

LAMPIRAN A

Universitas Kristen Maranatha

SIMULASI FDTD.M

```
% 2D, simulasi FDTD spherical dari antena UWB conical.
% Simulasi ini menggunakan 2D dari 3D persamaan spherical Maxwell.
% Komponen medan listrik dan medan magnet dikalkulasi untuk propagasi
% gelombang.
% Eksitasi atau pembangkit gelombang dapat dipilih antara gelombang sinus
% atau pulsa Gaussian
%
% Tuti A. Harefa
%
function [Et] = FDTD12(tmax)
% Initialize standard free space constants
uo = 4*pi*1e-7;
eo = 1e-9/(36*pi);
Z = 120*pi;
c = 3e+8;
% Definisikan dimensi antena. Ini adalah 15dr (1 lambda_nom) panjangnya
% dengan sudut 30 derajat. Tambahkan nilai satu sebagai referensi matriks.
ant_length = 15;
ant_angle = 31;
% Definisikan batas sisi. Rmax terjadi pada 10(lambda_nominal) dimana lambda
% nominal adalah panjang gelombang untuk frekuensi 6.5GHz. Bandwidth yang
% diharapkan adalah 3,1-10,6 GHz, with 3,1 GHz being lambda_max dan 10.6
% GHz menjadi lambda_min.
% Jarak ini didefinisikan untuk FCC untuk transmisi UWB . Theta
% luasnya dari 0 ke 90 derajat. Disini, dr = 0.003 m (lambda_min/10), dan
% dtheta = 1 derajat. Penggunaan dt = 0.2 psec akan menyederhanakan batas
% Courant untuk semua frekuensi.
dr = 0.003; % radius sel
dth = 1.0*pi/180; % sudut sel dalam radian
dt = 0.2e-12; % time step
Rmax = 151; % solusi radius ruang
THmax = 91; % solusi sudut ruang
b = 0.006; % radius coaxial input terluar
a = 0.003; % radius coaxial input terdalam
% Set up konstanta untuk daerah ruang:Npml=# layer,R0= koefisien refleksi
% pada sudut 90 derajat, sigma_space = konduktansi pada ruang
% bebas (0 layer), sigmaM_space = konduktansi magnet pada ruang bebas, L=1
% untuk profile konduktansi linear, L=2 untuk profile konduktansi parabolik,
% untuk sebuah konduktansi geometri
R0 = 1e-14; % pilih refleksi yang diinginkan
```

```

Npml = 20; % pilih jumlah layer yang digunakan
L = 2; % pilih profile konduktiviti
g = 2; % pilih faktor geometri
% Kalkulasikan konduktiviti ruang bebas untuk profile linear atau parabolik
sigma_space = - eo*c*log(R0)/(2^(L+2)*dr*Npml^(L+1));
% Kalkulasikan konduktiviti ruang bebas untuk profile geometri
% sigma_zero = - eo*c*log(g)*log(R0)/(2*dr*(g^(Npml)-1));
% sigma_space = sigma_zero*(sqrt(g) - 1)/log(g);
sigmaM_space = uo*sigma_space/eo; % kondidi matching impedansi
% Kalkulasikan final konduktiviti untuk profile linear atau parabolik final
sigmaPML(Npml) = sigma_space*(L+1)*2^(L+1)*Npml^L;
% Kalkulasikan final konduktiviti untuk profile geometri final
% sigmaPML(Npml) = sigma_zero*(g-1)*(g^Npml)/(sqrt(g)*log(g));
sigmaMPML(Npml) = uo*sigmaPML(Npml)/eo;
for I = 1:(Npml-1)
sigmaPML(I) = sigmaPML(Npml)*(I/Npml)^L; % untuk profile linear
% sigmaPML(I) = sigmaPML(Npml)*g^(I-Npml); % untuk profile geometric
sigmaMPML(I) = uo*sigmaPML(I)/eo;
end
% Inisialisasikan sub komponen H untuk daerah PML
for I = 1:Npml
for J = 1:THmax
Hpr(I,J) = 0;
Hpt(I,J) = 0;
end
end
% Set up pengkonduktansian permukaan antenna. Antena terbuat dari tembaga
% dengan konduktiviti of 5.8e+7 mho/m. ruang bebas diberi sebuah
% konduktiviti listrik kecil dan a konduktiviti magnet matching menurut
% kalkulasi PML yang telah dilakukan di atas.
sigma_cu = 5.8e+7;
sigmaM_cu = uo*sigma_cu/eo;
for I = 1:Rmax
for J = 1:THmax
if (J == ant_angle)
if (I <= ant_length)
sigma(I,J) = sigma_cu;
sigmaM(I,J) = sigmaM_cu;
elseif (I < Rmax-Npml+1)
sigma(I,J) = sigma_space;
sigmaM(I,J) = sigmaM_space;
else
sigma(I,J) = sigmaPML(I + Npml - Rmax);
sigmaM(I,J) = sigmaMPML(I + Npml - Rmax);
end
end
end

