Hardwarebeschleunigte elektromagnetische Simulation einer Drahtantenne

Bachelorarbeit – HTW Berlin

Lukas Brumm (S0553370)
1. und 2. Prüfer: Udo Pursche, Heiko Hübert
Datum: 11.03.19

Contents

1 Einleitung

Die vorliegende Arbeit 'Hardwarebeschleunigte elektromagnetische Simulation von Drahtantennen' wurde im Rahmen des Studienganges 'Informations- und Kommunikationstechnik' an der Hochschule für Technik und Wirtschaft Berlin verfasst. Ziel der Arbeit ist es die Methode der finiten Differenzen im Zeitbereich (FDTD-Methode) nachzuvollziehen und für eine hardwarebeschleunigte Simulation von Drahtantennen zu verwenden. Ich hiehlt mich bei der Herleitung der FDTD-Methode an die Vorlesung 'Electromagnetic Analysis Using Finite-Difference Time-Domain' samt veröffentlichtem Skript Raymond Rumpfs der Universität Texas in El Paso [Rumpf(a)], wobei ich mich entschloss die Maxwellgleichungen für ein normalisiertes H-Feld zu lösen und nur auf dem E- und H-Feld zu arbeiten. Dadurch lässt sich die numerische Lösung der Maxwellgleichungen durch nur zwei Formeln gänzlich beschreiben.

2 Motivation

Jede drahtlose Kommunikation benötigt eine Antenne auf der Sende und Empfangsseite. Durch die fortschreitende Miniaturisierung der Elektronik und eine daraus folgende Vernetzung wie man sie zum Beispiel im Internet of Things (IoT) beobachten kann, werden immer spezifischere Charakteristiken an die Sende- und Empfangsantenne gestellt. In solchen Einsatzgebieten sind konventionelle Antennendesigns häufig nicht anwendbar, sondern erfordern spezielle Antennen. Der Entwurf solcher Antennen passiert interaktiv mit einem Simulationsprogramm, wobei auch gezeigt wurde, dass evolutionäre Algorithmen (EA) zum automatisierten Antennendesign verwendet werden können. [Globus et al.()Globus, Linden, and John] Ein solcher EA muss mitunter hunderttausende Simulationen durchführen [Globus et al.()Globus, Linden, and John, p.6], weshalb jede Simulation so performant wie möglich sein sollte.

3 Stand der Technik

Das numerische Lösen von Differentialgleichungnen wird seit 1928 untersucht [R. Kourant(1928)] und hat sich seitdem kontinuierlich weiterentwickelt. [Hagness(2005)] Neben der FDTD existieren noch andere Verfahren, um Differentialgleichungen zu lösen, wie zum Beispiel die Finite-Elemente-Methode, die Finite-Volumen-Methode, oder die Randelement-Methode, die

häufig für Differentialgleichungen besonderer Form entwickelt wurde [Steunfeld()]. Die FDTD-Methode hingegen kann auf beliebige Differentialgleichungen angewandt werden und wird im Verlaufe der Arbeit eingehend behandelt. Auch wenn die FDTD-Methode seit 1928 erforscht wird, werden für die Einsatzgebiete der FDTD noch immer neue Erkenntisse gewonnen, so ist zum Beispiel die Simulation einer Antenneneinspeisung noch nicht vollständig geklärt und wird weiterhin untersucht [Luebbers and H.S()].

4 Lösen der Maxwellgleichungen durch die FDTD-Methode

Die FDTD-Methode, auch Yee-Methode genannt, löst Differentialgleichungen indem die als infinitesimal klein angenommenen Differentialquotienten durch finite Differenzenquotienten approximiert werden. [R. Kourant(1928)] Die Maxwellgleichungen werden daher nur in der Differentialform betrachtet. Die FDTD-Methode ist zu komplex und problemabhängig, als dass eine Einschätzung der Simulationsergebnise gelingen könnte, ohne die FDTD-Methode mathematisch für ein Problem nachvollzogen zu haben. Es werden daher nun kurz die Maxwellgleichungen eingeführt, allgemeine Stabilitätsbedingungen der FDTD-Methode behandelt und die FDTD-Methode genutzt, um eine numerische Lösung der Maxwellgleichungen sowie des 'Perfectly Matched Layer' (PML) herzuleiten.

4.1 Maxwellgleichungen

Unter den Maxwellgleichungen versteht man die folgenden Gleichungen, die zusammen alle elektromagnetischen Phänomene beschreiben. Das Gauß-Gesetz für Elektrizität [Rumpf(b)],

$$\nabla * \vec{D} = \rho_v \tag{1}$$

wobei $\vec{D}$ die elektrische Flussdichte und ρ_v die Ladungsdichte ist. Elektrische Felder divergieren bei positiven Ladungen und konvergieren bei negativen Ladungen. Das Gauß-Gesetz für Magnetismus [Rumpf(c)],

$$\nabla * \vec{B} = 0 \tag{2}$$

wobei $\vec{B}$ die magnetische Flussdichte ist. Es gibt keine magnetischen Monopole, Magnetfelder formen immer Schleifen. Das Amperesche Gesetz [Rumpf(d)],

$$\nabla \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \tag{3}$$

wobei $\vec{H}$ das magnetische Feld und $\vec{J}$ die Stromdichte ist. Zirkulierende magnetishe Felder induzieren Ströme sowie zeitabhängige elektrische Felder. Das Induktionsgesetz [Rumpf(e)],

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{4}$$

wobei $\vec{E}$ das elektrische Feld ist. Zirkulierende elektrische Felder induzieren zeitabhängige magnetische Felder. Weiterhin ist das D-Feld über die Permittivität mit dem E-Feld verbunden, [Rumpf(f)]

$$\vec{D}(t) = [\epsilon(t)] * \vec{E}(t) \tag{5}$$

sowie das B-Feld über die Permeabilität mit dem H-Feld verbunden ist. [Rumpf(f)]

$$\vec{B}(t) = [\mu(t)] * \vec{E}(t) \tag{6}$$

In beiden Fällen handelt es sich um eine Faltung mit einem Tensor. Im Verlauf der Arbeit wird angenommen, dass die Permittivität und die Permeabilität zeitlich konstant sind, damit wird die Faltung zu einer Multiplikation. Verwendet man die Gleichungen (5) und (6) um das D- und B-Feld zu eliminieren und gibt man die Permittivität und Permeabilität relativ zur Vacuumpermittivität und -permeabilität an, so ergeben sich folgende vier Formeln.

$$\nabla * ([\mu]\vec{H}(t)) = 0 \tag{7}$$

$$\nabla * ([\epsilon]\vec{E}(t)) = \rho_v \tag{8}$$

$$\nabla \times \vec{H} = \epsilon_0[\epsilon_r]\frac{\partial \vec{E}}{\partial t} \tag{9}$$

$$\nabla \times \vec{E} = -\mu_0[\mu_r]\frac{\partial \vec{H}}{\partial t} \tag{10}$$

Aus den beiden verschränkten Gleichungen (9) und (10) werden später die sog. Updategleichungen hergeleitet, also die Gleichungen die das E- bezw. H-Feld des nächsten Zeitschrittes liefern. Die Stromdichte $\vec{J}$ kann genutzt werden, um Verlust innerhalb von Materialien darzustellen. Im Verlauf der Arbeit wird angenommen, dass die Antennengeometrie nicht verlustbehaftet ist.

$$\vec{J} = 0 \tag{11}$$

Das E- und H-Feld sind durch die Materialimpedanz η mit einander verknüpft, wobei jede Impedanz relativ zur Freiraumimpedanz η_0 angegeben werden kann:

$$\eta = \frac{|\vec{E}|}{|\vec{H}|} \tag{12}$$

$$\eta = \eta_0 * \sqrt{\frac{\mu_r}{\epsilon_r}}, \eta_0 = \pi * 119,9169832\Omega \tag{13}$$

Im freien Raum ist das E- Feld also um den Faktor η_0 größer als das H-Feld. Diese Skalierung führt bei numerischen Lösungsverfahren zu Rundungsfehlern, weshalb das H-Feld normalisiert wird und somit die gleiche Größenordnung wie das E-Feld besitzt. [Rumpf(g)]

$$\widetilde{\vec{H}} = \eta_0 \vec{H} \tag{14}$$

$$\vec{H} = \frac{1}{\eta_0} \widetilde{\vec{H}} \tag{15}$$

Somit ergibt sich aus Gleichung (10):

$$\begin{aligned}\nabla \times \vec{E} &= [-\mu]\frac{\partial \widetilde{\vec{H}}}{\partial t} * \frac{1}{\eta_0} \quad |\eta_0 = \mu_0 * c \\ &= -\mu_0[\mu_r]\frac{1}{\mu_0 c}\frac{\partial \widetilde{\vec{H}}}{\partial t} \\ &= -\frac{[\mu_r]}{c}\frac{\partial \widetilde{\vec{H}}}{\partial t} \end{aligned} \tag{16}$$

Und aus Gleichung (9) ergibt sich:

$$\begin{aligned}\nabla \times \widetilde{\vec{H}} &= \eta_0\epsilon_0[\epsilon_r]\frac{\partial \vec{E}}{\partial t} \\ &= c\mu_0\epsilon_0[\epsilon_r]\frac{\partial \vec{E}}{\partial t} \quad |\mu_0 = \frac{1}{c^2\epsilon_0} \\ &= \frac{1}{c^2\epsilon_0}c\epsilon_0[\epsilon_r]\frac{\partial \vec{E}}{\partial t} \\ &= \frac{[\epsilon_r]}{c}\frac{\partial \vec{E}}{\partial t} \end{aligned} \tag{17}$$

4.2 Mathematische Stabilitätsbedingungen für finite Differenzen

Immer wenn die FDTD-Methode auf Differentialgleichungen angewendet wird muss jede finite-Differenz am selben Punkt im Raum und in der Zeit existieren, um die Differentialgleichung korrekt zu approximieren. [Rumpf(h)]

$$\frac{\partial f(x)}{\partial x} + f(x) = 0 | FDTD - Methode \tag{18}$$

$$\underbrace{\frac{x + \Delta x - f(x)}{\Delta x}}_{\text{Existiert am Punkt x}+\frac{\Delta x}{2}} + \underbrace{f(x)}_{Existiert\ am\ Punkt\ x} = 0 \quad |x = n\Delta x, x \in \mathbb{N} \tag{19}$$

Diese Gleichung wäre instabil, der Term $f(x)$ muss interpoliert werden um ebenfalls an den Punkten $x + \frac{\Delta x}{2}$ zu existieren:

$$\frac{f(x + \Delta x) - f(x)}{\Delta x} + \frac{f(x + \Delta x) + f(x)}{2} = 0 \tag{20}$$

4.3 Anwenden der FDTD auf die Maxwellgleichungen

Die Gleichungen (9) und (10) beinhalten eine offensichtliche Ableitung nach der Zeit und eine Ableitung nach allen drei Dimensionen, die im ∇-Operator steckt:

$$\nabla = (\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n}) \tag{21}$$

Als erstes wird die Ableitung nach der Zeit betrachted und mit der FDTD-Methode approximiert:

$$\nabla \times \vec{E}(t) = -\frac{[\mu_r]}{c} \frac{\partial \widetilde{\vec{H}}(t)}{\partial t} \quad |FDTD - Methode \tag{22}$$

$$\underbrace{\nabla \times \vec{E}(t)}_{\text{Zeitpunkt t}} = \underbrace{-\frac{[\mu_r]}{c} \frac{\widetilde{\vec{H}}(t + \Delta t) - \widetilde{\vec{H}}(t)}{\Delta t}}_{Zeitpunkt\ t+\frac{\Delta t}{2}} \tag{23}$$

$$\nabla \times \widetilde{\vec{H}}(t) = \frac{\epsilon_r}{c} \frac{\partial \vec{E}(t)}{\partial t} \quad |FDTD - Methode \tag{24}$$

$$\underbrace{\nabla \times \widetilde{\vec{H}}(t)}_{\text{Zeitpunkt t}} = \underbrace{\frac{\epsilon_r}{c} \frac{\vec{E}(t + \Delta t) - \vec{E}(t)}{\Delta t}}_{Zeitpunkt\ t+\frac{\Delta t}{2}} \tag{25}$$

$$\tag{26}$$

Diese Gleichungen sind aus den oben genannten Gründen instabil. Das Problem wird gelöst, indem man das H-Feld um $\Delta t/2$ verschoben definiert. [Rumpf(i)] Daraus ergibt sich:

$$\nabla \times \vec{E}(t) = -\mu \frac{\vec{H}(t + \frac{\Delta t}{2}) - \vec{H}(t - \frac{\Delta t}{2})}{\Delta t} \tag{27}$$

$$\nabla \times \vec{H}(t + \frac{\Delta t}{2}) = \epsilon \frac{\vec{E}(t + \Delta t) - \vec{E}(t)}{\Delta t} \tag{28}$$

