtwelle eine zulässige Vereinfachung.

[1] Diese Darstellung wurde gewählt, da ich in meinen Programmen schon früher die Geometrie so gewählt hatte, daß der Laser sich in x-Richtung ausbreitet.

2.1.1 Feldgleichungen

Breitet sich Licht im unterdichten Plasma aus, so wird in den Maxwellgleichungen die Kopplung zwischen Licht und Plasma über die elektrische Stromdichte erfaßt.

$$\begin{aligned} \partial_t \mathbf{E} &= c^2 \nabla \times \mathbf{B} - \frac{1}{\epsilon_0}\mathbf{j} && (2.1)\\ \nabla \cdot \mathbf{E} &= \frac{1}{\epsilon_0} \cdot \varrho && (2.2)\\ \partial_t \mathbf{B} &= -\nabla \times \mathbf{E} && (2.3)\\ \nabla \cdot \mathbf{B} &= 0 && (2.4) \end{aligned}$$

Für die Beschreibung des Plasmas wird im Folgenden die relativistische Formulierung der Fluidgleichungen des Einflüssigkeits-Modells verwendet:

$$\begin{aligned} \partial_t \mathbf{p} + \mathbf{v} \cdot \nabla \mathbf{p} &= -e(\mathbf{E} + \mathbf{v} \times \mathbf{B}) && (2.5)\\ \partial_t n + \nabla(n\mathbf{v}) &= 0 && (2.6)\\ \mathbf{v} &= \frac{c\mathbf{p}}{\sqrt{m^2c^2 + p^2}} && (2.7) \end{aligned}$$

Die für die Kopplung von Materie und Feld wichtigen Größen $\mathbf{j}$ und ϱ berechnen sich dabei wie folgt:

$$\begin{aligned} \varrho &= -en \quad \text{mit} && (2.8)\\ n &= n_e - n_0 \ , && (2.9)\\ \mathbf{j} &= \varrho\mathbf{v} \ . && (2.10) \end{aligned}$$

Die Ionen werden als unbeweglich angesehen, was wegen der kurzen Zeitskala des Lasers eine übliche Vereinfachung darstellt. Die Dichte n ist dabei die lokale Variation gegenüber einer als homogen angenommenen Plasmadichte n_0, die wiederum durch die Dichte der Ionen und deren Ladung gegeben ist: $n_0 = Zn_I$.

2.1.2 Verallgemeinerte Vortizität

Bildet man von Gleichung (2.5) die Rotation, nutzt die Beziehung $\nabla \times \nabla \times \mathbf{u} = \nabla(\nabla \cdot \mathbf{u}) - \nabla \cdot (\nabla \mathbf{u})$ aus und sortiert die Terme etwas um, so erhält man die Gleichung

$$\begin{aligned} \partial_t \mathbf{\Omega} &= \nabla \times (\mathbf{v} \times \mathbf{\Omega}) \ , && (2.11)\\ \mathbf{\Omega} &= \nabla \times \mathbf{q} - \frac{e}{mc}\mathbf{B} \ , && (2.12)\\ \mathbf{q} &= \frac{\mathbf{p}}{mc} \ , && (2.13) \end{aligned}$$

in welcher $\mathbf{\Omega}$ die verallgemeinerte Vortizität genannt wird. Die Gleichung ist so zu verstehen, daß der Fluß von $\mathbf{\Omega}$ durch eine Fläche, die sich mit der Strömungsgeschwindigkeit $\mathbf{v}$ der Fluidelemente bewegt, konstant ist. Dies bewirkt aber

auch, daß die partielle Zeitableitung von Omega verschwindet ($\partial_t \mathbf{\Omega} = 0$). Der Vektor $\mathbf{q}$ ist die auf c normierte Einsteingeschwindigkeit des Elektronenfluids.

Setzt man nun $\mathbf{\Omega} = 0$ an, so ergibt sich aus der Gleichung (2.11) die Beziehung

$$\mathbf{B} = \frac{mc}{e} \nabla \times \mathbf{q} \tag{2.14}$$

Setzt man nun diesem Wert für $\mathbf{B}$ in die Gleichung (2.5) ein, so erhält man als Gleichung für das elektrische Feld

$$\mathbf{E} = -\frac{mc}{e} \left(\partial_t \mathbf{q} + c \nabla \sqrt{1 + q^2} \right) \tag{2.15}$$

Der erste Term in der Klammer der rechten Seite ist zusammen mit der Beziehung (2.14) nötig, damit sich überhaupt elektromagnetische Wellen im Plasma ausbreiten können. Der zweite Term stellt eine relativistische Korrektur dar.

Bildet man nun die Divergenz von Gleichung (2.15), so erhält man eine Beziehung für die Elektronendichte n:

$$\frac{n}{n_0} = 1 + \frac{c}{\omega_p^2} \nabla \times \left(\partial_t \mathbf{q} + c \nabla \sqrt{1 + q^2} \right) \tag{2.16}$$

ω_p ist die Plasmafrequenz der Elektronen:

$$\omega_p = \sqrt{\frac{e^2 n_0}{\epsilon_0 m}} \tag{2.17}$$

Nun erhält man durch Einsetzen der Beziehungen aus den Gleichungen (2.14), (2.15) und (2.16) in die Gleichung (2.1) eine einzige Wellengleichung in $\mathbf{q}$, die allerdings in der exakten Form etwas unhandlich ist:

$$\begin{aligned} \partial_t^2 \mathbf{q} + c^2 \nabla \times \nabla \times \mathbf{q} + \frac{\omega_p^2 \mathbf{q}}{\sqrt{1+q^2}} &= -c \partial_t \nabla \sqrt{1+q^2} \\ &\quad - c \frac{\mathbf{q}}{\sqrt{1+q^2}} \nabla \left(\partial_t \mathbf{q} + c \nabla \sqrt{1+q^2} \right) \end{aligned} \tag{2.18}$$

Hier ist zu berücksichtigen, daß ω_p wegen lokaler Dichteunterschiede auch räumlich variieren kann. Dies kann dann zu Effekten wie der relativistischen Selbstfokussierung führen, was aber in diesem Fall nicht ausgearbeitet werden soll.

Um nun das Verhalten des Plasmas bei Anregung durch einen Laserpuls auch im schwach nichtlinearen Bereich beschreiben zu können, wird der nichtlineare Teil der Differentialgleichung (2.18) bezüglich der Variablen $\mathbf{q}$ entwickelt und der normalisierte Impuls $\mathbf{q}$ als eine Summe von Lösungen dargestellt: $\mathbf{q} = \mathbf{q}_1 + \mathbf{q}_2 + \mathbf{q}_3 + \mathbf{q}_4$. Dabei gibt der Index die Ordnung an, bis zu welcher die Gleichung (2.18) entwickelt werden muß, um das jeweilige Element zu erhalten.

Für die Entwicklung werden die bekannten Linearisierungsformeln für die beiden folgenden Wurzelterme verwendet:

$$\begin{aligned}\sqrt{1+q^2} &= 1+\frac{1}{2}q^2-\frac{1}{8}q^4+...\\ \frac{1}{\sqrt{1+q^2}} &= 1-\frac{1}{2}q^2+\frac{3}{8}q^4-...\end{aligned}$$

Mit dieser Ersetzung ergibt sich aus der Gleichung (2.18)

$$\begin{aligned}\partial_t^2\mathbf{q}+c^2\nabla\times\nabla\times\mathbf{q}+\omega_p^2\mathbf{q} &= -\frac{c}{2}\partial_t\nabla\left(q^2-\frac{q^4}{4}\right)\\ &\quad -c\mathbf{q}\left(1-\frac{q^2}{2}\right)\nabla\cdot\partial_t\mathbf{q}\\ &\quad +\frac{1}{2}\mathbf{q}(\omega_p^2-c^2\Delta)q^2\end{aligned}\tag{2.19}$$

Wenn wir nur die Terme betrachten, die linear von $\mathbf{q}$ abhängen, dann erhalten wir die einfachste Wellengleichung für ein Plasma:

$$\partial_t^2\mathbf{q}_1+c^2\nabla\times\nabla\times\mathbf{q}_1+\omega_p^2\mathbf{q}_1=0\tag{2.20}$$

Diese Gleichung hat zwei Lösungen, von denen die eine divergenzfrei ist und die andere verschwindende Rotation besitzt. Erstere beschreibt elektromagnetische Wellen im Plasma, die andere Elektronen-Plasma-Wellen. In erster Ordnung können beide aber nur unter ganz speziellen Bedingen miteinander wechselwirken, wobei direkt ein Lichtquant seine Energie ändert und ein Plasmon erzeugt oder vernichtet wird.

Bei der Erzeugung von Wakefields handelt es sich jedoch um andere Mechanismen: Dazu wird die Gleichung (2.19) zunächst bis zur 2. Ordnung entwickelt.

2.2 elektrische Wakefields

Elektrische Wakefields treten als Lösungen der Gleichung (2.19) bereits in 2. Ordnung auf. Dazu betrachtet man die Entwicklung von (2.19) mit der elektromagnetischen Welle (Lösung von (2.18) in 1. Ordnung) des Lasers als Treiber.

Wir suchen nun eine Lösung in zweiter Ordnung, die durch das Laserlicht induziert wird, das in erster Ordnung divergenzfrei ist. Dadurch bleibt als Treiber für q_2 nur der Term $\nabla\partial_t q_1^2$ übrig:

$$\partial_t^2\mathbf{q}_2+c^2\nabla\times\nabla\times\mathbf{q}_2+\omega_p^2\mathbf{q}_2=-\frac{c}{2}\nabla\partial q_1^2\tag{2.21}$$

Um die Gleichung (2.21) zu lösen, wird in [4] eine Fouriertransformation in Ort und Zeit verwendet. Dazu wird ausgenutzt, daß wegen der rechten Seite der Gleichung (2.21) die Rotation der Lösung $\mathbf{q}_2$ verschwinden sollte.

Das bedeutet, daß Gleichung (2.21) durch das Setzen von $\nabla \times \nabla \times \mathbf{q}_2 = 0$ vereinfacht werden kann. Fouriertransformation liefert dann

$$\left(\omega_p^2 - \omega^2\right) \tilde{\mathbf{q}}_2 = -\frac{c}{2}\, \mathbf{k}\, \omega\, \tilde{q}_1^2 \tag{2.22}$$

Teilt man die Gleichung durch den Faktor $\omega_p^2 - \omega^2$, so hat man eine Lösung für $\tilde{\mathbf{q}}_2$. Um nun zu verhindern, daß der Faktor zu 0 wird, fügt man in die Gleichung bei ω noch einen kleinen imaginären Anteil ϵ ein. Dieser muß vom Vorzeichen her so gewählt sein, daß die bei der Rücktransformation auftretenden Exponentialfunktionen für $t \to \infty$ verschwinden. Damit ergibt sich für $\tilde{\mathbf{q}}_2$:

$$\tilde{\mathbf{q}}_2(\mathbf{k}, \omega) = \frac{c}{2} \frac{\omega \mathbf{k}}{(\omega + i\epsilon)^2 - \omega_p^2} q_1^2(\mathbf{k}, \omega) \tag{2.23}$$

Die Funktion $\mathbf{q}_2(\mathbf{r}, t)$ erhält man nun durch Rücktransformation aus dem Wellenzahl-/Frequenzbereich in den Orts- und Zeitbereich:

$$\mathbf{q}_2(\mathbf{r}, t) = \int_{-\infty}^{\infty} \frac{d\omega}{2\pi} \int_{-\infty}^{\infty} \frac{d\mathbf{k}}{(2\pi)^3}\, e^{-i(\omega + i\epsilon)t}\, e^{i\mathbf{kr}}\, \tilde{\mathbf{q}}_2(\mathbf{k}, \omega) \tag{2.24}$$

Die Integration liefert dann als Resultat den Cauchyschen Hauptwert an der Polstelle bei $\omega = \omega_p$. Dieser ist

$$\mathbf{q}_2(\mathbf{r}, t) = -\frac{c}{4} \nabla \left(e^{-i\omega_p t} q_1^2(\mathbf{r}, \omega_p) + e^{i\omega_p t} q_1^2(\mathbf{r}, -\omega_p) \right) \ . \tag{2.25}$$

Die Bedingung, daß keine Störung in der Plasmadichte vor dem Puls erscheinen soll, liefert für den Treiber

$$q_1^2(\mathbf{r}, t) = \frac{i}{\omega_p} \int_{-\infty}^{t} d\tau e^{i\omega_p \tau} \partial_\tau q_1^2(\mathbf{r}, \tau) \ . \tag{2.26}$$

Wenn man die Gleichung (2.26) in die Gleichung (2.25) einsetzt, die Exponentialfunktionen in das Integral hineinzieht, danach die Integrale zusammenfaßt und den Differentialoperator aus dem Integral herauszieht erhält man:

$$\mathbf{q}_2(\mathbf{r}, t) = -\frac{c}{2\omega_p^2} \nabla \partial_t \Phi \ , \tag{2.27}$$

$$\Phi \equiv \omega_p \int_{-\infty}^{t} d\tau \sin \omega_p (t - \tau) q_1^2(\mathbf{r}, t) \ , \tag{2.28}$$

$$\omega_p^2 q_1^2 = \partial_t^2 \Phi + \omega_p^2 \Phi \ . \tag{2.29}$$

Die Gleichung (2.29) ergibt sich, wenn man die gefundene Lösung für $\mathbf{q}_2$ in die Gleichung (2.21) einsetzt. Die Funktion Φ beschreibt die Orts- und Zeitabhängigkeit der Plasmawelle in 2. Ordnung. Ebenso wie $\mathbf{q}_2$ lassen sich auch alle anderen

Größen aus dieser Funktion herleiten:

$$\mathbf{B}_2 = 0 , \tag{2.30}$$

$$\mathbf{E}_2 = -\frac{mc^2}{2e}\nabla\Phi , \tag{2.31}$$

$$n_2 = n_0\left(1+\frac{c^2}{2\omega_p^2}\nabla\Phi\right) , \tag{2.32}$$

$$\mathbf{v}_2 = -\frac{c^2}{2\omega_p^2}\partial_t\nabla\Phi . \tag{2.33}$$

Zum Herleiten der Beziehungen (2.30) bis (2.33) dienten die entsprechenden Beziehungen (2.14) bis (2.16) und die Definition des normierten Impulses $\mathbf{q}$. Beim Ausrechnen der Beziehungen wurden wiederum die Wurzelterme linearisiert und die zweite Ableitung nach der Zeit wurde durch die Gleichung (2.29) ausgedrückt.

Da die gefundene Lösung für $\mathbf{q}_2$ verschwindende Rotation besitzt, ist das magnetische Feld in zweiter Ordnung Null.

2.2.1 Gleichungen im mitbewegten System

Will man noch weitere Erkenntnisse über den Verlauf der Plasmawelle gewinnen, so muß man die Gleichungen im mitbewegten System lösen. Falls die Plasmafrequenz nicht zu hoch ist, ist es eine brauchbare Näherung, als Ausbreitungsgeschwindigkeit der Laserwelle die Vakuum-Lichtgeschwindikkeit anzunehmen. Auf die Verhältnisse bei der realen Gruppengeschwindigkeit des Lasers geht Kapitel 3 ein.

Zusätzlich ist es sinnvoll, für die Amplitude des Lasers Rotationssymetrie anzunehmen. Entsprechend wird senkrecht zur Ausbreitungssrichtung ein rotationssymetrisches Koordinatensystem verwendet. Wenn die Ausbreitung in x-Richtung erfolgt, so definiert sich das neue Koordinatensystem wie folgt[2].

$$\xi = ct - x \tag{2.34}$$

$$\varrho = \sqrt{y^2+z^2} \tag{2.35}$$

$$\varphi = \arctan\frac{y}{x} \tag{2.36}$$

Der Laserpuls breitet sich dabei von $x = -\infty$ in positive Richtung aus. In diesen Variablen ergeben sich für die elektrischen Feldkomponenten folgende Gleichungen:

$$E_{2\varrho} = -\frac{mc^2}{2e}\,\partial_\varrho\Phi , \tag{2.37}$$

[2]Die Vektoren $\mathbf{e}_x$, $\mathbf{e}_\varrho$ und $\mathbf{e}_\varphi$ besitzen in dieser Reihenfolge die korrekte Orientierung: es gilt $\mathbf{e}_x \times \mathbf{e}_\varrho = \mathbf{e}_\varphi$. Die Ableitungsoperatoren entsprechen denen der sonst üblichen Zylinderkoordinaten, lediglich z ist durch x ersetzt.

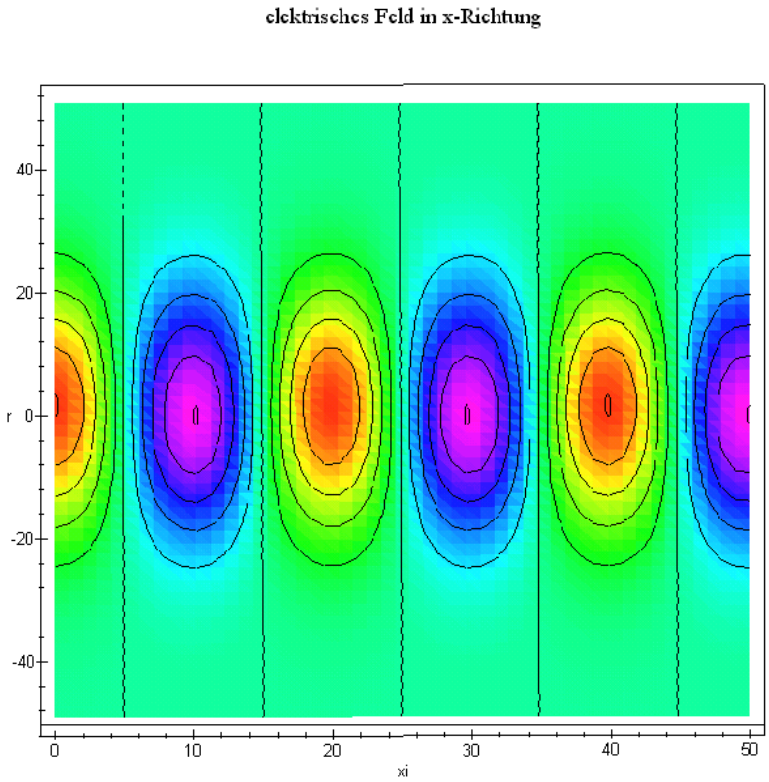

Abbildung 2.1: Lösung für die elektrische Feldkomponente in x-Richtung hinter dem Puls.

$$E_{2x} = \frac{mc^2}{2e}\,\partial_\xi \Phi \;, \tag{2.38}$$

$$\Phi = k_p \int_{-\infty}^{\xi} d\tau \sin k_p(\xi - \tau)\, q_1^2(\varrho, \tau) \;, \tag{2.39}$$

$$k_p = \frac{\omega_p}{c} \;, \tag{2.40}$$

dabei ist E_{2x} die Feldkomponente in Ausbreitungsrichtung (longitudinale Komponente), während $E_{2\varrho}$ die Feldkomponente in radialer Richtung ist (transversale Komponente). Die dritte Feldkomponente in Umfangsrichtung ($E_{2\varphi}$) verschwindet im Fall eines radialsymetrischen Laserpulses. Die Plasma-Wellenzahl k_p taucht in unseren Gleichungen auf, weil es sich bei den Wakefield-Phänomenen um Plasmaschwingungen mit der Plasmafrequenz ω_p handelt, deren Phasengeschwindigkeit aber gleich der Gruppengeschwindigkeit des Lasers ist.

2.2.2 Zeitlich begrenzter Laserpuls

In der Regel kann man einen zeitlich begrenzten Laserpuls annehmen, der außerdem noch symetrisch um $\xi = 0$ verteilt sein soll, das bedeutet, der Puls startet zu einem Wert $\xi = -L$ und endet zu einem Wert $\xi = L$. Das Integral, von dem die Funktion Φ abhängt, liefert also nur Beiträge für Werte von $-L < \xi < L$. Damit ergibt sich für die Funktion Φ hinter dem Laserpuls ($\xi > L$):

$$\Phi = \sin(k_p\xi)\,\psi_1(\varrho) - \cos(k_p\xi)\,\psi_2(\varrho)\;, \tag{2.41}$$

$$\psi_1(\varrho) = k_p \int_{-L}^{+L} d\tau\, \cos(k_p\tau)\, q_1^2(\varrho,\tau)\;, \tag{2.42}$$

$$\psi_2(\varrho) = k_p \int_{-L}^{+L} d\tau\, \sin(k_p\tau)\, q_1^2(\varrho,\tau)\;. \tag{2.43}$$

Hier wurde die Beziehung $\sin(x-y) = \sin(x)\cos(y) - \cos(x)\sin(y)$ verwendet und der nicht von der Integrationsvariable abhängige Teil des Ausdrucks aus dem Integral herausgezogen. Die Funktionen ψ_1 und ψ_2 sind nichts anderes als der Wert der Sinus- und Kosinustransformierten der Größe q_1^2 an der Stelle ω_p!

Das bedeutet, daß man die Gleichung (2.41) mit Hilfe komplexer Zahlen noch einfacher formulieren kann:

$$\Phi = \frac{1}{2}\left(ie^{-ik_p\xi}\,\psi(\varrho) + \text{cc.}\right)\;, \tag{2.44}$$

$$\psi(\varrho) = k_p \int_{-L}^{+L} d\tau\, e^{ik_p\tau}\, q_1^2(\varrho,\tau)\;. \tag{2.45}$$

Das kalte Plasma ist ein Medium, das mit der Plasmafrequenz stationär schwingen kann. Es wirkt auf die Schwingung des Lasers gewissermaßen wie ein schmalbandiges Filter, das nur auf die Plasmafrequenz reagiert. Alle anderen Frequenzkomponenten können keine Plasmaschwingung auf die beschriebene Art anregen.

Berechnet man nun mit Gleichungen (2.37) und (2.38) die longitudinale und die transversale Komponente des elektrischen Feldes hinter dem Laserpuls, so ergibt sich:

$$E_{2x} = -\frac{mc}{2e}\, k_p\, e^{-ik_p\xi}\,\psi(\varrho)\;, \tag{2.46}$$

$$E_{2\varrho} = i\frac{mc}{2e}\, e^{-ik_p\xi}\,\partial_\varrho\psi(\varrho)\;. \tag{2.47}$$

Aus den Gleichungen ist ersichtlich, daß longitudinale und transversale Komponenten eine Wellenlänge von $\lambda = 2\pi/k_p$ aufweisen und um eine Viertel Wellenlänge gegeneinander versetzt sind.

Während die longitudinale Komponente vom Amplitudenquadrat des Laserpulses abhängt und damit ihre Extrema in der Mitte des Strahlkanals hat, ist bei der transversalen Komponente eine Abhängigkeit von der radialen Ableitung

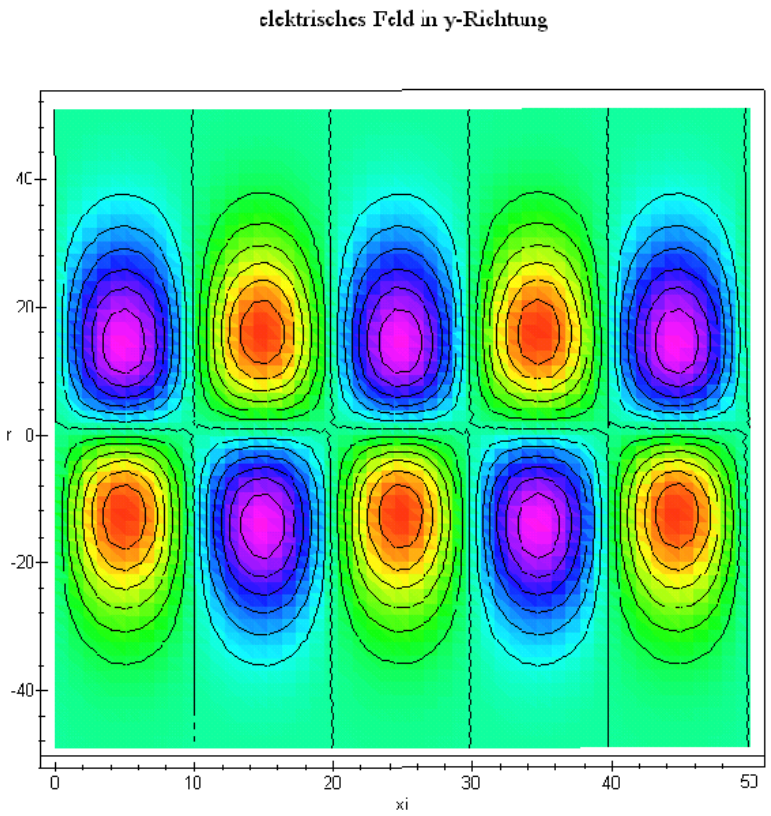

Abbildung 2.2: Lösung für die elektrische Feldkomponente senkrecht zur Ausbreitungsrichtung hinter dem Puls.

des Amplitudenquadrates gegeben. Entsprechend liegen deren Extrema dort, wo die Feldstärke des Lasers von der Mitte weg am stärksten abfällt, bei einem in erster Näherung gaußförmigen Strahlprofil ist dies bei etwas weniger als dem Abstand des halben Fokusdurchmessers der Fall. Die transversale Komponente verschwindet im Zentrum des Strahlkanals.

