بسم الله الرحمن الرحیم

شبیه سازی انتشار موج الکترومغناطیسی به روش FTDT

سیده پریسا داج خوش ۹۲۲۳۰۳۱، سیمین فر سماکوش ۹۲۲۳۰۵۱، فاطمه علی مرادی ۹۲۲۳۰۶۴

۱۶ تیر ۱۳۹۶

۱ مقدمه

دامنه اختلاف زمانی محدود (FTDT) یکی از متداول ترین روش های مفهومی است که از نظر پیاده سازی موج برای حل مشکلات الکترومغناطیس به کار می رود.این روش می تواند با دقتی بالا طیف وسیعی از مشکلات را حل نماید. روش (FTDT) می تواند مسائل پیچیده ای را حل نماید اما از نظر محاسبات عددی نیازمند حل محاسبات پیچیده ای است. راه حل این مسائل نیازمند حجم وسیعی از حافظه و محاسبات زمانی می باشد.

می دانیم انتشار موج الکترومغناطیس شامل دو میدان الکتریکی و مغناطیسی است که در پیش راندن موج موثر است. معادلات و ضرایب به صورت معادله ۱، معادله ۲، معادله ۳و معادله ۴ می باشند.

$$H_x^{q+\frac{1}{2}}\left[m,\ n+\frac{1}{2}\right]=\frac{1-\frac{\sigma_m\Delta_t}{2\mu}}{1+\frac{\sigma_m\Delta_t}{2\mu}}H_x^{q-\frac{1}{2}}\left[m,\ n+\frac{1}{2}\right]-\frac{1}{1+\frac{\sigma_m\Delta_t}{2\mu}}\frac{\Delta_t}{\mu\Delta_y}(E_z^q\left[m,\ n+1\right]-E_z^q\left[m,\ n\right])$$

$$H_y^{q+\frac{1}{2}}\left[m+\frac{1}{2},\ n\right]=\frac{1-\frac{\sigma_m\Delta_t}{2\mu}}{1+\frac{\sigma_m\Delta_t}{2\mu}}H_y^{q-\frac{1}{2}}\left[m+\frac{1}{2},\ n\right]+\frac{1}{1+\frac{\sigma_m\Delta_t}{2\mu}}\frac{\Delta_t}{\mu\Delta_x}(E_z^q\left[m+1,\ n\right]-E_z^q\left[m,\ n\right])$$

$$E_z^{q+1}\left[m,\ n\right]=\frac{1-\frac{\sigma\Delta_t}{2\epsilon}}{1+\frac{\sigma\Delta_t}{2\epsilon}}E_z^q\left[m,\ n\right]+$$

$$\frac{1}{1+\frac{\sigma\Delta_t}{2\epsilon}}\left(\frac{\Delta_t}{\epsilon\Delta_x}\left\{H_y^{q+\frac{1}{2}}\left[m+\frac{1}{2},\ n\right]-H_y^{q+\frac{1}{2}}\left[m-\frac{1}{2},\ n\right]\right.\right.$$

$$\left.\left.-\frac{\Delta_t}{\epsilon\Delta_y}\{H_x^{q+\frac{1}{2}}\left[m+\frac{1}{2},\ n\right]-H_x^{q+\frac{1}{2}}\left[m-\frac{1}{2},\ n\right]\right.\right.$$

$$C_{hxh}\left(m,\ n+1/2\right)=\left.\frac{1-\frac{\sigma_m\Delta_t}{2\mu}}{1+\frac{\sigma_m\Delta_t}{2\mu}}\right|_{m\delta,(n+1/2)\delta},$$

$$C_{hxe}\left(m,\ n+1/2\right)=\left.\frac{1}{1+\frac{\sigma_m\Delta_t}{2\mu}}\frac{\Delta_t}{\mu\delta}\right|_{m\delta,(n+1/2)\delta},$$

$$C_{hyh}\left(m+1/2,\ n\right)=\left.\frac{1-\frac{\sigma_m\Delta_t}{2\mu}}{1+\frac{\sigma_m\Delta_t}{2\mu}}\frac{\Delta_t}{\mu\delta}\right|_{(m+1/2)\delta,n\delta},$$

