% Author: Michael Doderer

% E-Mail: michael.doderer@ief.ee.ethz.ch

% Organization: ETHZ ITET IEF

% Version: 1.0

clear all

close all

%% Prepare Figure, GUI

f = figure('Name','Diffraction visualizer','units','normalized','Position',[0.2 0.2 0.6 0.7]);

bgcolor = f.Color;

domainsliderx = uicontrol('Parent',f,'Style','slider','units','normalized','Position',[0.05 0.52 0.15 0.02],...

'value',20, 'min',1, 'max',50,'SliderStep',[1/10 0.1]);

domainslidery = uicontrol('Parent',f,'Style','slider','units','normalized','Position',[0.05 0.45 0.15 0.02],...

'value',10, 'min',1, 'max',50,'SliderStep',[1/10 0.1]);

domainlabel = uicontrol('Parent',f,'Style','text','units','normalized','Position',[0.05 0.55 0.15 0.03],...

'String','Domain dimensions x and y','BackgroundColor',bgcolor);

modeselect = uicontrol('Parent',f,'Style','popupmenu','units','normalized','Position',[0.05 0.85 0.15 0.03],...

'String',{'single', 'double', 'grating'});

slideropen = uicontrol('Parent',f,'Style','slider','units','normalized','Position',[0.05 0.82 0.15 0.02],...

'value',1, 'min',0, 'max',30,'SliderStep',[1/40 0.1]);

imtext1 = uicontrol('Parent',f,'Style','text','units','normalized','Position',[0.05 0.79 0.15 0.03],...

'String',num2str(slideropen.Value),'BackgroundColor',bgcolor);

slidersep = uicontrol('Parent',f,'Style','slider','units','normalized','Position',[0.05 0.75 0.15 0.02],...

'value',2, 'min',0, 'max',30,'SliderStep',[1/60 0.1]);

imtext2 = uicontrol('Parent',f,'Style','text','units','normalized','Position',[0.05 0.72 0.15 0.03],...

'String',num2str(slidersep.Value),'BackgroundColor',bgcolor);

sliderwl = uicontrol('Parent',f,'Style','slider','units','normalized','Position',[0.05 0.69 0.15 0.02],...

'value',500, 'min',400, 'max',1500,'SliderStep',[1/(11\*5) 0.1]);

imtext1 = uicontrol('Parent',f,'Style','text','units','normalized','Position',[0.05 0.66 0.15 0.03],...

'String',num2str(slideropen.Value),'BackgroundColor',bgcolor);

ax(1) = axes('Parent',f,'position',[0.25 0.55 0.4 0.4]);

ax(2) = axes('Parent',f,'position',[0.25 0.05 0.4 0.4]);

ax(3) = polaraxes('Parent',f,'position',[0.55 0.4 0.5 0.5]);

ax(4) = axes('Parent',f,'position',[0.7 0.05 0.25 0.25]);

params.mode = (modeselect.String{modeselect.Value});

params.widthopen = slideropen.Value;

params.widthsep = slidersep.Value;

%% Initialize plots

[x,y] = domainCallback(f,ax,20,10,modeselect.String{modeselect.Value},slideropen.Value,slidersep.Value,sliderwl.Value);

%% Define the two callback functions for the domain and parameter changes

domainsliderx.Callback = @(es,ed) domainCallback(f,ax,es.Value,domainslidery.Value,modeselect.String{modeselect.Value},slideropen.Value,slidersep.Value,sliderwl.Value);

domainslidery.Callback = @(es,ed) domainCallback(f,ax,domainsliderx.Value,es.Value,modeselect.String{modeselect.Value},slideropen.Value,slidersep.Value,sliderwl.Value);

modeselect.Callback = @(es,ed) bCallback(f,ax,es.String{es.Value},slideropen.Value,slidersep.Value,sliderwl.Value);

slideropen.Callback = @(es,ed) bCallback(f,ax,modeselect.String{modeselect.Value},es.Value,slidersep.Value,sliderwl.Value);

slidersep.Callback = @(es,ed) bCallback(f,ax,modeselect.String{modeselect.Value},slideropen.Value,es.Value,sliderwl.Value);

sliderwl.Callback = @(es,ed) bCallback(f,ax,modeselect.String{modeselect.Value},slideropen.Value,slidersep.Value,es.Value);

%% domainCallback which controlls changes of the domain size and does the initial plotting

function [x,y] = domainCallback(f,ax,xx,yy,mode,widthopen,widthsep,wavelength)

bx = [0 xx].\*1e-6; % bounding box in x

by = [-yy yy].\*1e-6; % bounding box in y

dd = 100e-9; % grid resolution

x = bx(1):dd:bx(2);

y = by(1):dd:by(2);

imagesc(ax(1),x\*1e6,y\*1e6,ones(length(x),length(y)))

colorbar(ax(1))

title(ax(1),'E-Field real part')

axis(ax(1),'image')

xlabel(ax(1),'Position z after screen [\mum]')

ylabel(ax(1),'Position y [\mum]')

colormap(ax(1),mycolormapfieldssymmetric)

imagesc(ax(2),x\*1e6,y\*1e6,ones(length(x),length(y)))

colorbar(ax(2))

title(ax(2),'normalized E-field')

axis(ax(2),'image')

xlabel(ax(2),'Position z after screen [\mum]')

ylabel(ax(2),'Position y [\mum]')

polarplot(ax(3),linspace(-pi/2,pi/2,1e4),ones(1,1e4))

title(ax(3),'Farfield normalized E-field')

thetalim(ax(3),[-90 90])

plot(ax(4),linspace(-pi/2,pi/2,1e4)\*180/pi,ones(1,1e4))

title(ax(4),'Farfield normalized E-field')