Die Abbildungen 2.1 und 2.2 zeigen die Feldkomponeten des elektrischen Wakefields in Ausbreitungsrichtung und senkrecht dazu. Die Parameter dazu waren: Intensität $1 \cdot 10^{18}\mathrm{W/cm}^2$, $\omega = 1.886 \cdot 10^{15}1/\mathrm{s}$, $n = 0.1 n_{crit}$, Pulsdauer war $5\ /\ \omega$, Pulsbreite lag bei $20\ /\ k$. Die Darstellung erfolgt in den im Anhang beschriebenen angepaßten Einheiten. Die Amplitude der E_x-Komponente des elektrischen Feldes lagen bei 0.05 $m_e c\, \omega_l / e$, die der E_y-Komponente entsprechend bei 0.006 $m_e c\, \omega_l / e$, was bis auf weniger als 10% Fehler mit den Simulationsergebnissen übereinstimmt.

Die Abbildungen 2.1 und 2.2 sind Darstellungen der Funktionen aus den Gleichungen (2.46) und (2.47), wobei die Berechnung der Gleichungen und die Dar-

stellung mit dem Mathematikprogramm **Maple V** erfolgte. Eine gute Einführung in die Programmierung mit Maple V liefert [14].

2.3 magnetische Wakefields

Während elektrische Wakefields schon in zweiter Ordnung auftreten, zeigen die Gleichungen in dieser Ordnung verschwindende Rotation und damit ein Fehlen magnetischer Felder.

Um diese ausrechnen zu können, muß man die Terme dritter und vierter Ordnung in Gleichung (2.18) wie in (2.19) gezeigt entwickeln. Um nun die Beiträge höherer Ordnung zu errechnen, geht man davon aus, daß die Lösung $\mathbf{q}_3$ die Lösung der Gleichung (2.19) ist, wenn diese bis zur dritten Ordnung entwickelt wurde und auf der rechten Seite als Treiber die Beiträge von $\mathbf{q}_1$ und $\mathbf{q}_2$ stehen. Als Gleichung für $\mathbf{q}_3$ ergibt sich dann:

$$\begin{aligned} \partial_t^2\mathbf{q}_3 + c^2\nabla\times\nabla\times\mathbf{q}_3 + \omega_p^2\mathbf{q}_3 &= -c\partial_t\nabla\mathbf{q}_1\mathbf{q}_2 - c\mathbf{q}_1\nabla\cdot\partial_t\mathbf{q}_2 \\ &\quad +\frac{1}{2}\mathbf{q}_1\left(\omega_p^2 - c^2\Delta\right)q_1^2 \,. \end{aligned} \tag{2.48}$$

Für die Komponente $\mathbf{q}_4$ ergibt sich zuletzt die Gleichung

$$\begin{aligned} \partial_t^2\mathbf{q}_4 + c^2\nabla\times\nabla\times\mathbf{q}_4 + \omega_p^2\mathbf{q}_4 &= -\frac{c}{2}\partial_t\nabla\left(q_2^2 + 2\mathbf{q}_1\mathbf{q}_3 - \frac{1}{4}q_1^4\right) \\ &\quad +\mathbf{q}_1\left(\left[\omega_p^2 - c^2\Delta\right]\mathbf{q}_1\mathbf{q}_2 - c\partial_t\nabla\cdot\mathbf{q}_3\right) \\ &\quad +\mathbf{q}_2\left(\frac{1}{2}\left[\omega_p^2 - c^2\Delta\right]q_1^2 - c\partial_t\nabla\cdot\mathbf{q}_2\right) \,. \end{aligned} \tag{2.49}$$

Das Lösen dieser Gleichungen gestaltet sich etwas schwieriger als bei der Gleichung für $\mathbf{q}_2$. Als erster Schritt bietet es sich an, die Rotation von Gleichung (2.49) zu bilden, und mit der Beziehung (2.14) eine magnetische Feldstärke zu errechnen.

Man erhält dadurch die Gleichung

$$\left(\partial_t^2 - c^2\Delta + \omega_p^2\right)\mathbf{B}_4 = \frac{1}{\epsilon_0}\nabla\times(\mathbf{j}_1 + \mathbf{j}_2) \text{ mit} \tag{2.50}$$

$$\mathbf{j}_1 = \frac{\epsilon_0 mc}{e}\mathbf{q}_2\left(\frac{1}{2}\left[\omega_p^2 - c^2\Delta\right]q_1^2 - c\partial_t\nabla\cdot\mathbf{q}_2\right) \,, \tag{2.51}$$

$$\mathbf{j}_2 = \frac{\epsilon_0 mc}{e}\mathbf{q}_1\left(\left[\omega_p^2 - c^2\Delta\right]\mathbf{q}_1\cdot\mathbf{q}_2 - c\partial_t\nabla\cdot\mathbf{q}_3\right) \,. \tag{2.52}$$

Um nun die Größe $\mathbf{j}_2$ berechnen zu können, fehlt uns noch ein Lösung für $\nabla\cdot\mathbf{q}_3$. Diese können wir durch eine Fouriertransformation im Zeitbereich aus der Gleichung (2.48) gewinnen:

$$\begin{aligned} \left(\partial_t^2 + \omega_p^2\right)\nabla\cdot\mathbf{q}_3 &= \mathbf{q}_1\cdot\nabla\left(\frac{1}{2}\left[\omega_p^2 - c^2\Delta\right]q_1^2 - c\nabla\cdot\partial_t\mathbf{q}_2\right) \\ &\quad -c\partial_t\Delta(\mathbf{q}_1\cdot\mathbf{q}_2) \,. \end{aligned} \tag{2.53}$$

Damit ergibt sich als Lösung für die Größe $\nabla \cdot \mathbf{q}_3$:

$$\begin{aligned} \nabla \cdot \mathbf{q}_3 &= \frac{1}{\omega_p} \int_{-\infty}^{t} d\tau \sin\left(\omega_p(t-\tau)\right) \\ &\quad * \left\{ \mathbf{q}_1 \cdot \nabla \left(\frac{1}{2} \left[\omega_p^2 - c^2\Delta\right] q_1^2 - c\partial_t \nabla \cdot \mathbf{q}_2 \right) - c\partial_t \Delta(\mathbf{q}_1 \cdot \mathbf{q}_2 \right\} . \end{aligned} \tag{2.54}$$

Um nun den näherungsweisen Verlauf der säkularen magnetischen Felder zu kennen, muß man eine Lösung für $\mathbf{q}_1$ kennen. Dazu genügt es aber, wenn wir uns zur Beschreibung der Wakefield-Phänomene auf den quasistatischen Anteil des Stromes beschränken. Diesen erhält man, indem man für das Vektorpotential $\mathbf{A}$ der Welle eine Funktion der Form

$$\mathbf{A}(\varrho, x, t) = \mathbf{A}_0(\varrho, x, t) \exp\left(-\frac{\varrho^2}{2\varrho_0^2}\right) \sin(\omega t - kx) \tag{2.55}$$

ansetzt.

Hierbei ist ϱ_0 die Halbwertsbreite des Laserpulses, k die Vakuum-Wellenzahl. Die langsam in Ort und Zeit veränderliche Envelope ist durch die Funktion $\mathbf{A}_0$ beschrieben. Damit kann man nun den Strom in Gleichung (2.50) mit Hilfe des Vektorpotentials ausdrücken, wenn man die Definition des Vektorpotentials und die Gleichung (2.14) benutzt:

$$\mathbf{j} = \frac{n_0 e c^2}{4\omega_p^2} \left(\nabla \partial_t \phi_0\right) \left[\frac{|\mathbf{A}_0|^2}{2} - \frac{c^2}{\omega_p} \Delta\phi_0 \right] \quad \text{mit} \tag{2.56}$$

$$\phi_0 = \frac{\omega_p}{2} \int_{-\infty}^{t} d\tau \sin \omega_p(t-\tau) |\mathbf{A}_0|^2 \tag{2.57}$$

Im mitbewegten System ergeben sich daraus für $\mathbf{A}_0$ und ϕ_0 die Gleichungen

$$\mathbf{A}(\varrho, \xi, x) = \mathbf{A}_0(\varrho, \xi) \exp\left(-\frac{\varrho^2}{2\varrho_0^2}\right) \sin(k\xi) \tag{2.58}$$

$$\phi_0 = \frac{k_p}{2} \int_{-\infty}^{\xi} d\tau \sin k_p(\xi - \tau) |\mathbf{A}_0|^2 . \tag{2.59}$$

Nutzt man nun noch die Beziehung

$$\partial_\xi^2 \phi_0 = \frac{k_p^2}{2} |\mathbf{A}_0|^2 - k_p^2 \phi_0 \quad \text{aus,} \tag{2.60}$$

so ergibt sich für den Strom $\mathbf{j}$ der Ausdruck

$$\mathbf{j} = \frac{n_0 e c}{4k_p^4} \left((\partial_\varrho - \partial_\xi) \partial_\xi \, \phi_0 \right) \left[(k_p^2 - \Delta_\perp) \phi_0 \right] \quad . \tag{2.61}$$

Der Ableitungsoperator $\Delta_\perp$ hat im betrachteten Koordinatensystem bei Anwendung auf einen Skalar die folgende Struktur:

$$\Delta_\perp = \partial_\rho^2 + \frac{1}{\rho^2} \partial_\varphi^2 \tag{2.62}$$

Der Strom **j** hat somit Beiträge in der j_x- und der j_ϱ-Komponente:

$$j_x = \frac{n_0 ec}{4k_p^4}\left(\partial_\xi^2\ \phi_0\right)\left[(k_p^2-\Delta_\perp)\phi_0\right] \quad \text{und} \tag{2.63}$$

$$j_\varrho = \frac{n_0 ec}{4k_p^4}\left(\partial_\varrho\partial_\xi\ \phi_0\right)\left[(k_p^2-\Delta_\perp)\phi_0\right] \quad . \tag{2.64}$$

Mit diesem Strom läßt sich nun auch das magnetische Feld der Welle im Plasma ausrechnen, indem man die Rotation des Stromes bildet und diesen Ausdruck in die Gleichung (2.50) einsetzt. Dabei stellt man fest, daß das Magnetfeld nur eine B_φ-Komponente besitzt, da sowohl die partiellen Ableitungen der Stromkomponenten in φ-Richtung verschwinden als auch die Stromkomponente selbst. Für die magnetische Feldkomponente B_φ ergibt sich damit:

$$\left(\frac{1}{\varrho}\partial_\varrho\varrho\partial_\varrho - \frac{1}{\varrho^2} - k_p^2\right)B_\varphi = -\frac{1}{4\epsilon_0}\frac{n_0 e}{ck_p^4}F(\varrho,\xi) \quad \text{mit} \tag{2.65}$$

$$\begin{aligned} F(\varrho,\xi) = {} & \left(\partial_\varrho\partial_\xi\ \psi_0\right)\partial_\xi\left[(k_p^2-\Delta_\perp)\ \psi_0\right] - \\ & \left(\partial_\xi^2\ \psi_0\right)\partial_\varrho\left[(k_p^2-\Delta_\perp)\ \psi_0\right] \quad , \end{aligned}$$

wobei die Funktion $F(\varrho,\xi)$ durch das Bilden der Rotation des Stromes zustandekommt[3].

Um nun eine Lösung für das Magnetfeld zu finden, ist es sinnvoll, die Differentialgleichung auf dimensionslose Größen zu bringen und eine mathematische Funktionensammlung wie [10] zu bemühen. Als Transformationsgleichungen eignen sich

$$r = \frac{k_p}{\varrho} \quad \text{und} \tag{2.66}$$

$$W(r) = \frac{\epsilon_0 ck_p}{en_0}B_\phi(\varrho)\,. \tag{2.67}$$

Damit ergibt sich für die Gleichung (2.65) für die r-Richtung die folgende Differentialgleichung:

$$\left(\frac{1}{r}\partial_r r\partial_r - \frac{1}{r^2} - 1^2\right)W(r) = -F(r) \quad \text{mit} \tag{2.68}$$

$$\begin{aligned} F(r) = {} & \left(\partial_r\partial_\xi\ \psi_0\right)\partial_\xi\left[(1-\Delta_\perp)\ \psi_0\right] - \\ & \left(\partial_\xi^2\ \psi_0\right)\partial_r\left[(1-\Delta_\perp)\ \psi_0\right] \quad , \end{aligned}$$

Den in dieser Gleichung enthaltenen Ableitungsoperator kann man mit Hilfe der Produktregel noch einmal umformen zu

$$\frac{1}{r}\partial_r\ r\partial_r = \partial_r^2 + \frac{1}{r}\partial_r \tag{2.69}$$

[3]Die Produktregel bei der Bildung der Differentiale auf die Stromkomponenten angewandt, bewirkt daß die ersten der beiden Summanden eines jeden Teilausdrucks sich beim Bilden der Rotation wegheben.

und erhält dann die Gleichung

$$\left(\partial_r^2 + \frac{1}{r}\partial_r - \left(\frac{1}{r^2} + 1\right)\right) W(r) = -F(r) \; . \tag{2.70}$$

Indem man nun die Gleichung (2.70) mit dem Faktor r^2 multipliziert, gelangt man zur besselschen Differentialgleichung 1. Ordnung in der modifizierten Form:

$$r^2 \frac{d^2 w}{dr^2} + r\frac{dw}{dr} - (r^2 + 1)w = 0 \; . \tag{2.71}$$

Die Lösungen von Gleichung (2.71) sind die Bessel-Funktionen $I_{\pm 1}$ und K_1. Für uns interessant sind die Lösungen, die für $r \to 0$ beziehungsweise $r \to \infty$ verschwinden; dies sind I_1 und K_1.

Aus [10] sind in [4] darüberhinaus noch die folgenden Beziehungen für die Lösungen K_1 und I_1 hergeleitet:

$$\begin{aligned} r\partial_r K_1 + K_1 &= -rK_0 \; , & (2.72) \\ r\partial_r I_1 + I_1 &= rI_0 & (2.73) \end{aligned}$$

Dabei sind die Funktionen K_0 und I_0 definiert als die Lösungen der besselschen Differentialgleichung 0. Ordnung. Zwischen den Lösungen zweier solcher Differentialgleichung von um 1 verschiedener Ordnung gibt es eine weitere Beziehung, nämlich

$$I_n K_{n+1} + I_{n+1} K_n = \frac{1}{r} \tag{2.74}$$

Mit den in Gleichungen (2.71) bis (2.74) beschriebenen Eigenschaften kann man durch Nachrechnen verifizieren, daß die Lösung für Gleichung (2.70) sich als Summe aus zwei Integralausdrücken der Funktionen I_1 und K_1 mit der Funktion $F(r, \xi)$ in der folgenden Art zusammensetzen muß:

$$W(r) = -I_1(r) \int_\infty^r d\rho \; \rho K_1(\rho) F(\rho, \xi) + K_1(r) \int_0^r d\rho \; \rho I_1(\rho) F(\rho, \xi) \; . \tag{2.75}$$

2.3.1 Magnetfelder hinter einem Laserpuls endlicher Länge

Ähnlich wie im Fall der elektrischen Wakefields vereinfachen sich die Verhältnisse auch bei magnetischen Wakefields, wenn man einen Puls von endlicher Dauer annimmt und nur an den Verhältnissen hinter dem Puls interessiert ist. Angenommen, der Puls ist wieder symetrisch um $\xi = 0$ verteilt, so läßt sich die Funktion ϕ_0 aus Gleichung (2.59) schreiben als:

$$\begin{aligned} \phi_0 &= A \sin(k_p \xi - \theta) \quad \text{mit} & (2.76) \\ A &= \sqrt{A_1^2 + A_2^2} \quad \text{und} \\ \tan\theta &= \frac{A_1}{A_2} \, , \end{aligned}$$

die beiden Zahlenwerte A_1 und A_2 sind die Werte der im Ortsraum Sinus- und Kosinustransformierten des Quadrates des Vektorpotentials des Laserpulses an der Stelle k_p:

$$A_1 = \frac{k_p}{2}\int_{-L}^{+L} d\tau \cos k_p\tau |A_0(\rho,\tau)|^2 \tag{2.77}$$

$$A_2 = \frac{k_p}{2}\int_{-L}^{+L} d\tau \sin k_p\tau |A_0(\rho,\tau)|^2 \tag{2.78}$$

Nun kann man damit die Funktion F der Gleichung (2.66) schreiben als

$$F(\rho,\xi) = \frac{k_p^2}{2}\Big(\partial_\rho\Big(A^2 - A\Delta_\perp A\Big) + (A\partial_\rho\Delta_\perp A - (\partial_\rho A)\,\Delta_\perp A) \cos 2(k_p\xi - \theta)\Big)\,. \tag{2.79}$$

Man erkennt in Gleichung (2.79) sowohl einen konstanten Anteil des Magnetfeldes als auch einen Anteil, der mit der doppen Plasmafrequenz schwingt. Das Zustandekommen dieser Anteile ist einsichtig, wenn man berücksichtigt, daß das magnetische Wakefield ein Resultat aus der Verknüpfung zweier Schwingungen mit der Frequenz ω_p ist: Deren Produkt enthält sowohl Komponenten der Frequenz 0 als auch solche der Frequenz $2\omega_p$. Die beiden Schwingungen sind die Veränderung der Dichte und die Geschwindigkeit des Elektronenfluids. Beide bilden zusammen den Strom in der Form $\mathbf{j} = -en_0\delta n_2\mathbf{v_2}$.

Vergleicht man die hier gefundenen Ergebnisse mit den magnetischen Feldern, wie sie bei Simulationen auftreten, so findet man in beiden Fällen statische Komponenten und solche mit der beschriebenen $2\omega_p$-Abhängigkeit. In den Simulationen tauchen aber auch Komponenten mit einer direkten $2\pi/k_p$-Periodizität auf, die man in dem hier beschriebenen Modell nur so erklären kann, daß die Stromdichte aus den elektrischen Wakefields in 2. Ordnung möglicherweise nicht völlig rotationsfrei ist und deshalb auch eine Magnetfeldkomponente mit der $2\pi/k_p$-Periodizität des elektrischen Wakefields auftaucht. Allerdings sind diese Komponenten nicht intensiver als die Komponenten der doppelten Frequenz, können aber dazu führen, daß einzelne Extrema überdeckt werden.

Der statische Anteil des magnetischen Wakefiels ist in der Nähe der Strahlachse so beschaffen, daß er Elektronen bündeln kann, die sich in Richtung der Ausbreitung des Lasers bewegen.

Kapitel 3

Teilchenbeschleunigung in der Plasmawelle

Unter bestimmten Umständen können in einer genügend intensiven Plasmawelle auch Teilchen beschleunigt werden. Dazu sind zwei Bedingungen nötig:

- Es muß Teilchen geben, die eine Anfangsgeschwindigkeit nahe an der Gruppengeschwindigkeit der Plasmawelle besitzen und
- die Intensität der Welle muß ausreichend groß sein, so daß diese Teilchen während einer halben Wellenlänge ausreichend beschleunigt werden können.

3.1 Mechanismus der Teilchenbeschleunigung

Die elektrostatische Plasmawelle besitzt ein Potential ϕ, das sich im Fall der Wake-Field-erzeugten Welle mit der Gruppengeschwindigkeit des Lasers in dessen Ausbreitungsrichtung durch das Plasma bewegt. Die Gruppengeschwindigkeit des Lasers ist gegeben durch:

$$v_g = \frac{\partial \omega}{\partial k}. \tag{3.1}$$

Für die Ausbreitung elektromagnetischer Wellen im Plasma gilt die Dispersionsrelation

$$c^2 k^2 = \omega^2 - \omega_p^2. \tag{3.2}$$

Damit läßt sich die Frequenz in Abhängigkeit der Wellenzahl ausdrücken:

$$\omega = \sqrt{c^2 k^2 + \omega_p^2}, \tag{3.3}$$

womit sich für die Gruppengeschwindigkeit aus Gleichung (3.1) die Beziehung

$$v_g = \frac{2c^2 k}{\sqrt{c^2 k^2 + \omega_p^2}} \tag{3.4}$$

ergibt. Drückt man nun die Wellenzahl k noch durch die Größen ω und ω_p aus, so ergibt sich die Beziehung

$$v_g = c\sqrt{1 - \frac{\omega_p^2}{\omega^2}}. \tag{3.5}$$

Für ein Plasma, das 1/10 der kritischen Dichte des Lasers besitzt, bedeutet das immerhin noch eine Gruppengeschwindigkeit von $0,9c$. Bei Werten von $\omega/\omega_p > 10$ ergeben sich entsprechend Werte von $v_g > 0,99c$. Für die Wellenzahl k des Wakefields ergibt sich damit

$$k = \frac{\omega_p}{v_g} = \frac{1}{c}\frac{\omega\omega_p}{\sqrt{\omega^2 - \omega_p^2}} \tag{3.6}$$

Da sich der Laserpuls mit der Geschwindigkeit v_g durch das Plasma bewegt, ist dies wegen der Ankopplung des Wake-Fields an den Laser die *Phasengeschwindigkeit* des Potentials des elektrostatischen Wakefields. Mit dieser etwas abweichenden Phasengeschwindigkeit läßt sich die Gleichung (2.44) nun wie folgt formulieren:

$$\Phi = \frac{1}{2}\left(ie^{-ik\xi}\,\psi(\varrho) + \text{cc.}\right)\ , \tag{3.7}$$

wobei ψ hinter dem Wellenpuls als einfache Funktion der radialen Entfernung von der Strahlachse betrachtet werden kann. Für Elektronen bedeutet das, daß die maximale Energie, die sie aus der Welle gewinnen können gegeben ist durch

$$\Delta E = 2 \cdot e\ \hat{\Phi}\ , \tag{3.8}$$

wobei $\hat{\Phi}$ der maximale Betrag der Funktion $\Phi(\varrho)$ ist.

Bewegt sich nun ein Teilchen mit der Phasengeschwindigkeit des elektrischen Wakefields durch das Plasma, so sieht es ein zeitlich konstantes Potential. Da das Potential in erster Näherung einen sinusförmigen Verlauf besitzt, können Teilchen in seinem Minimum mit konstantem Impuls bewegt werden. Für diese Teilchen gilt dann $v = v_g$.

Betrachtet man die Zusammenhänge zwischen der Geschwindigkeit eines Teilchens und seinem Impuls, so gibt es aus der speziellen Relativitätstheorie den Zusammenhang

$$\gamma = \frac{1}{\sqrt{1 - \frac{v^2}{c^2}}} = \sqrt{1 + \frac{p^2}{m^2c^2}}\ . \tag{3.9}$$

Für ein mit der Welle bewegtes Teilchen bedeutet dies, daß der Impuls des Teilchens gegeben ist durch

$$p = mc\sqrt{\frac{v_g^2}{c^2 - v_g^2}} \tag{3.10}$$

Drückt man den Impuls durch die Plasmafrequenz und die Laserfrequenz aus, so ergibt sich

$$p = mc\sqrt{\frac{\omega^2 - \omega_p^2}{\omega_p^2}} \tag{3.11}$$

Erreichen die Elektronen die Phasengeschwindigkeit der Welle im mitbewegten Potential an einer Stelle, die über dem Minimum des Potentials liegt, so können die Elektronen auf dem Weg zum Minimum hin noch mehr Energie aufnehmen, die Welle also gewissermaßen überholen. Erst wenn sie dann das Minimum wieder durchlaufen haben, werden sie abgebremst und verringern ihre Energie wieder. Aus diesem Grund gibt der Impuls in Formel (3.11) eigentlich nur einen groben Anhaltspunkt für den Impuls eines mit relativistischer Geschwindigkeit in einer Welle gefangenen Teilchens an.

Kapitel 4

Numerische Plasmasimulation

Wie simuliert man mit vertretbarem Rechenaufwand und Speicherbedarf einen Aggregatzustand, dessen vorrangiges Kennzeichen es ist, daß er aus einer Vielzahl geladener Teilchen besteht und deren kollektive Wechselwirkung den größten Teil der Physik dieses Aggregatzustandes ausmacht?

Doch halt, man kann ja mal klein anfangen: Warum sich nicht mit der Simulation einiger tausend Teilchen begnügen, deren Freiheitsgrade bis auf das unbedingt Nötige vereinfachen und die Bewegungsgleichungen auf die elektromagnetischen Kräfte beschränken, die den Großteil der Plasmaphysik bestimmen?

Auf diese Art und Weise entstanden in den 70er Jahren die ersten PIC-Codes, die konsistent Bewegungs- und Feldgleichungen für das Plasma lösten. Was damals die Rechenleistung eines Großrechners erforderte, läßt sich Dank der rapiden Entwicklung auf dem Gebiet der Rechnerhardware heute auf einem PC rechnen.

Der während dieser Diplomarbeit geschriebene PIC-Code bewegt die simulierten Teilchen entsprechend den elektrischen und magnetischen Feldern am Ort der Teilchen und löst die Maxwellgleichungen unter Berücksichtigung der Materiebewegungen.