```

```

elseif (I == ant_length)
if (J <= ant_angle)
sigma(I,J) = sigma_cu;
sigmaM(I,J) = sigmaM_cu;
else
sigma(I,J) = sigma_space;
sigmaM(I,J) = sigmaM_space;
end
else
if (I < Rmax-Npml+1)
sigma(I,J) = sigma_space;
sigmaM(I,J) = sigmaM_space;
else
sigma(I,J) = sigmaPML(I + Npml - Rmax);
sigmaM(I,J) = sigmaMPML(I + Npml - Rmax);
end
end
end
end
end
% Untuk inisialisasi, nolkan medan untuk semua node bebas pada masalah tersebut.
for I = 1:Rmax
for J = 1:THmax
if ((I >= ant_length) | (J >= ant_angle))
Er(I,J) = 0;
Et(I,J) = 0;
Hp(I,J) = 0;
end
end
end
% Tetapkan parameter driving pulsa Gaussian. Centernya
% 200ps out, it has a decay time of 50ps and frekuensi center
% 6.5 GHz.
tstart = 200e-12;
tdecay = 50e-12;
tsample = 183e-12;
once = 0;
fc = 6.5e+9;
Vocw = 0.01;
Vouwb = 0.06;
Rs = 50;
Zin = 0;
% Set up beberapa konstanta untuk waktu iterasi
t = 0;
g1 = dt/(2*uo); % g1 = 7.6e-8
g2 = dt/(2*eo); % g2 = 0.0113

```

```

g3 = dt/(dr*eo); % g3 = 7.53
g4 = dt/(dr*uo); % g4 = 5.3e-5
sphere_factor = 1/(b*log(2));
I_factor = 2*pi*b;
volt_factor = (log(sin(ant_angle*pi/180))-log(1-cos(ant_angle*pi/180)))/log(2);
% Mulai waktu iterasi
while (t < (tmax*dt))
% Untuk pertamanya 1/2 timestep, update medan Hphi dimana saja.
% Jangan step kesingularan sumber, point ground plane,
% jalur/garis simetri, atau daerah PML.
t = t + 0.5*dt;
for I = 3:(Rmax-Npml)
for J = 2:(THmax-1)
if ((I >= ant_length) | (J >= ant_angle))
g5 = sin(J*dth)/sin((J-1)*dth);
Da = (1-sigmaM(I,J)*g1)/(1+sigmaM(I,J)*g1);
Db = g4/(1+sigmaM(I,J)*g1);
ER1 = (g5*Er(I,J+1)-Er(I,J))/((I-1/2)*dth);
ET1 = (I/(I-1))*Et(I+1,J) - Et(I,J);
Hp(I,J) = Da*Hp(I,J) + Db*(ER1 - ET1);
end
end
end
% Hitung medan H pada ground plane, sisi luar daerah PML
J = THmax;
for I = 3:(Rmax-Npml)
Da = (1-sigmaM(I,J)*g1)/(1+sigmaM(I,J)*g1);
Db = g4/(1+sigmaM(I,J)*g1);
ET1 = (I/(I-1))*Et(I+1,J) - Et(I,J);
Hp(I,J) = Da*Hp(I,J) - Db*ET1;
end
% Hitung medan H sepanjang garis simetri.
for I = ant_length:(Rmax-Npml)
Hp(I,1) = Hp(I,2);
end
% Update medan H di daerah PML, kecuali ground plane
layer = 1;
for I = (Rmax-Npml+1):Rmax
Dar = (1-sigmaMPML(layer)*g1)/(1+sigmaMPML(layer)*g1);
Dbr = g4/(1+sigmaMPML(layer)*g1);
Dat = Dar;
Dbr = Dbr;
for J = 2:(THmax-1)
g5 = sin(J*dth)/sin((J-1)*dth);
ER1 = (g5*Er(I,J+1)-Er(I,J))/((I-1/2)*dth);