Nun existieren alle Terme einer Gleichung zu denselben Zeitpunkten. Das gleiche Problem tritt bei der Ableitung nach dem Ort auf, doch erst muss der ∇-Operator aufgelöst werden, was zu den folgenden sechs Gleichungen führt:

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = -\frac{1}{c}(\mu_{xx}\frac{\partial \widetilde{H_x}}{\partial t} + \mu_{xy}\frac{\partial \widetilde{H_y}}{\partial t} + \mu_{xz}\frac{\partial \widetilde{H_z}}{\partial t}) \tag{29}$$

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = -\frac{1}{c}(\mu_{yx}\frac{\partial \widetilde{H_x}}{\partial t} + \mu_{yy}\frac{\partial \widetilde{H_y}}{\partial t} + \mu_{yz}\frac{\partial \widetilde{H_z}}{\partial t}) \tag{30}$$

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = -\frac{1}{c}(\mu_{zx}\frac{\partial \widetilde{H_x}}{\partial t} + \mu_{zy}\frac{\partial \widetilde{H_y}}{\partial t} + \mu_{zz}\frac{\partial \widetilde{H_z}}{\partial t}) \tag{31}$$

$$\frac{\partial \widetilde{H_z}}{\partial y} - \frac{\partial \widetilde{H_y}}{\partial z} = \frac{1}{c}(\epsilon_{xx}\frac{\partial E_x}{\partial t} + \epsilon_{xy}\frac{\partial E_y}{\partial t} + \epsilon_{xz}\frac{\partial E_z}{\partial t}) \tag{32}$$

$$\frac{\partial \widetilde{H_x}}{\partial z} - \frac{\partial \widetilde{H_z}}{\partial x} = \frac{1}{c}(\epsilon_{yx}\frac{\partial E_x}{\partial t} + \epsilon_{yy}\frac{\partial E_y}{\partial t} + \epsilon_{yz}\frac{\partial E_z}{\partial t}) \tag{33}$$

$$\frac{\partial \widetilde{H_y}}{\partial x} - \frac{\partial \widetilde{H_x}}{\partial y} = \frac{1}{c}(\epsilon_{zx}\frac{\partial E_x}{\partial t} + \epsilon_{zy}\frac{\partial E_y}{\partial t} + \epsilon_{zz}\frac{\partial E_z}{\partial t}) \tag{34}$$

Es ist immer möglich ein Koordinatensystem zu wählen, sodass alle nicht diagonalen Anteile der relativen Permittivität und Permeabilitiät null sind, sie können daher als null angenommen werden, wodurch sich die Gleichungen vereinfachen [Rumpf(j)]:

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = -\frac{1}{c}\mu_{xx}\frac{\partial \widetilde{H_x}}{\partial t} \tag{35}$$

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = -\frac{1}{c}\mu_{yy}\frac{\partial \widetilde{H_y}}{\partial t} \tag{36}$$

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = -\frac{1}{c}\mu_{zz}\frac{\partial \widetilde{H_z}}{\partial t} \tag{37}$$

$$\frac{\partial \widetilde{H_z}}{\partial y} - \frac{\partial \widetilde{H_y}}{\partial z} = \frac{1}{c}\epsilon_{xx}\frac{\partial E_x}{\partial t} \tag{38}$$

$$\frac{\partial \widetilde{H_x}}{\partial z} - \frac{\partial \widetilde{H_z}}{\partial x} = \frac{1}{c}\epsilon_{yy}\frac{\partial E_y}{\partial t} \tag{39}$$

$$\frac{\partial \widetilde{H_y}}{\partial x} - \frac{\partial \widetilde{H_x}}{\partial y} = \frac{1}{c}\epsilon_{zz}\frac{\partial E_z}{\partial t} \tag{40}$$

Wendet man nun die FDTD-Methode erneut an, um die Ableitung nach dem Ort zu approximieren, ergeben sich:

$$\frac{E_z^{i,j+1,k}|_t - E_z^{i,j,k}|_t}{\Delta y} - \frac{E_y^{i,j,k+1}|_t - E_y^{i,j,k}|_t}{\Delta z} = -\frac{\mu_{xx}^{i,j,k}}{c}\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{41}$$

$$\frac{E_x^{i,j,k+1}|_t - E_x^{i,j,k}|_t}{\Delta z} - \frac{E_z^{i+1,j,k}|_t - E_z^{i,j,k}|_t}{\Delta x} = -\frac{\mu_{yy}^{i,j,k}}{c}\frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{42}$$

$$\frac{E_y^{i+1,j,k}|_t - E_y^{i,j,k}|_t}{\Delta x} - \frac{E_x^{i,j+1,k}|_t - E_x^{i,j,k}|_t}{\Delta y} = -\frac{\mu_{zz}^{i,j,k}}{c}\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{43}$$

$$\frac{\widetilde{H_z}^{i,j+1,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k+1}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta z} = -\frac{\epsilon_{xx}^{i,j,k}}{c}\frac{E_x^{i,j,k}|_{t+\Delta t} - E_x^{i,j,k}|_t}{\Delta t} \tag{44}$$

$$\frac{\widetilde{H_x}^{i,j,k+1}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta z} - \frac{\widetilde{H_z}^{i+1,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x} = -\frac{\epsilon_{yy}^{i,j,k}}{c}\frac{E_y^{i,j,k}|_{t+\Delta t} - E_y^{i,j,k}|_t}{\Delta t} \tag{45}$$

$$\frac{\widetilde{H_y}^{i+1,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x} - \frac{\widetilde{H_x}^{i,j+1,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} = -\frac{\epsilon_{zz}^{i,j,k}}{c}\frac{E_z^{i,j,k}|_{t+\Delta t} - E_z^{i,j,k}|_t}{\Delta t} \tag{46}$$

Wobei i, j, k den diskreten Punkt im Raum angeben und $\Delta x, \Delta y, \Delta z$ sowie Δt die Größe der finiten Differenzen sind. Betrachtet man zum Beispiel die Gleichung (41), so fällt auf, dass auch hier die Stabilitätsbedingung der FDTD-Methode nicht eingehalten wird:

$$\underbrace{\frac{E_z^{i,j+1,k}|_t - E_z^{i,j,k}|_t}{\Delta y}}_{\text{Existiert am Ort i,j+}\frac{\Delta y}{2}\text{,k}} - \underbrace{\frac{E_y^{i,j,k+1}|_t - E_y^{i,j,k}|_t}{\Delta z}}_{\text{Existiert am Ort i,j,k+}\frac{\Delta z}{2}} = -\frac{\mu_{xx}^{i,j,k}}{c}\underbrace{\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t}}_{\text{Existiert am Ort i,j,k}} \tag{47}$$

Dieses Problem kann gelöst werden, indem man definiert, dass das H-Feld dem E-Feld um den Vector $\left(\frac{\Delta x}{2}, \frac{\Delta y}{2}, \frac{\Delta z}{2}\right)$ verschoben ist. Man bezeichnet die zeitlich und räumlich verschobenen Bezugsrahmen nach dem Mathematiker Kane S. Yee Yee-Gitter. [Yee(1966)] Durch die Verwendung des Yee-Gitters

werden die Gleichungen (1) und (2) sowie die Kontinuitätsbedingung an Mediumsübergängen implizit eingehalten und müssen nicht mehr explizit behandelt werden. Für die räumliche Ableitung muss nun ein positiver und ein negativer Offset verwendet werden. Es handelt sich dabei jedoch weiterhin um eine rechtsseitige Ableitung.

$$\frac{E_z^{i,j+1,k}|_t - E_z^{i,j,k}|_t}{\Delta y} - \frac{E_y^{i,j,k+1}|_t - E_y^{i,j,k}|_t}{\Delta z} = -\frac{\mu_{xx}^{i,j,k}}{c} \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{48}$$

$$\frac{E_x^{i,j,k+1}|_t - E_x^{i,j,k}|_t}{\Delta z} - \frac{E_z^{i+1,j,k}|_t - E_z^{i,j,k}|_t}{\Delta x} = -\frac{\mu_{yy}^{i,j,k}}{c} \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{49}$$

$$\frac{E_y^{i+1,j,k}|_t - E_y^{i,j,k}|_t}{\Delta x} - \frac{E_x^{i,j+1,k}|_t - E_x^{i,j,k}|_t}{\Delta y} = -\frac{\mu_{zz}^{i,j,k}}{c} \frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{50}$$

$$\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z} = -\frac{\epsilon_{xx}^{i,j,k}}{c} \frac{E_x^{i,j,k}|_{t+\Delta t} - E_x^{i,j,k}|_t}{\Delta t} \tag{51}$$

$$\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z} - \frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i-1,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x} = -\frac{\epsilon_{yy}^{i,j,k}}{c} \frac{E_y^{i,j,k}|_{t+\Delta t} - E_y^{i,j,k}|_t}{\Delta t} \tag{52}$$

$$\frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i-1,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x} - \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} = -\frac{\epsilon_{zz}^{i,j,k}}{c} \frac{E_z^{i,j,k}|_{t+\Delta t} - E_z^{i,j,k}|_t}{\Delta t} \tag{53}$$

Nun können die Gleichungen nach den Feldstärkewerten des jeweils nächsten Zeitschritts umgestellt werden. Es ergeben sich folgende Updategleichungen:

$$\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} = \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}} - \frac{c\Delta t}{\mu_{xx}^{i,j,k}}\left(\frac{E_z^{i,j+1,k}|_t - E_z^{i,j,k}|_t}{\Delta y} - \frac{E_y^{i,j,k+1}|_t - E_y^{i,j,k}|_t}{\Delta z}\right) \tag{54}$$

$$\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} = \widetilde{H_y}^{i,j,k}|_{t-\frac{\Delta t}{2}} - \frac{c\Delta t}{\mu_{yy}^{i,j,k}}\left(\frac{E_x^{i,j,k+1}|_t - E_x^{i,j,k}|_t}{\Delta z} - \frac{E_z^{i+1,j,k}|_t - E_z^{i,j,k}|_t}{\Delta x}\right) \tag{55}$$

$$\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} = \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}} - \frac{c\Delta t}{\mu_{zz}^{i,j,k}}\left(\frac{E_y^{i+1,j,k}|_t - E_y^{i,j,k}|_t}{\Delta x} - \frac{E_x^{i,j+1,k}|_t - E_x^{i,j,k}|_t}{\Delta y}\right) \tag{56}$$

$$E_x^{i,j,k}|_{t+\Delta t} = E_x^{i,j,k}|_t - \frac{c\Delta t}{\epsilon_{xx}^{i,j,k}}\left(\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z}\right) \tag{57}$$

$$E_y^{i,j,k}|_{t+\Delta t} = E_y^{i,j,k}|_t - \frac{c\Delta t}{\epsilon_{yy}^{i,j,k}}\left(\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z} - \frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i-1,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x}\right) \tag{58}$$

$$E_z^{i,j,k}|_{t+\Delta t} = E_x^{i,j,k}|_t - \frac{c\Delta t}{\epsilon_{zz}^{i,j,k}}\left(\frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i-1,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x} - \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y}\right) \tag{59}$$

Einige Probleme, wie zum Beispiel das Simulieren eines Hornstrahlers, lassen sich aufgrund von Symmetrien im zwei- oder eindimensionalen Raum hinreichend genau beschreiben. Das reduziert die nötige Rechenleistung und wird, wann immer möglich, angewandt. Für eine Reduktion auf den zwei- oder eindimensionalen Fall nimmt man an, dass die Differenzenquotienten der nichtbetrachteten Dimensionen gleich null sind. Jede Geometrie wird dadurch in den nichtbetrachteten Dimensionen unendlich ausgedehnt. [Rumpf(k)] Für die Simulation von Drahtantennen ist diese Art von Vereinfachung leider nicht möglich, da die Geometrie eines Drahts in jede Dimension begrenzt ist und keine Symmetrie aufweist, die eine solche Annäherung erlauben würde.