4.1 PIC - „Particle in Cell“

PIC ist die Abkürzung für den englischen Ausdruck „Particle in Cell“, zu deutsch: „Teilchen in einer Zelle“. PIC ist eine Methode, mit einfachem Aufwand Vielteilchenrechnungen durchzuführen. Es werden hier einige Tausend bis zu mehrere Millionen Teilchenbahnen berechnet. Der Rechenaufwand hält sich aber in Grenzen, weil die Teilchen nicht direkt miteinander in Wechselwirkung treten, sondern mit einem auf Stützstellen definierten elektromagnetischen Feld. Dabei muß beachtet werden, daß ein berechnetes Teilchen nicht exakt einem Teilchen in der Realität entspricht, vielmehr wird hier über die Wahl von Skalierungsfaktoren beispielsweise nur jede 1000ste Bahn berechnet. Ein berechnetes Teilchen verursacht dann die Feldveränderung, die 1000 Teilchen mit dieser Bewegungsrichtung

und einem Ort innerhalb derselben Gitterzelle verursachen würden.

Grundlage von PIC ist die Repräsentation des elektrischen und magnetischen Feldes durch Werte an Stützstellen auf einem Gitter von fester Schrittweite. An diesen Stützstellen werden die Feldstärken zu jedem Zeitschritt neu berechnet. Um Speicherplatz zu sparen, sind die Algorithmen dabei so angelegt, daß die Berechnung die alten Werte an Ort und Stelle überschreibt. Es sind also immer nur zu einem einzigen Zeitschritt die Daten im Speicher. Werden die Daten für Auswertungen benötigt, so muß man sie von Zeit zu Zeit in Dateien abspeichern.

Die Teilchen sind in ihrer Bewegung natürlich nicht an das diskrete Gitter gebunden; das bedeutet, daß die Feldstärken am Ort der Teilchen auf eine bestimmte Weise aus den Werten an den Gitterpunkten ermittelt werden muß, um die Impulsänderung des Teilchens möglichst exakt errechnen zu können.

Im folgenden wird für die Indizierung in x-Richtung immer der Index i verwendet, während der Index j für die Indizierung in y-Richtung steht. Ein hochgestellter Index n steht für den Zeitschritt der jeweiligen Größe.

Abbildung der Feldgrößen auf ein Gitter

Zunächst einmal stellt sich die Frage, wie das elektromagnetische Feld aus Sicht der Teilchen berechnet wird und welche Einschränkungen sich aus der Berechnungsmethode ergeben.

Die Ortsabhängigkeit der Vektoren und der skalaren Dichten wird durch die Werte an unterschiedlichen Gitterstellen repräsentiert:

$$\begin{aligned} \mathbf{E}(\mathbf{x},t) &\simeq \mathbf{E}^n_{i,j} \\ \mathbf{B}(\mathbf{x},t) &\simeq \mathbf{B}^n_{i,j} \\ \mathbf{j}(\mathbf{x},t) &\simeq \mathbf{j}^n_{i,j} \\ \rho(\mathbf{x},t) &\simeq \rho^n_{i,j} \end{aligned}$$

Die Ortsableitung, die in den Maxwellgleichungen auftauchen, werden durch Differenzen zwischen Werten benachbarter Gitterpunkte ersetzt. Auch die totalen Zeitableitungen, die in den Maxwellgleichungen (2.1) und (2.3) stehen, werden zu Differenzen. Auf diese Art und Weise wird aus dem System partieller Differentialgleichungen ein System von linearen partiellen Differenzengleichungen.

4.2 Elektrostatischer Teil des PIC-Codes

Unter diesem Aspekt des PIC-Codes möchte ich die Umrechnungen zwischen Gitter- und Teilchenmodell sowie das Lösen der Poisson-Gleichung für das elektrostatische Feld der Teilchen betrachten. Zwischen den Gitterpunkten muß der

Wert der Feldstärke durch eine geeignete Interpolation ermittelt werden. Umgekehrt müssen die Teilchen und ihre Bewegung wieder auf ein Gitter abgebildet werden. Diesen Vorgang bezeichnet man deswegen als Gewichtung, wobei diese Gewichtung in beiden Richtungen erfolgen muß. Sinnvollerweise verwendet man in beide Richtungen gleiche Gewichte, um Effekten wie eine Selbstbeschleunigung von Teilchen vorzubeugen.

Für ein zweidimensionales Modell mit Teilchenbewegungen in der x-y-Ebene sind auf jeden Fall zwei Komponenten des elektrischen Feldes notwendig: dies sind E_x und E_y.

4.2.1 Gewichtung der Teilchen auf das Gitter

Unter der Gewichtung der Teilchen kann man sich folgendes vorstellen: Jedes PIC-Teilchen besteht aus einer Anzahl realer Teilchen mit einer bestimmten Verteilung in Orts- und Impulsraum. Im Impulsraum wird eine δ-förmige Verteilung angenommen, d.h., alle Teilchen besitzen gleichen Impuls und bewegen sich nie auseinander (dies ist der Unterschied zu komplexeren Codes, bei denen auch im Impulsraum eine bestimmte Verteilung angenommen werden kann und die simulierten Teilchen dann auch eine individuelle Temperatur bekommen). Die Verteilung der simulierten Teilchen im Ortsraum wird durch die Gewichtung von Teilchen auf das Gitter bestimmt.

Gewichtung 0. Ordnung

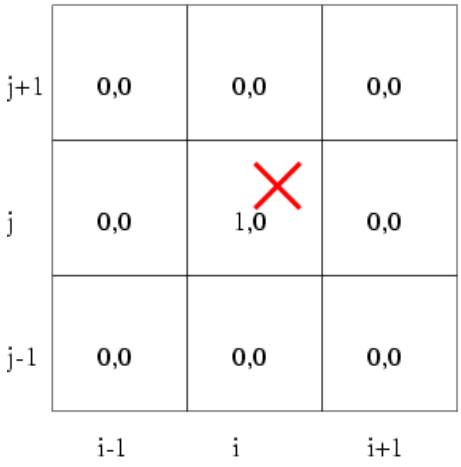

Abbildung 4.1: Direkte Gewichtung des Teilchens auf eine Stützstelle

Die einfachste Form der Gewichtung ist eine Gewichtung zu 100 % auf eine Stützstelle. Das bedeutet im Fall der Feldstärken, daß die Feldstärke am Ort eines Teilchens nur aus der Feldstärke an der nächstgelegenen Stützstelle besteht.

Umgekehrt wirkt jedes Teilchen auch nur auf die nächste Stützstelle. Das Bild 4.1 veranschaulicht dies: das Gewicht des Punktes (i, j) ist 1.0, alle anderen Punkte besitzen kein Gewicht. Diese Art der Gewichtung bringt einige Nachteile mit sich: So ist beispielsweise die Kraft auf ein Teilchen während der Bewegung nicht stetig, sondern „springt“ beim Wechsel von einer Stützstelle zur nächsten. Insgesamt führt diese einfachste Art der Gewichtung zu relativ starkem „Rauschen“ in der Kraftwirkung, die die Teilchen aufeinander ausüben. Dieses Rauschen ist z. B. auch in der Energie sichtbar.

Die direkte Gewichtung bedeutet im Grunde, daß sich alle durch ein PIC-Teilchen simulierte realen Teilchen am selben Ort des Gitters aufhalten. Außer im Impulsraum besitzen die simulierten Teilchen auch eine δ-Verteilung im Ortsraum.

Lineare Gewichtung

Die bessere Alternative ist eine lineare Gewichtung. Man kann sich dazu das Feld in Kacheln mit der Breite dx und der Tiefe dy aufgeteilt vorstellen. In der Mitte der Kachel befindet sich die Stützstelle des Feldgitters. Ein Teilchen ist nun kein Punkt, sondern ebenfalls eine Zelle der Breite dx und der Tiefe dy (daher die Bezeichnung „Particle in Cell“).

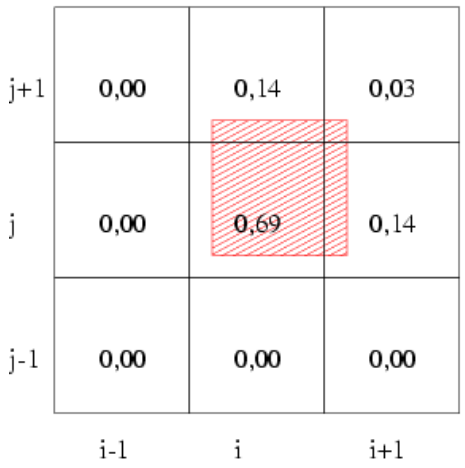

Abbildung 4.2: Lineare Gewichtung

Die Gewichtung des Teilchens auf die Stützstellen ergibt sich nun aus den Flächenanteilen, mit der die Teilchenzelle die Gitterzelle überdeckt. Ein Teilchen wirkt damit auf minimal eine und maximal vier Gitterzellen. Mit dem gleichen Verfahren werden auch die Felder am Ort des Teilchens ermittelt. Hierdurch ändert sich die Kraft nicht sprunghaft wie bei der direkten Gewichtung, sondern bei Annäherung zweier Teilchen eher allmählich. Das Bild 4.2 veranschaulicht

diesen Zusammenhang. Hier sind die Gewichtungsfaktoren für eine Überlappung der Teilchen- und Gitterkoordinaten im Verhältnis 5/6 eingetragen.

Die lineare Gewichtung bedeutet im Grunde, daß sich alle durch ein PIC-Teilchen simmulierte realen Teilchen homogen verteilt auf der Fläche einer Gitterzelle aufhalten. Der Mittelpunkt der Gitterzelle ist die Teilchenkoordinate.

Gewichtungen höherer Ordnung

Es gibt noch weitergehende Möglichkeiten, die auch entferntere Stützstellen noch in die Gewichtung mit einfließen lassen. Diese verwenden sogenannte Spline-Kurven, um die Feldstärke am Ort des Teilchens aus den Feldstärken an den Stützstellen zu ermitteln. Hier kommen quadratische Gleichungen oder Gleichungen noch höherer Ordnung zum Einsatz. Da aber die Gewichtung pro Teilchen mehrmals zu jedem Zeitschritt erfolgen muß, sollte man hier sehr überlegen, ob sich der Aufwand lohnt.

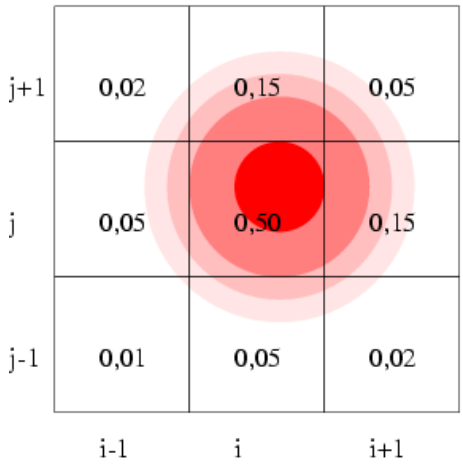

Abbildung 4.3: Gewichtung höherer Ordnung

Das Bild 4.3 veranschaulicht diese Art der Gewichtung, hier am Beispiel einer Gewichtung 2. Ordnung mit quadratischen Splines in jede Richtung. Durch die Splines wird im Prinzip eine Ladungsverteilung innerhalb der „Teilchenzelle“ vorgegeben, die dann die Gewichtung auf die Gitterzellen bestimmt.

4.2.2 Physikalische Darstellung der PIC-Teilchen

Aus physikalischer Sicht stellt ein PIC-Teilchen in einem 2D-Code eine geladene „Kachel“ einer bestimmten Masse und Ladungsdichte dar, wobei die Verteilung der Ladung von der gewählten Gewichtung abhängt.

Das Verhältnis von Ladung zu Masse ist dabei von der simulierten Teilchenart abhängig. Die Ladungsdichte der PIC-Teilchen kann entweder für alle Teilchen

global festgelegt werden, oder sie kann für jedes Teilchen individuell z.B. zur Modellierung von Dichteprofilen variiert werden. In meiner Diplomarbeit habe ich mich für die erste Methode entschieden, da sich Dichtevariationen auch über eine Variation der PIC-Teilchendichten erzeugen lassen.

4.2.3 Notwendige Größen für die Berechnung

Teilchendaten

Für jedes berechnete Teilchen benötigt man Speicherplatz für Orts- und Impulskoordinaten. Bei zweidimensionaler Berechnung sind dies vier Zahlenwerte (im Fall eines $2\frac{1}{2}$-dimensionalen Codes sind es fünf Zahlenwerte).

Verwendet man zum Abspeichern der Daten einfach genaue Fließkommazahlen, so ergibt sich ein Speicherplatzbedarf von 128 Bit oder 16 Byte für ein Teilchen. Bei doppelt genauen Fließkommazahlen verdoppelt sich der Speicherbedarf auf 32 Byte pro Teilchen. Der relativistische Gammafaktor wird nicht abgespeichert, da er aus den Werten für des Impulses berechnet werden kann. Zwar verlangsamt dies etwas die Berechnung, verringert aber den Speicherplatzbedarf für die Teilchendaten.

Felddaten

Für das elektrostatische Feld werden die Feldkomponenten E_x und E_y benötigt. Diese werden aus dem elektrostatischen Potential ermittelt, das wiederum aus der Ladungsverteilung erstellt wird. Es wird also für die volle Gitterauflösung ein zweidimensionales Feld der elektrostatischen Feldstärke, des Potentials und der Ladungsverteilung benötigt. Durch das Lösen der Poisson-Gleichung wird das elektrostatische Potential einer gegebenen Ladungsverteilung und daraus dann die longitudinalen Feldanteile des elektrischen Feldes berechnet. Um Feldberechnung und Teilchenbewegung zu koppeln, wird zusätzlich noch die von den Teilchen generierte Stromdichte in der Auflösung des Feldgitters benötigt.

4.2.4 Lösen der Poisson-Gleichung in einer Dimension

Für die Initialisierung der elektrostatischen Felder zu Begin der Simulation und um während der Simulation die Quasi-Neutralität des Plasmas aufrechtzuerhalten sowie zur Ermittelung der potentiellen Energie der Teilchen, ist es nötig, die Poisson-Gleichung für das elektrostatische Potential zu lösen zu können:

$$\Delta\phi = -\frac{1}{\epsilon_0}\rho \tag{4.1}$$

Schreibt man den Laplace-Operator in Form eines Differenzenquotienten, so sieht man, daß die Gleichung (4.1) in einer Dimension auf ein lineares Gleichungssystem mit einer tridiagonalen Matrix führt:

$$\phi_{i-1} - 2 \cdot \phi_i + \phi_{i+1} = -\frac{\Delta x^2}{\epsilon_0} \cdot \rho_i \tag{4.2}$$

Für den ersten bzw. letzten zu berechnenden Wert muß man noch Randwerte einführen, dann ist die Gleichung mit einem Aufwand zu lösen, der nur linear mit der Anzahl Stützstellen ansteigt.

4.2.5 Lösen der Poisson-Gleichung in mehreren Dimensionen

Die Vorgehensweise für das Lösen einer zwei- oder dreidimensionalen Poisson-Gleichung ist meist wie folgt: Reduktion der Ordnung des Problems durch Fouriertransformation der Ladungsdichte ρ in allen bis auf eine Dimension. Damit dies möglich ist, muß das Problem natürlich in den Richtungen, die durch die Fouriertransformation gelöst werden sollen, periodisch sein. Ansosten kann man die direkte Fouriertransformation nicht anwenden und muß auf andere Lösungsmöglichkeiten ausweichen.

In der letzten Dimension erhält man nach der Fouriertransformation eine Gleichung, die in ihrer Struktur der Gleichung (4.2) ähnlich ist, aber noch für jede der Ortsfrequenzen etwas unterschiedliche Faktoren in der Diagonalen der Matrix trägt. Diese Gleichung löst man unter Berücksichtigung passender Randbedingungen und transformiert zum Schluß das Potential wieder in den Ortsraum zurück.

Sinnvollerweise verwendet man für die Transformation die schnelle Fouriertransformation anstelle der diskreten, was allerdings für die Anzahl der Gitterpunkte in y-Richtung bedeutet, daß diese eine Potenz von 2 sein muß. Dies stellt jedoch keine gravierende Einschränkung dar. Man spart dafür deutlich an Rechenzeit, da die schnelle Fouriertransformation ein $n \log(n)$-Algorithmus ist, während bei der diskreten Fouriertransformation der Rechenaufwand mit n^2 ansteigt.

Im folgenden wird der Index i benutzt, um ein Feld in x-Richtung zu indizieren, während j für die Indizierung in y-Richtung verwendet wird. Falls eine in y-Richtung fouriertransformierte Größe verwendet wird, wird dies durch die Indizierung mit dem Buchstaben m angedeutet. Zur besseren Übersichtlichkeit wurden die hochgestellten Indizes n für den Zeitschritt weggelassen, da der Poisson-Solver immer nur auf Feldern innerhalb desselben Zeitschrittes rechnet.

Zunächst wird eine diskrete Fouriertransformation der Ladungsdichte in y-Richtung durchgeführt:

$$\rho_{i,m} = \sum_{j=0}^{N_y-1} \rho_{i,j} \cdot e^{-i\frac{2\pi m}{N_y} j} \tag{4.3}$$

N_y ist hierbei die Anzahl der Gitterpunkte in y-Richtung. Wenn man den üblichen 5-Punkt-Laplace-Operator in zwei Dimensionen verwendet, ergibt sich die folgende Darstellung der Poisson-Gleichung:

$$\phi_{i-1,m} - 2d_m \cdot \phi_{i,m} + \phi_{i+1,m} = -\frac{\Delta x^2}{\epsilon_0} \cdot \rho_{i,m} \; . \tag{4.4}$$

Die Zahlenwerte d_m berechnen sich wie folgt:

$$d_m = 1 + 2 \cdot \left(\frac{\Delta x}{\Delta y \sin \frac{\pi m}{N_y}} \right)^2 \; . \tag{4.5}$$

Um nun die Gleichung (4.4) lösen zu können, benötigt man noch Bedingungen für die Randwerte. Diese kann man aus folgender Überlegung gewinnen:

Wir können der Einfachheit halber annehmen, daß es keine Teilchen auf dem Randbereich des Gitters in x-Richtung gibt (was i. d. Regel keine zu große Einschränkung darstellt, immerhin sollen unsere Teilchen ja das Feld nicht verlassen). Dann kennen wir aber den Verlauf des Feldes in x-Richtung aus der Gleichung (4.4), es ist nämlich

$$\phi_{i,m} = a r_m^i + b r_m^{-i} \tag{4.6}$$

Wobei r die größere der Wurzeln aus der quadratischen Gleichung

$$r_m^2 - 2d_m r_m + 1 = 0 \tag{4.7}$$

ist. (Die zweite Wurzel ist r^{-1}). Da jeweils nur die Lösung interessant ist, für die das Potential für $r \to \infty$ nicht divergiert, ergibt sich für die Potentiale am Rand des berechneten Bereiches

$$-\left(2d_m - r\right) \phi_{1,m} + \phi_{2,m} = -\frac{\Delta x^2}{\epsilon_0} \rho_{1,m} \tag{4.8}$$

und

$$\phi_{N_x-1,m} - \left(2d_m - r\right) \phi_{N_x,m} = -\frac{\Delta x^2}{\epsilon_0} \rho_{N_x,m} \tag{4.9}$$

Wie man sieht, ändert sich an der tridiagonalen Struktur des Gleichungssystems durch das Einfügen der Randbedingungen nichts. Mittels eines geeigneten Gleichungslösers für tridiagonale Matrizen läßt sich der Wert für das in y-Richtung fouriertransformierte Potential ermitteln, entsprechende Algorithmen finden sich beispielsweise in [11]. Nun muß man nur noch mit einer inversen Fouriertransformation in y-Richtung das Potential im Ortsraum berechnen:

$$\phi_{i,j} = \frac{1}{\Delta y N_y} \sum_{m=0}^{N_y-1} \phi_{i,m} \cdot e^{i \frac{2\pi j}{N_y} m} \tag{4.10}$$

4.2.6 Berechnen des elektrostatischen Feldes

Das elektrostatische Feld läßt sich durch Bilden der Ableitung einfach aus dem Potential ermitteln. Hier verwendet man die Gleichung

$$\mathbf{E} = -\nabla\phi \tag{4.11}$$

In Form von Differenzenquotienten[1] ausgedrückt ergibt sich daraus

$$E_{x\,i,j} = -\frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x} \text{ und} \tag{4.12}$$

$$E_{y\,i,j} = -\frac{\phi_{i,j+1} - \phi_{i,j}}{\Delta y}\,. \tag{4.13}$$

4.3 Elektrodynamische Rechnungen in 2d

Für die Berechnung der elektromagnetischen Felder in den Fällen von Lichtausbreitung muß man auch die magnetischen Felder sowie die transversalen elektrischen Felder berücksichtigen, denn der Strahlungstransport beruht auf der gegenseitigen Wechselwirkung von elektrischen und magnetischen Feldern, wie sie in den vollständigen Maxwellgleichungen enthalten ist.

4.3.1 Die Integration der Feldgleichungen

Die Maxwellschen Gleichungen bestehen sowohl für das elektrische als auch für das magnetische Feld aus je einer vektoriellen und einer skalaren Gleichung. Für die hier durchzuführende numerische Integration sind zunächst einmal die Vektorgleichungen interessant, weil sie für jede Komponente von elektrischem und magnetischem Feld einen Ausdruck für die Zeitableitung liefern.

Diese Gleichungen sind unter (2.1) und (2.3) aufgeführt. Die Integration der Bewegungsgleichungen soll 2-dimensional erfolgen, wobei aber auch örtlich variable Felder in Betracht gezogen werden müssen. Das bedeutet, daß das elektromagnetische Feld mindestens zweidimensional mit den Komponenten E_x, E_y und B_z berechnet werden muß. Die Komponente E_x ist dabei nötig, um z.B. eine Fokussierung des Laserstrahls im berechneten Bereich zu ermöglichen sowie um die säkularen Effekte des Laserlichtes in Ausbreitungsrichtung des Lasers zu erfassen. In [3] wird diese Komponente als *transversal magnetic* (TM) bezeichnet, weil hier die magnetische Feldkomponente als einzige senkrecht auf der berechneten Ebene steht[2].

[1] Durch den Versatz der Stützstellen zwischen dem elektrostatischen Potential und dem elektrischen Feld ist der Ableitungsoperator hier gegenüber dem des elektrischen Feldes mit um den Wert 1 erhöhten Indexwerten definiert

[2] In der Literatur wird hier auch oft von p-polarisiertem Licht gesprochen, in Anlehnung an die Notation in den Fresnelschen Formeln bei der Lichtbrechung und -reflexion an einer Ebene.

Um die Wellen der darauf senkrecht stehenden Polarisationsrichtung rechnen zu können, sind die drei restlichen Feldkomponenten E_z, B_x und B_y nötig. In [3] wird diese Komponente als *transverse electric* (TE) bezeichnet, weil man hier die elektrische Feldkomponente der Welle senkrecht auf der berechneten Ebene stehen hat[3].

Die Gleichungen für die TM-polarisierte Welle koppeln wegen der Homogenität in z-Richtung nicht mit denen für die TE-polarisierte Welle. Die beiden Sätze von Gleichungen für TM- und TE-polarisierte Welle lassen sich also völlig unabhängig voneinander berechnen.

Lokalisierung der Stützstellen

Besondere Aufmerksamkeit muß der Platzierung der Stützstellen gelten, da diese für die Bereitstellung der Anschlußbedingungen an den Rändern des berechneten Bereichs wichtig sind, sowie für die Abfolge der Indizierung bei der Bildung der räumlichen Ableitung.

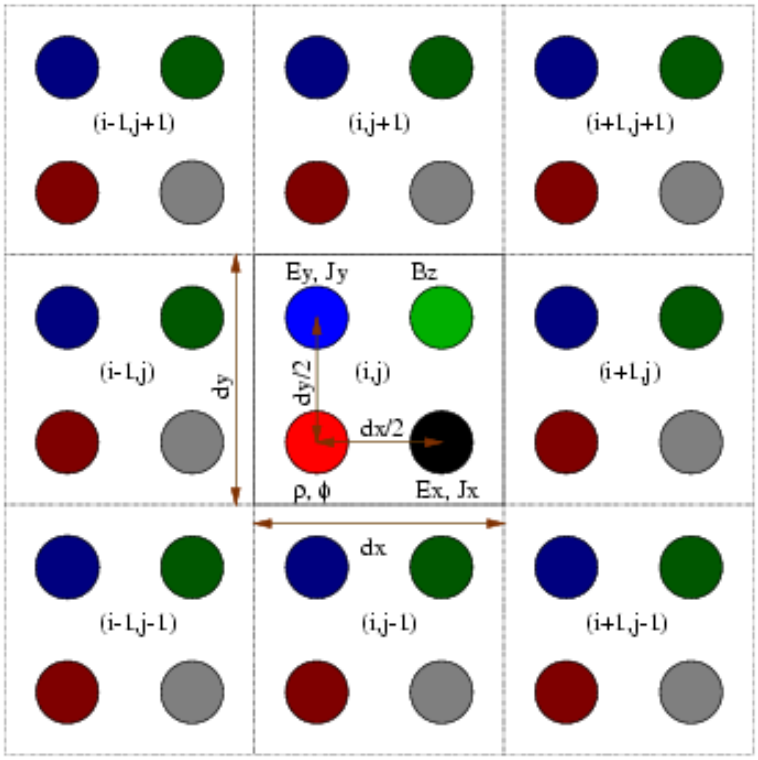

Abbildung 4.4: Die Konfiguration der Stützstellen

[3]Hier wird oft die Bezeichnung *s*-polarisiertes Licht benutzt, in Analogie zu der Notation bei den Fresnelschen Formeln bei der Lichtbrechung und -reflexion an einer Ebene.