```

```

if (I == Rmax)
ET1 = 0;
else
ET1 = (I/(I-1))*Et(I+1,J) - Et(I,J);
end
Hpr(layer,J) = Dar*Hpr(layer,J) - Dbr*ET1;
Hpt(layer,J) = Dat*Hpt(layer,J) + Dbt*ER1;
Hp(I,J) = Hpr(layer,J) + Hpt(layer,J);
end
layer = layer + 1;
end
% Update medan H pada daerah PML sepanjang ground plane
J = THmax;
layer = 1;
for I = (Rmax-Npml+1):Rmax
Dar = (1-sigmaMPML(layer)*g1)/(1+sigmaMPML(layer)*g1);
Dbr = g4/(1+sigmaMPML(layer)*g1);
Dat = Dar;
if (I == Rmax)
ET1 = 0;
else
ET1 = (I/(I-1))*Et(I+1,J) - Et(I,J);
end
Hpr(layer,J) = Dar*Hpr(layer,J) - Dbr*ET1;
Hpt(layer,J) = Dat*Hpt(layer,J);
Hp(I,J) = Hpr(layer,J) + Hpt(layer,J);
layer = layer + 1;
end
% Untuk keduanya 1/2 timestep, update medan E dimana saja. Sekali lagi,
% jangan step driver pada sumber atau point ground plane.
t = t + 0.5*dt;
% Step sumber eksitasi, pulsa Gaussian, or the steady
% state driver sinusoidal 6.5GHz.
% if (t < 1.0e-9)
% Vsrc = Vocw*cos(2*pi*fc*t); % CW source
% else
% Vsrc = 0;
% end
% Vsrc = Vouwb*exp(-((t-tstart)/tdecay)^2)*cos(2*pi*fc*(t-tstart));
Vsrc = -Vouwb*exp(-((t-tstart)/tdecay)^2)*sin(2*pi*fc*(t-tstart));
% Sekarang update medan E. Step pertama driving sphere...
Vin = 0; Iins = 0;
for J = ant_angle:THmax
Iin = I_factor*sin((J-1/2)*dth)*Hp(3,J);
Vdrv = Vsrc - Rs*Iin;