4.4 Physikalische Stabilitätsbedingungen für finite Differenzen

Wie bei der Einführung der FDTD-Methode erwähnt, approximiert die FDTD-Methode Differentialgleichungen, indem die Differentialquotienten durch Differenzenquotienten ersetzt werden. Sie kann daher nur dann korrekte Ergebnisse liefern, wenn die betrachteten Differenzen hinreichend klein sind, so sollte die Gittergröße in jede Dimension so klein gewählt werden, dass die kleinste vorkommende Wellenlänge noch mindestenz 10mal abgetastet wird. Für die räumliche Abtastrate N_λ muss daher $N_\lambda \geq 10$ gelten [Rumpf(l)]. Für die Gittergrößen Δx, Δx, Δx gilt daher

$$\lambda_{min} = \frac{c}{f_{max} * \eta_{max}} \tag{60}$$

$$\Delta x, \Delta y, \Delta z \leq \frac{\lambda_{min}}{N_\lambda} \tag{61}$$

$$\Delta x, \Delta y, \Delta z \leq \frac{c}{f_{max}\eta_{max}N_\lambda} \tag{62}$$

Weiterhin muss die Gittergröße klein genug gewählt sein, dass die minimalste Strukturbreite mindestenz viermal abgetastet wird [Rumpf(l)]. Aus der ersten Bedingung heraus folgt, dass die Breite des Yee-Gitters die maximale Frequenz vorgibt,die korrekt simuliert werden kann. Das wird später für die Wahl einer geeigneten Quellenfunktion relevant werden. Da das E- und H-Feld zeitversetzt berrechnet werden ist es in der numerischen Lösung nicht möglich, dass sich eine Welle schneller als eine Zelle pro Zeitschritt ausbreitet. Der Zeitschritt Δt muss so klein gewählt werden, dass sich auch eine physikalische Welle in einem Zeitschritt nicht weiter als eine Zelle ausbreiten würde. Für den dreidimensionalen Fall wird das durch die Courant-Stabilitätsbedingung sichergestellt [Rumpf(m)]:

$$\Delta t \leq \frac{1}{c\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}}} \tag{63}$$

4.5 Geometrie in der FDTD

Geometrien können durch die ortsabhängige relative Permittivität und Permeabilität gesetzt werden, sie geben die Materialeigenschaften an einem Gitterpunk an. Für Metalle sind keine relativen Permittivitäten definiert, sie werden als unendlich groß angenommen. Daraus ergibt sich, dass die elektrische Feldstärke an Metallen null ist:

$$\lim_{\epsilon \to \infty} \frac{c\Delta t}{\epsilon} = 0 \tag{64}$$

Metallgeometrien können platziert werden, indem die elektrischen Feldstärkewerte nach der Berrechnung eines Zeitschrittes wieder auf null gesetzt werden, oder man einen Metallgeometriefaktor $m \in \{0, 1\}$ einführt:

$$E_x^{i,j,k}|_{t+\Delta t} = m * (E_x^{i,j,k}|_t - \frac{c\Delta t}{\epsilon_{xx}^{i,j,k}}(\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z})) \tag{65}$$

$$E_y^{i,j,k}|_{t+\Delta t} = m * (E_y^{i,j,k}|_t - \frac{c\Delta t}{\epsilon_{yy}^{i,j,k}}(\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z} - \frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i-1,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x})) \tag{66}$$

$$E_z^{i,j,k}|_{t+\Delta t} = m * (E_x^{i,j,k}|_t - \frac{c\Delta t}{\epsilon_{zz}^{i,j,k}}(\frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i-1,j,k}|_{t+\frac{\Delta t}{2}}}{\Delta x} - \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y})) \tag{67}$$

Dieser Faktor wird in der Implementierung der FDTD-Methode genutzt, der Übersichtlichkeit halber aber nicht weiter mitgeführt.

4.6 Quellen in der FDTD-Methode

Neben der Frage, wie sich Geometrien in der FDTD-Methode setzen, lassen stellt sich die Frage nach Quellen in der FDTD-Methode. Eine Quelle ist eine, wie auch immer geartete Erregung, des elektrischen oder magnetischen Feldes. Dabei kann eine Erregung als Addition passieren, man spricht dann von einer sog. 'Soft Source' oder als Zuweisung, man spricht dann von einer sog. 'Hard Source'. [Rumpf(n)] Da bei einer Hard Source, ähnlich wie bei einem Metall, ein Feldstärkewert vorgegeben wird, kommt es an ihr zu Reflektionen. Haüfig wird eine Quelle auch als Spannungsdifferenz zwischen zwei Gitterpunken implementiert, man spricht dann von einer sog. 'Gap-Source', die sich für eine Punktquellle in X-Richtung des E-Feldes an der Stelle q wie folgt berrechnet [Luebbers and H.S()]:

$$E_x^{q_x,q_y,q_z}|_t = u_q(t)/\Delta x \tag{68}$$

Wobei $u_q(t)$ die Quellspannung ist. Es wurde gezeigt, dass sich die nötigen Zeitschritte für einen eingeschwungenen Zustand einer Antenne durch die Wahl einer geeigneten Quelle reduzieren lassen [Luebbers and H.S()], etwas worauf der Einfachkeit halber verzichtet wurde. Neben der Art der Quelle beeinflusst auch das Quellensignal die Simulation. Da die FDTD nur bis zu einer maximalen Frequenz stabil ist, darf kein Quellensignal verwendet werden, dass Frequenzen überhalb der maximalen Frequenz aufweist. Zeitlich stark begrenzte Signale mit einem unendlich ausgedehnten Spektrum, wie zum Beispiel ein Rechteck- oder Deltaimpuls, führen immer zu instabilem Verhalten. Soll betrachtet werden, wie sich ein System bei unterschiedlichen Frequenzen verhält, kann ein Gauzimpuls genutzt werden. Seine Breite muss so gewaehlt werden, dass dessen Fouriertransformierte, wieder ein Gaußimpuls, für die maximale Frequenz eine Amplitude nahe null angenommen hat. Für die Simulation von Antennen bietet sich ein Sinus mit der gewünschten Sendefreuenz an. Da die FDTD-Methode ein numerisches Lösungsverfahren darstellt, beginnt die Berrechnung an einem diskreten Zeitpunkt. Für die sinusförmige Quellenspannung bedeutet das $q(t) = 0, \forall t \in [-\infty, 0]$. Analytisch entspricht dies einer Multiplikation mit der Heaviside-Funktion, die zu einer Faltung mit $\frac{1}{j\omega}$ im Frequenzbereich korrespondiert. Um die Verzerrung des Quellenspektrums möglichst gering zu halten, wird daher meist ein Sinus verwendet, dessen Amplitude sich kontinuierlich auf die gewünschte Zielamplitude vergrößert [Rumpf(o)].

4.7 Numerisches Randwertproblem

Eine numerische Lösung einer Differentialgleichung ist immer nur in einem begrenzten Problembereich möglich. Betrachtet man jedoch die Updatefunktionen, so fällt auf, dass die räumliche Ableitung für einen Feldstärkewert am Rand des Problembereichs einen Feldstärkewert von außerhalb des Problembereichs benötigt. Die einfachste Möglichkeit dieses Randwertproblem zu lösen ist die sog. Dirichlet-Randbedingung. Es wird angenommen, dass die Feldstärkewerte außerhalb des Problembereichs null sind [Rumpf(p)]. Das entspricht, wie oben beschrieben, jedoch der Eigenschaft eines perfekten Leiters. Durch die Dirichlet-Randbedingungen gibt es eine Totalreflektion am Rand des Problembereichs. Neben der Dirichlet-Randbedingung gibt es noch sog. Absorbierende Randbedingungen (ABCs), die versuchen die Ausbreitung der EM-Wellen vorrauszusagen und Feldstärken außerhalb des Problembereichs annährern können, abgestrahlte EM-Wellen würden so den Problembereich verlassen.

4.8 Perfectly Matched Layer (PML)

ABCs fordern dass der Winkel oder die Frequenz der EM-Welle bekannt ist, um einen Anhaltspunk zu haben anhand dessen die Ausbreitung der EM-Welle vorausgesagt werden kann. Für die Simulation von Antennen können solche Angaben jedoch nicht getroffen werden, es können daher keine ABCs verwendet werden. Um zu verhindern, dass abgestrahlte Wellen in den Problembereich zurückreflektiert werden und die Abstrahlcharakteristik verfälschen, muss der unendlich ausgedehnte Freiraum simuliert werden. Das geschieht durch ein sog. 'Pefectly Matched Layer' (PML), ein nur in der Theorie existierendes anisotropes Material, das Wellen aller Frequenzen und unter jedem Winkel reflektionsfrei passieren lässt und dennoch verlustbehaftet ist [Schneider()]. Abgestrahlte Wellen werden absorbiert, bevor sie am Rand des Problembereichs reflektiert werden. Folgend wird das PML hergeleitet, dafür werden die Maxwellgleichungen zuerst im Frequenzbereich betrachtet und anschließend wieder in den Zeitbereich zurücktransformiert [Rumpf(q)]. Im Frequenzbereich lassen sich verlustbehaftete Materialien durch eine komplexe Permittivität ausdrücken. [Rumpf(r)] Die Herleitung wird nur für X-Komponente des E- und des H-Feldes passieren da sie auf gleiche Weise für die anderen Komponenten durchgeführt werden kann. Sie werden folgend mit I und II nummeriert. Damit keine Reflektion und keine Brechung stattfindet muss die Impedanz des PML gleich der Impedanz des umliegenden Materials sein, da der unendliche Freiraum simuliert werden

soll, wird es zu einer Mediumsgrenze von Vakuum zu PML geben. Die Impedanz des PML muss daher überall 1 sein. Aus der Formel der Impedanz folgt, dass dafür die relative Permittivität gleich der relativen Permeabilität sein muss. [Rumpf(s)]

$$\eta = \sqrt{\mu_r}\epsilon_r \stackrel{!}{=} 1 \tag{69}$$
$$\Rightarrow \mu_r = \epsilon_r \tag{70}$$

Da μ_r und ϵ_r gleich sind wird, ein neuer Parameter s eingeführt. Dieser hat die wie μ_r und ϵ die Form einer Diagonalmatrix:

$$[s] = [\mu_r] = [\epsilon_r] \tag{71}$$
$$[s] = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} \tag{72}$$

Aus dem Selliuschen Brechungsgesetz für anisotrope Medien in Z-Richtung und der Bedingung $\eta_1 = 1$ folgt:

$$\eta_1 \sin\theta 1 = \sqrt{bc}\sin\theta_2 \quad , \eta_1 = 1 \tag{73}$$
$$\sin\theta 1 = \sqrt{bc}\sin\theta_2 \tag{74}$$

Da keine Brechung vorliegen soll, muss $\theta_1 = \theta_2$ gelten, wodurch:

$$\sqrt{bc} = 1 \tag{75}$$
$$c = \frac{1}{b} \tag{76}$$

gelten muss. Mit $\theta_1 = \theta_2$ vereinfacht sich das Fresnel-Gesetz für anisotrope Medien in Z-Richtung zu einer winkelunabhängigen Form:

$$r_{TE} = \frac{\sqrt{a}\cos\theta_1 - \sqrt{b}\cos\theta_2}{\sqrt{a}\cos\theta 1 + \sqrt{b}\cos\theta_2} \tag{77}$$

$$= \frac{\sqrt{a} - \sqrt{b}}{\sqrt{a} - \sqrt{b}} \tag{78}$$

$$r_{TM} = \frac{\sqrt{a}\cos\theta_2 - \sqrt{b}\cos\theta_1}{\sqrt{a}\cos\theta 1 + \sqrt{b}\cos\theta_2} \tag{79}$$

$$= \frac{-\sqrt{a} + \sqrt{b}}{\sqrt{a} + \sqrt{b}} \tag{80}$$

Da $r_{TE} = 0$ und $r_{TM} = 0$ gelten soll muss:

$$\sqrt{a} - \sqrt{b} = 0 \tag{81}$$

$$\sqrt{a} = \sqrt{b} \tag{82}$$

$$a = b \tag{83}$$

gelten. Mit Gleichung (76) ergibt sich nun für $[s_z]$:

$$[s_z] = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & \frac{1}{a} \end{bmatrix} = \begin{bmatrix} b & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & \frac{1}{b} \end{bmatrix} \quad |s_z = a = b \in \mathbb{C} \tag{84}$$

$$[s_z] = \begin{bmatrix} sz & 0 & 0 \\ 0 & sz & 0 \\ 0 & 0 & \frac{1}{sz} \end{bmatrix} \tag{85}$$

Durch Rotation erhält man die Parameter $[s_x]$ und $[s_y]$ die eine Medium mit relfektionsfreier Mediumsgrenze in X- und Y-Richtung beschreibt.

$$[s_x] = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & s_x & 0 \\ 0 & 0 & s_x \end{bmatrix} \tag{86}$$

$$[s_y] = \begin{bmatrix} s_y & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & s_y \end{bmatrix} \tag{87}$$

Durch eine Multiplikation erhält man einen Parameter $[s]$, der ein reflektionsfreies Material für alle Richtungen beschreibt:

$$[s] = [s_x] * [s_y] * [s_y] \tag{88}$$

$$= \begin{bmatrix} \frac{s_y s_z}{s_x} & 0 & 0 \\ 0 & \frac{s_x s_z}{s_y} & 0 \\ 0 & 0 & \frac{s_x s_y}{s_z} \end{bmatrix} \tag{89}$$