$$C_{hye}\left(m+1/2,\ n\right)=\left.\frac{1}{1+\frac{\sigma_m\Delta_t}{2\mu}}\frac{\Delta_t}{\mu\delta}\right|_{(m+1/2)\delta,n\delta},$$

$$C_{eze}\left(m,\ n\right)=\left.\frac{1-\frac{\sigma\Delta_t}{2\epsilon}}{1+\frac{\sigma\Delta_t}{2\epsilon}}\right|_{m\delta,n\delta},$$

$$C_{ezh}\left(m,\ n\right)=\left.\frac{1}{1+\frac{\sigma\Delta_t}{2\epsilon}}\frac{\Delta_t}{\epsilon\delta}\right|_{m\delta,n\delta}$$

۲ روش پیاده سازی

در این پروژه پس از قرار دادن یک منبع تولید موج سینوسی، در محیط اتاق انتشار موج به سه شکل مختلف بررسی می گردد. در مرحله نخست، انتشار به گونه ای است که با برخورد موج به دیواره ها، تمام موج به طور کامل بازتاب می گردد. در مرحله ی دوم، موج به طور کامل از دیواره ها عبور می نماید. در روش سوم، دیواره ها تمام موج را جذب کرده و بدین ترتیب نه موج عبور داده می شود و نه بازتاب می گردد.

نکته ی قابل توجه در برنامه نویسی مربوط به مشخصات اتاق، این است که با توجه به اینکه، محدوده ی فرکانسی امواج الکترومغناطیس، طیف گسترده ای از مقادیر را شامل می شود، در صورت تغییر فرکانس، طول موج نیز تغییر پیدا کرده و در صورت ثبات طول اتاق، در برخی فرکانس ها، موجی قابل مشاهده نیست. بنابراین در قسمت تعیین ابعاد اتاق، ابعاد را وابسته به فرکانس تعریف کرده ایم به این معنی که با تغییر فرکانس، ابعاد به گونه ای تغییر نماید که موج در هر صورت به راحتی قابل مشاهده باشد.

۱.۲ منبع الکترومغناطیسی

در ابتدا برای انتشار موج در اتاق بایستی مشخصات اتاق مانند ابعاد، تعداد سلول ها و مشخصات موج مانند فرکانس، دامنه و سایر مقادیر اولیه تعریف گردد. همچنین یک منبع سینوسی در نقطه ای داخل اتاق تعریف می

شود به این گونه که مقدار آن را برابر میدان الکتریکی موج در نقطه موردنظر قرار داده می شود. منبع مذکور به صورت معادله ۵ تعریف می شود.

$$src(t) = A\sin(2\pi ft)$$

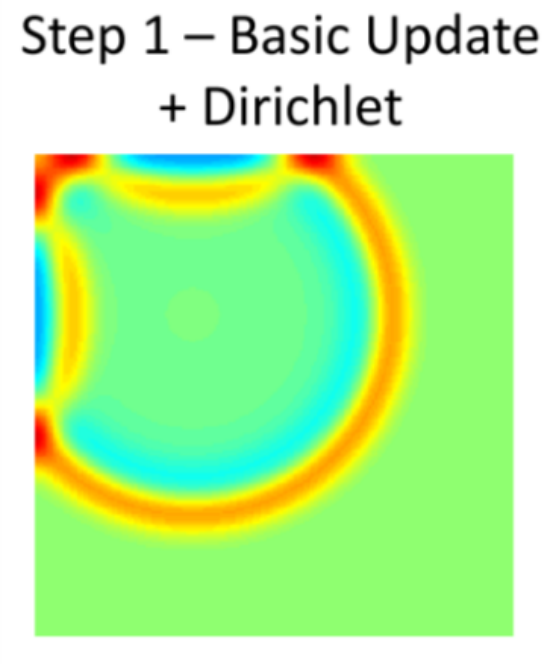
حال شبیه سازی را با یک حلقه در زمان شروع می کنیم: منبع سینوسی را قرار می دهیم و برای ادامه شبیه سازی منبع را برمی داریم. سپس مقدار میدان مغناطیسی در جهت x, y با توجه به مقدار فعلی میدان الکتریکی اپدیت می شوند. و این مقادیر جدید، خود میدان الکتریکی را اپدیت می کند.