Prizipiell gibt es hier zwei Möglichkeiten: zum einen kann man alle Feldkomponenten an einer Stützstelle innerhalb der Gitterzelle positionieren, man kann aber auch die Stützstelle innerhalb der Gitterzelle so verschieben, daß sich die Differentialformen für die in den Gleichungen auftauchenden Ortsableitungen möglichst einfach formulieren lassen. Diese Verschiebung wurde in der vorliegenden Arbeit verwendet. Die Abbildung 4.4 veranschaulicht die in dieser Arbeit verwendete Anordnung der Stützstellen für das Feld.

Leapfrog-Verfahren

Bei der Integration der Feldgleichungen hat man zwei vektorielle Größen, deren zeitliche Veränderung von der Rotation der anderen Größe am gleichen Ort abhängt. Damit hat man aber das Problem, daß zum Zeitpunkt t^n alle Werte zum Zeitpunkt t^{n-1} bekannt sein müssen: man müßte zumindest für eines der Felder einen Zwischenspeicher anlegen.

Man kann aber auch die Zeitintegration der Felder verschachteln: Man teilt einen Zeitschritt in zwei Halbschritte auf, im ersten Halbschritt berechnet man die magnetischen Felder, im zweiten Halbschritt dann die elektrischen. Damit hat man auf der Zeitachse etwas ähnliches gemacht wie im Ort, wo man die Stützstellen um die halbe Schrittweite verschoben hat, um eine räumliche Zentrierung des Ableitungsoperators zu erreichen.

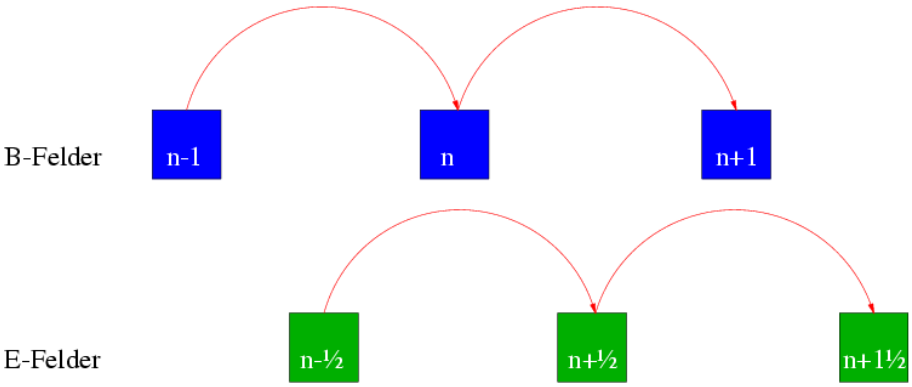

Abbildung 4.5: Der Leapfrog-Algorithmus des Maxwell-Solvers

Das Bild 4.5 zeigt, wie die Integration von elektrischen und magnetischen Feldern zeitlich verschachtelt abläuft.

4.3.2 Integrationsgleichungen

Damit ergeben sich die folgenden Gleichungen für die Integration der Felder, zunächst für die TM-polarisierte Welle (magnetischer Feldstärkevektor senkrecht

zur berechneten Ebene):

$$\begin{aligned}
\frac{\Delta E_{x\,i,j}^{n+\frac{1}{2}}}{\Delta t} &= c^2 \frac{B_{z\,i,j}^{n} - B_{z\,i-1,j}^{n}}{\Delta y} - \frac{1}{\epsilon_0} J_{x\,i,j}^{n} \\
\frac{\Delta E_{y\,i,j}^{n+\frac{1}{2}}}{\Delta t} &= -c^2 \frac{B_{z\,i,j}^{n} - B_{z\,i,j-1}^{n}}{\Delta x} - \frac{1}{\epsilon_0} J_{y\,i,j}^{n} \\
\frac{\Delta B_{z\,i,j}^{n+1}}{\Delta t} &= -\frac{E_{y\,i+1,j}^{n+\frac{1}{2}} - E_{y\,i,j}^{n+\frac{1}{2}}}{\Delta x} + \frac{E_{x\,i,j+1}^{n+\frac{1}{2}} - E_{x\,i,j}^{n+\frac{1}{2}}}{\Delta y}
\end{aligned} \tag{4.14}$$

Man erkennt, daß die Ortsableitungen der elektrischen Felder und des magnetischen Feldes in der Indizierung um 1 verschoben sind. Dies ist eine Folge des Versatzes der Felder um eine halbe Schrittweite. Die z-Komponente des Ableitungsoperators $\nabla\times$ weicht gegenüber den x- und y-Komponenten in der Indizierung um 1 ab.

Beachtet werden muß das insbesondere bei den Werten an den Rändern des zu berechnenden Bereiches: während für die Bildung des elektrischen Feldes Randwerte am Anfang des zu berechnenden Bereiches benötigt werden (bei den Indexwerten 1), sind dies beim magnetischen Feld Randwerte am Ende des zu berechnenden Bereiches (Bei den Indexwerten N_x und N_y).

Für die TE-polarisierte Welle (elektrischer Feldstärkevektor senkrecht zur berechneten Ebene) ergibt sich:

$$\begin{aligned}
\frac{\Delta E_{z\,i,j}^{n+\frac{1}{2}}}{\Delta t} &= c^2 \frac{B_{y\,i+1,j}^{n} - B_{y\,i,j}^{n}}{\Delta x} - c^2 \frac{B_{x\,i,j+1}^{n} - B_{x\,i,j}^{n}}{\Delta y} - \frac{1}{\epsilon_0} J_{z\,i,j}^{n} \\
\frac{\Delta B_{x\,i,j}^{n+1}}{\Delta t} &= -\frac{E_{z\,i,j}^{n+\frac{1}{2}} - E_{z\,i-1,j}^{n+\frac{1}{2}}}{\Delta y} \\
\frac{\Delta B_{y\,i,j}^{n+1}}{\Delta t} &= \frac{E_{z\,i,j}^{n+\frac{1}{2}} - E_{z\,i,j-1}^{n+\frac{1}{2}}}{\Delta x}
\end{aligned} \tag{4.15}$$

4.3.3 Festlegen der Randbedingungen für den Maxwell-Solver

Wie bei jedem Algorithmus, der sich mit der Integration von Feldgleichungen auf einem Gebiet befaßt (unabhängig davon, ob dieses Gebiet ein- oder mehrdimensional ist), muß man sich auch beim Maxwell-Solver Gedanken über die Festlegung der Randbedingungen machen, die auf die Problemstellung angepaßt ist.

Die Geometrie ist so gewählt, daß der Laser in x-Richtung propagiert. In y-Richtung ergibt sich ggf. ein geringfügiger Energietransport durch das Defokussieren des Lasers. Teilchen sollen in y-Richtung nicht reflektiert, sondern wieder in das Gitter eingespeist werden (und zwar am gegenüberliegenden Rand,

damit ihr Impuls beibehalten werden kann). In x-Richtung sollen die Teilchen dagegen reflektiert werden.

Das führt auf folgende Festlegungen für die Randwerte: in y-Richtung ist das Gitter periodisch, Teilchen behalten ihre Bewegungsrichtung bei und werden beim Überschreiten des Randes zum gegenüberliegenden Rand transportiert. In x-Richtung dagegen werden Teilchen beim Überschreiten des Randes durch Negieren der x-Komponente ihres Impulsvektors reflektiert[4].

Verhinderung von Reflexionen an den Grenzen in x-Richtung

Betrachtet man den Energietransport an den Grenzen in x-Richtung, so wird er getragen vom elektrischer und magnetischer Feldstärke, und zwar nur von den Komponenten senkrecht zur x-Richtung.

Will man Reflexionen minimieren, so muß man die Komponente, die aus dem berechneten Gitter hinaus gerichtet ist, über den Rand des Bereichs hinaus propagieren, während die einfallende Komponente an einer Grenzfläche gleich der Intensität des einfallenden Lasers gesetzt wird. An der gegenüberliegenden Grenzfläche wird die einfallende Komponente gelöscht.

Zu diesem Zweck rechnet man die elektrischen und magnetischen Feldkompomponenten in einfallende und ausfallende Komponente der beiden Polarisationsrichtungen am Rand des gerade noch mit den Gleichungen (4.14) und (4.15) berechneten Bereichs um:

$$S_{s,+x} = \sqrt{\frac{1}{2}}\ (E_y + c \cdot B_z)\ , \tag{4.16}$$

$$S_{s,-x} = \sqrt{\frac{1}{2}}\ (E_y - c \cdot B_z)\ , \tag{4.17}$$

$$S_{p,+x} = \sqrt{\frac{1}{2}}\ (E_z - c \cdot B_y)\ , \tag{4.18}$$

$$S_{p,-x} = \sqrt{\frac{1}{2}}\ (E_z + c \cdot B_y)\ . \tag{4.19}$$

Dabei bestimmt der erste Index die Polarisation, während das Vorzeichen des zweiten Indexes angibt, in welche Richtung sich die Komponente ausbreitet.

Um nun die Reflexionen zu unterdrücken, addiert man den aus dem berechneten Bereich heraus propagierenden Teil zu der einfallenden Welle am Rand. Das bedeutet, daß dieser Anteil um genau einen Gitterpunkt in seine Ausbreitungsrichtung propagiert wird. Weiter muß man ihn nicht verschieben, da für die Feldintegration nur die Randbedingungen eine Stützstelle außerhalb des berechneten Bereiches interessant sind. Der in den berechneten Bereich hinein propagierende Anteil wird dabei von der Feldintegration korrekt behandelt.

[4]Gegen Ende der Arbeit wurde auch ein einfaches Verfahren für ein Wiedereinspeisen der in x-Richtung entkommenen Elektronen mit thermischer Anfangsgeschwindigkeit implementiert.

Grenzwerte für die Schrittweiten im Maxwell-Solver

C.K. Birdsall und A.B. Langdon beschreiben in [3] detailliert die Grenzwerte für die Wahl der Schrittweiten in Ort und Zeit. Neben den Bedingungen, daß die Schrittweiten in Ort und Zeit geeignet sein müssen, eine bestimmte Maximalfrequenz auflösen zu können, tritt bei der hier gewählten Art der Integration der Maxwellgleichungen noch ein sogenanntes *Courant*-Kriterium auf, das sich in die Gleichung

$$\frac{1}{c^2\Delta t^2} > \frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \tag{4.20}$$

fassen läßt. Grund für diese Einschränkung ist die Stabilität des Integrationsverfahrens: Insbesondere die Moden mit höchster Frequenz wachsen innerhalb weniger Rechenschritte, wenn Gleichung (4.20) verletzt ist.

4.4 Das Lösen der Bewegungsgleichungen: Der Teilchenmover

Der Teilchenmover hat die Aufgabe, die simulierten Teilchen entsprechend der Bewegungsgleichung

$$\frac{\partial \mathbf{p}}{\partial t} = \frac{\mathbf{F}}{m} , \tag{4.21}$$

$$\frac{\partial \mathbf{x}}{\partial t} = \frac{\mathbf{p}}{\gamma} \tag{4.22}$$

zu bewegen.

Bei PIC-Berechnungen steckt ein Großteil des Rechenaufwandes im Lösen der Bewegungsgleichungen für die Berechnung der Teilchenbahnen. Dies rührt daher, daß die Teilchenzahl meist ein Mehrfaches der Zahl der Gitterpunkte beträgt und die Gleichungen für das Bewegen der Teilchen deutlich aufwendiger sind als die Differenzengleichungen für die Zeitentwicklung der elektromagnetischen Felder. Gleichzeitig liefert aber auch die Schrittweite des Gitters des elektromagnetischen Feldes eine Obergrenze für die Schrittweite dt in der Gleichung (4.20): wird das Courrant-Kriterium verletzt, dann ist die Iteration nicht mehr stabil. Das bedeutet umgekehrt, daß man einen Algorithmus benötigt, der die Teilchenbewegung mit möglichst geringem Rechenaufwand bei einer akzeptablen Genauigkeit durchführen kann; dafür kann umgekehrt inkauf genommen werden, daß die Schrttweite für das Bewegen der Teilchen nicht zu groß sein darf.

4.4.1 Prinzipielle Arbeitsweise des Teilchenmovers

Ein PIC-Teilchen besteht – programmtechnisch gesehen – aus der Kenngröße Ladungs/Masseverhältnis, Flächendichte der Ladung, dem Ort und dem Impuls.

Ort und Impuls können dabei 1, 2 oder 3-dimensional sein. Bei Laser-Plasma-Simulationen kommt man im einfachsten Fall mit einer Dimension im Ort und zwei Dimensionen im Impulsraum aus, für genauere Analysen insbesondere von örtlich variablen Intensitäten kann man eine weitere Dimension im Ort hinzunehmen. Falls noch spezielle Effekte in die Berechnung eingehen sollen, kann es sinnvoll sein, auch die dritte Dimension noch im Impulsraum mit zu betrachten. Die elektromagnetischen Felder müssen mit ebenso hoher Dimension abgebildet werden wie die Teilchen im Ortsraum besitzen, da eine weitere Ortskoordinate sonst keinen Sinn macht.

Das für die Integration der Bewegungsgleichungen verwendete Verfahren sollte zwei Anforderungen erfüllen:

1. es soll möglichst wenig Information pro Teilchen benötigen und
2. es sollte möglichst schnell ablaufen, und das bei einer noch akzeptablen Genauigkeit.

Das bedeutet aber auch, daß Verfahren höherer Ordnung wie Runge-Kutta ausscheiden. Diese kommen zwar mit einer größeren Schrittweite in der Zeit zurecht, benötigen aber dafür möglichst exakte Informationen über die in dieser Zeit wirkenden Kräfte (zusätzlicher Speicherplatz für Feldstärken) und über die vergangenen Zeitschritte. Was bei PIC-Rechnungen benötigt wird, ist ein Verfahren, das möglichst nur die Kräfte zu einem Zeitschritt benötigt und ohne Information aus früheren Zeitschritten auskommt.

4.4.2 Das Leapfrog-Verfahren

Kern des Leapfrog-Verfahrens ist ein Verschachteln der Rechenschritte: Es werden nicht in einem Rechenschritt Ort und Impuls der Teilchen neu berechnet, sondern es wird zunächst einmal nur der Impuls neu berechnet. Erst einen halben Zeitschritt später wird der Ort des Teilchens neu berechnet, einen weiteren halben Zeitschritt später wird der Impuls des Teilchens zum nächsten Zeitschritt berechnet.

Mathematisch bedeutet das Verwenden eines Leapfrog-Algorithmus nichts anderes, als daß man den neuen Ort nicht mit den alten Werten des Impulses berechnet, sondern schon die Werte des Impulses aus dem aktuellen Zeitschritt zum Bewegen der Teilchen verwendet. Die beiden grundsätzlichen Gleichungen, die pro Zeitschritt für jedes Teilchen gelöst werden müssen, sind in Gleichung (4.21) dargestellt. Werden diese Gleichungen durch finite Differenzen ersetzt und umgeschrieben, ergibt sich

$$\frac{\Delta \mathbf{p}^n}{\Delta t} = \mathbf{F}^{n-\frac{1}{2}} \quad \text{und} \tag{4.23}$$

$$\frac{\Delta \mathbf{x}^{n+\frac{1}{2}}}{\Delta t} = \frac{\mathbf{p}^n}{\gamma^n} \ . \tag{4.24}$$

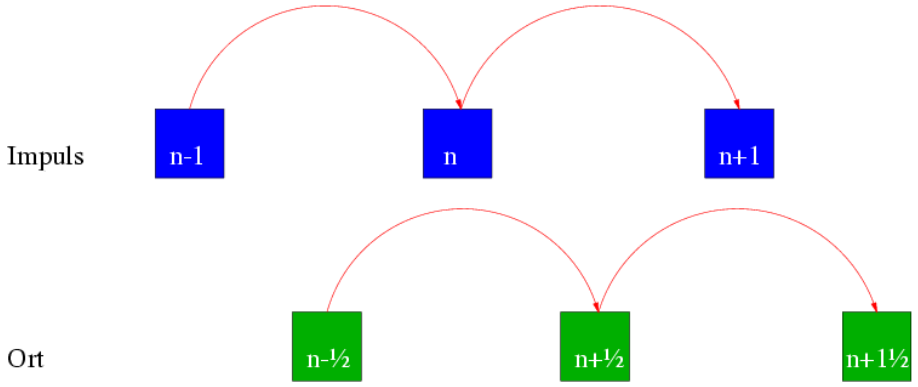

Abbildung 4.6: Der Leapfrog-Algorithmus des Teilchenmovers

Das bedeutet, daß der alte Impuls **p** zu dem Zeitpunkt, zu dem der neue Ort **x** berechnet wird, schon nicht mehr bekannt ist. Sinngemäß das gleiche gilt für die Geschwindigkeiten: Die Kraft auf das Teilchen wird an einem Ort bestimmt, der nicht mehr identisch ist mit dem Ort, an dem das Teilchen zu dem Zeitpunkt war, als die alte Geschwindigkeit berechnet wurde.

Allerdings laufen im Programm die Schritte in den Gleichungen (4.23) und (4.24) unmittelbar hintereinander ab, ohne daß in der Zwischenzeit der Maxwell-Gleichungslöser gerechnet wird. Der Grund dafür liegt einmal darin, daß die Neuberechnung des Ortes nur die ohnehin gerade berechneten Impulse benötigt und keine Felddaten, zum anderen läßt sich so das bei modernen Mikroprozessoren aufwendige Durchlaufen des Datenfeldes für die Teilchen auf einen einzigen Durchlauf pro Zeitschritt reduzieren.

Zeitliche Abfolge der Rechenschritte von Teilchenmover und Maxwellsolver

In den Abschnitten 6 und 4.4.2 wurden jeweils Verfahren vorgestellt, die Werte für Ort und elektrische Feldstärke zu halben Zeitschritten liefern, Werte für Impuls und magnetische Feldstärke aber entsprechend zu ganzen Zeitschritten. Zum Bewegen der Teilchen werden jedoch die magnetischen Felder zu halben Zeitschritten benötigt.

Um nun zu vermeiden, daß man die Werte zweier Zeitschritte speichern muß, um sie zu mitteln, verwendet man nicht die volle Schrittweite, sondern avanciert die magnetischen Felder nur zur Hälfte, bewegt dann die Teilchen und avanciert die magnetischen Felder anschließend nochmals zur Hälfte.

Das Bild 4.7 verdeutlicht dieses Verfahren. Die Integration der magnetischen Feldstärke wird doppelt so häufig ausgeführt wie die restlichen Integrationsschritte, aber das ist insbesondere in dem Fall, wo nur mit TM-polarisiertem Licht gerechnet wird, der Teilschritt mit dem geringsten Rechenaufwand.

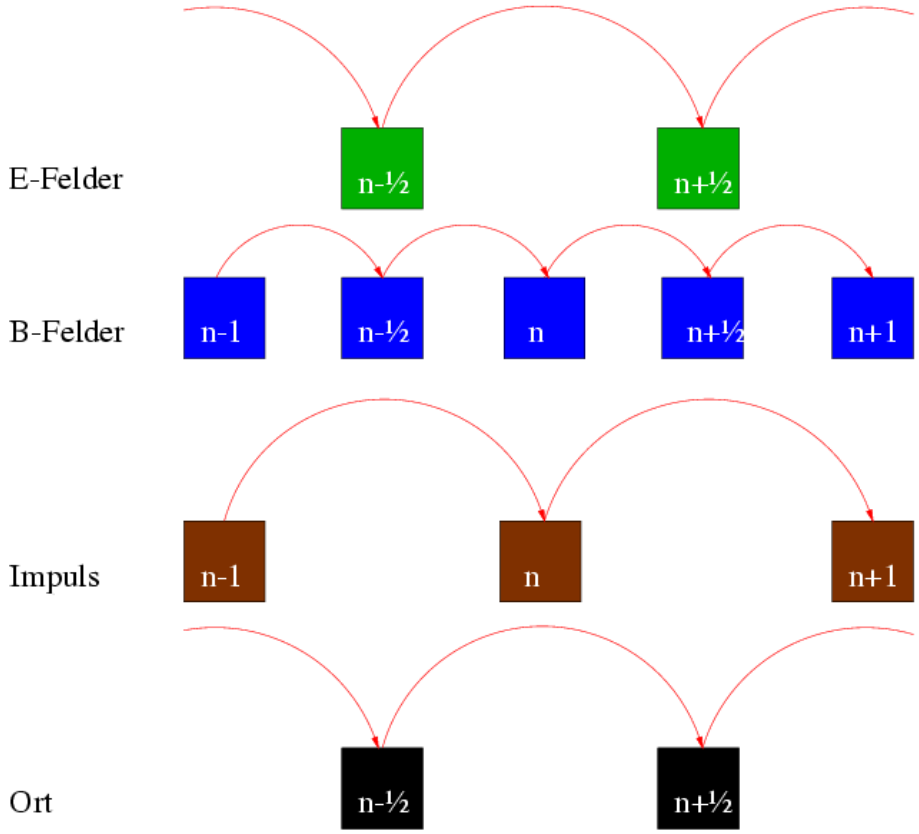

Abbildung 4.7: Der Leapfrog-Algorithmus für gleiche Schrittweite in Teilchenmover und Maxwell-Solver

Mehrfache Schrittweite im Teilchenmover

Da das Bewegen der Teilchen mit Abstand den rechenintensivsten Teil des PIC-Codes darstellt und zusätzlich nicht durch das für den Maxwell-Solver geltende Courant-Kriterium in der Schrittweite eingeschränkt ist, ergibt sich noch eine weitere Möglichkeit, die Rechengeschwindigkeit des Programms zu steigern.

Dazu wählt man die Schrittweite von Maxwell-Solver und Teilchenmover unterschiedlich, und zwar so, daß die Schrittweite im Teilchenmover eine ganzzahlige Vielfache der Schrittweite im Maxwell-Solver ist. Das bedeutet z.B. im Fall der doppelten Schrittweite im Teilchenmover, daß der Ort der Teilchen zu ungeraden Werten des Schrittzählers berechnet wird, während der Impuls zu geraden Werten des Schrittzählers berechnet wird. Beim Maxwellsolver rechnet man so wie bisher, allerdings kann man die zwei Halbschritte bei der Integration des magnetischen Feldes durch einen vollen Zeitschritt ersetzen, wenn keine Neuberechnung der Teilchenimpulse ansteht.

Das Bild 4.8 zeigt diese Zusammenhänge. Die Unterschiede sind bei den

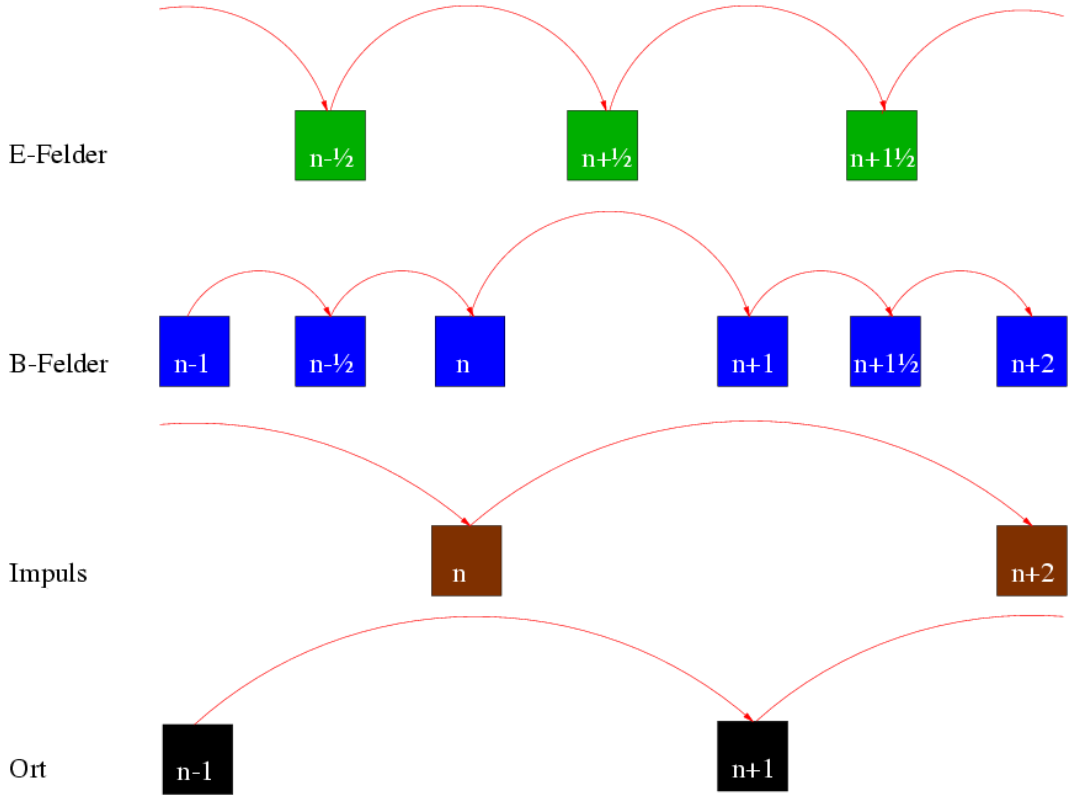

Abbildung 4.8: Der Leapfrog-Algorithmus für doppelte Schrittweite im Teilchenmover gegenüber dem Maxwell-Solver

üblicherweise verwendeten Schrittweiten (ca. 50 Schritte im Ort und 70 Schritte in der Zeit während einer Vollwelle des Lasers) vernachlässigbar gering, man gewinnt aber fast den Skalierungsfaktor in der Geschwindigkeit.