Die Parameter s_x, s_y und s_z sind komplex und werden nur innerhalb ihrer jeweiligen PML-Schichten ungleich 1. Im nun etwas kleiner gewordenen Problembereich gilt: $s_x = s_y = s_y = 1$. Innerhalb des PML-Materials steigen die Parameter kontinuierlich an:

$$s_x(x) = 1 + \frac{\sigma`(x)}{j\omega\epsilon_0}, \quad \sigma`(x) = \frac{\epsilon_0}{2\Delta t}\frac{x}{L_x}^3, \quad x \in [0, L_x] \tag{90}$$

Wobei L_x die Länge des PML-Materials in X-Richtung ist. Die Gleichung gilt analog für s_y und s_y. Dieser kontinuierliche Anstieg ist nur aufgrund des Diskretiesierungsfehlers nötig. Der nun vollständig beschriebene Parameter $[s]$ wird in den Maxwellgleichungen als zusätzliche Materialeigenschaft verwendet. Im Frequenzbereich ergibt sich dadurch:

$$\nabla \times \vec{H} = j\omega\epsilon_0[\epsilon_r][s]\vec{E} \tag{91}$$

$$\nabla \times \vec{E} = -j\omega\mu_0[\mu_r][s]\vec{H} \tag{92}$$

Normalisiert man das H-Feld ergibt sich:

$$\nabla \times \vec{H} = j\omega \frac{[\epsilon_r][s]}{c} \vec{E} \tag{93}$$

$$\nabla \times \vec{E} = -j\omega \frac{[\mu_r][s]}{c} \vec{H} \tag{94}$$

Für die Rücktransformation umgestellt ergibt sich:

$$\mathrm{I} : j\omega(1 + \frac{\sigma_x^{'}}{j\omega\epsilon_0})^{-1}(1 + \frac{\sigma_y^{'}}{j\omega\epsilon_0})(1 + \frac{\sigma_z^{'}}{j\omega\epsilon_0})E_x = \frac{c}{\epsilon_{xx}}(\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z}) \quad | * (1 + \frac{\sigma_x^{'}}{j\omega\epsilon_0}) \tag{95}$$

$$\mathrm{II} : j\omega(1 + \frac{\sigma_x^{'}}{j\omega\epsilon_0})^{-1}(1 + \frac{\sigma_y^{'}}{j\omega\epsilon_0})(1 + \frac{\sigma_z^{'}}{j\omega\epsilon_0})H_x = -\frac{c}{\mu_{xx}}(\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z}) \quad | * (1 + \frac{\sigma_x^{'}}{j\omega\epsilon_0}) \tag{96}$$

$$\mathrm{I} : j\omega(1 + \frac{\sigma_y^{'}}{j\omega\epsilon_0})(1 + \frac{\sigma_z^{'}}{j\omega\epsilon_0})E_x = \frac{c}{\epsilon_{xx}} \underbrace{(\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z})}_{\mathrm{C_x^H}}(1 + \frac{\sigma_x^{'}}{j\omega\epsilon_0}) \tag{97}$$

$$\mathrm{II} : j\omega(1 + \frac{\sigma_y^{'}}{j\omega\epsilon_0})(1 + \frac{\sigma_z^{'}}{j\omega\epsilon_0})H_x = -\frac{c}{\mu_{xx}} \underbrace{(\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z})}_{\mathrm{C_x^E}}(1 + \frac{\sigma_x^{'}}{j\omega\epsilon_0}) \tag{98}$$

$$\mathrm{I} : j\omega E_x + \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x + \frac{1}{j\omega}\frac{\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} E_x = \frac{c}{\epsilon_{xx}} C_x^H + \frac{1}{j\omega}\frac{c\sigma_x^{'}}{\epsilon_{xx}\epsilon_0} C_x^H \tag{99}$$

$$\mathrm{II} : j\omega H_x + \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} H_x + \frac{1}{j\omega}\frac{\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} H_x = \frac{c}{\mu_{xx}} C_x^E - \frac{1}{j\omega}\frac{c\sigma_x^{'}}{\mu_{xx}\epsilon_0} C_x^E \tag{100}$$

Die Rücktransformation kann, aufgrund der Linearitätseigenschaft der Fouriertransformation, für jeden Term einzeln durchgeführt werden. Der Uebersichtlichkeit halber wird die Rücktransformation für die beiden Gleichungen getrennt vorgenommen.

Die Korrespondenztabelle für I ist:

$$j\omega E_x(\omega) \circ\!\!-\!\!\bullet \frac{\partial E_x(t)}{\partial t} \tag{101}$$

$$\frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x(\omega) \circ\!\!-\!\!\bullet \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x(t) \tag{102}$$

$$\frac{1}{j\omega} \frac{E_x(\omega)\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} \circ\!\!-\!\!\bullet \int_{-\infty}^{t} \frac{\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} E_x(\tau) d\tau \tag{103}$$

$$\frac{c}{\epsilon_{xx}} C_x^H(\omega) \circ\!\!-\!\!\bullet \frac{c}{\epsilon_{xx}} C_x^H(t) \tag{104}$$

$$\frac{1}{j\omega} \frac{c}{\epsilon_{xx}} \frac{\sigma_x^{'}}{\epsilon_0} C_x^H(\omega) \circ\!\!-\!\!\bullet \int_{-\infty}^{t} \frac{c\sigma_x^{'}}{\epsilon_{xx}\epsilon_0} C_x^H(\tau) d\tau \tag{105}$$

Daraus ergibt sich für I:

$$\frac{\partial E_x(t)}{\partial t} + \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x(t) + \int_{-\infty}^{t} \frac{\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} E_x(\tau) d\tau = \frac{c}{\epsilon_{xx}} C_x^H(t) + \int_{-\infty}^{t} \frac{c\sigma_x^{'}}{\epsilon_{xx}\epsilon_0} C_x^H(\tau) d\tau \tag{106}$$

Für II ist die Korrespondenztabelle:

$$j\omega H_x(\omega) \circ\!\!-\!\!\bullet \frac{\partial H_x(t)}{\partial t} \tag{107}$$

$$\frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} H_x(\omega) \circ\!\!-\!\!\bullet \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} H_x(t) \tag{108}$$

$$\frac{1}{j\omega} \frac{H_x(\omega)\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} \circ\!\!-\!\!\bullet \int_{-\infty}^{t} \frac{\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} H_x(\tau) d\tau \tag{109}$$

$$-\frac{c}{\mu_{xx}} C_x^E(\omega) \circ\!\!-\!\!\bullet -\frac{c}{\mu_{xx}} C_x^E(t) \tag{110}$$

$$-\frac{1}{j\omega} \frac{c}{\mu_{xx}} \frac{\sigma_x^{'}}{\epsilon_0} C_x^E(\omega) \circ\!\!-\!\!\bullet -\int_{-\infty}^{t} \frac{c\sigma_x^{'}}{\epsilon_{xx}\epsilon_0} C_x^E(\tau) d\tau \tag{111}$$

Daraus ergibt sich für II:

$$\frac{\partial H_x(t)}{\partial t} + \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} H_x(t) + \int_{-\infty}^{t} \frac{\sigma_y^{'}\sigma_z^{'}}{\epsilon_0^2} H_x(\tau) d\tau = -\frac{c}{\mu_{xx}} C_x^E(t) - \int_{-\infty}^{t} \frac{c\sigma_x^{'}}{\epsilon_{xx}\epsilon_0} C_x^E(\tau) d\tau \tag{112}$$

Die transformierten Terme können nun seperat über die FDTD-Methode angenähert werden.

Dadurch ergibt sich für I:

$$\frac{\partial E_x(t)}{\partial t} \approx \frac{E_x^{i,j,k}|_{t+\Delta t} - E_x^{i,j,k}|_t}{\delta t} \tag{113}$$

$$\frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x(t) \approx \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x^{i,j,k}|_t \tag{114}$$

$$\int_{-\infty}^{t} \frac{\sigma_z^{'}\sigma_y^{'}}{\epsilon_0^2} E_x(\tau) d\tau \approx \frac{\sigma_z^{'}\sigma_y^{'}}{\epsilon_0^2} \sum_{T=0}^{t} E_x^{i,j,k}|_T \Delta t \tag{115}$$

$$\frac{c}{\epsilon_{xx} i} C_x^H(t) \approx \frac{c}{\epsilon_{xx}} \left(\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z}\right) \tag{116}$$

$$\int_{-\infty}^{t} \frac{c\sigma_x^{'}}{\epsilon_{xx}\epsilon_0} C_x^H(\tau) d\tau \approx \frac{c\sigma_x^{'}\Delta t}{\epsilon_{xx}\epsilon_0} \sum_{T=0}^{t} \left(\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z}\right) \tag{117}$$

Insgesamt also:

$$\frac{E_x^{i,j,k}|_{t+\Delta t} - E_x^{i,j,k}|_t}{\delta t} + \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} E_x^{i,j,k}|_t + \frac{\sigma_z^{'}\sigma_y^{'}}{\epsilon_0^2} \sum_{T=0}^{t} E_x^{i,j,k}|_T \Delta t = \tag{118}$$

$$\frac{c}{\epsilon_{xx}} \left(\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z}\right) \tag{119}$$

$$+ \frac{c\sigma_x^{'}\Delta t}{\epsilon_{xx}\epsilon_0} \sum_{T=0}^{t} \left(\frac{\widetilde{H_z}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_z}^{i,j-1,k}|_{t+\frac{\Delta t}{2}}}{\Delta y} - \frac{\widetilde{H_y}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_y}^{i,j,k-1}|_{t+\frac{\Delta t}{2}}}{\Delta z}\right) \tag{120}$$

Für II ergibt sich:

$$\frac{\partial H_x(t)}{\partial t} \approx \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} \tag{121}$$

$$\frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} H_x(t) \approx \frac{\sigma_z^{'} + \sigma_y^{'}}{\epsilon_0} \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} + \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{2} \tag{122}$$

$$\int_{-\infty}^{t} \frac{\sigma_z^{'}\sigma_y^{'}}{\epsilon_0^2} H_x(\tau)d\tau \approx \frac{\sigma_z^{'}\sigma_y^{'}\Delta t}{\epsilon_0^2}(\sum_{T=\frac{\Delta t}{2}}^{t+\frac{\Delta t}{2}} \widetilde{H_x}^{i,j,k}|_T \tag{123}$$

$$-\frac{c}{\mu_{xx}} C_x^E(t) \approx -\frac{c}{\mu_{xx}}(\frac{E_z^{i,j,k}|_{t+\Delta t} - E_z^{i,j-1,k}|_{t+\Delta t}}{\Delta y} - \frac{E_y^{i,j,k}|_{t+\Delta t} - E_y^{i,j,k-1}|_{t+\Delta t}}{\Delta z}) \tag{124}$$

$$-\int_{-\infty}^{t} \frac{c\sigma_x^{'}}{\mu_{xx}\epsilon_0} C_x^E(\tau)d\tau \approx -\frac{c\sigma_x^{'}\Delta t}{\mu_{xx}\epsilon_0} \sum_{T=0}^{t}(\frac{E_z^{i,j,k}|_t - E_z^{i,j-1,k}|_t}{\Delta y} - \frac{E_y^{i,j,k}|_t - E_y^{i,j,k-1}|_t}{\Delta z}) \tag{125}$$

Betrachtet man die Gleichung (123) so fällt auf, dass die magnetischen Feldstärkewerte bis zum Zeitpunkt $t+\frac{\Delta t}{2}$ aufsummiert werden, es wird einen halben Zeitschritt zu weit integriert. Das Problem wird gelöst, indem man bis zum Zeitpunkt $t-\frac{\Delta t}{2}$ integriert und den letzten halben Zeitschritt interpoliert.

$$\begin{aligned}
&\int_{-\infty}^{t} \frac{\sigma_z^{'}\sigma_y^{'}}{\epsilon_0^2} H_x(\tau)d\tau \approx \frac{\sigma_z^{'}\sigma_y^{'}\Delta t}{\epsilon_0^2}(\sum_{T=\frac{\Delta t}{2}}^{t+\frac{\Delta t}{2}} \widetilde{H_x}^{i,j,k}|_T \\
&= \frac{\sigma_z^{'}\sigma_y^{'}\Delta t}{\epsilon_0^2} \sum_{T=\frac{\Delta t}{2}}^{t-\frac{Deltat}{2}} \widetilde{H_x}^{i,j,k}|_T + \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} + \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{2} \frac{\Delta t}{2} \\
&= \frac{\sigma_z^{'}\sigma_z^{'}\Delta t}{\epsilon_0^2}(\sum_{T=\frac{\Delta t}{2}}^{t-\frac{Deltat}{2}} \widetilde{H_x}^{i,j,k}|_T + \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} + \widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}}}{4})
\end{aligned} \tag{126}$$