```

```

if ((t >= tsample) & (J > ant_angle))
Vin = Vin + b*Et(3,J)*dth;
lins = lins + lin;
if (J == (THmax - 1))
tsample = 1e-6;
lins = lins/(THmax - ant_angle - 1);
exVin = Vin
exlin = lin
Zin = abs(Vin/lin)
end
end
Et(3,J) = Vdrv*sphere_factor/sin((J-1/2)*dth);
end
% Sekarang step medan E untuk node bebas
for I = 4:(Rmax-Npml)
for J = 2:(THmax-1)
if ((I >= ant_length) | (J >= ant_angle))
g6 = sin((J-1/2)*dth)/sin((J-3/2)*dth);
Ca = (1 - g2*sigma(I,J))/(1 + g2*sigma(I,J));
Cb = g3/(1 + g2*sigma(I,J));
HPHI1 = (g6*Hp(I,J) - Hp(I,J-1))/((I-1/2)*dth);
HPHI2 = Hp(I-1,J) - ((I-1/2)/(I-3/2))*Hp(I,J);
Er(I,J) = Ca*Er(I,J) + Cb*HPHI1;
Et(I,J) = Ca*Et(I,J) + Cb*HPHI2;
end
end
end
% Hitung medan E pada ground plane
J = THmax;
for I = 4:(Rmax-Npml)
Ca = (1 - g2*sigma(I,J))/(1 + g2*sigma(I,J));
Cb = g3/(1 + g2*sigma(I,J));
HPHI2 = Hp(I-1,J) - ((I-1/2)/(I-3/2))*Hp(I,J);
Er(I,J) = 0;
Et(I,J) = Ca*Et(I,J) + Cb*HPHI2;
end
% Hitung medan E sepanjang garis simetri.
for I = ant_length:(Rmax-Npml)
Ca = (1 - g2*sigma(I,1))/(1 + g2*sigma(I,1));
Cb = g3/(1 + g2*sigma(I,J));
HPHI1 = (Hp(I,2) - Hp(I,1))/((I-1/2)*dth);
Er(I,1) = Ca*Er(I,1) + Cb*HPHI1;
Et(I,1) = 0;
end
% Update medan E di daerah PML, kecuali ground plane

```

```

layer = 1;
for I = (Rmax-Npml+1):Rmax
for J = 2:(THmax-1)
g6 = sin((J-1/2)*dth)/sin((J-3/2)*dth);
Car = (1 - g2*sigmaPML(layer))/(1 + g2*sigmaPML(layer));
Cbr = g3/(1 + g2*sigmaPML(layer));
Cat = Car;
Cbt = Cbr;
HPHI1 = (g6*Hp(I,J) - Hp(I,J-1))/((I-1/2)*dth);
HPHI2 = (Hp(I-1,J) - ((I-1/2)/(I-3/2))*Hp(I,J));
Er(I,J) = Cat*Er(I,J) + Cbt*HPHI1;
Et(I,J) = Car*Et(I,J) + Cbr*HPHI2;
end
layer = layer + 1;
end
% Hitung medan E daerah PML pada ground plane
layer = 1;
J = THmax;
for I = (Rmax-Npml+1):Rmax
Car = (1 - g2*sigmaPML(layer))/(1 + g2*sigmaPML(layer));
Cbr = g3/(1 + g2*sigmaPML(layer));
HPHI2 = (Hp(I-1,J) - ((I-1/2)/(I-3/2))*Hp(I,J));
Er(I,J) = 0;
Et(I,J) = Car*Et(I,J) + Cbr*HPHI2;
layer = layer + 1;
end
end
% Plot hasilnya
% Ubah kutub koordinat ke koordinat rectangular dan cerminkan
% data simulasi antena untuk menunjukkan +x and -x dari peradiasian
% medan.
for I = 1:Rmax
for J = 1:THmax-1
x = 151 + round((I*sin((J-1)*dth)));
x2 = 303 - x;
y = 1 + round((I*cos((J-1)*dth)));
Ecart(x,y) = Et(I,J);
Ecart(x2,y) = Et(I,J);
end
end
% Load sebuah matrik Cartesian untuk menyisipkan elemen yang hilang
for I = 1:(2*Rmax-1)
for J = 1:Rmax
EcartNew(I,J) = Ecart(I,J);
end