Falls man mehr als zwei Schritte des Maxwellsolvers zwischen einem Schritt des Teilchen-Movers hat, dann werden die Schritte des Maxwell-Solvers bei $n+\frac{1}{2}$ und $n+1$ so oft wiederholt, bis wieder ein Schritt des Teilchenmovers ansteht.

Berechnung des Stromes

Besondere Sorgfalt muß der Berechnung des Stromes $\mathbf{j}$ gelten, da dieser die Verbindung zwischen den beiden algorithmischen Teilen des PIC-Codes – Teilchen und elektromagnetischem Feld – bildet.

Der Beitrag eines Teilchens zum Strom errechnet sich zu

$$\mathbf{j}^n = \frac{q}{\gamma^n}\frac{\mathbf{p}^n}{mc}. \tag{4.25}$$

Dabei muß aber berücksichtigt werden, daß der Ort, an dem dieser Vektor lokalisiert ist, durch $\mathbf{x}^n$ gegeben ist. Allerdings wird durch das Leapfrog-Verfahren die-

ser Wert nie ausgerechnet: Es stehen nur Werte für $\mathbf{x}^{n-\frac{1}{2}}$ und $\mathbf{x}^{n+\frac{1}{2}}$ zur Verfügung. Man kann entweder durch lineare Interpolation den benötigten Wert nach der Formel

$$\mathbf{x}^n = \frac{\mathbf{x}^{n-\frac{1}{2}} + \mathbf{x}^{n+\frac{1}{2}}}{2} \tag{4.26}$$

erhalten, oder man bildet den Strom aus zwei Teilwerten, indem man die Hälfte des Beitrags *vor* dem Bewegen des Teilchens (also am Ort $\mathbf{x}^{n-\frac{1}{2}}$) und die zweite Hälfte des Beitrags *nach* dem Bewegen des Teilchens (also am Ort $\mathbf{x}^{n+\frac{1}{2}}$) zum Strom hinzuaddiert. Im Endergebnis hat man damit den Strom am Ort $\mathbf{x}^n$ zum Gesamtstrom hinzuaddiert.

Kräfte auf ein geladenes Teilchen in einem Plasma

Auf ein geladenes Teilchen in einem Plasma wirken in der Hauptsache nur die Kräfte, die aufgrund der elektrischen und magnetischen Felder auftreten. Falls es sich um ein vollionisiertes Plasma mit ausreichend großer kinetischer Energie und nicht zu hoher Dichte handelt (d.h., die mittlere kinetische Energie der Teilchen des Plasmas beträgt ein Vielfaches der mittleren potentiellen Energie), dann kann man Kräfte aufgrund von quantenmechanischen Effekten (Entartungsdruck, Spin-Spin-Wechselwirkung, etc.) getrost vernachlässigen. Die Berechnungen sind dann gültig für vollionisierte, nichtentartete Plasmen.

Daraus folgt, daß für die in der Berechnung benötigte Kraft auf ein Teilchen gilt:

$$\mathbf{F}^{n-\frac{1}{2}} = q \cdot \left(\mathbf{E}^{n-\frac{1}{2}} + \frac{1}{m\gamma} \left(\mathbf{p}^{n-\frac{1}{2}} \times \mathbf{B}^{n-\frac{1}{2}} \right) \right) \tag{4.27}$$

Die zu lösenden Bewegungsgleichung für den Impuls ist dann

$$\frac{\Delta \mathbf{p}^n}{\Delta t} = \frac{q}{m} \cdot \left(\mathbf{E}^{n-\frac{1}{2}} + \frac{1}{m\gamma} \left(\mathbf{p}^{n-\frac{1}{2}} \times \mathbf{B}^{n-\frac{1}{2}} \right) \right) \ . \tag{4.28}$$

Betrachtet man sich die Änderung des Impulses genauer, dann stellt man fest, daß es sich dabei um eine Verschiebung des Vektors in Richtung des elektrischen Feldes handelt, verbunden mit einer Drehung um die Achse, die durch die Richtung des Magnetfeldes bestimmt ist. Der Drehwinkel hängt dabei von dem Produkt aus der Ladung und der Beträge des Vektors des Impulses und des magnetischen Feldes ab, dividiert durch den relativistischen γ-Faktor.

Das bedeutet aber zweierlei: einmal läßt sich eine Drehung numerisch exakter berechnen, wenn man sie nicht als Addition eines einzigen kleinen, senkrecht auf dem ursprünglichen Vektor stehenden Vektors ausführt, sondern sie wirklich als eine Rotation des betreffenden Vektors implementiert. Zum anderen kann man die Verschiebung nicht von der Drehung komplett getrennt betrachten, sondern muß beide Operationen im Prizip *gleichzeitig* ausführen. Dies ist wichtig, weil

der auf der ursprünglichen Geschwindigkeit senkrecht stehende Teil, der für die Drehung addiert wird, auch von der Geschwindigkeit selbst abhängt.

4.4.3 Impulsänderungen durch elektromagnetische Felder

Buneman hat in [1] einen sehr effizienten Algorithmus vorgestellt, der bei der Beschränkung auf 2 Dimensionen im Impulsraum zu den Standard-Algorithmen bei der Integration von Teilchenbahnen bei PIC-Berechnungen gehört.

Boris' Algorithmus aus [2] nutzt die Struktur der in der elektromagnetischen Wechselwirkung auftauchenden Kräfte dafür aus, eine erhöhte Genauigkeit ohne Rückgriff auf zusätzliche Information zu erreichen.

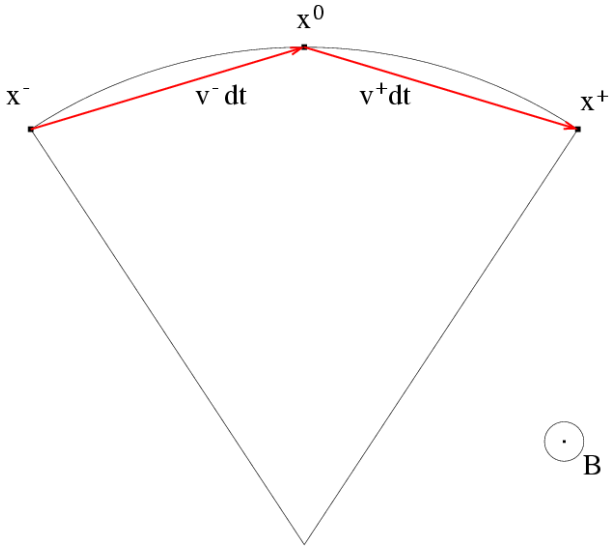

Abbildung 4.9: Die Gyration einer positiven Ladung im magnetischen Feld, aufgeteilt in zwei Teilschritte $v^- \frac{\Delta t}{2}$ und $v^+ \frac{\Delta t}{2}$.

Dazu wird die Berechnung der Impulsänderung in die auf dem elektrischen Feld basierende Änderung und die auf dem magnetischen Feld beruhende Änderung aufgeteilt.

Diese Aufteilung ist deswegen naheliegend, weil die aus dem elektrischen und dem magnetischen Feld stammenden Kräfte unterschiedliche Eigenschaften haben:

- Kräfte aus dem elektrischen Feld können eine beliebige Richtung relativ zum Impuls des Teilchens aufweisen
- Kräfte aus dem magnetischen Feld stehen immer senkrecht auf dem Impulsvektors des Teilchens und ändern nichts an der Energie des Teilchens, sondern bewirken eine Drehung des Impulsvektors

Kernstück ist folgendes Vorgehen: Die Beschleunigung aufgrund des elektrischen Feldes wird aufgeteilt in zwei Hälften. Die erste Hälfte wird zu Beginn des Neuberechnens des Impulses addierte:

$$\mathbf{p}^{-} = \mathbf{p}^{n-1} + \frac{q}{2m}\mathbf{E} \cdot \Delta t \tag{4.29}$$

Die eigentliche Drehung aufgrund des magnetischen Feldes wird nun in zwei Schritte aufgespalten. Der Grund ist folgender: Das magnetische Feld bewirkt eine Rotation des Impulsvektors. Infinitesimal wird eine Rotation dadurch beschrieben, daß ein betragsmäßig eben infinitesimal kleiner Vektor auf den Impuls addiert wird, der aber senkrecht auf diesem steht.

Das klappt aber bei Inkrementen mit endlicher Länge nicht mehr: Will man eine Drehung auf diese Art und Weise bewerkstelligen, so stellt man fest, daß der Impulsvektor langsam, aber kontinuierlich an Betrag zunimmt. Der Grund für dieses Verhalten: Der resultierende Vektor ist die Hypothenuse eines rechtwinkligen Dreiecks, korrekt wäre allerdings ein gleichseitiges Dreieck mit den beiden Impulsvektoren vor und nach dem Zeitschritt als gleichlange Schenkel, so wie es in Abbildung 4.9 ersichtlich ist. Das gleichseitige Dreieck wird hier durch die Punkte $\mathbf{x}^{-}$, den tiefsten Punkt der Figur und $\mathbf{x}^{+}$ definiert.

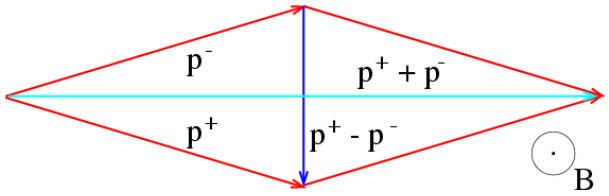

Abbildung 4.10: Die Drehung des Impulsvektors einer positiven Ladung im magnetischen Feld von Abbildung 4.9

Dabei ist der Impulsvektor vor der Drehung durch $\mathbf{p}^{-}$ gegeben, der Impulsvektor nach der Drehung durch $\mathbf{p}^{+}$. In Abbildung 4.10 erkennt man, daß der für die Rotation benötigte Vektor gegeben ist durch

$$\frac{\mathbf{p}^{+} - \mathbf{p}^{-}}{\Delta t} = \frac{q}{2\gamma m}(\mathbf{p}^{+} + \mathbf{p}^{-}) \times \mathbf{B}\,, \tag{4.30}$$

allerdings ist zu Zeitpunkt der Berechnung $\mathbf{p}^{+}$ noch nicht bekannt. Das Vorgehen ist daher wie folgt:

Das magnetische Feld bewirkt eine Drehung des Impulsvektors um einen Winkel θ. Angenommen, wir haben nur eine Komponente des magnetischen Feldes $B = B_z$, so ist der Tangens des halben Drehwinkels in der x-y-Ebene gegeben durch

$$\tan\frac{\theta}{2} = -\frac{qB}{\gamma m}\frac{\Delta t}{2}\,. \tag{4.31}$$

Indem man

$$t = -\tan\frac{\theta}{2} \tag{4.32}$$

setzt und

$$s = -\sin\theta = \frac{2t}{1+t^2} \tag{4.33}$$

$$c = \cos\theta = \frac{1-t^2}{1+t^2} \tag{4.34}$$

kann man die Werte für Sinus und Kosinus des Drehwinkels erhalten, ohne daß man trigonometrische Funktionen auswerten muß. Die Rotation in der Ebene läßt sich dann durchführen mit

$$p_x^+ = c\,p_x^- + s\,p_y^- \quad \text{und} \tag{4.35}$$

$$p_y^+ = -s\,p_x^- + c\,p_y^- \ . \tag{4.36}$$

Dabei müssen jedoch Sinus *und* Kosinus des Drehwinkels ausgerechnet werden. Schneller geht es, wenn man Bunemans Algorithmus aus [1] verwendet: Dieser macht einen Zwischenschritt bei der Berechnung, wodurch die Berechnung des Kosinus entfällt:

$$p'_x = p_x^- + p_y^- t \tag{4.37}$$

$$p_y^+ = p_y^- - p'_x s \tag{4.38}$$

$$p_x^+ = p'_x + p_y^+ t \tag{4.39}$$

Dies ist *keine* Näherung, sondern es wurden die beiden Identitäten

$$\cos\theta = 1 - \sin\theta\,\tan\frac{\theta}{2} \quad \text{und} \tag{4.40}$$

$$\sin\theta = (1+\cos\theta)\tan\frac{\theta}{2} \ . \tag{4.41}$$

verwendet.

Für eine Rotation durch einen beliebig orientierten Magnetfeldvektor bietet sich Boris' Algorithmus aus [2] an. Hier wird in einem ersten Schritt ein Vektor $\mathbf{t}$ berechnet:

$$\mathbf{t} = \frac{q}{2m}\frac{\mathbf{B}}{\gamma\mu_0}\Delta t \tag{4.42}$$

Mit diesem Vektor wird ein temporärer Vektor $\mathbf{p}'$ erzeugt:

$$\mathbf{p}' = \mathbf{p}^- + \mathbf{p}^- \times \mathbf{t} \tag{4.43}$$

Der Vektor $\mathbf{p}^- \times \mathbf{t}$ steht senkrecht auf dem Vektor $\mathbf{p}^-$ und wäre im Fall einer infinitesimalen Drehung gerade die Hälfte des Inkrementes, um die Drehung des Impulsvektors im Magnetfeld zu bewerkstelligen.

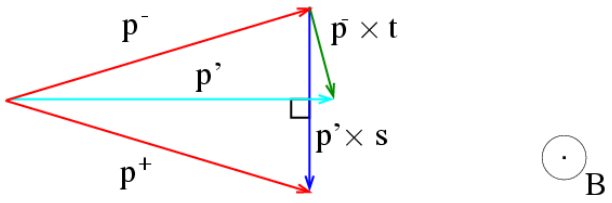

Abbildung 4.11: Die Drehung des Impulsvektors durch Boris' Algorithmus

Bildet man nun die Summe aus dem ursprünglichen Vektor vor der Drehung und dem halben Inkrement, so hat man den ursprünglichen Vektor um den halben Winkel gedreht. Zwar ist der Vektor $\mathbf{p}'$ etwas länger als der Vektor $\mathbf{p}^-$, aber das gesuchte Inkrement, um von $\mathbf{p}^-$ nach $\mathbf{p}^+$ zu gelangen. steht genau senkrecht auf dem Vektor $\mathbf{p}'$. Man kann also das Inkrement erzeugen, indem man das Kreuzprodukt dieses Vektors $\mathbf{p}'$ mit einem Vektor parallel zu $\mathbf{B}$ bildet, wie es aus der Abbildung 4.11 ersichtlich ist.

Der dazu benötigte Vektor $\mathbf{s}$ ist gegeben durch

$$\mathbf{s} = \frac{2\mathbf{t}}{1 - t^2} \,. \tag{4.44}$$

Mit Hilfe dieses Vektors wird ein dritter temporärer Wert für den Impuls errechnet, der die volle Drehung des Impulsvektors durch das magnetische Feld enthält.

$$\mathbf{p}^+ = \mathbf{p}^- + \mathbf{p}' \times \mathbf{s} \tag{4.45}$$

Zum Schluß wird noch die zweite Hälfte der Impulsänderung durch das elektrische Feld addiert[5]:

$$\mathbf{p}^n = \mathbf{p}^+ + \frac{q}{2m}\mathbf{E} \cdot \Delta t \tag{4.46}$$

4.5 Kopplung der elektrodynamischen und elektrostatischen Gleichungen

Bei der Kopplung von elektrodynamischen und elektrostatischen Gleichungen tritt insbesondere nach einer großen Anzahl Rechenschritten das Problem auf, daß die Poisson-Gleichung nicht mehr erfüllt ist, da sich beim Aufsummieren der Ströme kleine Fehler ergeben, die im Endeffekt ein Auseinanderlaufen der Werte von ρ und $\nabla \cdot \mathbf{E}$ bewirken.

Um nun zu verhindern, daß sich Fehler in der elektrischen Feldstärke ergeben, gibt es verschiedene mögliche Wege:

[5] Die Vektoren $\mathbf{s}$ und $\mathbf{t}$ können in denselben Variablen berechnet werden, da sie nur nacheinander, aber nicht gleichzeitig benötigt werden. Das gleiche gilt für die Vektoren $\mathbf{p^{n-1}}$, $\mathbf{p}^-$, $\mathbf{p}^+$ und $\mathbf{p^n}$.

1. Man berechnet die elektrostatischen Felder mittels des Potentials ϕ getrennt von den elektrodynamischen Feldern. In den Maxwellgleichungen muß man darauf achten, daß die Divergenz des elektromagnetischen Anteils des Feldes verschwindet (Aufteilung des elektrischen Feldes in einen longitudinalen und einen transversalen Anteil)

2. Man initialisiert die elektrostatischen Felder zu Anfang korrekt über das Potential ϕ. Um nun die Erfüllung der Poisson-Gleichung sicherzustellen, benutzt man die Kontinuitätsgleichung der Ladung zur Korrektur des Stromes

3. Man addiert eine Korrektur zur longitudinalen Komponente des elektromagnetischen Feldes. Diese Korrektur ist so bemessen, daß eine Differenz in der Poisson-Gleichung, die sich im Laufe der Zeit über Rechenungenauigkeiten bei der Bestimmung des Stromes ergibt, eleminiert wird

4.5.1 Getrennte Berechnung der longitudinalen und transversalen elektrischen Felder

Die Aufteilung des elektrischen Feldes nach **longitudinalen** und **transversalen** Anteilen geschieht über die Divergenz des Feldes: Die transversalen Anteile sind die Anteile des elektrischen Feldes, deren Divergenz verschwindet, während die Divergenz der longitudinalen Felder gleich der Ladungsdichte ist.

Da diese Aufteilung nur sehr aufwendig anhand der Gesamtfeldstärke zu bewältigen ist, führt man sie entweder so durch, daß man zur Berechnung der Maxwellgleichungen die Potentiale $\mathbf{A}$ und ρ in der Coulomb-Eichung benutzt, oder indem man die Maxwellgleichungen für die Felder $\mathbf{E}$ und $\mathbf{B}$ alleine mit der transversalen Komponente des elektrischen Stromes rechnet. Falls man das Vektorpotential benutzt, reduzieren sich die Maxwellgleichungen für die Strahlungsausbreitungen auf das Lösen *einer* vektoriellen Gleichung, dafür müssen *zwei* skalare Potentiale integriert werden. Außerdem müssen die Feldstärken für den Teilchenmover durch Differentiation aus den Potentialen gewonnen werden.

Für die Wellenausbreitung kann man dann aus den beiden Gleichungen (2.1) und (2.3) eine Gleichung für die zeitliche Änderung des Vektorpotentials $\mathbf{A}$ ableiten:

$$\frac{\partial \mathbf{A}}{\partial t} = c^2 \cdot \nabla^2 \mathbf{A} + \frac{1}{\epsilon_0} \mathbf{J}_T \tag{4.47}$$

$\mathbf{J}_T = \mathbf{J} - \nabla\eta$ ist hier der transversale Strom. Dieser wird gebildet, indem der longitudinale Anteil des Stromes abgezogen wird. Dazu dient das Potential η:

$$\nabla^2 \eta = \nabla \mathbf{J}_L = \nabla \left(\mathbf{J} - \mathbf{J}_T \right) \tag{4.48}$$

Durch diese Aufteilung des Stromes bleibt die Eichung des Vektorpotentials erhalten. Die für die Teilchenbewegung benötigten Felder erhält man aus den

Gleichungen

$$\mathbf{E} = -\nabla\phi - \epsilon_0 \frac{\partial \mathbf{A}}{\partial t} \qquad \text{und} \tag{4.49}$$

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{4.50}$$

Man muß bei diesem Verfahren die Poisson-Gleichung zu jedem Zeitschritt zweimal lösen: einmal für das elektrostatische Potential ϕ und einmal für das Korrekturpotential η.

Da der Poisson-Solver bei der Integration der Feldgleichungen den höchsten Aufwand darstellt, ist dies ein Nachteil des Verfahrens. Von Vorteil ist das Verfahren dann, wenn man etwa für andere Operationen das elektrostatische Potential auf jeden Fall berechnen muß.

4.5.2 Integration der Maxwellgleichungen unter Erhaltung der Poisson-Gleichung

Der im Absatz 4.5.1 vorgestellte Weg funktioniert mit allen Feldern, die sich aus einem Potential errechnen: Man kann mittels eines Korrekturpotentials eine bestimmte „Eichung“ des Potentials erhalten, indem man nämlich die Treiber bei der Integration der Gleichungen so wählt, daß die „Eichung“ nicht zerstört wird. Und die „Eichung“ des elektrischen Potentials steckt in der Poisson-Gleichung. Der Treiber bei der Integration des elektischen Feldes sind aus Gleichung (2.1) ersichtlich: Die Rotation des magnetischen Feldes und der Strom.

Während die Divergenz der Rotation des magnetischen Feldes stets verschwindet, ist die elektrische Stromdichte, wie sie aus dem Teilchenmover geliefert wird, nicht geeignet, die Divergenz des elektrischen Potentials für alle Zeiten zu erhalten.

Das bedeutet, daß die Korrektur über den Strom stattfinden muß. Zu diesem Zweck arbeiten wir auch wieder mit einem Korrekturpotential ξ, das diesmal der Gleichung

$$\nabla^2 \xi = \nabla \mathbf{J} + \frac{\partial \rho}{\partial t} \tag{4.51}$$

genügt. Der bei der Integration des elektromagnetischen Feldes verwendete Strom ist nun $\mathbf{J}' = \mathbf{J} - \nabla\xi$. Hierdurch bleibt bei der Zeitintegration der elektrischen Felder die Poisson-Gleichung weiterhin erhalten, sofern sie zu Beginn der Berechnung erfüllt war. Der Grund hierfür ist einfach: Die Zeitintegration und der Divergenzoperator vertauschen, dadurch macht es keinen Unterschied, ob man zuerst die Zeitintegration durchführt und dann die Divergenz bildet, oder ob man die Divergenz aufrechterhält und dann die Zeitintegration durchführt.

Was das eingeführte Potential bewirkt ist nun, daß die Änderung der Divergenz, die bei der Zeitintegration des elektrischen Feldes auftritt, exakt mit der Änderung der elektrischen Ladungsdichte übereinstimmt. Genau das ist aber die

Variation auf beiden Seiten der Poisson-Gleichung im Verlauf eines Zeitschrittes! Dadurch, daß die Variation auf beiden Seiten der Poisson-Gleichung über jeden Zeitschritt hinweg gleich bleibt, bleibt auch die Poisson-Gleichung als solche erhalten.

4.5.3 Direkte Korrektur der elektrischen Feldstärke zur Erhaltung der Poisson-Gleichung

Auch bei dieser Methode wird mit einem Korrekturterm gearbeitet, der jedoch die Felder selbst und nicht die felderzeugenden Ströme korrigiert. Die Korrektur entspricht der im letzten Kapitel, es werden allerdings nicht die Ableitungen der elektrischen Felder, sondern die Felder selbst korrigiert.

Hierzu berechnet man die Abweichung zwischen den beiden Seiten der Poisson-Gleichung, womit sich eine Defekt-Ladungsdichte ρ' ergibt:

$$\rho' = \rho - \epsilon_0 \nabla \mathbf{E} \tag{4.52}$$

Mit dieser Ladungsdichte löst man nun wiederum die Poisson-Gleichung für ein dazu passendes Potential und addiert dann die Ableitung dieses Potentials zu den Feldern hinzu:

$$\nabla^2 \phi' = -\frac{1}{\epsilon_0} \rho' \tag{4.53}$$

damit ergibt sich für das korrigierte elektrische Feld:

$$\mathbf{E}' = \mathbf{E} - \nabla \phi' \tag{4.54}$$

Dieses Feld erfüllt nun die Poisson-Gleichung.

Ein Vorteil des in Abschnitt 4.5.3 vorgestellten Verfahrens ist es, daß sich damit auch jederzeit nachträglich die longitudinalen Feldanteile korrigieren lassen. Das bedeutet: Wenn man den aus dem Teilchenmover gelieferten Strom so errechnet, daß die Kontinuitätsgleichung für die Ladung möglichst exakt eingehalten wird, dann genügt es, wenn nur bei jedem 10. oder 100. Zeitschritt eine Korrektur der longitudinalen Feldanteile stattfindet.