Insgesamt ergibt sich also:

$$\begin{aligned}
&\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} - \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{\Delta t} + \frac{\overset{\backsim}{\sigma_z} + \overset{\backsim}{\sigma_y}}{\epsilon_0}\frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} + \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{2} \\
&+ \frac{\overset{\backsim}{\sigma_z}\overset{\backsim}{\sigma_y}\Delta t}{\epsilon_0^2}(\sum_{T=\frac{\Delta t}{2}}^{t-\frac{\Delta t}{2}} \widetilde{H_x}^{i,j,k}|_T + \frac{\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} + \widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}}}{4}) \\
&= -\frac{c}{\mu_{xx}}(\frac{E_z^{i,j,k}|_{t+\Delta t} - E_z^{i,j-1,k}|_{t+\Delta t}}{\Delta y} - \frac{E_y^{i,j,k}|_{t+\Delta t} - E_y^{i,j,k-1}|_{t+\Delta t}}{\Delta z}) \\
&- \frac{c\overset{\backsim}{\sigma_x}\Delta t}{\mu_{xx}\epsilon_0}\sum_{T=0}^{t}(\frac{E_z^{i,j,k}|_t - E_z^{i,j-1,k}|_t}{\Delta y} - \frac{E_y^{i,j,k}|_t - E_y^{i,j,k-1}|_t}{\Delta z}) \qquad (127)
\end{aligned}$$

Löst man nun I nach $E_x^{i,j,k}|_{t+\Delta t}$ und II nach $\widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}}$ auf, erhält man die folgenden Updategleichungen:

$$\begin{aligned}
\text{I: } E_x^{i,j,k}|_{t+\Delta t} &= E_x^{i,j,k}|_t(1 - \frac{\Delta t(\overset{\backsim}{\sigma_z} + \overset{\backsim}{\sigma_y})}{\epsilon_0}) - \frac{\overset{\backsim}{\sigma_z}\overset{\backsim}{\sigma_y}\Delta t^2}{\epsilon_0^2}\sum_{T=0}^{t} E_x^{i,j,k}|_T \\
&+ \frac{c\Delta t}{\epsilon_{xx}} C_x^H|_{t+\frac{\Delta t}{2}}^{i,j,k} + \frac{c\overset{\backsim}{\sigma_x}\Delta t^2}{\epsilon_{xx}\epsilon_0}\sum_{T=0}^{t} C_x^H|_{T+\frac{\Delta t}{2}}^{i,j,k} \qquad (128) \\
\text{II: } \widetilde{H_x}^{i,j,k}|_{t+\frac{\Delta t}{2}} &= \frac{-\frac{\overset{\backsim}{\sigma_y}+\overset{\backsim}{\sigma_z}}{2\epsilon_0} + \frac{\overset{\backsim}{\sigma_y}\overset{\backsim}{\sigma_z}\Delta t}{4\epsilon_0} + \frac{1}{\Delta t}}{\frac{\overset{\backsim}{\sigma_y}+\overset{\backsim}{\sigma_z}}{2\epsilon_0} + \frac{\overset{\backsim}{\sigma_y}\overset{\backsim}{\sigma_z}\Delta t}{4\epsilon_0} + \frac{1}{\Delta t}}\widetilde{H_x}^{i,j,k}|_{t-\frac{\Delta t}{2}} - \frac{\frac{x}{\mu_{xx}}}{\frac{\overset{\backsim}{\sigma_y}+\overset{\backsim}{\sigma_z}}{2\epsilon_0} + \frac{\overset{\backsim}{\sigma_y}\overset{\backsim}{\sigma_z}\Delta t}{4\epsilon_0} + \frac{1}{\Delta t}} C_x^E|_t^{i,j,k} \\
&- \frac{\frac{c\overset{\backsim}{\sigma_x}\Delta_t}{\mu_{xx}\epsilon_0}}{\frac{\overset{\backsim}{\sigma_y}+\overset{\backsim}{\sigma_z}}{2\epsilon_0} + \frac{\overset{\backsim}{\sigma_y}\overset{\backsim}{\sigma_z}\Delta t}{4\epsilon_0} + \frac{1}{\Delta t}}\sum_{T=0}^{t} C_x^E|_t^{i,j,k} - \frac{\frac{\overset{\backsim}{\sigma_y}\overset{\backsim}{\sigma_z}\Delta t}{\epsilon_0^2}}{\frac{\overset{\backsim}{\sigma_y}+\overset{\backsim}{\sigma_z}}{2\epsilon_0} + \frac{\overset{\backsim}{\sigma_y}\overset{\backsim}{\sigma_z}\Delta t}{4\epsilon_0} + \frac{1}{\Delta t}}\sum_{T=\frac{\Delta t}{2}}^{t-\frac{\Delta t}{2}} \widetilde{H_x}^{i,j,k}|_T \qquad (129)
\end{aligned}$$

Wenn diese Gleichungen (128) und (129) korrekt sind muessen sie sich fuer σ-Terme gleich null zu den Gleichungen (54) und (57) vereinfachen. Das ist gegeben, die bisherige Herleitung ist korrekt. Diese beiden Updategleichungen beschreiben kompakt die Lösung der Maxwellgleichungen mit beliebigen diagonal-anistropen nicht frequenz- und nicht zeitabhängigen Geometrien. Im folgenden Kapitel werden diese Implementiert und anschlieszend genutzt um eine λ-Dipolantenne zu simulieren.

5 Implementierung der FDTD

In dieser Arbeit wurde die FDTD einmal sequentiell in C und auf zwei unterschiedliche Weisen parallelisiert in C und OpenCL implementiert. Die Referenzimplementierung in C wurde genutzt, um die Ergebnisse der parallelen Implementierungen zu überprüfen und die Performanz der Implementierungen vergleichen zu können. Alle Implementierungen wurden für Fließkommazahlen einfacher und doppelter Genauigkeit definiert.

5.1 OpenCL

OpenCL ist eine von der Khronos Group standardisierte Architektur für parallele Programmierung. [Group()] Eine OpenCL Applikation besteht aus einem Hostprogramm, der auf der CPU ausgeführt wird und einem oder mehrerer sog. Kernel, die parallelisiert auf der Beschleunigerhardware ausgeführt werden. [Munshi et al.(2011)Munshi, Gaster, Mattson, Fung, and Ginsburg, p.13] In der OpenCL Architektur existiert immer nur ein Host, der mit einem oder mehreren 'Compute Devices' verbunden ist. Ein Compute Device kann eine CPU, GPU oder ein Digitaler Signal Prozessor sein. Ein Compute Device ist unterteilt in sog. Compute Units, die in sog. Processing Elements unterteilt sind, dort werden die Kernel tatsächlich ausgeführt. [Munshi et al.(2011)Munshi, Gaster, Mattson, Fung, and Ginsburg, p.12] Möchte man einen Algorithmus in OpenCL implementieren so muss ein Kernel in der gleichnamigen Sprache OpenCL implementiert werden und ein globaler Arbeitsbereich definiert werden. Dieser globale Arbeitsbereich ist ein-, zwei-, oder dreidimensional und kann in lokale Arbeitsbereiche unterteilt werden. Lokale Arbeitsbereiche werden parallel von Compute Units ausgeführt, die ihrerseits Kernel konkurrierend auf ihren Processing Units ausführt. Eine Instanz eines lokalen Arbeitsbereichs nennt man Workgroup, die Instanz eines Kernels nennt man Workitem. Jedem Workitem ist bekannt zu welcher Workgroup er gehört und an welcher Stelle er im globalen Problembereicht er steht. Da Workgroups auf getrennten Compute Units ausgeführt werden ist keine Synchronisation unter Kerneln unterschiedlicher Workgroups möglich.

5.2 Implementieren der Dirichlet-Randbedingung

Die Dirichlet-Randbedingung löst den Spezialfall für Zellen am Rand des simulierten Raums. Mathematisch vereinfachen sich dort die Updategleichungen, da Feldstärkewerte außerhalb des Simulationsbereichs als null angenommen werden. Für die Implementierung der Dirichlet-Randbedingung gibt es sequentiell und parallel mehrere Möglichkeiten, welche innerhalb der Arbeit detaillierter behandelt werden, da sie die Form der letztendlichen Implementierung beeinflussten. Sequentiell kann die Dirichlet-Randbedingung implementiert werden indem die Berrechnung der Randfeldstärkewerte ausgelagert und die vereinfachten Formeln genutzt werden. Das bietet sich für eine parallele Implementierung nicht an, da entweder für jeden Rand ein eigener Kernel implementiert und aufgerufen werden müsste, was umständlich ist, oder aber eine Kontrollstruktur genutzt werden müsste die Randfälle erkennt und behandelt, was unperformant ist. Eine weitere Möglichkeit ist es in der sequentiellen Implementierung die Iteration über den Simulationsraum nur von dem Index $(1, 1, 1)$ bis $(N_x - 1, N_y - 1, N_z - 1)$ laufen zu lassen. Somit ist der tatsächliche Simulationsraum in jede Dimension um zwei kleiner, da der Rand nicht mit simuliert wird. Die Feldstärkewerte am Rand bleiben bei ihrem Intialisierungswert null, wodurch die Dirichlet-Randbedingung implizit eingehalten wird. Dieses Verfahren kann ähnlich auch in der parallelen Implementierung genutzt werden. Dafuer muss der globale Arbeitsbereich in jeder Dimension als zwei kleiner definiert werden, als das zugrundeliegende Feldstärkearray. Innerhalb des OpenCL-Kernels kann nun der globale Index mit einem Offset von eins für jede Dimension errechnet werden. Dadurch bleiben die Ränder der Feldstärkewerte unsimuliert und die Dirichlet-Randbedingung ist implizit implementiert. Zwischenzeitlich erschien es auch möglich die Dirichlet-Randbedingung durch die OpenCL-Texturroutinen implementieren zu können. Dafür hätten die alten und neün E- und H-Felder als Texturen mit einem RGB-Farbkanal und einem float-Pixelformat dargestellt werden müssen. Für den Zugriff auf Texturen muss in OpenCL ein sog. Sampler definiert werden dem CLK_ADDRESS_CLAMP als Parameter übergeben werden kann. Das veranlasst den Sampler bei Zugriffen auf Texturen außerhalb des Texturbereichs einen definierten Randwert zurückzugeben. Standartmäßig ist dieser als $(0, 0, 0)$ definiert, was exakt der Dirichlet-Randbedingung entspräche. Leider sind in OpenCL 1.1 Texturen als read- oder write-only definiert, wodurch sie nicht für die Implementierung der FDTD genutzt werden können, da lesend und schreibend auf die Feldstärkewerte zugegriffen wird. Aufgrund dieser Überlegungen wurde sich innerhalb dieser Arbeit dafür entschieden

den Simulationsraum künstlich zu verkleinern.