```



```

end
% Sisipkan elemen yang hilang di dalam matrik Cartesian
Imin = 2; Imax = 2*Rmax - 2; Jmin = 2; Jmax = Rmax - 1;
for I = Imin:Imax
for J = Jmin:Jmax
if ((Ecart(I,J) == 0) & (Rmax*cos((I-Rmax)*dth*91/151) + 25 >= J))
ItempLo = I - 1;
ItempHi = I + 1;
while ((Ecart(ItempLo,J) == 0) & (ItempLo > 1))
ItempLo = IntempLo - 1;
end
while ((Ecart(ItempHi,J) == 0) & (ItempHi < 2*Rmax - 1))
ItempHi = IntempHi + 1;
end
M = Ecart(ItempLo,J);
N = Ecart(ItempHi,J);
if (M == 0)
temp1 = N;
elseif (N == 0)
temp1 = M;
else
temp1 = sign(M+N)*sqrt(abs(M*N));
end
JtempLo = J - 1;
JtempHi = J + 1;
while ((Ecart(I,JtempLo) == 0) & (JtempLo > 1))
JtempLo = JtempLo - 1;
end
while ((Ecart(I,JtempHi) == 0) & (JtempHi < Rmax))
JtempHi = JtempHi + 1;
end
M = Ecart(I,JtempLo);
N = Ecart(I,JtempHi);
if (M == 0)
temp2 = N;
elseif (N == 0)
temp2 = M;
else
temp2 = sign(M+N)*sqrt(abs(M*N));
end
if (temp1 == 0)
EcartNew(I,J) = temp2;
elseif (temp2 == 0)
EcartNew(I,J) = temp1;
else

```

```

EcartNew(I,J) = sign(temp1+temp2)*sqrt(abs(temp1 *temp2));
end
end
end
EcartNew(1,151) = 1.2; % point palsu ditambahkan kekuatan penskalaan
EcartNew(2,151) = -1.0; % final plot
end
I = 1:301;
J = 1:151;
x(I) = I;
y(J) = J;
surfl(y(J), x(I), EcartNew(I,J))
xlabel('Y-axis cm'), ylabel('X-axis cm')
zlabel('Etheta V/m')
% title('FDTD Antena Conical 6.5GHz: CW Driver')
title('FDTD Antena Conical 6.5GHz: UWB Driver')
shading interp
colormap pink

```

SIMULASI IDEALCONE.M

```
% Solusi medan bentuk tertutup untuk infinite antena conical
%
function [Et] = IdealCone14()
% Gunakan parameter diskritisasi yang sama (dr, dth, dt, ranges, dll.) saat
% digunakan pada simulasi FDTD.
dr = 0.003;
dth = 1;
dt = 5e-12;
Z = 377;
c = 3e+8;
% Jarak memperluas panjang gelombang 15 median dalam R dan dari 0-90
% derajat dalam theta. Sudut infinite antena cone sama dengan sebelumnya
% untuk simulasi FDTD.
Rmax = 151;
THmax = 91;
cone_angle = 31;
theta = cone_angle;
r = dr;
% Inisialisasi parameter gelombang, frekuensi (fo), amplituda (Eo),
% jumlah gelombang (k), dll.
Vo = 0.033;
Eo = Vo/(2*log(cot((pi/180)*(cone_angle-1)/2)));
fo = 6.5e+9; % 6.5GHz
k = 2*pi*fo/c; % 136.1
t = 0;
for i = 1:Rmax
theta = cone_angle;
for j = 1:(THmax-1)
if (j >= cone_angle)
wave = cos(2*pi*fo*t - k*r);
Et(i,j) = (1/(r)*sin((theta-1)*pi/180))*(Eo*wave);
else
Et(i,j) = 0.0;
end
Hp(i,j) = Et(i,j)/Z;
theta = theta + dth;
end
r = r + dr;
end
% Plot hasilnya
```

```

% Ubah koordinat polar ke koordinat rectangular dan cerminkan data simulasi
% antenna untuk menunjukkan +x dan -x dari peradiasian
% medan.
% Solusi ruang harus dikompres 5x untuk menyelesaikan perubahan polar ke
% rectangular.
for i = 1:Rmax
for j = 1:THmax-1
x = 32 + round((i*sin((j-1)*dth*pi/180))/5);
x2 = 63 - x;
y = 1 + round((i*cos((j-1)*dth*pi/180))/5);
Ecart(x,y) = Et(i,j);
Ecart(x2,y) = Et(i,j);
end
end
i = 1:62;
j = 1:31;
x(i) = i*1.5;
y(j) = j*1.5;
surfl(y(j), x(i), Ecart(i,j))
xlabel('Y-axis cm'), ylabel('X-axis cm')
zlabel('Etheta V/m')
title('Infinite Antena Cone sudut 30 derajat pada 6.5GHz')
shading interp
colormap pink