۲.۲ شبیه سازی نخست : شرایط مرزی دیریکله

در این روش موج پس از برخورد با دیواره بازتاب می شود. برای شبیه سازی این قسمت، کافی است فرمول های ذکر شده ی معادله ۱، معادله ۲، معادله ۳و معادله ۴ را به صورت برنامه به زبان Python پیاده سازی نماییم. از آنجایی که محیط های خارج از اتاق تعریف نشده اند، می توان گفت به صورت ثابت مقدار «صفر» را دارا هستند؛ بدین معنی که هیچ موجی در این نقاط موجود نیست. بنابراین با رسیدن موج به حاشیه های اتاق، با توجه به اینکه اجازه ی ورود به محیط های خارجی را ندارد، به ناچار به طور کامل بازتاب می گردد. مطابق (شکل ۱)

۳.۲ شبیه سازی دوم : شرایط مرزی استوانه ای (Periodic Boundary Conditions)

همانطور که توضیح داده شد در این قسمت نیاز است شرایط مرزی به گونه ای اعمال گردد که به جای بازتاب، موج در برخورد با دیواره ها عبور داده شود. اما در حالت واقعی، به دلیل محدود بودن صفحه ی نمایش اتاق، موج عبور داده شده از دیوار نبایستی نمایش داده شود و در ظاهر با حالت جذب کامل موج توسط دیوار، تفاوتی نخواهد داشت. برای اینکه بتوان این تفاوت را در ظاهر نمایان کرد، برنامه را به گونه ای می نویسیم که در هنگام برخورد موج به دیوار، امتداد آن از دیوار مقابل خارج شود. به این صورت می توان تصور عبور را مطابق شکل ۲ برای بیننده فراهم نمود. برای اعمال این شرایط، بدین صورت عمل می کنیم که هر گاه موج به دیواره ها رسید، مختصات آن را به گونه ای تغییر می دهیم که به مختصات دیواره ی مقابل منتقل شده و ادامه ی انتشارش را از آن نقطه دنبال می نماییم.



شکل ۱: قدم نخست ـ به روز رسانی پخش موج در اتاق و اعمال شرایط مرزی دیریکله

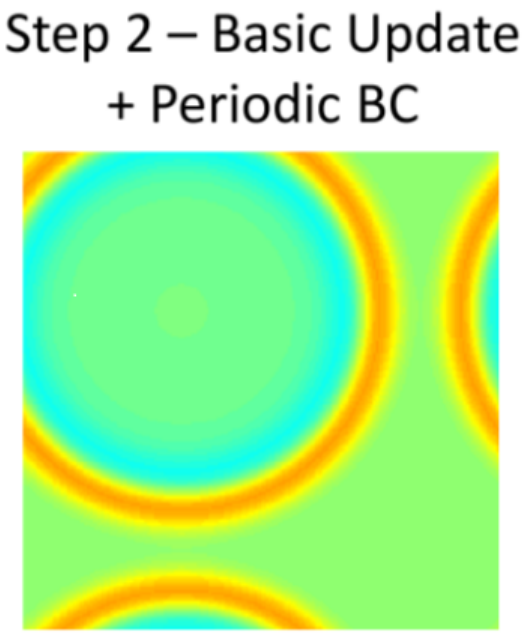
۴.۲ شبیه سازی سوم : اعمال لایه های جذب (PML)

مرحله بعدی پیشرفت برنامه، قرار دادن لایه های جاذب موج است (PML = Perfectly Matched Layer) با قرار دادن این لایه ها در مرز ها، موج الکترومغناطیسی به طور کامل جذب شده و از بازتاب و عبور آن ها جلوگیری به عمل می آید. (شکل ۳) محاسبات فیزیکی مربوط به انتشار موج در این حالت مورد مطالعه قرار گرفته و توضیح آن از حوصله ی این گزارش به دور است.