5.3 Implementierung für einen Zeitschritt

Die erste, leichtere und wie sich herausstellen soll weniger performante Implementierungsmöglichkeit lagert nur die Berrechnung eines Zeitschrittes auf die GPU aus. Für jeden Zeitschritt werden die Daten des E- und H-Feldes an die GPU gesendet, berrechnet und das Ergebnis gelesen. Danach kann hostseitig eine oder mehrere Quellen injiziert werden und ein weiterer Zeitschritt berrechnet werden. Das hat den Vorteil, dass Quellen flexibler gestaltet werden können und dass die Feldstärken für jeden Zeitschritt bekannt sind. Eine Visualisierung in einem Video wäre damit möglich. Die optimale Größe des lokalen Problembereichs kann von OpenCL bestimmt werden und da keine Synchronisation nötig ist, können alle Compute Units verwendet werden. Als nachteilig stellte sich heraus, dass für jeden Zeitschritt Daten von und auf die GPU kopiert werden müssen was im Vergleich zu Rechenoperationen ein langsamer Prozess ist. Der Kernelcode ist hier nur für Fließkommazahlen einfacher Genauigkeit angegeben:

```
1  // Get the actual id
2  const int id = get_global_id(0) + get_global_id(1) * uwidth +
       get_global_id(2) * uwidth * udepth + 1 + uwidth + uwidth *
       udepth;
3
4  // Calculate the current magnetic field from the old electric
       field
5  float hx_curl_term = ((old_e[id + uwidth].z - old_e[id].z) /
       grid_width_y) - ((old_e[id + uwidth * udepth].y - old_e[id].y
       ) / grid_width_z);
6  float hy_curl_term = ((old_e[id + uwidth * udepth].x - old_e[id
       ].x) / grid_width_z) - ((old_e[id + 1].z - old_e[id].z) /
       grid_width_x);
7  float hz_curl_term = ((old_e[id + 1].y - old_e[id].y) /
       grid_width_x) - ((old_e[id + uwidth].x - old_e[id].x) /
       grid_width_y);
8  h_curl_integral[id].x += hx_curl_term;
9  h_curl_integral[id].y += hy_curl_term;
10 h_curl_integral[id].z += hz_curl_term;
11 h_field_integral[id] += old_h[id];
12 current_h[id].x = + hx_factor[id].x * old_h[id].x - hx_factor[id
       ].y * hx_curl_term
13 - hx_factor[id].z * h_curl_integral[id].x - hx_factor[id].w *
       h_field_integral[id].x;
14 current_h[id].y = + hy_factor[id].x * old_h[id].y - hy_factor[id
       ].y * hy_curl_term
```

```
15 - hy_factor[id].z * h_curl_integral[id].y - hy_factor[id].w *
       h_field_integral[id].y;
16 current_h[id].z = + hz_factor[id].x * old_h[id].z - hz_factor[id
       ].y * hz_curl_term
17 - hz_factor[id].z * h_curl_integral[id].z - hz_factor[id].w *
       h_field_integral[id].z;
18
19 // Calculate the current electric field from the current
       magnetic field
20 float ex_curl_term = ((current_h[id].z - current_h[id - uwidth].
       z) / grid_width_y) - ((current_h[id].y - current_h[id -
       uwidth * udepth].y) / grid_width_z);
21 float ey_curl_term = ((current_h[id].x - current_h[id - uwidth *
        udepth].x) / grid_width_z) - ((current_h[id].z - current_h[
       id - 1].z) / grid_width_x);
22 float ez_curl_term = ((current_h[id].y - current_h[id - 1].y) /
       grid_width_x) - ((current_h[id].x - current_h[id - uwidth].x)
        / grid_width_y);
23 e_curl_integral[id].x += ex_curl_term;
24 e_curl_integral[id].y += ey_curl_term;
25 e_curl_integral[id].z += ez_curl_term;
26 e_field_integral[id] += old_e[id];
27 current_e[id].x = ex_factor[id].x * old_e[id].x - ex_factor[id].
       y * e_field_integral[id].x
28 + ex_factor[id].z * ex_curl_term + ex_factor[id].w *
       e_curl_integral[id].x;
29 current_e[id].y = ey_factor[id].x * old_e[id].y - ey_factor[id].
       y * e_field_integral[id].y
30 + ey_factor[id].z * ey_curl_term + ey_factor[id].w *
       e_curl_integral[id].y;
31 current_e[id].z = ez_factor[id].x * old_e[id].z - ez_factor[id].
       y * e_field_integral[id].z
32 + ez_factor[id].z * ez_curl_term + ez_factor[id].w *
       e_curl_integral[id].z;
33 current_e[id] *= geometry[id];
```

In Zeile 2 wird die globale X- Y- und Z-Position erfragt, um den Index des Workitems in den globalen Feldstärkearrays zu errechnen. Durch den Offset wird sichergestellt, dass der linksseitige Rand der Feldstärkearrays nicht mitsimuliert wird. Dieser Offset ist die oben erwähnte implizite Implementierung der Dirichlet-Randbedingung. Die numerische Integration der Gleichungen (128) und (129) passiert in den Zeilen 8 bis 11 und 23 bis 26. Die X-Komponente des H-Feldes wird nach der Gleichung (129) in Zeile 12 und 13 errechnet, die Zeilen 14 bis 17 beinhalten die Formeln für die Y- und Z-Komponente des H-Feldes. Die X-Komponente des E-Feldes wird nach der Gleichung (128) in Zeile 29 errechnet, die Zeilen 28 und 29 beinhalten die Berrechnung der Y- und Z-Komponente des E-Feldes. Der eingeführt

Geometriefaktor m wird in Zeile 33 genutzt, um das E-Feld für Zellen mit Geometrie, d.h. $m = 0$, auf null zu setzen.

5.4 Implementierung für mehrere Zeitschritte

Sollen gleich mehrere Zeitschritte auf der GPU verarbeitet werden, müssen die Kernel synchronisiert werden, um sicherzustellen, dass die Felder für einen Zeitschritt schon vollständig berrechnet wurden, bevor die nächste Iteration beginnt. Weiterhin muss die Quelle innerhalb des Kernels implementiert sein. Da eine Synchronisation nur zwischen Kerneln einer Workgroup möglich ist, darf nur eine Workgroup existieren. Das bedeutet, dass der lokaler Problembereich und der globale Problembereich gleichgroß sein müssen. Die innerhalb der Arbeit zu Verfügung stehende GPU hatte eine maximale Workgroupsize von 256. Demzufolge können nur 256 Kernel verwendet werden. Ein dreidimensionaler Problembereich kann somit nur $256^{(1/3)} = 6.349 \approx 6 \Rightarrow 6^3$ groß sein. Da es möglich sein soll einen Bereich von mehr als 6^3 Zellen zu simulieren, muss ein Kernel die Berrechnung für mehrere Zellen durchführen. Das führt zu folgendem Code, der auch hier nur für Fließkommazahlen einfacher Genauigkeit angegeben ist.

```
1  // get the actual id
2  const int  ID = get_global_id(0) * batch_size_x + get_global_id
       (1) * batch_size_y * uwidth + get_global_id(2) *
       batch_size_z * uwidth * udepth + 1 + uwidth + uwidth *
       udepth;
3
4  int id = ID;
5
6  for (float current_time = 0; current_time < max_time;
       current_time += time_step)
7  {
8   for (int z = 0; z < batch_size_z; z++)
9   {
10   for (int y = 0; y < batch_size_y; y++)
11   {
12    for (int x = 0; x < batch_size_x; x++)
13    {
14     id = ID + x + y * uwidth + z * uwidth * udepth;
15     // calculate the current magnetic field from the old
           electric field
16     float hx_curl_term = ((old_e[id + uwidth].z - old_e[id].z)
           / grid_width_y) - ((old_e[id + uwidth * udepth].y -
           old_e[id].y) / grid_width_z);
17     float hy_curl_term = ((old_e[id + uwidth * udepth].x -
           old_e[id].x) / grid_width_z) - ((old_e[id + 1].z - old_e
           [id].z) / grid_width_x);
```

```
18        float hz_curl_term = ((old_e[id + 1].y - old_e[id].y) /
              grid_width_x) - ((old_e[id + uwidth].x - old_e[id].x) /
              grid_width_y);
19        h_curl_integral[id].x += hx_curl_term;
20        h_curl_integral[id].y += hy_curl_term;
21        h_curl_integral[id].z += hz_curl_term;
22        h_field_integral[id] += old_h[id];
23        current_h[id].x = + hx_factor[id].x * old_h[id].x -
              hx_factor[id].y * hx_curl_term
24          - hx_factor[id].z * h_curl_integral[id].x - hx_factor[id
                ].w * h_field_integral[id].x;
25        current_h[id].y = + hy_factor[id].x * old_h[id].y -
              hy_factor[id].y * hy_curl_term
26          - hy_factor[id].z * h_curl_integral[id].y - hy_factor[id
                ].w * h_field_integral[id].y;
27        current_h[id].z = + hz_factor[id].x * old_h[id].z -
              hz_factor[id].y * hz_curl_term
28          - hz_factor[id].z * h_curl_integral[id].z - hz_factor[id
                ].w * h_field_integral[id].z;
29      }
30     }
31    }
32
33    barrier(CLK_LOCAL_MEM_FENCE | CLK_GLOBAL_MEM_FENCE);
34
35    // calculate the current electric field from the current
          magnetic field
36    for (int z = 0; z < batch_size_z; z++)
37    {
38     for (int y = 0; y < batch_size_y; y++)
39     {
40      for (int x = 0; x < batch_size_x; x++)
41      {
42       id = ID + x + y * uwidth + z * uwidth * udepth;
43       float ex_curl_term = ((current_h[id].z - current_h[id -
             uwidth].z) / grid_width_y) - ((current_h[id].y -
             current_h[id - uwidth * udepth].y) / grid_width_z);
44       float ey_curl_term = ((current_h[id].x - current_h[id -
             uwidth * udepth].x) / grid_width_z) - ((current_h[id].z
             - current_h[id - 1].z) / grid_width_x);
45       float ez_curl_term = ((current_h[id].y - current_h[id - 1].
             y) / grid_width_x) - ((current_h[id].x - current_h[id -
             uwidth].x) / grid_width_y);
46       e_curl_integral[id].x += ex_curl_term;
47       e_curl_integral[id].y += ey_curl_term;
48       e_curl_integral[id].z += ez_curl_term;
49       e_field_integral[id] += old_e[id];
50       current_e[id].x = ex_factor[id].x * old_e[id].x - ex_factor
             [id].y * e_field_integral[id].x
```

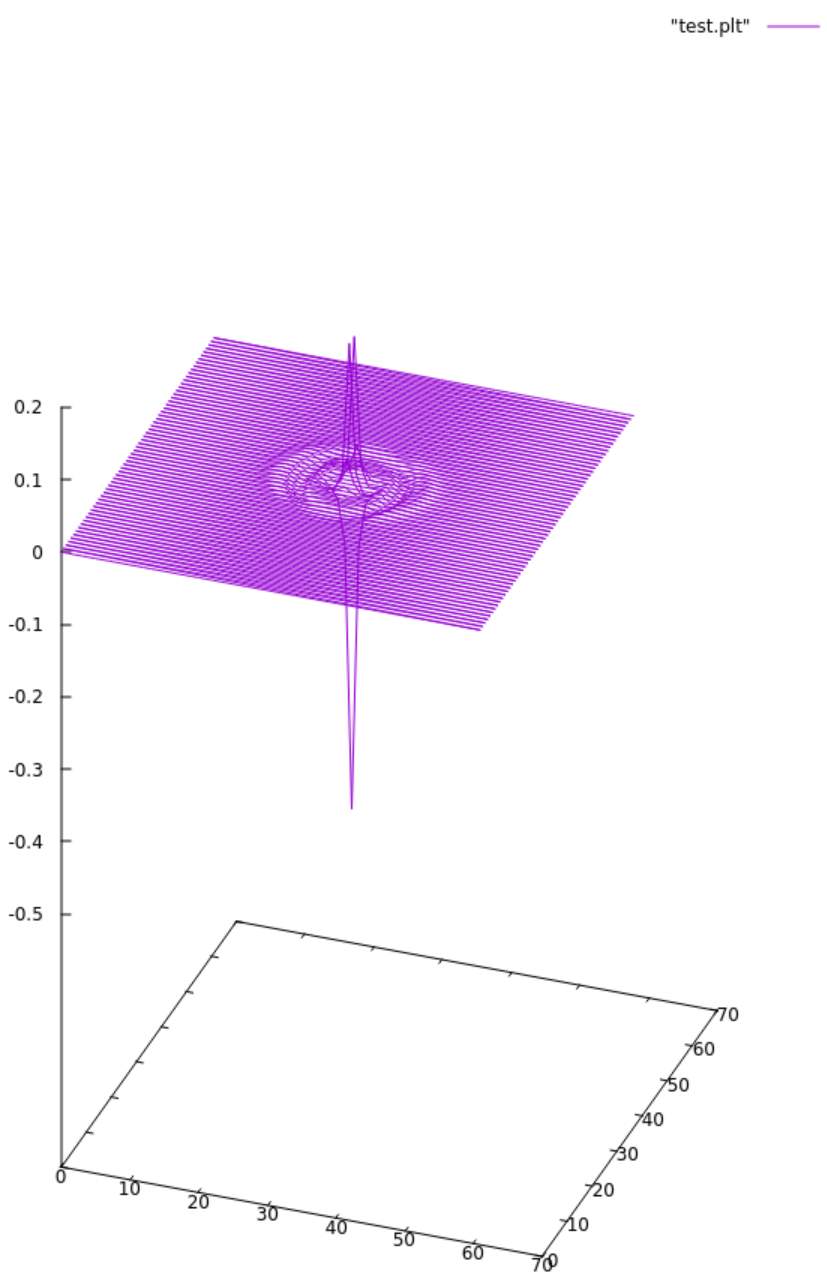
```
51          + ex_factor[id].z * ex_curl_term + ex_factor[id].w *
                e_curl_integral[id].x;
52        current_e[id].y = ey_factor[id].x * old_e[id].y - ey_factor
              [id].y * e_field_integral[id].y
53          + ey_factor[id].z * ey_curl_term + ey_factor[id].w *
                e_curl_integral[id].y;
54        current_e[id].z = ez_factor[id].x * old_e[id].z - ez_factor
              [id].y * e_field_integral[id].z
55          + ez_factor[id].z * ez_curl_term + ez_factor[id].w *
                e_curl_integral[id].z;
56
57        // zero ez out if the soure_factor is 1
58        current_e[id].z -= abs(source_factor[id]) * current_e[id].z
              ;
59        // than add the source. this way a hard source is created!
60        current_e[id].z += source_factor[id] * source_amplitude *
              ramped_sin_fp32(current_time, source_ramp_len,
              source_frequency);
61        current_e[id] *= geometry[id];
62      }
63     }
64    }
65
66    // synchronize the workitems after calculating the current
          fields
67    barrier(CLK_LOCAL_MEM_FENCE | CLK_GLOBAL_MEM_FENCE);
68
69    // now memory consistency is assured and the fields can be
          copied
70    for (int z = 0; z < batch_size_z; z++)
71    {
72     for (int y = 0; y < batch_size_y; y++)
73     {
74      for (int x = 0; x < batch_size_x; x++)
75      {
76       id = ID + x + y * uwidth + z * uwidth * udepth;
77       old_h[id] = current_h[id];
78       old_e[id] = current_e[id];
79      }
80     }
81    }
82    barrier(CLK_LOCAL_MEM_FENCE | CLK_GLOBAL_MEM_FENCE);
83   }
```