```

SIMULASI PLOT.M

```
% Menghasilkan sebuah plot polar untuk menunjukkan pola radiasi medan jauh.
%
% Fungsi ini menerima input matrik dari run simulation FDTD12 dan
% membangkitkan sebuah plot polar 2-D. Run FDTD71 seharusnya untuk 8000
% timesteps pada dt = 0.2psec.
%
function [] = Pplot73(Ft)
% Defenisikan beberapa konstanta: Jmax adalah sudut maximum, Rview adalah
% hasil radius medan jauh untuk memplot medan.
Jmax = 91;
Rview = 121;
tstep = pi/180;
tmax = pi/2;
t = tstep:tstep:tmax+tstep;
% Secara manual enter data radiasi medan.
B(1:10) = [0 0.02 0.03 0.04 0.06 0.07 0.08 0.09 0.10 0.11];
B(11:21) = [0.12 0.13 0.15 0.16 0.18 0.19 0.2 0.21 0.23 0.24 0.25];
B(22:33) = [0.26 0.27 0.29 0.3 0.31 0.32 0.33 0.35 0.36 0.37 0.38 0.4];
B(34:45) = [0.41 0.42 0.44 0.45 0.46 0.48 0.49 0.5 0.52 0.53 0.55 0.59];
B(46:57) = [0.60 0.62 0.63 0.65 0.66 0.67 0.69 0.7 0.71 0.73 0.74 0.75];
B(58:69) = [0.77 0.78 0.79 0.8 0.81 0.82 0.83 0.85 0.86 0.87 0.88 0.88];
B(70:81) = [0.89 0.9 0.91 0.91 0.92 0.93 0.94 0.94 0.95 0.96 0.96 0.97];
B(82:91) = [0.97 0.98 0.98 0.99 0.99 0.99 0.99 1.00 1.00 1.00];
% Skala data in amplituda
B = B * 0.043;
% Plot hasil simulasi
% polar(t, Ft(Rview,1:Jmax),'r')
C(1:Jmax) = Ft(Rview, 1:Jmax);
C(Jmax+1:2*Jmax) = B;
t2(1:Jmax) = t;
t2(Jmax+1:2*Jmax) = -t;
polar(t2, C, 'r')
```

SIMULASI VSRC.M (PULSA UWB)

```
% Fungsi ini memodelkan pulsa bentuk tegangan UWB Gaussian untuk
% mendrivekan antenna UWB.
% Ini menerima sebuah maximum. timestep "Nmax"
% Outputnya sebuah array. Ini seharusnya dipanggil sekali, and hasil output
% matriknya seharusnya di referensikan.
%
function [Vout] = Vsrc13(Nmax,dt)
% Set time step pada 1psec
dt = 1e-12;
% Letakkan pulsa waktu center (nstart) = 200psec
nstart = 200;
% Waktu 1/e (ndecay) diambil 10dt
ndecay = 50;
% Frekuensi pulsa center (fc) adalah 6.5GHz, center dari FCC 3,1-10,6 GHz
fc = 6.5*10^9;
% Pulsa amplituda maximum (Vo) diambil 1 volt
Vo = 1.0;
for n = 1:Nmax
Vout(n) = Vo*exp(-((n-nstart)/ndecay)^2)*cos(2*pi*fc*(n-nstart)*dt);
end
plot(Vout)
% title('Pulsa bentuk Gaussian:-Vo*exp(-((n-200)/50)^2)*sin(2*pi*fc*(n-200)dt)')
xlabel('Time (psec)');
ylabel('Vout(Volt)');
title('Pulsa bentuk Gaussian: Vo*exp(-((n-200)/50)^2)*cos(2*pi*fc*(n-200)dt)')
```