۳ افزودن مانع

به منظور مشاهده نحوه تغییر مسیر انتشار موج، از یک مانع مستطیلی شکل استفاده می شود. لذا یک patch مستطیل شکل را با مختصات دلخواه در شکل قرار می دهیم و مقدار

$$E_z$$

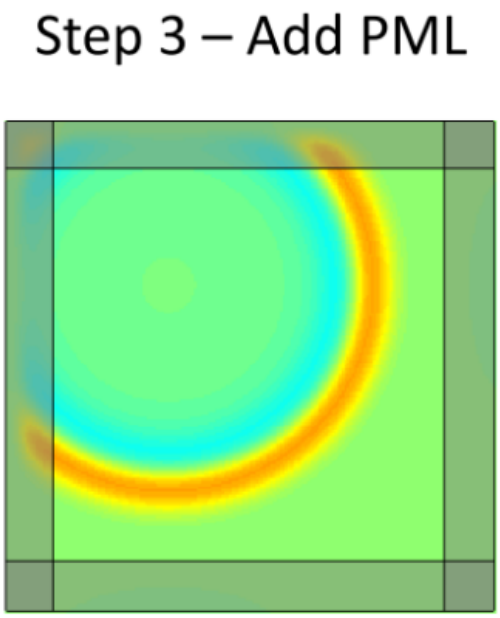


شکل ۲: قدم دوم ـ به روز رسانی پخش موج در اتاق و اعمال شرایط مرزی استوانه ای

را در این محدوده اتاق برابر صفر می نماییم. مشابه آنچه که در قسمت بازتاب کامل توضیح داده شد، با صفر قرار دادن کل محدوده ی دیوار، موج، از دیوار به طور کامل بازتاب خواهد شد. در حقیقت میزان بازتاب یا عبور موج از هر مانع را ضریب شکست جنس آن مانع تعیین می نماید. برای همین، قابلیتی قرار داده شده تا کاربر خود، ضریب شکست را تعیین نماید و مطابق با آن میزان بازتاب و عبور مانع را مشاهده نماید. اما به علت ذیق وقت، برنامه ی مربوط به این قسمت نا تمام رها شده است و امکان تغییر ضریب شکست در حال حاضر برای کاربر وجود ندارد.

۴ انتقال به محیط Qt

کد نهایی که شامل نمایش نمودار در Qt نیز می باشد، در مجموع شامل دو کلاس به نام های `MyWindow` و `PlotThread` است که به ترتیب مربوط به نمایش نمودار در widget ، مدیریت زمان بندی اجرای کد، یک تابع برای آپدیت مقادیر میدان های الکتریکی و مغناطیسی و تنظیم ضرایب مربوطه می باشد. که در ادامه به تفصیل در زیر به آنها پرداخته می شود.



شکل ۳: قدم سوم ـ افزودن لایه جاذب موج

۱.۴ محیط کاربری گرافیکی

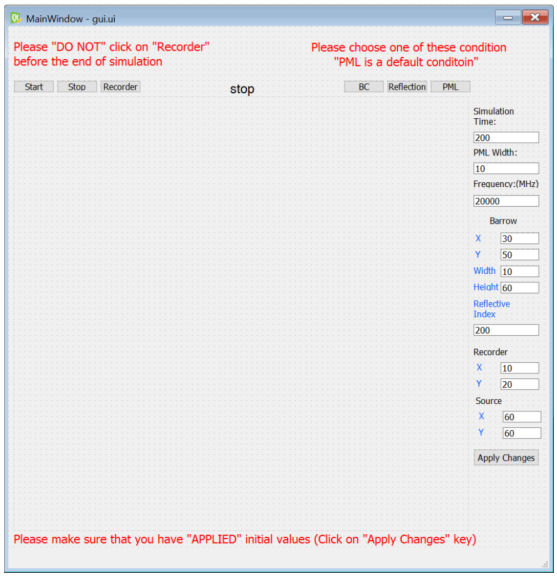
در ابتدا بایستی برای انتقال، یک محیط کاربری گرافیکی تعیین گردد تا ارتباط میان کاربر و برنامه را به صورت شمایی گرافیکی فراهم نماید. مطابق شکل ۴ یک محیط گرافیکی طراحی شده تا تعدادی ورودی از کاربر دریافت کرده و با توجه به مقادیر دلخواه کاربر شبیه سازی را انجام دهد. همچنین دکمه هایی برای شروع، توقف و ضبط برنامه و همچنین تغییر شرایط مرزی قرار داده شده است.