In Zeile 2 wird wieder der Index für die Feldstärkearrays berrechnet. Da nun jedoch nicht für jede Zelle ein Workitem existiert, sondern ein Workitem mehrere Zellen berrechnet, muss die Menge der zu errechnenden Zellen mit einbezogen werden, um den Index der Startzelle auf dem Feldstärkearray

zu ermitteln. Von dort aus läuft das Workitem über einen Kubus von Zellen und berrechnet das H-Feld. Bevor nun das E-Feld berrechnet werden kann, muss sichergestellt werden, dass das H-Feld für jede Zelle innerhalb des Simulationsbereichs errechnet worden ist. Das geschieht durch den Befehl 'barrier' in Zeile 33, der angibt, dass kein Workitem das Programm weiter ausführt, bis nicht alle Workitems Zeile 33 erreicht haben. Der Parameter 'CLK_LOCAL_MEM_FENCE | CLK_GLOBAL_MEM_FENCE' gibt weiterhin an, dass alle Speicherzugriffe auf den lokalen und globalen Speicher abgeschlossen sein müssen. Gleiches tut die Barriere in Zeile 67 für das E-Feld. Haben alle Workitems das neue E- und H-Feld berrechnet, können die alten Feldstärkewerte überschrieben werden und ein neuer Zeitschritt berrechnet werden. Die Barriere in Zeile 82 stellt sicher, dass alle Workitems synchron die neue Iteration beginnen. Die Quelle wurde durch einen Quellenfaktorarray implementiert das jeder Zelle einen Quellenfaktor zuweist. Dadurch ist es möglich, ohne die Verwendung einer if-Abfrage die elektrischen Feldstärkewerte von Zellen mit einer Quelle auf null zusetzen, um dann den aktuellen Wert des Quellensignals aufzuaddieren,.Dies geschieht in Zeile 58 bis 60. Kontrollstrukturen wie if-Abfragen werden in optimiertem Code vermieden, da sie im Vergleich zu mathematischen Operationen langsam sind. Der Kode der Funktion 'ramped_sin_fp32' ist:

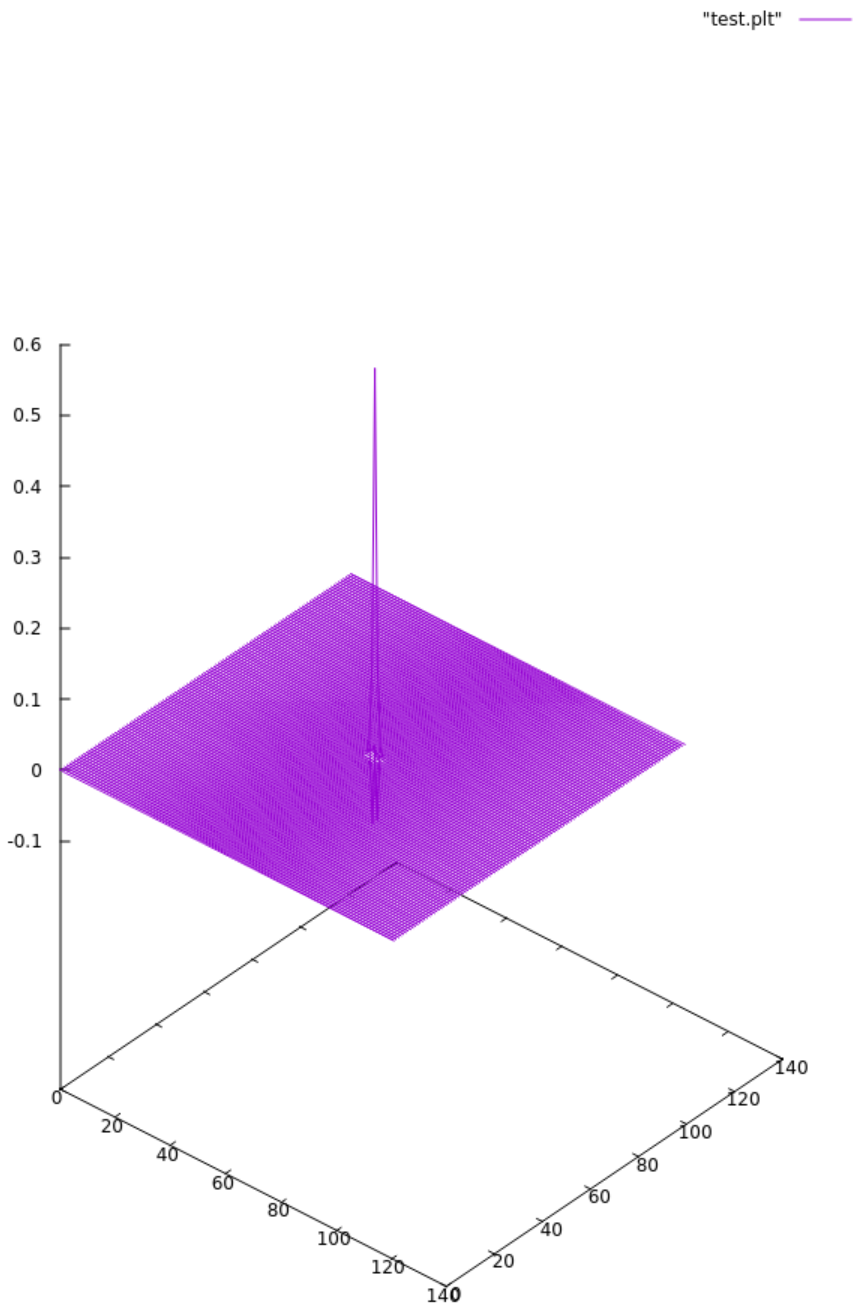
```
1 inline float ramped_sin_fp32(float t, float ramp_len, float
      frequency)
2 {
3         return smoothstep(0, ramp_len, t) * sin(2.0 * M_PI *
              frequency * t);
4 }
```

Da jedes Workitem die gleiche Anzahl an Zellen berrechnet und in jeder Dimension sechs Workitems plaziert sind, muss der zu simulierende Bereich in jede Dimension ein vielfaches von 6 lang sein. Die zusätzlichen Zellen für die Dirichlet-Randbedingung wurden hierbei nicht betrachtet. Ist das nicht gegeben, kann es zu illegalen Speicherzugriffen kommen. Leider liefert die Implementierung bis jetzt nur für gewissen Simulationsbereichsgrößen korrekte Werte, so zum Beispiel für ein Simulationsbereich von 62**3. Zum Vergleich sind hier daher ein korrektes und ein fehlerhaftes Simulationsergebnis angegeben:



Source: ./main –usize 62 –samplerate_space 20 –samplerate_time 10 –max_time 200 –ramped_sin_length 10 –ramped_sin_amplitude 1 –ramped_sin_frequency 2400000000

Figure 1: Korrekter XY-Querschnit des E_z-Feldes eines isotropen Rundstrahlers



Source: ./main –usize 122 –samplerate_space 20 –samplerate_time 10 –max_time 200 –ramped_sin_length 10 –ramped_sin_amplitude 1 –ramped_sin_frequency 2400000000

Figure 2: Fehlerhafter XY-Querschnit des E_z-Feldes eines isotropen Rundstrahlers

Da nur der Parameter usize verändert wurde, sollte sich die Form des E_z-Feldes nicht verändern.

6 Benchmarking

Die Laufzeiten der sechs Implementierungen wurden für die selben Parameter mit dem Programm time gemessen. Das gibt eine reale Laufzeit, sowie die Laufzeit des Prozesses im User- und im Kernelland an. Da die Berrechnung auf die Graphikkarte ausgelagert wurden, ist nur die reale Laufzeit aussagekräftig. Alle sechs Impementierungen wurden mit den selben Parametern aufgerufen:

```
./main --usize 62 --samplerate_space 10 --samplerate_time 20
--max_time 10000 --ramped_sin_length 100 --ramped_sin_amplitude 1
--ramped_sin_frequency 2400000000
```

Das führte zu folgenden Laufzeiten:

Fließkommagenauigkeit	CPU	GPU ein Zeitschritt	GPU alle Zeitschritte
float	11429100us	11353us	701052us
double	11348500us	91852us	1305535us

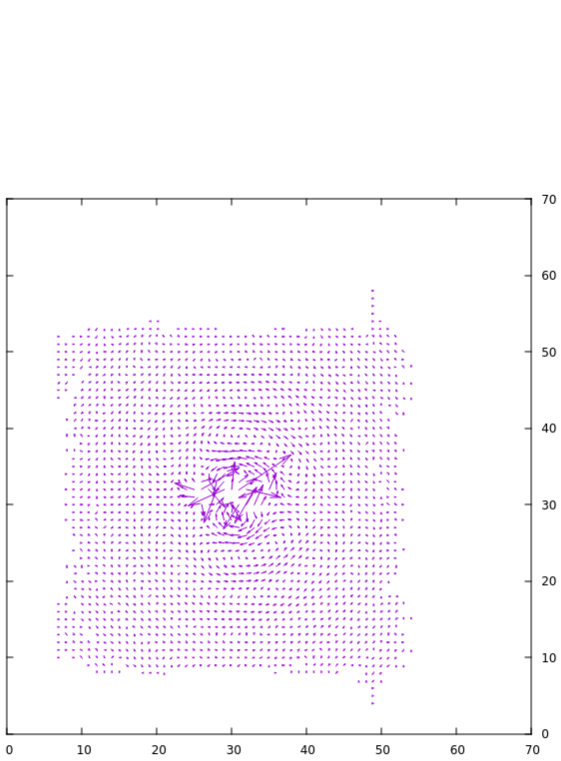
Daraus lässt sich der Performanzgewinn errechnen

Fließkommagenauigkeit	CPU/GPU_{ein}	CPU/GPU_{alle}	GPU_{ein}/GPU_{alle}
float	16.302784957463924	1006.7030740773364	61.750374350391965
double	8.692604947397044	123.55201846448635	14.213462962156513

Wird die Berrechnung aller Zeitschritte auf die GPU ausgelägert und Fließkommazahlen einfacher Genauigkeit verwendet ist die Simulation ungefaher um den Faktor 1006 beschleunigt. Sollen dagegen Fließkommazahlen doppelter Genauigkeit verwendet werden, so ist sie nur ungefähr um den Faktor 123 beschleunigt. Fließkommazahlen doppelter Genauigkeit sind auf GPUs immernoch deutlich schlechter unterstützt.