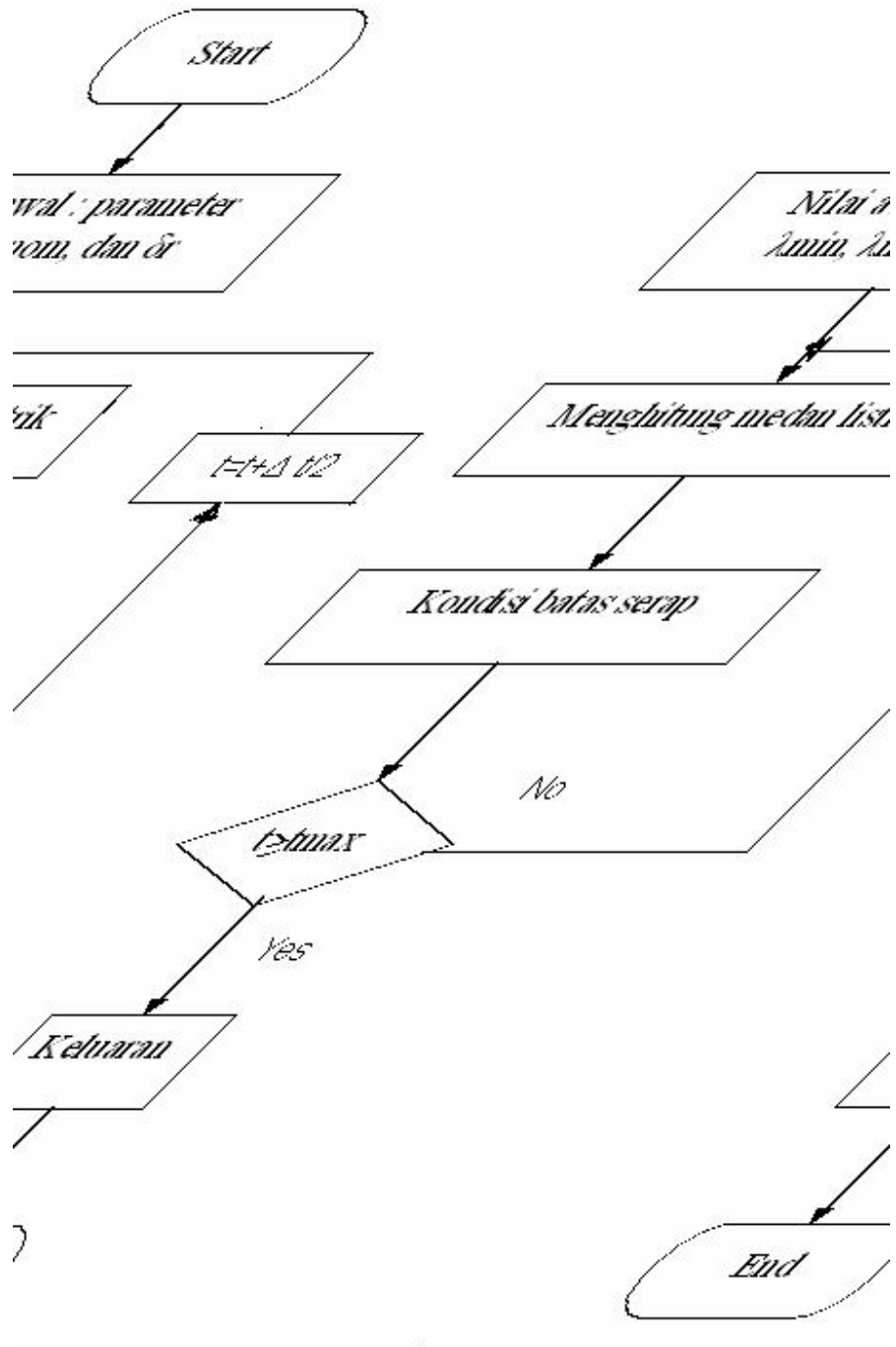
SIMULASI CONEGEOM.M

```
% Fungsi ini memplot sketsa secara fisik dari antenna conical.
%
function [Z] = ConeGeom15()
ant_angle = 31;
R= 15;
Ha = cos((ant_angle - 1)*pi/180);
Zmax = R*Ha;
N = 60;
Zprev = 0;
for i = 1:N+1
for j = 1:N+1
if (Zprev < Zmax)
Z(i,j) = Ha*R*(i-1)/N;
term = R*(i-1)/N;
Zprev = term;
else
Z(i,j) = R*cos(pi/2*(2*(i-1)/N - 1)- pi/2);
term = (1/Ha)*R*cos(pi/2*(2*(i-1)/N - 1));
end
arg = 2*pi*(j-1)/N;
Y(i,j) = term*cos(arg + pi/2);
X(i,j) = term*cos(arg + pi);
end
end
surf(X,Y,Z)
hidden off
```