۲.۴ MyWindow

این کلاس از `QMainWindow` و `Form` که مربوط به فراخوانی فایل `gui` است ارث می برد. توابع مورد استفاده در این کلاس عبارتند از :

۱.۲.۴ تابع `__init__`

- صدا زدن constructor دو کلاسی که از آن ارث می برد.
- مقداردهی اولیه شرایط محیطی و فرکانس، دامنه و مکان منبع و سایر مقادیر ثابت



شکل ۴: رابط کاربری گرافیکی

- صفر کردن مقدار تمام میدان ها در تمام نقاط به منظور مشخص شدن سایز ماتریس
- نمایش نموداری شامل اتاق و دیوارها و مانع در پنجره
- ایجاد ارتباط میان pushBotton های پنجره

۲.۲.۴ توابع BC، Reflection و PML

این توابع مشخص میکنند که با فشردن دکمه مربوطه در پنجره چه شرایط مرزی ای به اتاق اعمال شود. به این صورت که یک متغیر condition را تغییر داده و هنگام تعریف ضرایب در تابع FDTD مورد استفاده قرار می گیرد. Flag2 برای تغییر شرایط مرزی است که در کلاس PlotThread به آن اشاره خواهد شد.

۳.۲.۴ تابع start

هنگام فشرده شدن دکمه start موجود در پنجره صدا زده می شود. حال چناچه مقدار flag برابر True باشد، دو دیکشنری شامل مقادیر متغیر ها ساخته و این مقادیر به یک object از کلاس PlotThread داده می شود و تنظیم می شود که برای آپدیت کردن یا اعلام اتمام اجرای دستور به ترتیب این دو سیگنال به update_plot و stop متصل شوند. اما اگر flag، برابر False این thread بسته و یک thread دیگر ایجاد می شود.

۴.۲.۴ تابع update_plo

این تابع مقدار E_z را توسط canvas نمایش می دهد.

۵.۲.۴ تابع stop

flag2 را True می کند به این معنا که در این حالت امکان تغییر در شرایط مرزی توسط کاربر وجود دارد و thread فعلی را می بندد.

۶.۲.۴ تابع Plot_record

این تابع نمودار فوریه رکوردر موجود در صفحه را که با دایره قرمز نشان داده شده است را رسم می کند. توجه شود که برای مشاهده نمودار باید فرآیند انتشار تمام شده باشد در غیر اینصورت چنانچه به صورت دستی روند اجرای برنامه متوقف شود، امکان نمایش نمودار وجود ندارد.

۷.۲.۴ تابع apply

این تابع وظیفه ی تعیین شرایط اولیه ای را به عهده دارد، که در محیط کاربری گرافیکی، توسط کاربر تعیین و تایید می گردند. در واقع با فشردن دکمه ی تایید، محل منبع، مانع و ضبط کننده ی موج مشخص می گردد. همچنین فرکانس موج، زمان شبیه سازی، عرض لایه ی جاذب و ضریب شکست دیوار تایین می گردد.

۳.۴ کلاس PlotThread

ابتدا دو سیگنالی ه در کلاس قبلی به آن اشاره شد تعریف می شود. هم چنین از این کلاس نیز ارث بری شده است. و آرایه ۳۰۰ تایی شامل ۳۰۰ بار آپدیت کردن میدان ها برای recordrf اختصاص داده می شود. توابع مورد استفاده در این کلاس عبارتند از :

۱.۳.۴ تابع __init__

در این تابع constructor مربوط به کلاس QtCore.QThread که از آن ارث برده می شود صدا زده می شود. سپس متغیرهایی مشابه متغیرهای که در کلاس MyWindow داشتیم را تعریف و مقدار آن را برابر مقادیر در کلاس مذکور قرار می دهیم. این روش برای جلوگیری ازتعریف متغیر global انجام گرفته است. همچنین flag برابر False قرار داده می شود تا امکان خروج از thread را فراهم نماید.