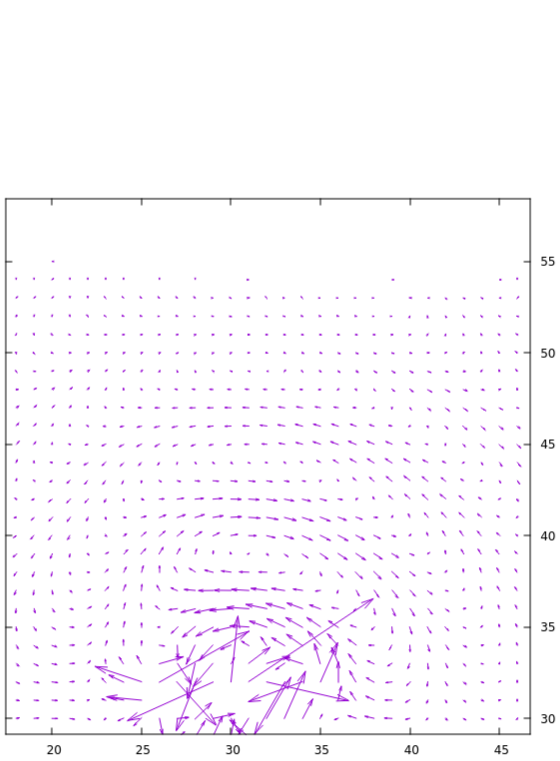
7 Simulation einer λ-Dipolantenne

Eine λ-Dipolantenne ist ein Dipolantenne mit einer Drahtlänge gleich der Wellenlänge der Sende- und Empfangsfrequenz. Bei einer symmetrischen Einspeisung weist sie eine keulenförmige Richtcharakteristik auf. Die Gittergröße wurde, wie weiter oben besprochen, so gewählt, dass die kleinste Wellenlänge noch N_λ mal abgetastet wird. Das bedeutet um eine λ-Dipolantennengeometrie zu setzen muss der Geometriefaktor von genau N_λ Zellen null sein. Der Simulationsbereich ist so groß zu wählen, dass sich das PML nicht im Nahfeld der Antenne befindet, da das die Abstrahlcharakteristik verfälschen würde. Der Übergang von Nah- zu Fernfeld ist dabei fließend, für gewöhnlich wird angenommen, dass das Fernfeld im Radius von der dreifachen Wellenlänge anfängt. Die Lücke zwischen den beiden Armen des λ-Dipols wurde nicht mitsimuliert, da sich die Antennengeometrie nicht symmetrisch teilen ließ. Die Einspeisung des λ-Dipols passierte durch eine E_x-Gap Source direkt unterhalb der Mitte der Antennengeometrie, da so die Einspeisung durch ein Koaxialkabel hinreichend gut abbildet wird.



Source: $./main--usize62--samplerate\_space20--samplerate\_time10--max\_time1000\ -\ -ramped\_sin\_length10\ -\ -ramped\_sin\_amplitude1\ --ramped\_sin\_frequency2400000000 > extest.plt2 > ehtest.plt$

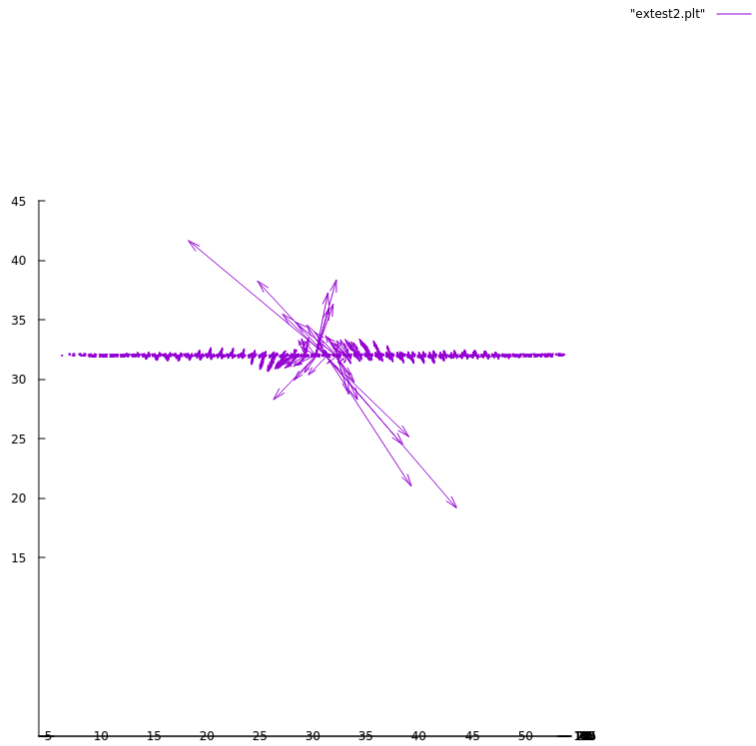
Figure 3: Das E-Feld einer λ-Dipolantenne bei symmetrischer E_x-Einspeisung



Source: $./main --usize 62 --samplerate\_space 20 --samplerate\_time 10 --max\_time 1000 - -ramped\_sin\_length 10 - -ramped\_sin\_amplitude 1 - -ramped\_sin\_frequency 2400000000 > extest.plt 2 > ehtest.plt$

Figure 4: Das E-Feld einer λ-Dipolantenne bei symmetrischer E_x-Einspeisung (vergrößer)

Das E-Feld wird in radial von der λ-Dipolantenne abgestrahlt. Die Richtcharakteristik lässt sich erahnen.



Source: $./main --usize 62 --samplerate\_space 20 --samplerate\_time 10 --max\_time 1000 - -ramped\_sin\_length 10 - -ramped\_sin\_amplitude 1 - -ramped\_sin\_frequency 2400000000 > extest.plt 2 > ehtest.plt$

Figure 5: Das H-Feld einer λ-Dipolantenne bei symmetrischer E_x-Einspeisung

Auch wenn man die Richtcharakteristik erahnen kann ist eine Einschätzung der Abstrahlcharakteristik mit diesen Simulationsergebnissen noch nicht möglich. Dafür müsste aus dem E- und H-Feld das Poyntingvektorfeld errechnet und in Bezug zum Feld des isotropen Rundstrahler gesetzt werden. Auch das könnte hardwarebeschleunigt implementiert werden.

8 Zusammenfassung und Ausblicke

Die FDTD-Methode kann theoretisch beliebig komplexe Antennegeometrien simulieren, bei gekrümmten Geometrien kommt es jedoch aufgrund des Yee-Gitters zu Ungenauigkeiten. Diese können reduziert werden, indem die räumliche Abtastrate erhöht wird, oder aber Techniken wie 'Dielectrical Smoothing' verwendet werden. [Rumpf(t)]. Sollen ebenfalls nichtmetallische Strukturen wie zum Beispiel das Gehäuse der Antenne mitsimuliert werden, so kann es ebenfalls aufgrund des Yee-Gitters zu Ungenauigkeiten kommen. Da das E-Gitter und das H-Gitter um die Hälfte der Gittergröße verschoben sind, kann es sein, das Zellen gleichen Indices unterschiedliche Materialien repräsentieren. Dieses Problem kann durch die '2xGrid Technique' gelöst werden. Dabei wird die Geometrie auf einem Yee-Gitter doppelter Auflösung definiert und dann in ein Yee-Gitter einfacher Auflösung transformiert. Nutzt man die FDTD-Methode, um unbekannte Antennen zu simulieren, hat man weiterhin das Problem, dass aus einer Simulation nicht geschlossen werden kann, ob sich die Antenne schon eingeschwungen hat. Dieses Problem kann gelöst werden, indem man die Abstrahlcharakteristik errechnet und die Simulationsdauer so lange erhöht, bis sich die Abstrahlcharakteristik nicht mehr ändert. Sollte man die FDTD-Methode als Grundlage für einen evolutionären Algorithmus nutzen, müsste dies automatisiert passieren. Mathematisch hat sich das Lösen der Maxwellgleichungen auf Additionen und Multiplikationen heruntergebrochen, es wäre daher denkbar die FDTD-Methode auf einem FPGA zu implementieren. Jede Zelle des Yee-Gitters könnte durch die Instanz einer Entity beschrieben werden, diese müsste nur mit den sechs benachbarten Instanzen sowie einem Kontrolleur kommunizieren können. Zu Beginn der Simulation würde jede Instanz die PML-Vorfaktoren vom Kontrolleur erhalten, danach könnten alle Instanzen durch einen Takt synchronisiert das H-Feld und E-Feld errechnen und ihre intern gespeicherten alten Feldstärken überschreiben. Da moderne FPGA-Boards neben der programmierbaren Logik meist noch einen Prozessor besitzen, könnte die Interaktion mit dem Benutzer, oder aber eventuell das Zuweisen einer Fitnessfunktion auf dem Prozessor passieren. Abschliesßend lässt sich sagen, die FDTD-Methode ist ein sehr mächtiges Verfahren und in der Lage beliebige Anntennengeometrien zu simulieren, sie ist dabei jedoch auch so rechenintensiv, dass eine sequentielle Implementierung kaum zu verwenden ist.

References

[Rumpf(a)] Prof. Raymond Rumpf. Electormagnetic analzsis using finite-differenze time-domain, a.

[Globus et al.()Globus, Linden, and John] Al Globus, Derek S. Linden, and Jason D. John. Automated antenna design with evolutionary algorithms. "http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.102.9841&rep=rep1&type=pdf".

[R. Kourant(1928)] H. Lewy R. Kourant, K. Friedrichs. Über die partiellen differenzengleichungen der mathematischen physik, 1928.

[Hagness(2005)] Hagness. *Computational Electrodynamics: The Finite-Difference Time-Domain Method, 3rd ed.* Artech House, 2005.

[Steunfeld()] Thomas Steunfeld. Liste numerischer verfahren. "https://mathepedia.de/Numerische$_V erfahren.html$".

[Luebbers and H.S()] R.J Luebbers and Langdon H.S. A simple feed model that reduces time steps needed for fdtd antenna and microstip calculations. "http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.461.3208&rep=rep1&type=pdf".

[Rumpf(b)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", b. Seite 3.

[Rumpf(c)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", c. Seite 4.

[Rumpf(d)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", d. Seite 4.

[Rumpf(e)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", e. Seite 5.

[Rumpf(f)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", f. Seite 9.

[Rumpf(g)] Prof. Raymond Rumpf. Lecture 5 – formulation of 1d fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 5 – Formulation of 1D FDTD.pdf", g. Seite 11.

[Rumpf(h)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", h. Seite 24.

[Rumpf(i)] Prof. Raymond Rumpf. Lecture 4 – electromagnetics and fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 4 – Electromagnetics and FDTD.pdf", i. Seite 26.

[Rumpf(j)] Prof. Raymond Rumpf. Lecture 5 – formulation of 1d fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 5 – Formulation of 1D FDTD.pdf", j. Seite 13.

[Yee(1966)] K. Yee. Numerical solution of inital boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14:302–307, May 1966. doi: 10.1109/TAP.1966.1138693.

[Rumpf(k)] Prof. Raymond Rumpf. Topic 2b building geometries into data arrays. "http://emlab.utep.edu/ee4386$_5$301*cmee/Topic*2*b* − −*BuildingGeometriesintoDataArrays.pdf*", *k. Seite*7.

[Rumpf(l)] Prof. Raymond Rumpf. Topic 2b building geometries into data arrays. "http://emlab.utep.edu/ee5390fdtd/Lecture 6 – Implementation0lf 1D FDTD.pdf", l. Seite 13.

[Rumpf(m)] Prof. Raymond Rumpf. Topic 2b building geometries into data arrays. "http://emlab.utep.edu/ee5390fdtd/Lecture 6 – Implementation0lf 1D FDTD.pdf", m. Seite 18.

[Rumpf(n)] Prof. Raymond Rumpf. Lecture 6 – implementation of 1d fdtd.pdf. "http://emlab.utep.edu/ee5390fdtd/Lecture 6 – Implementation of 1D FDTD.pdf", n. Seite 25.

[Rumpf(o)] Prof. Raymond Rumpf. Implementation of 1d fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 6 – Implementation of 1D FDTD.pdf", o. Youtube Video.

[Rumpf(p)] Prof. Raymond Rumpf. Implementation of one dimensional fdtd. "http://emlab.utep.edu/ee5390fdtd/Lecture 6 – Implementation of 1D FDTD.pdf", p. Seite 10.

[Schneider()] John Schneider. Chapter 11 perfectly matched layer. "https://www.eecs.wsu.edu/ schneidj/ufdtd/chap11.pdf". Seite 307.

[Rumpf(q)] Prof. Raymond Rumpf. The perfectly matched layer (pml). http://emlab.utep.edu/ee5390fdtd/Lecture 13 – The Perfectly Matched Layer.pdf, q.

[Rumpf(r)] Prof. Raymond Rumpf. The perfectly matched layer (pml). "http://emlab.utep.edu/ee5390fdtd/Lecture 13 – The Perfectly Matched Layer.pdf", r. Seite 9.

[Rumpf(s)] Prof. Raymond Rumpf. The perfectly matched layer (pml). "http://emlab.utep.edu/ee5390fdtd/Lecture 13 – The Perfectly Matched Layer.pdf", s. Seite 17.

[Group()] Khronos Group. Opencl overview. "https://www.khronos.org/opencl/".

[Munshi et al.(2011)Munshi, Gaster, Mattson, Fung, and Ginsburg] Aaftab Munshi, Benedic R. Gaster, Timothy G. Mattson, James Fung, and Dan Ginsburg. *OpenCL Programming Guide*. Pearson Education, 2011.

[Rumpf(t)] Prof. Raymond Rumpf. Windowing and grid techniques. "http://emlab.utep.edu/ee5390fdtd/Lecture 12 – Windowing and grid techniques.pdf", t. Seite 20.