SIMULASI FILM.M

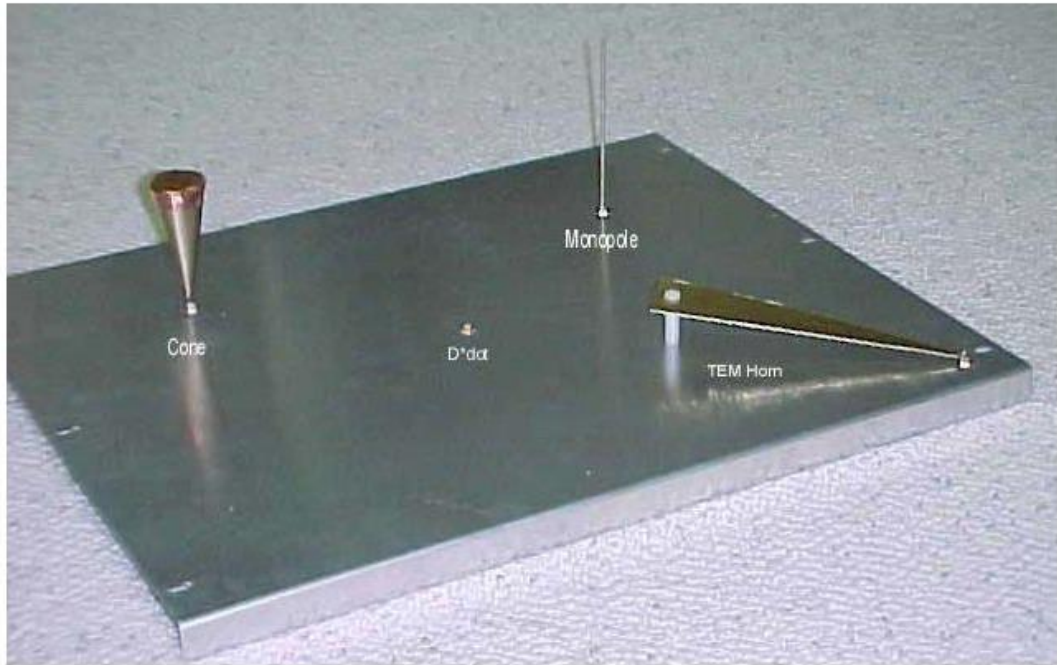
```
% Film.m : Menghasilkan sebuah movie dari simulasi FDTD antenna Conical
% dengan memanggil callinfg FDTD12 secara berulang dengan interval waktu
% yang berbeda. Panggil routine ini dengan m = film; jika ingin untuk
% menyimpan file movie, m, untuk mereplay lagi.
% Untuk mengeksekusi hasil dari fungsi ini, buka data file Matlab pulsa
% Gaussian.
%
function [m] = Film17()
tcount = 0;
for I = 1:23
if (I < 7)
tinc = 250;
else
tinc = 500;
end
tcount = tcount + tinc;
FDTD1(tcount);
view(-90,60);
axis([0 200 0 300 -1 2]);
m(I) = getframe;
end
cla
movie(m,10,3)
```


LAMPIRAN B



Gambar B-1 Flowchart Cara Kerja

LAMPIRAN C



Gambar C-1 Miniatur Jarak Antena UWB