۲.۳.۴ تابع stop

در این تابع flag برابر False قرار داده می شود تا امکان خروج از thread را در کلاس MyWindow ایجاد کند.

۳.۳.۴ تابع run

این تابع به منظور اجرای دستوراتی است که هنگام فراخوانی thread اجرا می شود. در thread اگر flag برابر True باشد امکان آپدیت مقادیر وجود دارد. به این صورت که در یک حلقه ۳۰۰ تایی، تابع FTDT فراخوانی می شود تا میدان ها را به روز کند. اما قبل از آن موارد زیر انجام می گردد: مقدار میدان در محدوده مانع برابر صفر می گردد. اگر condition در زمانی که اجرای برنامه متوقف بوده توسط کابر تغییر کرده باشد مقادیر میدان برابر صفر قرار می گیرد تا کاربر بتواند انتشار موج را این بار با شرایط مرزی جدید مشاهده کند. همچنین برای مشاهده بهتر انتشار موج تنها منبع در ۲۰ تکرار اول وجود دارد و پس از آن منبع برداشته می شود. سپس این مقدار میدان در رکوردر ذخیره شده و هر سه تکرار یکبار هم مقدار میدان در نمودار نمایش داده می شود. بعد از اتمام حلقه مقدار فوریه محاسبه شده و سیگنال اتمام دستور منتشر می گردد.

۴.۴ تابع FTDT

در این تابع متغیرهایی که در کلاس MyWindow مقداردهی شده اند به این تابع داده شده و این تابع با توجه به condition ای که کاربر تعریف کرده است ابتدا ضرائب راتعیین سپس مقادیر جدید میدان ها را محاسبه می کند برای حالت reflection میدان در نزدیکی دیواره همیشه مقداری ثابت دارد لذا موجب بازگشت موج از دیواره می شود. حالت های PBC و PML ضرائب متفاوتی دارند اما همانطور که در بخش قبلی نیز توضیح داده شده بود، نحوه آپدیت مقادیر به این صورت است که چنانچه مقدار میدان در دیواره را بخواهیم از مقدار میدان در دیواره مقابل آن کمک می گیریم به گونه ای که گویا دیواره مقابل در ادامه مسیر موج قرار دارد.

پس از این تعاریف مشابه دستور های گفته شده در کلاس، یک شئ از QApplication ساخته می شود سپس یک شئ از کلاس MyWindow ساخته می شود و با دستور show نمایش داده می شود. که با اجرای این کلاس عملیات به روز رسانی مقادیر میدان ها و استفاده از thread ها از داخل همین کلاس صدا زده می شوند.

۵ انتقال محاسبات عددی سنگین به cyton

برای سریع تر شدن اجرای برنامه، بهتر است که محاسبات ریاضی برنامه در یک کلاس قرار داده شده و به محیطی سریع تر منتقل گردد. برای این کار از زبان سایتون استفاده می کنیم که بینابین دو زبان پایتون و سی

است و کلاس مذکور را به صورت یک کتابخانه داخل برنامه `import` می نماید.

با استفاده از فایل html به دست آمده از سایتون، اینگونه پیش می رویم که خطوط زرد آن را به گونه ای بهینه سازی می کنیم که عملکرد برنامه سریع تر شده، بیشتر شبیه به کد سی گردد و به رنگ سفید در آید. برای این کار، نوع متغیر هایی که در پایتون، نامشخص بود را تعیین نمودیم. همچنین توابعی را که می توانستیم به زبان سی بنویسیم را به زبان سی نوشتیم. همچنین حلقه هایی که به صورت list comprehention بودند را به صورت حلقه های تو در تو نوشتیم که البته باعث ایجاد هیچ بهبودی در برنامه نشد.