Dies ist insbesondere dann von Vorteil, wenn man den PIC-Code auf einer großen Anzahl von Prozessoren parallel rechnet: Von allen Algorithmen in einem typischen PIC-Programm ist das Lösen der Poisson-Gleichung mit Abstand am schlechtesten parallelisierbar (wegen der auftretenden schnellen Fouriertransformation) und erfordert zumindest einen hohen Kommunikationsaufwand. Können in der Zwischenzeit aber 100 Schritte gerechnet werden, dann kann man auch die Korrektur der longitudinalen Felder einem Knoten überlassen,der diese dann praktisch asynchron erledigt (wegen der Linearität der Maxwellgleichungen und der nur schwachen Korrektur ändert sich nichts wesentliches an den Ergebnissen,

wenn die Korrektur erst einige Rechenschritte später als eigentlich vorgesehen in die weiteren Berechnungen einfließt).

In der parallelisierten Version meines PIC-Codes wurde dieses Verfahren verwendet, um die Erhaltung der Poisson-Gleichung zu gewährleisten.

4.6 Parallelisierung des Codes

Vielteilchenrechnungen nach der PIC-Methode gehören zu dem am besten parallelisierbaren Algorithmen. Der Grund liegt in der Vernachlässigung der direkten Teilchen-Teilchen-Wechselwirkung und der Abbildung der elektromagnetischen Wechselwirkungen auf das Feldgitter.

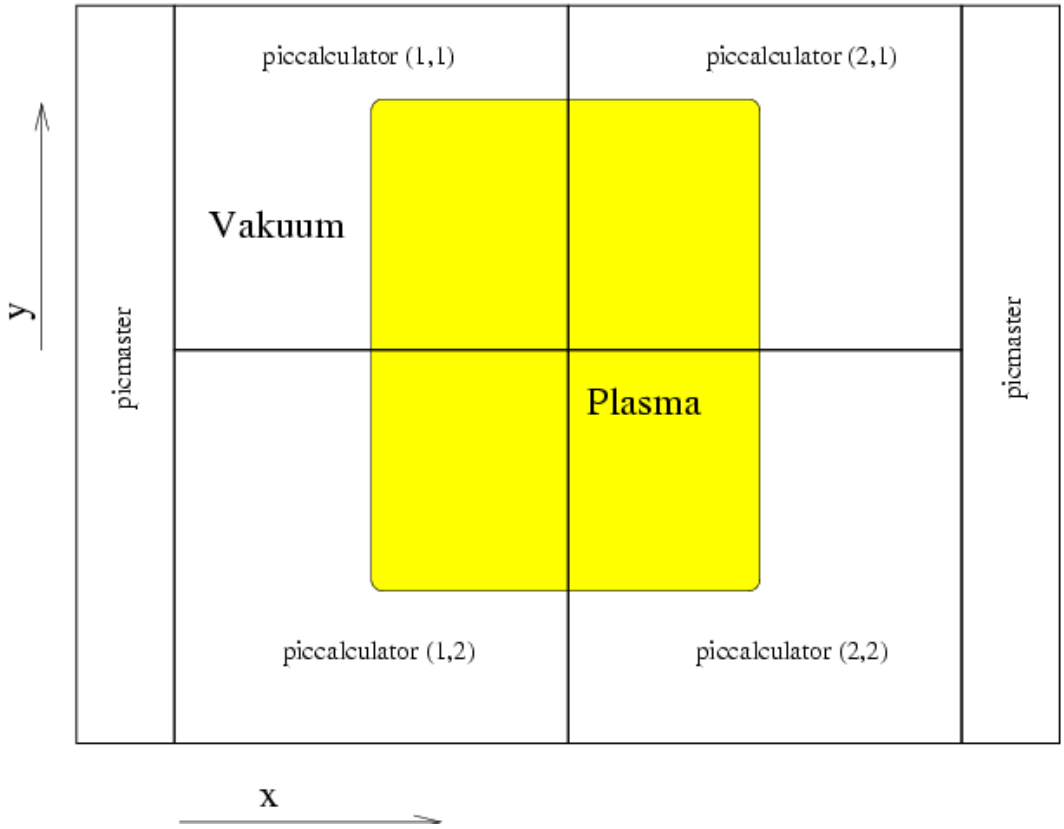

Abbildung 4.12: Aufteilung des berechneten Bereiches auf vier Prozesse

Um die Arbeit während der Berechnung auf mehrere Prozesse aufzuteilen, teilt man das Feld geometrisch in Abschnitte ein. Jeder Prozess erhält eine bestimmte Fläche zugewiesen, auf der er seine Berechnungen durchführt. Zu Ende jedes Rechenschrittes werden die Daten über Feldstärken oder Ströme an den Rändern der Bereiche ausgetauscht und Teilchen übertragen, die den Bereich des Prozesses

verlassen haben, womit auch eine Synchronisation der Prozesse sichergestellt ist. Die Aufteilung des zu berechnenden Bereiches in zwei Dimensionen zeigt das Bild 4.12. Dort, wo der zu berechnende Bereich in x-Richtung zuende ist, werden die Feldränder vom Master geliefert. Hierüber ist es möglich, Randbedingungen wie z.B. den Einfall des Lasers zu definieren.

4.6.1 Realisierung

Die parallelisierte Variante des PIC-Codes besteht aus drei verschiedenen Programmen:

1. Dem eigentlichen Master-Programm, das vom Anwender gestartet wird und seinerseits Kindprozesse startet und diese überwacht und steuert sowie die Ausgabe der Ergebnisse durchführt.

2. Dem Berechnungsprogramm, das vom Master einmal für jeden Berechnungsprozess gestartet wird und die eigentlichen Berechnungen durchführt.

3. Dem Poisson-Solver, der aus der Differenz zwischen Ladungsverteilung und der Divergenz des elektrischen Feldes ein Korrekturpotential bildet.

Die Kommunikation der Prozesse erfolgt mittels des an den Oak Ridge National Laboratories entwickelten Tool PVM. Hauptgründe für die Verwendung von PVM gegenüber dem ähnlich gelagerten Tool MPI war die etwas einfachere Handhabung und das fortgeschrittene Entwicklungsstadium von PVM; während MPI sich noch immer in der Entwicklungsphase befindet, liegt PVM bereits in der Version 3.4 vor.

4.6.2 PVM - Parallel Virtual Machine

Die Entwicklung von PVM wurde in den 80er Jahren initiiert, als klar wurde, daß sogenannte Mini- und Microcomputer innerhalb kurzer Zeit in Leistungsbereiche vorstoßen würden, die der damaligen Rechenleistung eines Großrechners entsprachen.

Ziel der Entwicklng war es, eine einfache Möglichkeit zu schaffen, verschiedene Rechner auch unterschiedlicher Architektur zur Zusammenarbeit zu nutzen, um so größere Rechenaufgaben bewältigen zu können.

Dabei sollte eine Programmierschnittstelle sowohl für vernetzte PC's als auch für parallele Großrechner zur Verfügung stehen. Das Ergebnis ist heute die Version 3.4 von PVM, welches unter den verschiedensten Betriebssystemen und Rechnerarchitekturen läuft und auch die vernetzte Programmierung auf Clustern unterschiedlichster Architektur ermöglicht.

Eine gute Einführung in PVM gibt [6], diese ist im Internet auch auf der PVM-Webseite [7] erreichbar und liegt vielen Linux-Distributionen bei.

Grundlegende Funktionsweise von PVM

PVM stellt die Kommunikationsfunktionen für sogenannte *Message–Passing–Architekturen* bereit. Das bedeutet, die Prozesse, die parallel arbeiten, besitzen jeder für sich einen eigenen Satz von Daten und können auch aus unterschiedlichem Programmcode heraus ablaufen, kommunizieren jedoch mittels von PVM bereitgestellten Funktionen und tauschen darüber Nachrichten über den Zustand des Programms aus.

Darüber hinaus kann PVM auch Gruppen von Prozessen definieren, denen Prozesse auch dynamisch beitreten und sie verlassen können. Nachrichten können an einzelne Prozesse gesendet werden, und es können Nachrichten an alle Mitglieder einer Gruppe versendet werden.

Die Kommunikation selbst kann entweder auf der Basis eines TCP/IP–Netzwerkes erfolgen oder aber über einen Bereich gemeinsam genutzten Speichers, falls die beteiligten Prozesse auf einer Maschine laufen. Dabei macht es für die Programmierung keinerlei Unterschied, PVM wählt von sich aus die entsprechende Methode der Nachrichtenübermittlung aus.

Das Sicherstellen der Synchronisation von Berechnungsschritten geschieht bei der Programmierung unter PVM in der Regel über ein blockierendes Warten auf eine bestimmte Anzahl Prozesse in einer Gruppe oder über das blockierende Warten auf eine bestimmte Nachricht. Wie immer bei der Programmierung von verteilten Anwendungen muß man sich dabei Gedanken darüber machen, wie man einen sogenannten „Deadlock“ verhindert, bei dem Prozesse sich durch das gegenseitige Warten auf Nachrichten blockieren.

PVM ist unter UNIX als ein Daemon[6] realisiert, der üblicherweise von einem speziellen PVM-Programm, der Konsole mit Namen *pvm* gestartet wird. Der Daemon kann jedoch auch unabhängig von der Konsole laufen. Die Konsole wiederum stellt Funktionen bereit, mit deren Hilfe sich *virtuelle Maschinen* definieren lassen. Diese bestehen dann aus einer Anzahl Rechnern, den *Hosts*. Alternativ zur Befehlszeilenversion der PVM-Konsole gibt es auch die unter dem X Windows System laufende Konsole *xpvm*, die eine grafische Oberfläche für PVM zur Verfügung stellt und z.B. auch grafisch die Kommunikation der Prozesse untereinander visualisieren kann.

4.6.3 Ansatz der Parallelisierung

Um die PIC-Berechnung zu parallelisieren, wurde das Master-Worker-Konzept verwendet. Dieses Konzept besteht aus einem Hauptprogramm, dem sogenannten Master, der einen oder mehrere Rechenprozesse, eben die Worker, startet. Dabei kann es sich bei den Workern um Prozesse desselben Typs handeln, oder es können

[6]Ein Daemon ist ein Programm, das im Hintergrund läuft und Dienste für andere Prozesse zur Verfügung stellt.

auch verschiedene Arten von Prozessen gemeinsam an einem Problem rechnen, wobei jede Art unterschiedliche Teilaufgaben wahrnimmt. In dieser Arbeit wurde der letztere Fall realisiert.

Das Masterprogramm mit dem Namen *picmaster* wird vom Anwender entweder direkt oder über die PVM-Konsole gestartet. Es verarbeitet die übergebenen Parameter und überprüft sie auf sinnvolle Werte. Danach wird die Aufteilung des zu berechnenden Bereiches unter den verschiedenen Prozessen bestimmt und die Berechnungsprozesse (*piccalculator*) werden gestartet.

Die Parameterübergabe an die Prozesse erfolgt ebenfalls über PVM. Zunächst einmal werden globale Daten wie die Schrittweiten in Ort und Zeit per Broadcast (eine Art Rundruf) an alle Berechnungsprozesse versendet, danach werden die individuellen Parameter wie die Gitterpositionen an jeden Berechnungsprozess separat gesendet. Danach wird noch eine Instanz des Poisson-Solvers (*piccorrector*) gestartet, dieser erhält eine Liste der Berechnungsprozesse, damit er mit diesen direkt ohne die Vermittelung durch den Master kommunizieren kann.

Wenn die Initialisierung durchgeführt ist, dann laufen alle Prozesse in einer Schleife. Während die Schleifen bei *picmaster* und *piccalculator* synchron laufen, hat *piccorrector* eine deutlich einfachere Hauptprogrammschleife.

Die Schleife bei Master und Rechenprozessen sieht so aus, daß eine bestimmte Anzahl Durchläufe der eigentlichen Rechenschleife erfolgt, bestehend aus einem halben Zeitschritt des Maxwellsolvers für die Magnetfelder, dem Bewegen der Teilchen, der zweiten Hälfte des Zeitschrittes für die Magnetfelder und einem Zeitschritt für die elektrischen Felder. In bestimmten Abschnitten werden dann vor dem Bewegen der Teilchen von den Rechenprozesse die Verteilungen gebildet, an den Master versendet und von diesem ausgegeben.

4.6.4 Prinzip der Kommunikation

Kommunikation beim Start

Alle Rechenprozesse melden sich nach der Initialisierung beim PVM-Daemon des Rechners an, lesen von diesem die Prozess-ID des Masters und melden sich bei der Gruppe „Running“ an. Der Master ist solange blockiert, bis alle Rechenprozesse Mitglieder dieser Gruppe sind. Dann werden globale Parameter an alle Rechenprozesse gesendet. Dies beinhaltet die Schrittweiten im Ort für die Berechnung und die davon abweichende Schrittweite für die Datenausgabe, die Schrittweite in der Zeit und einige Daten, die für die Berechnung wichtig sind.

Wenn diese globalen Parameter gesendet sind, sendet der Master die individuellen Daten zu jedem Rechenprozess. Zu diesen Daten gehören z.B. die Größe des von diesem Prozess zu bearbeitenden Bereichs der Daten, die Anzahl der Teilchen für diesen Prozess, die Prozess-ID's der Nachbarprozesse etc.

Hierauf beginnen die Rechenprozesse, ihre Datenbereiche zu initialisieren und das numerische Plasma entweder per Zufallsgenerator oder mit einem festen Git-

ter relativ zu den Stützstellen des Feldes aufzubauen. Dabei wird auch die Ladungsverteilung ermittelt.

Der Master startet auch einen weiteren Prozess, der später das Lösen der Poisson-Gleichung übernimmt (*piccorrector*). Dieser meldet sich nach dem Start beim Master an und erhält auch einen Satz der für ihn wichtigen Parameter, die u.a. das globale Layout der Rechenprozesse enhalten.

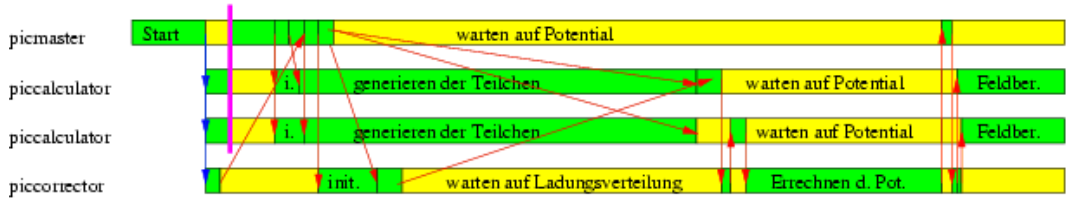

Abbildung 4.13: Start des Programmes mit 2 Rechenprozessen

Nachdem die Teilchen von den Rechenprozessen generiert wurden, erfolgt ggf. noch eine Berechnung des tatsächlichen Potentials durch den Poisson-Solver, bevor das erste Mal die Randbedingungen ausgetauscht werden. Dann beginnen die Rechenprozesse mit dem Bewegen der Teilchen.

Kommunikation nach Feldberechnung

Wenn eine Neuberechnung einer Feldkomponente durchgeführt wurde, dann werden mit den benachbarten Prozessen die Feldränder auf einer Breite von einem Gitterpunkt ausgetauscht. Steht nach der Neuberechnung der Feldkomponente noch das Bewegen der Teilchen an, so werden die Feldränder auf einer Breite von drei Gitterpunkten ausgetauscht.

Der Grund für den Austausch von drei Gitterpunkte: Ein Teilchen kann während eines Zeitschrittes des Teilchenmovers maximal n Gitterpunkte weit bewegt werden, falls der Teilchenmover mit der n-fachen Schrittweite des Maxwellsolvers arbeitet. Zwei Gitterpunkte sind nötig, da ein Teilchen immer auf zwei Gitterpunkte je Richtung wirkt (lineare Gewichtung). Nimmt man nun einen weiteren Gitterpunkt hinzu, so kann das Teilchen von einer Position innerhalb des berechneten Bereiches aus zwei Gitterpunkte weit bewegt werden, ohne daß es den Rand des berechneten Bereiches überschreitet. Wenn das Teilchen früh genug auf den Nachbarknoten transportiert wird, also bereits bei Annäherung an den Rand und nicht erst bei Überschreiten des Randes, dann kann ein Teilchen sogar bis zu vier Gitterpunkte zurücklegen, bevor es den gemappten Bereich des Feldes verläßt. Dies ist in der Regel ausreichend bis zu einer Skalierung von 1:4 zwischen den Schrittweiten im Maxwellsolver und im Teilchenmover.

Der Datenaustausch läuft so ab, daß der Prozess zunächst allen Nachbarn die eigenen Feldränder sendet, danach auf die Daten der Nachbarprozesse wartet. Ist

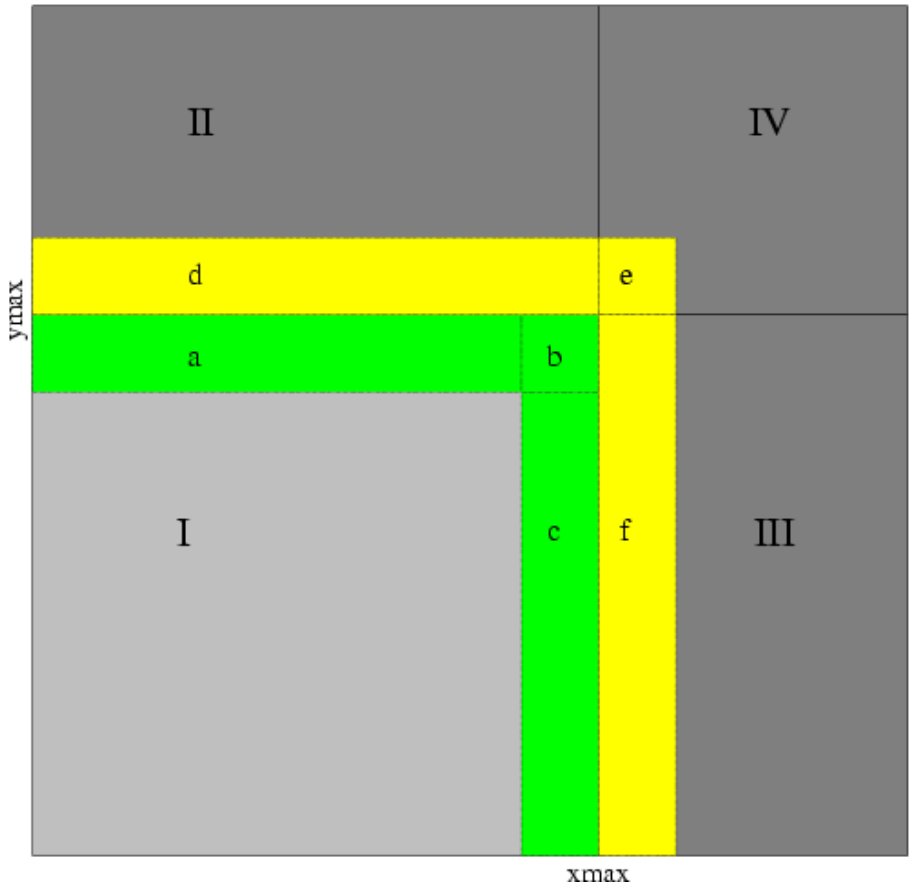

Abbildung 4.14: Rand des zu berechnenden Bereiches bei ($xmax$, $ymax$)

der Prozeß in x-Richtung am oberen oder unteren Rand, so sind drei der Nachbarknoten jeweils der Master, der sich aber auch komplett wie ein Rechenprozess verhält, also auch erst an die beteiligten Partner die Felder sendet und danach die Felder von den Partnern empfängt.

Über diese Schnittstelle mit dem Master speist dieser die Randbedingungen für die Felder in x-Richtung in die Berechnung ein.

Die periodische Randbedingung in y-Richtung wird dabei durch die Anordnung der Nachbarprozesse in y-Richtung bewirkt: An den letzten Rechenprozeß in y-Richtung grenzt in $+y$-Richtung wieder der erste Prozeß an, an den ersten Rechenprozeß in $-y$-Richtung grenzt der letzte Rechenprozeß an. Werden Teilchen über die Grenze zwischen y_{max} und y_{min} transferiert, so wird auf ihre y-Koordinate ein Offset addiert, der betragsmäßig gleich der Gesamtbreite des Feldes in y-Richtung ist und dafür sorgt, daß das übertragene Teilchen auch tatsächlich im Rechenbereich des Zielprozesses liegt.

Welche Bereiche bei der Kommunikation nach der Feldberechnung wohin

übertragen werden, veranschaulicht Bild 4.14. Mit **I** ist der eigene Rechenprozess bezeichnet, **II** ist der Rechenprozess in positive y-Richtung, **III** ist der Rechenprozess in positive x-Richtung und **IV** liegt der Ecke zu positiven x- und y-Werten gegenüber.

Jeder Prozess berechnet die Felder in seinem Bereich vollständig. Dieser Bereich umfaßt im Fall des Prozesses **I** den grauen Bereich mit dem grünen Rand. Der gelbe Rand wird von **I** für die Zeitentwicklung der Felder und für das Bewegen der Teilchen benötigt, wird aber auf den Nachbarknoten berechnet. Daher muß **I** den mit **a** bezeichneten Randbereich an **II** weitergeben (der auch den Bereich **b** einschließt !) und erhält von diesem im Gegenzug die Daten aus dessen Randbereich **d**. genauso wird mit den Daten aus dem Bereich **c** verfahren (auch der Bereich **b** gehört hier mit dazu). Diese werden an den Prozeß **III** weitergegeben, der dann die Daten aus seinem Bereich **f** zusendet. Damit die Daten an den Rändern stimmen, müssen noch an den Prozess **IV** die Daten aus dem Bereich **b** versandt werden, wobei wiederum **IV** Daten aus dem Bereich **e** sendet.

Kommunikation nach Teilchenbewegung

Falls eine Teilchenbewegung durchgeführt wurde, werden von den Prozessen die Teilchen aus ihrem Teilchenpool heraussortiert, die sich an den Rändern des berechneten Bereiches befinden und sich aus den Bereich bewegen wollen. Diese werden für jede Richtung in einer Liste gespeichert und zum Abschluß des Bewegens der Teilchen werden alle diese selektierten Teilchen zu den entsprechenden Nachbarknoten transferiert. Falls dabei ein Teilchen in y-Richtung über den Rand zwischen erstem und letztem Prozess wechselt, wird die Position des Teilchens so korriegiert, daß es wieder in einem gültigen Bereich liegt (periodische Randbedingung).

Gleichzeitig mit den Elektronen werden auch die Ströme an den Rändern zu den Nachbarknoten hin verschickt. Dies ist notwendig, da die Feldintegration in diesen Randbereichen auf den Nachbarknoten erfolgt und nicht auf dem eigenen Knoten.

Die nötigen Bereiche erkennt man auf Bild 4.14. **I** hat von den eigenen Teilchen auch Ströme in den Bereichen **d**, **e** und **f**. Diese müssen daher an die Knoten **II**, **III** und **IV** versendet werden, um dort zu deren Strömen hinzuaddiert werden zu können. Von den Nachbarknoten kommen umgekehrt Werte für die Ströme in den Bereichen **a**, **b** und **c**, wobei hier der Bereich **b** wieder Daten von drei Nachbarknoten beinhaltet.

Auch Ionen können versendet werden, allerdings ist hier wegen der wesentlich langsameren Bewegung mit einem deutlich geringeren Teilchenaufkommen zu rechnen, entsprechend kleiner sind die Datenfelder für den Transfer.

Ist einer der benachbarten Prozesse der Master, dann werden keine Daten dorthin versendet und die Teilchen werden an der Grenze reflektiert, d.h., ihre Impulskomponente in x-Richtung wird invertiert.

Da während des Bewegens der Teilchen auch die einzelnen Energieanteile des Rechenprozesses zur Gesamtenergie berechnet werden, wird zum Abschluß noch eine Statistik von jedem Rechenprozess an den Master gesendet. Diese Statistik enthält neben der Zahl transferierter Teilchen noch die einzelnen Energiebeiträge für diesen Prozess. Im einzelnen wird unterschieden nach:

- kinetischer Energie der Elektronen
- kinetischer Energie der Ionen
- Energie des elektrischen Feldes
- Energie des magnetischen Feldes
- ein- oder ausgestrahlte Energie an jeder Kante

Kommunikation für die Korrektur von $\nabla \cdot \mathbf{E}$

Einmal während eines vollständigen Rechenschrittes (Teilchenmover und entsprechende Schritte der Feldintegration) sendet der Master an alle Rechenprozesse und an den Poisson-Solver eine Statusvariable, deren Wert bestimmt, ob eine Korrektur der Gröse $\nabla \cdot \mathbf{E}$ durchgeführt werden soll.