% axis(ax(4),'image')

xlabel(ax(4),'Farfield angle \theta [deg]')

ylabel(ax(4),'relative Intensity')

axis(ax(4),[-90 90 -inf inf])

% axis(ax(3),'image')

% xlabel(ax(3),'Position z after screen [\mum]')

% ylabel(ax(3),'Position y [\mum]')

bCallback(f,ax,mode,widthopen,widthsep,wavelength);

end

%% second callback which does the calculations and changes the plots upon parameter changes

function bCallback(f,ax,simtype,widthopen,widthsep,lambdain)

% get x and y vectors from the axes objects

x = ax(1).Children.XData\*1e-6;

y = ax(1).Children.YData\*1e-6;

lambda = round(lambdain).\*1e-9; % Wavelength conversion

opening\_a = round(widthopen,2).\*1e-6; % Opening of each slit

separation\_d = round(widthsep,2).\*1e-6; % Separation between slits

dd = x(2)-x(1);

theta = linspace(-pi/2,pi/2,1e4); % vector for farfield calculations

[X,Y] = meshgrid(x,y);

E = zeros(length(y),length(x));

% Calculate k

c0 = 3e8;

k = 2\*pi/lambda;

% Create source vector

nopen = round(opening\_a/dd);

nsep = round(separation\_d/dd);

source = zeros(1,length(y));

switch simtype

case 'single'

source(floor(length(source)/2-nopen/2):ceil(length(source)/2+nopen/2)) = 1;

case 'double'

source(floor(length(source)/2-nopen/2-nsep/2):ceil(length(source)/2+nopen/2-nsep/2)) = 1;

source(floor(length(source)/2-nopen/2+nsep/2):ceil(length(source)/2+nopen/2+nsep/2)) = 1;

case 'grating'

nperiod = nsep;

for i = ceil(length(source)/6):floor(length(source)\*5/6)

if mod(i,nperiod)<=nopen

source(i) = 1;

end

end

end

%% Loop calculation

j = 1;

for i = 1:length(source)

if source(i)==1

% full field calculation for each source 'point'

chi = atan((Y-y(i))./(X+dd));

r = sqrt((X+dd).^2+(Y-y(i)).^2);

E(:,:,j) = cos(chi)./r.\*exp(1i\*(k.\*r));

% Farfield calculation for each source 'point'

r2 = sqrt((cos(theta)).^2+(sin(theta)-y(i)).^2);

Eff(:,j) = cos(theta)./r2.\*exp(1i.\*(k.\*r2));

j = j+1;

end

end

% Summing over all the single sources (Huygens Principle)

Etot = sum(E,3);

Etot = Etot./max(Etot,[],'all');

Efarfield = sum(Eff,2);

% %% Non-loop calculation, needs lots of ram and is not really faster

% openings = logical(source);

% X2 = repmat(X,[1 1 sum(openings)]);

% Y2 = repmat(Y,[1 1 sum(openings)]);

% y2 = reshape(y(openings),[1 1 sum(openings)]);

%

% chi = atan((Y2-y2)./(X2+dd));

% r = sqrt((X2+dd).^2+(Y2-y2).^2);

% E = cos(chi+pi)./r.\*exp(1i\*(k.\*r));

%

% r2 = sqrt((cos(theta)).^2+(sin(theta)-y2).^2);

% Eff = cos(theta)./r2.\*exp(1i.\*(k.\*r2));

% Etot = sum(E,3);

% Etot = Etot./max(Etot,[],'all');

% Efarfield = sum(Eff,3);

%% Update the plotting

f.Children(5).String = ['Slit width: ' num2str(opening\_a\*1e6) ' um'];

f.Children(3).String = ['Slit separation: ' num2str(separation\_d\*1e6) ' um'];

f.Children(1).String = ['Wavelength: ' num2str(lambda\*1e9) ' nm'];

Eplot = real(Etot);

ax(1).Children.CData = Eplot;

ax(1).CLim = [-median(median(Eplot(Eplot>0))).\*20 median(median(Eplot(Eplot>0))).\*20];

Eplot = (Etot).\*conj(Etot);

ax(2).Children.CData = Eplot;

ax(2).CLim = [0 median(Eplot,'all').\*20];

Eplot = ((Efarfield).\*conj(Efarfield));

Eplot = 10\*log10(Eplot./max(Eplot,[],'all'));

Eplot(Eplot<-50) = -50;

ax(3).Children.RData = Eplot;

ax(3).RLim = [-50 inf];

ax(4).Children.YData = Eplot;

% ax(4).Lim = [-50 inf];

end

%% Pretty colormap

function out = mycolormapfieldssymmetric(size)

if nargin<1

size = 2^10;

end

size2 = size/2;

out = ones(size,3);

for i = 0:size2-1

out(size-i+1,:) = [(i+(size2-1)\*1.5)/2.5 i (i+(size2-1)\*0.0)/1.0]./(size2-1);

end

for i = 0:size2-1

out(i+1,:) = [(i+(size2-1)\*0.0)/1.0 i (i+(size2-1)\*1.5)/2.5]./(size2-1);

end

out = flip(out.^1.6);

end