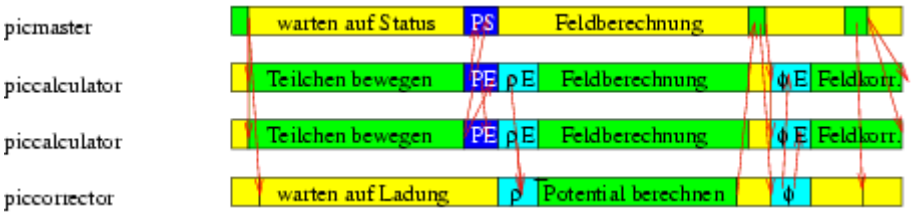

Abbildung 4.15: Schema der Kommunikation für die Korrektur von $\nabla \cdot \mathbf{E}$

Die Abbildung 4.15 zeigt das Schema der Kommunikation für diese Korrektur. Falls die Korrektur erfolgen soll, so generieren die Rechenprozesse während des Bewegens der Teilchen die notwendige Verteilung der Ladungsdichte. Nach dem Bewegen der Teilchen wird durch Berechnen der Gleichung (4.52) die Abweichung beider Seiten der Poisson-Gleichung voneinander ermittelt. Das Kürzel PS steht in Abbildung 4.15 für den Teilchenstatus, den der Master an dieser Stelle der Berechnung erwartet, während das Kürzel PE für den Teilchenaustausch der Rechenprozesse untereinander steht.

Der Poisson-Solver gibt nun den Rechenprozessen nacheinander die Anweisung, ihre Abweichung zu übermitteln. Diese senden ihm die Größe ρ', danach berechnet der Solver das Korrekturpotential, während die restlichen Prozesse weiterrechnen. Erst zu Beginn des nächsten Rechenschrittes testet der Master, ob

der Poisson-Solver das Potential berechnet hat. Ist dies der Fall, so wird den Rechenprozessen die Anweisung gegeben, auf das Potential zu warten, während der Solver-Prozess die Anweisung erhält, seinerseits das Potential zu senden.

Haben die Rechenprozesse das Potential erhalten, so korrigieren sie ihre Felder, bevor sie ganz normal mit dem Bewegen der Teilchen fortfahren.

Kommunikation zur Übertragung der Daten

Da Master und Rechenprozesse einen gleichartigen Zähler führen, wissen die Rechenprozesse, wann der Master die Verteilungen zum Abspeichern der Werte haben möchte. Sie warten dann auf ein Signal des Master-Prozesses, bevor sie beginnen, die von ihnen gebildeten Verteilungen zu versenden. Dies hat den Sinn, einen zu großen Bedarf an Pufferspeicher seitens des PVM-Daemons zu verhindern.

Wurden alle Daten übertragen, so beginnt der betreffende Rechenprozeß bereits wieder mit dem Bewegen der Teilchen. Hier ließe sich noch etwas Zeit gewinnen, wenn man das Warten auf das Datenausgabesignal des Masters nicht blockierend macht, sondern während des Bewegens der Teilchen die Warteschlange des Prozesses abfragt und ggf. in die Routine zum Senden der Verteilungsfunktion hineinspringt. Während die Rechenprozesse die Teilchen bewegen, gibt der Master die Verteilungen aus.

Da die Datenausgabe in Form von `ASCII`-Tabellen erfolgt, entsteht hier doch ein erheblicher Rechenzeitbedarf, da je nach Umfang der Simulation Datenmengen bis zu einigen Megabyte von der internen Fließkommadarstellung des Rechners in das Textformat gewandelt werden. Hier läßt sich ggf. noch Rechenzeit und Speicherbedarf sparen, wenn die Ausgabe nicht in Textform, sondern binär erfolgt. Allerdings sind die Daten dann nicht mehr mit einem einfachen Textbetrachter lesbar, sondern müssen mit einem geeigneten Programm visualisiert werden.

4.6.5 Behandlung eines Programmabbruchs

Bei der parallelisierten Variante des PIC-Codes ist das Master-Programm in der Regel das einzige der Programme, das eine Standard-Ein/Ausgabe in dem Sinn hat, daß es mit dem Anwender kommunizieren kann. Es kann Ausgaben über den Fortgang der Berechnungen liefern, und es kann auch Eingaben des Anwenders entgegennehmen. Das bedeutet aber auch, daß ein Programmabbruch durch die Tastenkombination `CTRL-C` nur den Master betrifft: alle von ihm gestarteten Prozesse empfangen dieses Signal nicht und warten von da an vergeblich auf weitere Daten vom Master. Erst wenn diese Prozesse ein explizites `kill` erhalten, werden sie beendet. Dies kann insbesondere beim Rechnen auf einer Anzahl Maschinen lästig sein, da auf jeder Maschine mittels `killall` die betreffenden Prozesse gestoppt werden müssen.

Um den Programmabbruch des Masters etwas komfortabler zu handhaben, wurde in diesem Programm ein sogenannter Signal-Handler implementiert, der eine globale Variable setzt. Dieser Signal-Handler fängt die Signale `TERM` und `INT` ab. Ersteres wird bei einem Aufruf von `kill` erzeugt, letzteres bei einem Betätigen der Tastenkombination `CTRL-C` im Fenster des Prozesses.

In der Rechenschleife des Hauptprogramms wird dann das Kommando zum Beenden der Rechenprozesse gesendet. Dazu wurde der Befehl benutzt, der das Signal für die Korrektur von $\nabla \cdot \mathbf{E}$ gibt. Auf diese Art erfahren auch alle Prozesse vom Programmabbruch und können sich geordnet beenden.

Dieser Mechanismus funktioniert allerdings *nicht*, wenn der Master über das Signal `KILL` beendet wird, da der Signalhandler des Programms dieses Signal nicht abfangen kann und der Master sofort beendet wird, ohne daß von Seiten des Programms noch irgendwelche Eingriffsmöglichkeiten bestehen.

4.7 Ausgabe der Daten

4.7.1 Energiebilanz

Die Energiebilanz einer Berechnung stellt eine sehr nützliche Informationsquelle zur Beurteilung des Ergebnisses dar. Praktisch jeder kleinste Fehler in der Simulation hat auch Auswirkung auf die Energieerhaltung. Die Energiebilanz wird zu jedem Zeitschritt des Teilchenmovers aufgestellt, wobei der Master die Beiträge aller Rechenprozesse addiert.

Die Einheit, in der die einzelnen Energiebeiträge ausgegeben werden, ist wegen der verwendeten Einheiten gegeben durch

$$[E] = n_{crit} \cdot (2\pi\lambda)^3 \cdot 511 \text{ keV} . \tag{4.55}$$

Um während einer Berechnung die Daten leicht einsehen zu können, wird ein Skript für das Utility `gnuplot` erstellt, das die Energiebilanz plottet und sich nach 30s Pause wieder aufruft. Wird dieses Skript innerhalb von `gnuplot` mit dem `load`-Befehl aufgerufen, so hat man im Grafikfenster stets den aktuellen Verlauf der Energie.

Teilchenenergie

Die Teilchenenergie wird getrennt für Elektronen und Ionen berechnet. Dies ist sinnvoll, weil die Elektronen wesentlich direkter auf die elektrischen und magnetischen Felder reagieren, die Ionen dagegen länger brauchen, bis sie auf eine höhere Energiedichte in den Feldern reagieren.

Zur Berechnung der kinetischen Energie eines Teilchens wird nicht die naheliegende Formel

$$E_{kin} = mc^2(\gamma - 1) \tag{4.56}$$

verwendet, weil diese bei nichtrelativistischen Teilchen wegen des Terms $\gamma - 1$ sehr fehlerträchtig ist (γ ist in diesem Fall fast gleich 1!). Stattdessen wird der Ausdruck

$$E_{kin} = mc^2 \frac{\mathbf{q}^2}{1+\gamma} \tag{4.57}$$

verwendet. Da hier keine Differenz zwischen zwei fast gleichgroßen Werten auftaucht, ist diese Form der kinetischen Energie unempfindlich gegenüber Rundungsfehlern. Da das Impulsquadrat und der relativistische γ-Faktor auch zum Lösen der Bewegungsgleichung benötigt werden, fällt nur wenig zusätzliche Rechenarbeit zur Ermittelung der Teilchenergie an.

Für ein einzelnes Teilchen bedeutet dies, daß sich seine Teilchenenergie in der unter (4.55) beschriebenen Einheit nach der Formel

$$E_i = \frac{q_{x,i}^2 + q_{y,i}^2 + q_{z,i}^2}{1+\gamma_i} \cdot dx \cdot dy \cdot n \tag{4.58}$$

berechnet (i ist hier der Teilchenindex, dx und dy sind die Abmessungen des Teilchens, n beschreibt die Dichte der physikalischen Partikel im PIC-Teilchen relativ zur kritischen Dichte n_{crit}). Die Gesamtenergie einer Teilchensorte ist dann

$$E_{kin} = \sum_{i=1}^{N} E_i \tag{4.59}$$

Der Anteil der potentiellen Energie der Teilchen an der Gesamtenergie wird nicht separat ermittelt; wegen der Berechnung der elektrischen Feldstärken als Gesamtfeldstärke (longitudinale + transversale Feldstärke) ist dieser Anteil bereits in der Feldenergie enthalten.

4.7.2 Energiebilanz des Maxwellsolvers

Der Maxwellsolver berechnet ebenfalls bei jedem Rechenschritt eine Bilanz der durch die Grenzen des berechneten Bereichs eingestrahlten Energie. Hierzu wird der Poynting-Strom durch die Grenze aufintegriert, gegeben durch

$$E_S = \frac{\epsilon_0}{c} \int d\mathbf{A}\, dt\, \mathbf{E} \times \mathbf{B} \; . \tag{4.60}$$

Da die Grenzen bei einem Programm in 2d nur eindimensional sind, beschränkt sich die Integration der Energie entlang der Grenze auf ein Aufsummieren des Kreuzproduktes aus elektrischem und magnetischem Feldes entlang des Randes. Multipliziert mit der Schrittweite in der Zeit wird dieser Wert dann bei jedem Rechenschritt des Maxwellsolvers zum gesamten Energiestrom durch diese Grenzfläche aufsummiert. In Formeln gefaßt bedeutet dies, daß sich der Energiestrom pro Zeitschritt Δt durch eine Grenze in y-Richtung berechnet aus

$$E_s = \frac{\epsilon_0}{c} \sum_{i=n_x}^{N_x} (E_{y\,i,j} \cdot B_{z\,i,j} - E_{z\,i,j} \cdot B_{y\,i,j}) \cdot dy \; , \tag{4.61}$$

durch eine Grenze in x-Richtung berechnet sich die Energie zu

$$E_s = \frac{\epsilon_0}{c} \sum_{j=n_y}^{N_y} (E_{z\,i,j} \cdot B_{x\,i,j} - E_{x\,i,j} \cdot B_{z\,i,j}) \cdot dx \ . \tag{4.62}$$

Ein positives Vorzeichen des Energiestroms bedeutet dabei einen Energiefluß in positive Richtung.

Die Gesamtenergie des elektromagnetischen Feldes wird nach den Formeln

$$E_{el} = \frac{1}{2}\,\epsilon_0 \int d^3V \mathbf{E}^2 \tag{4.63}$$

$$E_{mag} = \frac{1}{2}\,\mu_0 \int d^3V \mathbf{B}^2 \tag{4.64}$$

berechnet. Dies bedeutet eine Summation über das gesamte Feldgitter und ergibt in der diskretisierten Form

$$E_{el} = \frac{1}{2}\,\epsilon_0 \sum_{i=n_x}^{N_x} \sum_{j=n_y}^{N_y} \left(E_{x\,i,j}^2 + E_{y\,i,j}^2 + E_{z\,i,j}^2\right) \cdot dx \cdot dy \ \text{ und} \tag{4.65}$$

$$E_{mag} = \frac{1}{2}\,\mu_0 \sum_{i=n_x}^{N_x} \sum_{j=n_y}^{N_y} \left(B_{x\,i,j}^2 + B_{y\,i,j}^2 + B_{z\,i,j}^2\right) \cdot dx \cdot dy \ . \tag{4.66}$$

4.7.3 Ausgabe der Flächendiagramme

Die Ausgabe der Daten des PIC-Programmes erfolgte in Form von ASCII-Tabellen und dazu passenden Skriptdateien für das Datenvisualisierungsprogramm IDL der Firma RSI. Dazu werden in bestimmten Abständen in Zeilen und Spalten aufgeteilte Tabellen der betreffenden Verteilung geschrieben. Diese Dateien tragen alle die Endung `.DAT`, enthalten einen Teil ihres Namens durch die Art der Verteilung und tragen eine fortlaufende Nummer.

Zu jeder der Verteilungen gibt es eine Skriptdatei, die nach dem Ausgeben einer Datendatei um die nötigen Anweisungen ergänzt wird, damit IDL diese Datendatei in einem Diagramm darstellen kann. Durch ein Leerschreiben der Puffer wird dafür gesorgt, daß schon während der Berechnung ein Darstellen der Daten möglich ist. Die Skriptdatei trägt die Endung `.idl`.

Die Weiterverarbeitung der Daten erfolgte über einen Export nach PostScript, so daß die entstehenden PostScript-Dateien auf dem Rechner mit einem entsprechenden Programm (in der Regel `ghostview` oder ähnliche) angesehen und auch ausgedruckt werden konnten. Dieser Export wurde bewältigt, indem die Ausgabe im Skript statt auf den Bildschirm in eine Datei umgelenkt wurde.

Anhang A

Parameter zu den gezeigten Diagrammen

Dieses Kapitel wurde wegen der Dateigröße in dieser Version der Arbeit ausgespart. Bei Interesse müssen Sie sich die komplette Arbeit herunterladen (ca. 3 MBytes).

Anhang B

Arbeiten mit dem parallelisierten PIC-Code

In diesem Kapitel des Anhangs soll beschrieben werden, wie mit der parallelisierten Variante meines PIC-Codes gearbeite werden kann und was dabei zu beachten ist.

B.1 Angepaßte Einheiten

Die in dieser Arbeit verwedete Version der Software arbeitet so, daß die Berechnungen und die Datenausgabe in sogenannten „angepaßten Einheiten“ erfolgt. Dazu werden einige relevante Naturkonstanten und Größen gleich 1 gesetzt, womit sich greifbarere Zahlenwerte als mit den SI-Eineheiten ergeben. Die betreffenden Konstanten sind:

c Vakuum-Lichtgeschwindigkeit

ω_l Laser-Kreisfrequenz

m_e Elektronenmasse

e Elementarladung

Damit ergeben sich folgende Skalierungen zwischen den Ausgabewerten des Programms und der Daten in SI-Einheiten:

- **Impulse** sind auf mc normiert[1]
- **Dichten** sind auf die kritische Dichte n_c normiert

[1]Einzige Ausnahme ist die Skalenteilung bei den Ionendichten auf den Impulsachsen. Hier ist der Skalierungsfaktor gegeben durch $\sqrt{m_e m_i c^2}$. Dadurch ist die thermische Geschwindigkeit v_{th} in den Diagrammen für Elektronen und Ionen gleich „groß“.

- **elektrische Feldstärken** sind auf den Faktor $m_e c\,\omega_l/e$ normiert
- **magnetische Feldstärken** sind auf den Faktor $m_e\omega_l/e$ normiert
- **Temperaturen** werden in keV angegeben
- **Stromdichten** sind auf $e/\omega_l c^2$ normiert
- **Energien** sind auf den Faktor $m_e c^2$ normiert

B.2 Vorbedingungen für das Starten des Programms

Da die parallelisierte Version des PIC-Codes die Bibliothek `PVM` zur Kommunikation verwendet, muß zunächst einmal `PVM` auf dem Rechner installiert sein und der `PVM`-Daemon muß laufen. Unter `Unix` erreicht man dies z.B., indem man das Programm `pvm` startet und die „virtuelle Maschine“ damit konfiguriert. Näheres dazu ist [6] zu entnehmen. Falls das `PVM`-Subsystem nicht arbeitet, stellt das Programm dies fest und beendet sich.

B.3 Arbeiten mit dem Programm picmaster

Die parallelisierte Version wird vom Anwender wie ein einfaches Berechnungsprogramm mit dem Befehl `picmaster` (bzw, `picmaster23d` im Fall der Version mit 3 Impulskoordinaten) gestartet. Über Kommandozeilenparameter wird dabei konfiguriert, was das Programm als Startbedingungen annehmen soll, wie der Laser wirkt und auch, mit welcher geometrischen Aufteilung die Berechnungen ablaufen sollen.

B.3.1 Parameter des Programms

Alle Parameter erwartet das Programm in der Form **Kürzel**:*Wert*. **Kürzel** ist dabei das Schlüsselwort des Parameters, z.B. **xc** für die Anzahl der Gitterpunkte in x-Richtung. Der Doppelpunkt dient dabei zur Abtrennung von Befehl und Parameter. Die meisten Parameter erwarten eine numerische Angabe, einige wenige erwarten einen Zahlenbereich in der Form `[min:max]`. Die meisten Paramter werden auf sinnvolle Werte überprüft und ggf. korrigiert.

Hinter dem Parameter wird die Art des Parameters angegeben, und zwar in der Form

n für einen ganzzahligen Wert

f.f für eine Fließkommazahl

txt für eine Zeichenkette

Nachfolgend sind die Parameter in alphabetischer Reihenfolge aufgelistet. Falls man das Programm mit dem Zeichen „?“ als einzigen Paramter startet, erhält man eine Auflistung der Parameter mit einer kurzen Beschreibung.

a : f.f Definition eines Dichteprofils

Der Parameter **a** stellt den Koeffizienten für ein exponentiell ansteigendes oder abfallendes Dichteprofil dar. Die Dichte verhält sich danach wie

$$n(x) = n_0 \, e^{a(x - x_{start})} \; . \tag{B.1}$$

Hierbei ist n_0 durch den Parameter m vorgegeben und x_{start} stellt die Anfangsposition des Plasmas dar. Vorgabe ist ein Wert von 0 (homogene Dichte des Plasmas).

Wurde ein direktes Mapping der Teilchen auf das Gitter gewählt, dann wird über den Parameter a definiert, wie stark das Dichteprofil an der Vorderkante des Plasmas ansteigt.

cs : n Rechenschritte zwischen Korrektur von $\nabla \mathbf{E}$

Für die Korrektur der Beziehung $\nabla \mathbf{E} = \varrho / \epsilon_0$ kann man hier die Anzahl von Rechenschritten vor einer Korrektur angeben.

d : f.f Pulsdauer des Lasers

Dies ist die Einstellmöglichkeit für die Pulsdauer des Lasers. Simuliert wird ein Puls mit der angegebenen Dauer, der mit einem gaußförmigen Anstieg und Abfall versehen ist.

dm : n Direktes Mapping von Teilchen

Dieser Parameter dient dazu, Teilchen direkt auf die Gitterpunkte des Feldgitters zu mappen. Der Parameter gibt an, wieviele Teilchen in jede Richtung auf das Gitter gemappt werden. Ein Wert von 2 erzeugt damit 4 Teilchen pro Gitterzelle.

dx : f.f Schrittweite in x-Richtung

Hiermit wird die Schrittweite in x-Richtung festgelegt. Der Zahlenwert muß größer als 0 sein, Vorgabe ist ein Wert von 0.2.

dy: f.f Schrittweite in y-Richtung

Dieser Parameter legt die Schrittweite in y-Richtung fest. Es gilt sinngemäß das gleiche wie für die Schrittweite in x-Richtung.

dt : f.f Schrittweite in der Zeit

Der Parameter dt legt die Schrittweite in der Zeit fest. Hierbei ist als Randbedingung das Courant-Kriterium für den Maxwell-Solver zu beachten:

$$dt < \frac{dxdy}{dx + dy} \tag{B.2}$$

e : [f.f:f.f] Darstellungsbereich für Energieverteilung

Dieser Parameter legt den Darstellungsbereich für die Verteilungen der Energiedichten gegen den Ort fest. Die Zahlenwerte sind Exponenten, d.h., eine Angabe von -4 bis 2 bedeutet einen Darstellungsbereich in der Energiedichte von 10^-4keV bis 10^2keV.

f : txt Dateiname für Datenausgabe

Dieser Parameter legt den konstanten Teil des Dateinamens für alle Ausgabedateien fest. Daraus ermittelt das Programm alle Dateinamen für die Ausgabedateien.

i : f.f Intensität des Lasers

Hiermit kann die Intensität des Laserlichtes an der Grenzschicht zum berechnete Bereich bestimmt werden. Die Angabe erfolgt in W/cm^2.

l : 1/0 Fokussierung des Lasers

Dieser Parameter dient als Schalter, um eine Fokussierung des Laserstrahls im berechneten Bereich oder ein senkrechtes Einstrahlen ohne Fokussierung auszuwählen.

m : f.f Elektronendichte der Partikel

Dieser Parameter legt die Elektronendichte in einem simulierten Teilchen in Einheiten der kritischen Dichte fest. Vorgabe ist hier ein Wert von $0.1n_{crit}$. Die tatsächliche Dichte ergibt sich dann aus der mittleren Anzahl von Teilchen pro Gitterzelle multipliziert mit der Elektronendichte eines PIC-Teilchens.

ms : n Multiple Schrittweite im Teilchenmover

Dieser Parameter legt fest, wieviele Schritte des Maxwellsolvers zwischen zwei Schritten des Teilchenmovers liegen. Sinnvoll sind Werte kleiner als 10.

ne : n Anzahl der Elektronen

Dieser Parameter legt die Anzahl der Elektronen fest. Falls gleichzeitig ein direktes Mapping von Teilchen auf das Gitter festgelegt ist, wird dieser Parameter ignoriert.

ni: n Anzahl der Ionen

Dieser Parameter legt die Anzahl der Ionen fest. Falls gleichzeitig ein direktes Mapping von Teilchen auf das Gitter festgelegt ist, dann wird dieser Parameter nur in dem Fall beachtet, wenn er 0 ist.

nx : n Anzahl der Rechenprozesse in x-Richtung

Hiermit wird die Anzahl der Rechenprozesse in x-Richtung festgelegt. Vorgabewert ist 2. Dieser Wert muß zwischen 1 und 64 liegen.

ny : n Anzahl der Rechenprozesse in y-Richtung

Legt die Anzahl der Rechenprozesse in y-Richtung fest. Vorgabewert ist 2, erlaubter Bereich ist von 1 bis 16.

o : f.f Laser-Kreisfrequenz

Hiermit wird die Laser-Kreisfrequenz ω_l in Einheiten von 1/s angegeben.

oc : n Anzahl der Ausgabezellen

Hier werden die Anzahl der Gitterzellen für die Ausgabe der Verteilungsfunktionen in den Impulsrichtungen und in der Energierichtung festgelegt.

os : n Skalierung der Ausgabe

Dieser Parameter legt die Skalierung zwischen Rechengitter und Ausgabegitter für x- und y-Richtung fest.

osx : n Skalierung der Ausgabe in x-Richtung

Dieser Parameter legt die Skalierung zwischen Rechengitter und Ausgabegitter für die x-Richtung fest.

osy : n Skalierung der Ausgabe in y-Richtung

Dieser Parameter legt die Skalierung zwischen Rechengitter und Ausgabegitter für die y-Richtung fest.

px : [f.f:f.f] Bereich für x-Impuls

Hiermit wird der Bereich für die Impulskomponente p_x bei der Ausgabe der Daten festgelegt.

py : [f.f:f.f] Bereich für y-Impuls

Dieser Paramter legt den Bereich für die Ausgabe der Impulskomponenten p_y fest.

pz : [f.f:f.f] Bereich für z-Impuls

Dieser Parameter legt den Bereich für die Ausgabe der Impulskomponenten p_z fest

s : f.f Intervall für Datenspeicherung

Dieser Parameter legt das Intervall für die Datenspeicherung fest. Die Angabe erfolgt in Einheiten von $1/\omega$.

sl : f.f Steilheit der Impulsflanke

Dieser Parameter bestimmt die Steilheit des gaußförmigen Ein- und Auslaufs des Laserpulses. Je kleiner der Wert ist, umso steiler sind die Pulsflanken.

t : f.f Endzeitpunkt der Berechnung

Hiermit wird der Endzeitpunkt festgelegt, bis zu dem die Berechnung erfolgt. Da immer nur in Vielfachen der Speicherzeit s gerechnet wird, kann das tatsächliche Ende der Berechnung auch deutlich später liegen.

te : f.f Temperatur des Plasmas

Hiermit wird die Temperatur des simulierten Plasmas eingestellt. Dieser Wert gilt sowohl für Ionen als auch für Elektronen. Für die Simulation einer Temperatur wird eine Maxwellverteilung der Geschwindigkeiten angenommen. Erreicht wird dies über die Vorbelegung der Geschwindigkeitswerte der Teilchen mit einer gaußförmigen Verteilung. Der Zahlenwert der Temperatur bestimmt dabei die Streubreite der Verteilung. Die Temperatur wird in **keV** angegeben.

w : f.f Breite des Laserpulses

Dieser Parameter legt die geometrische Breite des Laserpulses fest. Angegeben wird dabei der Abstand der Halbwertsbreite des Pulses von der Mitte des simulierten Bereichs in Vielfachen von $1/k_l$.

x : [f.f:f.f] **x-Bereich des Plasmas**

Hiermit werden die Abmessungen der plasmagefüllten Box in x-Richtung spezifiziert. Wenn einder der Grenzen die der Simulationsbox überschreitet, wird sie vom Programm entsprechend korrigiert.

xc: n **Anzahl Zellen in x-Richtung**

Hiermit wird die Anzahl der Gitterpunkte für das Feld in x-Richtung festgelegt.

y : [f.f:f.f] **y-Bereich des Plasmas**

Dieser Parameter legt die y-Abmessungen der plasmagefüllten Box fest. Falls der Bereich größer ist als die tatsächliche Simulationsbox, wird er entsprechend eingeschränkt.

yc: n **Anzahl Zellen in y-Richtung**

Dieser Parameter legt die Anzahl der Gitterpunkte in y-Richtung fest.

B.4 Hinweise zum Einsatz des Programms

In diesem Kapitel sollen einige Hinweise zu einem sinnvollen und nützlichen Einsatz des Programms gegeben werden. Dies gilt sowohl für die Vorgabe der Berechnungsparameter als auch für den Resourcenverbrauch des Programms.

B.4.1 Festlegung der Schrittweiten

Bei der Festlegung der Schrittweiten muß man oft einen Kompromiß zwischen der Rechengenauigkeit des Programms einerseits und dem Speicherverbrauch und dem Rechenzeitbedarf andererseits eingehen. Sinnvoll ist es daher, vorab Überlegungen über einen sinnvollen Bereich der Schrittweiten anzustellen. Dabei ist es entscheidend, ob man sich im Bereich unter- oder überkritischen Plasmas befindet, weil dies die minimal aufzulösende Frequenz bestimmt.

Berechnungen bei $\omega_p < \omega_l$

Im Bereich des unterkritischen Plasmas ist die Frequenz ω_l des Lasers die höchste Frequenz, die in der Zeit noch sinnvoll aufgelöst werden muß. Im Ort ist es die Laserwellenlänge λ_l, die die Auflösung bestimmen sollte. Entsprechend ist es sinnvoll, die Schrittweiten in Ort und Zeit so zu wählen, daß eine Vollwelle des Lasers auf 20 bis 50 Gitterpunkte abgebildet wird.

Die Bewegung der Elektronen wird hingegen von zwei Größenordnungen beeinflußt: dies ist zum einen die Laserfrequenz, die auch bei der Teilchenbewegung

noch aufgelöst werden muß, um z.B. den Brechungsindex des Plasmas realistisch simulieren zu können, andererseits laufen alle kollektiven Phänomene auf der Zeitskala der Plasmafrequenz. Aus diesem Grund ist es durchaus möglich, beim Bewegen der Teilchen eine schlechtere Auflösung der Laserfrequenz zu haben und die Schrittweite hier hochzuskalieren.

Sinnvolle Bereiche der Parameter sind:

$$\begin{aligned} dx &= 0.1...0.5 \\ dy &= 0.1...0.5 \\ dt &= 0.07...0.35 \\ ms &= 1...4 \end{aligned}$$

Der Skalierungsfaktor in der Zeit zwischen Teilchenschrittweite und Maxwellschrittweite sollte umso kleiner sein, je größer die Schrittweite in Zeit und Ort ist, so daß im Endeffekt eine Schrittweite von ca. 0.2 – 0.35 in der Zeit für die Teilchenbewegung resultiert.

Berechnung bei $\omega_p > \omega_l$

Bei Simulationen im Bereich überdichten Plasmas bestimmen die Plasmafrequenz ω_p und die Debye-Länge λ_D die Größenordnung des Geschehens. Mehrfache Schrittweite im Teilchenmover hat hier bei Testrechnungen meist eine große Instabilität der Berechnung mit sich gebracht, da die Teilchen im Bereich relativistischer Geschwindigkeiten Schwingungsmoden treiben können, die sich selbst verstärken. Entsprechend sollte man sehen, daß sowohl in Ort als auch in der Zeit die Plasmafrequenz gut aufgelöst wird, im Ort sollte man zusätzlich beachten, daß die Debyelänge aufgelöst werden kann.

Sinnvolle Bereiche der Parameter sind:

$$\begin{aligned} dx &= 0.1...0.5/k_p \\ dy &= 0.1...0.5/k_p \\ dt &= 0.07...0.35/\omega_p \\ ms &= 1 \text{ oder } 2 \end{aligned}$$

Die Temperatur sollte so hoch gewählt sein, daß sich eine Debyelänge über zwei bis fünf Gitterzellen erstreckt.

B.4.2 Verteilung der Rechenlast auf verschiedene Rechner

Um die Prozesse definiert auf verschiedenen Rechnern einer `pvm`-Maschine starten zu können, kann man eine Textdatei namens `hosts` verwenden, die vom Master-Programm beim Start gelesen wird. In jeder Zeile dieser Datei sollte ein

Computer der virtuellen Maschine stehen. Dann werden auf dieser Maschine alle Prozesse eines Abschnitts in x-Richtung gestartet (Dies hat den Sinn, daß die Komunikation mit anderen Maschinen nur notwendig ist, wenn Grenzen in x-Richtung betroffen sind).

Hat man Maschinen unterschiedlichen Leistungsvermögens, so kann man die schnelleren Maschinen auch häufiger in dieser Datei aufführen. Anzumerken ist aber, daß zu große Unterschiede in der Rechenleistung nur ausgeglichen werden können, wenn der zu berechnende Bereich in sehr kleine Kacheln zerlegt wird; ansonsten führt der Einsatz leistungsschwacher Rechner in einem Cluster sehr schnell zu einem sehr effektiven Ausbremsen der schnelleren Rechner, da für jeden Rechenschritt auf die Ergebnisse des langsamsten Rechners gewartet werden muß.

Fehlt die Datei `hosts`, so entscheidet `pvm` darüber, auf welchem Rechner der virtuellen Maschine der nächste Prozeß gestartet wird.

B.5 Anmerkungen zum Quellcode und der Compilierung

Das Programm besteht aus verschiedenen Quelltexten und Kopfdateien, die alle in C++ abgefaßt wurden. Hauptgrund für die Verwendung von C++ war die leichte Portierbarkeit von einer Rechnerarchitektur zu einer anderen, die insbesondere auf UNIX-Sytemen durch den frei erhältlichen GNU-Compiler und dessen Weiterentwicklung EGCS gegeben ist.

Ein weiterer Grund waren meine langjährigen Programmierkenntnisse in dieser Sprache sowie die kostenlose Verfügbarkeit leistungsfähiger Compiler für einfache PCs. C++ ist darüberhinaus meiner Ansicht nach eine sehr viel modernere und flexiblere Sprache als Fortran.

Alle Kommentare des Quelltextes wurden in englischer Sprache verfaßt, gleiches gilt auch für die wenigen Textausgaben der Programme. Lediglich die Diagrammüberschriften für die Skripte generiert das Programm in deutscher Sprache.

B.5.1 Hinweise zu den Klassen

Jedes der Programme besitzt eine zentrale Klasse, die in einer einzigen Instanz erzeugt wird. Der Sinn und Zweck dieser Klasse ist es, das prozedurale Schema beim Start von C++-Programmen in ein Schema abzubilden, das im Bereich des objektorientierten Programmierung üblich ist - nämlich Initialisierungen in einen Konstruktor zu packen, den Programmablauf einer `Run()`-Methode zu überlassen und alle Aufräumarbeiten danach von einem Destruktor erledigen zu lassen.

Zu diesem Zweck gibt es nur einen Konstruktor, der die Parameter der `main()`-Routine übergeben bekommt, mit der ein jedes C- und C++-Programm startet. Hier erfolgen die Initialisierungen wie der Start der Tochterprozesse, Belegen von

Speicher und Öffnen von Dateien. Aus diesem Grund wird auch die Instanz des Programmobjekts in der `main()`-Routine angelegt, und zwar ein einziges Exemplar davon. Da das Objekt so angelegt ist, daß es nicht übermäßig groß ist, wird es sinnvollerweise als `auto`-Variable auf dem Stack der Funktion angelegt.

Die eigentliche Arbeit übernimmt dann die Methode `Run()` der Klasse. Kehrt das Programm aus dieser Methode zurück, dann wird deren Rückgabewert als Rückgabewert der `main()`-Funktion des Programms an das Betriebssystem weitergeleitet.

Der Destruktor wird beim Verlassen der `main()`-Routine automatisch aufgerufen, wenn der Gültigkeitsbereich des Objektes verlassen wird.

Array- und Vektor-Klassen

Es wurde zwei Klassen `vector` und `array` geschrieben, die für die Speicherung von ein- und zweidimensionalen Datenfelder verwendet wurden. Für diese Klassen wurde das sogenannte `template`[2]-Konzept verwendet, wie es auch in [12] beschrieben ist. Das Konzept basiert darauf, daß für die Deklaration der Klasse kein fester Datentyp angegeben wird, sondern ein generischer Typ `<T>`. Erst dann, wenn eine Instanz der Klasse angelegt wird, wird der Typ in spitzen Klammern angegeben: `array<double>` ist z.B. die Array-Klasse mit Elementen vom Typ `double`.

Die Klassen besitzen überladene Operatoren `[]` und `()`, so daß die Feldzugriffe entweder nach C-Art mit einer eckigen Klammer pro Index abgewickelt werden können oder wie bei Pascal mit einer runden Klammer, die alle Indizes einschließt. Letztere Methode ist dabei sinnvoller, da bei der `array`-Klasse sonst zwei Operatoren, nämlich der der Klasse `array` und der der Klasse `vector` nacheinander aufgerufen werden müssen.

Der `operator ()` der Klasse `array` arbeitet im übrigen mit der Datenausrichtung von Pascal und C (letzter Index ändert sich am schnellsten) und nicht mit der von Fortran (erster Index ändert sich am schnellsten). Anfangs- und Endindex in jeder Dimension können entweder beim Erzeugen der Daten oder später festgelegt werden. Wurde das Makro `DEBUG` bei der Compilierung definiert, dann werden die Operatoren für die Datenzugriffe mit einer Indexüberprüfung compiliert. Wird ein Zugriffsversuch auf einen ungültigen Index festgestellt, so löst dies eine Exception[3] aus (siehe auch [12]). In dem Exception-Element werden Informationen zu dem fehlgeschlagenen Zugriffsversuch abgelegt.

Insbesondere die optionale Indexüberprüfung erleichtet die Fehlersuche sehr. Läuft das Programm dann, wird die Definition des `DEBUG`-Makros wieder ent-

[2]Der Begriff stammt aus dem englischen Sprachraum und läßt sich im hier verwendeten Zusammenhang wohl am besten mit *Vorlage* übersetzen.

[3]Eine Ausnahme (Exception) in C++ führt zu einem Abbruch der gerade ausgeführten Anweisung. Das Programm verzweigt zu einem Programmteil, das die Ausnahme behandelt (sofern programmiert) oder beendet sich.

fernt und das Programm neu compiliert. Damit läuft das Programm mit voller Geschwindigkeit.

B.5.2 Aufteilung der Quelltexte

Prinzipiell wurde der Quelltext so aufgeteilt, daß jedes der drei Programmteile einen Quelltext zugeteilt bekam. Ausnahme sind kleinere Module, die von mehreren Programmen benötigt werden, z.B. das Modul `names.cpp`, das die Gruppenamen für die Prozessgruppen definiert. Globale Kopfdateien wurden in einem separaten `include`-Verzeichnis untergebracht.

Das Programm picmaster

Der Quelltext für dieses Programm ist die Datei `picmaster.cpp`. Hierin wird als Kopfdatei `picmaster.hpp` eingebunden, das die Klasse `PICMaster` deklariert.

Das Programm piccalculator

Der Quelltext für dieses Programm ist die Datei `piccalculator.cpp`, die Deklaration der Laufzeitklasse ist die Datei `piccalculator.hpp`.

Das Programm piccorrector

Der Quelltext für dieses Programm ist die Datei `piccorrector.cpp`, die Deklaration für die Laufzeitklasse befindet sich in der Datei `piccorrector.hpp`.

B.5.3 Compilierung der Programme

Ein an Linux angepaßtes `makefile` ist im Quellcodeverzeichnis enthalten. Es wurde nicht das mit `PVM` verfügbare Skript `aimk` verwendet, das ein plattformabhängiges `make` unterstützt, da ich mit den Programmen nur unter Linux gearbeitet habe. Das `makefile` führt dazu, daß im Quellcodeverzeichnis die Objektdateien und auch die ausführbaren Dateien angelegt werden.

Mittels zweier Skripte werden die ausführbaren Dateien dann unter dem Verzeichnis `~/pvm3/bin/LINUX` abgelegt, wo der `PVM`-Daemon nach den ausfürbaren Dateien für den `pvm_spawn`-Befehl sucht.

Dabei existieren zwei Skripte zum Kopieren der Programmdateien, je nachdem, ob die 2d- oder die $2\frac{1}{2}$d-Version des Programms installiert werden soll. In der Version mit der z-Komponente des Impulses startet das Masterprogramm die Programme `piccalculator23d` und `piccorrector23d`, weswegen hier andere Skripte nötig sind. Andernfalls wäre es nur möglich, eine Version (2d oder $2\frac{1}{2}$d) installiert zu haben. Die Skripte heißen `instprgs` und `instprgs23d`.

Bei der Compilierung treten einige Warnmeldung des Compilers bezüglich der Konvertierung von `double` oder `float` nach `int` auf, diese können aber ignoriert werden.

Für vier verschiedene Grundeinstellungen befinden sich Einstellungen für die Compiler-Flags im Makefile; die jeweils nicht passenden Einstellungen müssen dabei auskommentiert werden. Die Einstellungen sind: je eine Debug-Einstellung mit und ohne z-Komponente des Impulses und je eine Release-Einstellung mit und ohne z-Komponente des Impulses. Die Unterschiedung zwischen dem Programm mit und ohne z-Komponente des Impulses wird über die Definition des Makros `MOMENT3D` durchgeführt.

Dies geschiet über die Compileroption `-DMOMENT3D` im `makefile`. Der Vorteil einer solchen Umschaltung liegt darin, daß ein Quellcode für beide Versionen genügt. Da 95% des Quellcodes in beiden Versionen ohnehin identisch ist, spart dies sehr viel Arbeit, wenn es darum geht, die Versionen konsistent zu halten.

B.6 Mögliche Weiterentwicklungen des Codes

B.6.1 Erweiterung des Codes auf 3D

Der Code ist im Moment auf Berechnungen in 2D ausgelegt. Allerdings wäre der Aufwand einer Portierung in 3D nicht extrem hoch, da lediglich eine neue Klasse für 3D-Matrizen benötigt würde und die Zugriffe auf die 2D-Felder durch solche auf 3D-Felder geändert werden müßten.

Änderung an den Datenstrukturen

Die Datenstrukturen für die Teilchen müßten dahingehend geändert werden, daß eine zusätzliche Komponente für den Ort in z-Richtung aufgenommen wird. Die Datenstrukturen für die elektrischen und magnetischen Felder können bereits mit allen Komponenten verwendet werden, falls das Programm in der Version für $2\frac{1}{2}$ Dimensionen compiliert wird, auch sind alle Komponenten des Stromes dann schon vorhanden.

Was fehlt, ist eine Datenstruktur für die Speicherung von 3-dimensionalen Feldern. Diese sollte sinnvollerweise in der Art realisiert werden, wie die Datenstruktur für die Speicherung und den Zugriff auf 2-dimensionale Felder realisiert ist, mit einem zentralen Speicherplatz für die Daten und einem überladenen `operator ()` für den Zugriff auf ein einzelnes Element. Der `operator []` sollte hingegen nicht überladen sein, denn es macht keinen Sinn, den Zugriff auf ein einzelnes Feldelement in der Art `a[i][j][k]` zu bewerkstelligen, da hier nacheinander *drei* Operatoren aufgerufen werden statt eines einzigen bei einem Zugriff über `a(i,j,k)`.

Änderungen an den Algorithmen

Da im Programm bereits ein 3D-Teilchenmover vorhanden ist, müßte lediglich der Maxwellsolver noch um die fehlenden Ableitungen der Feldkomponenten in z-Richtung ergänzt werden. Beim Teilchenmover genügt es, die Änderung des Ortes durch die p_z-Komponente zu berücksichtigen.

Die gravierendsten Änderungen betreffen die Kommunikation der Rechenprozesse untereinander und den Poisson-Solver. Die Kommunikation der Prozesse untereinander muß nun im Fall der Übertragung der Felddaten nicht nur 1-dimensionale Datenfelder versenden, sondern 2-dimensionale, was einen deutlich höheren Kommunikationsaufwand bedeuetet.

Sinnvoll ist es allerdings, nach wie vor nur eine 2-dimensionale Aufteilung des berechneten Bereiches durchzuführen, da ansonsten die Zahl der Nachbarbereiche eines Rechenprozesses noch einmal um die Hälfte ansteigt, die der Randbereiche sich gar verdreifacht und sogar acht neue Eckbereiche entstehen (die es in der 2d-Topologie gar nicht gibt).

In z-Richtung ist ebenso wie bereits in y-Richtung eine periodische Randbedingung sinnvoll, wobei entweder die Teilchenkoordinate korrigiert oder aber ein periodisches Mapping auf das Rechengitter durchgeführt werden müßte.

Änderung bei der Auswertung

Sinnvoll wäre es bei der Auswertung, die Bildung der Verteilungsfunktionen nicht dem Berechnungsprogramm zu überlassen, sondern einem Postprozessor, da die Datenausgabe in 3D sonst zu aufwendig würde. Immerhin würde dann aus jedem Flächendiagramm ein Volumendiagramm, oder man müßte die Anzahl der auszugebenden Verteilungsfunktionen extrem in die Höhe treiben.

Für einen Postprozessor würde es genügen, jede 10., 100. oder 1000. Teilchenbahn auszugeben und aus diesen Daten dann die Verteilungsfunktionen zu bilden. Hierbei verliert man zwar etwas Information, aber bei mehreren zehn- oder hundert Millionen Teilchen ist bei einer passablen Auswahl der Teilchen noch genügend Information vorhanden, so daß der Informationsverlust durch das Weglassen eines Teiles der Teilchen nicht den Informationsverlust durch das Bilden der Verteilungsfunktion überwiegt.

Abschätzung zum Speicher- und Rechenzeitverbrauch in 3D

Der größte Unterschied bei Rechnungen in 3D gegenüber 2D ist heute noch der Ressourcenverbrauch. Bei Simulationen in 2D mit 1024 Gitterpunkten in x-Richtung und 256 Gitterpunkten in y-Richtung genügt ein handelsüblicher PC, um einige 1.000 Rechenschritte pro Tag durchzuführen, das Programm paßt noch mitsamt allen Daten in den Hauptspeicher des Rechners.

Nimmt man jedoch eine dritte Dimension im Ort hinzu, so multiplizieren sich Speicher- und Rechenzeitverbrauch grob gesagt mit der Anzahl der Gitterpunke

in dieser Richtung: 256 zusätzliche Gitterpunkte in z-Richtung würden also das 256-fache an Speicher- und Rechenzeitverbrauch bedeuten. Auf absehbare Zeit ist nicht damit zu rechnen, daß Einzelplatzrechner diese Anforderungen erfüllen werden, sinnvoll ist also nur der Einsatz des Programms auf einem Großrechner oder einem Cluster in der Größenordnung von etwa 32 Einzelplatzrechnern aufwärts. Da das Programm bereits mittels PVM parallelisiert ist, wäre zumindest die Aufteilung der Rechenarbeit keine schwere Hürde mehr.

B.6.2 Verwenden anderer Algorithmen für das Lösen der Maxwellgleichungen

Das Programm ist im Moment durch den verwendeten Maxwellsolver recht eingeschränkt, was die Geschwindigkeit angeht. Grund ist das Courant-Kriterium für die Schrittweite des Maxwellsolvers in der Zeit. Hier könnte die Verwendung eines anderen Algorithmus' für den Maxwellsolver Abhilfe schaffen, insbesondere bei Berechnungen im Bereich überdichten Plasmas.

Eine Alternative wäre ein Charakteristiken-Verfahren, bei dem die longitudinalen und transversalen Felder getrennt berechnet werden. Strahlung wird hierbei entlang von sogenannten Charakteristiken propagiert, wodurch sich in diese Richtungen einmal die Lichtgeschwindikeit exakt erreichen läßt, zum anderen werden aber auch Reflexionen an den Rändern effektiv verhindert. Allerdings müssen diese Vorteile mit einem höheren Aufwand bei der Implementierung und einem größeren Rechenzeitbedarf erkauft werden, dafür verringert sich der Rechenaufwand bei den Teilchen, da diese mit größerer Schrittweite arbeiten könnnen.

Literaturverzeichnis

[1] O. Buneman, *Subgrid Resolution of Flow and Force Fields*, J. Computational Physics, 11, S, 250-268, Februar 1973

[2] J. P. Boris, *Relativistic plasma simulation-optimization of a hybrid code*, Proc. Fourth Conf. Num. Sim. Plasmas, Naval Res. Lab., Washington D.C., 3-67, 2-3 November 1970,

[3] C. K. Birdsall, A. B. Langdon, *Plasma Physics via Computer Simulation*, Institute of Physics Publishing, Bristol and Philadelphia, ISBN 0 7503 0117 1, 1991

[4] H. Ruhl. *Intense Laser Pulse Propagation through Underdense Plasma*, Journal of Plasma and Fusion Research, Vol. 74, No. 4(1998), pp. 322 - 335

[5] I.N.Bronstein, K.A.Semendjajew, *Taschenbuch der Mathematik*, Verlag Harri Deutsch, Thun und Frankfurt/Main, 23. Auflage

[6] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, V. Sunderan, *PVM 3 USER'S GUIDE AND REFERENCE MANUAL*, MIT Press, Cambridge Massachusets

[7] `http://www.csm.ornl.gov/pvm/`, offizielle Web-Seite des PVM-Projektes auf dem Server der Oak Ridge National Laboratories in Tennessee, USA

[8] E. Schmutzer, *Grundlagen der Theoretischen Physik, Teil I*, BI Wissenschaftsverlag, Mannheim/Wien/Zürich, ISBN 3-411-03145-X, 1989

[9] R.J.Goldston, P.H.Rutherford, *Introduction to Plasma Physics*, Institute of Physics Publishing Bristol and Philadelphia, ISBN 0-7503-0183-X, 2. Auflage 1997

[10] M. Abramowitz, I.A. Stegun, *Handbook of Mathematical Functions*, ISBN 486-61272-4, Dover Publications, Inc. New York, (1972)

[11] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical Recipes in C*, ISBN 0-521-43108-5, Cambridge University Press, Cambridge UK, (2. Auflage m. Korrekturen 1995)

[12] B. Stroustrup, *The C++ Programming Language*, ISBN 0-201-53992-6, Addison Wesley Publishing Company, New York USA, 2. Auflage 1991

[13] G. Joos, *Lehrbuch der Theoretischen Physik*, ISBN 3-89104-462-3, Aula-Verlag GmbH, Wiesbaden

[14] B.W. Char, K.O. Geldes, G.H. Gonnet, B.L. Leony, M.B. Monagan, S.M. Watt, *First Leafs: A Tutorial Introduction to Maple V*, ISBN 3-540-97621-3, Springer Verlag Berlin – Heidelberg – New York

[15] D. Louis, *C und C++: Programmierung und Referenz*, ISBN 3-8272-5066-8, Markt und Technik, Haar bei München

Hilfsmittel

Die für diese Arbeit verwendete Hardware war ein IBM-kompatibler PC mit zwei Prozessoren vom Typ Intel Pentium II mit 400 MHz Taktfrequenz. Das Betriebssystem war Linux, anfangs mit der Kernelversion 2.0.36, später 2.2.9 mit Multiprozessor-Unterstützung. Der verwendetete Compiler war **egcs-1.1.1**, ein frei erhältlicher C- und C++-Compiler.

Als Editor für das Programm und diese Arbeit kam die Version 20.4 des universellen Werkzeuges `xemacs` zum Einsatz.

Tests und einfachere Berechnungen wurden auch durchgeführt auf einer Kombination aus einem Notebook und einem Desktop mit Pentium-Prozessoren mit 100 bzw. 200 MHz. Mit dieser Kombination wurde überwiegend das Verhalten von PVM im Netzwerkbetrieb getestet. Allerdings liegt die Rechenleistung dieser Kombination etwa um den Faktor 5 bis 10 unter der des Dualprozessor-Systems. Das verwendete Betriebssystem war hier auch in jedem Fall Linux, wobei auf dem Desktop die Kernelversion 2.0.36 zum Einsatz kam, während das Notebook später mit der Kernelversion 2.2.9 betrieben wurde..

Für das Erstellen dieser Arbeit wurde ausschließlich auf frei erhältliche Software unter Linux zurückgegriffen, mit Ausnahme des Visualisierungstools IDL, das im Demo-Mode verwendet wurde.

Danken möchte ich an dieser Stelle auch allen Menschen, die durch ihre größtenteils unentgeltliche Arbeit ein freies Betriebssystem wie Linux überhaupt erst möglich gemacht haben.

Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig angefertigt zu haben.

Darmstadt, im August 1999.