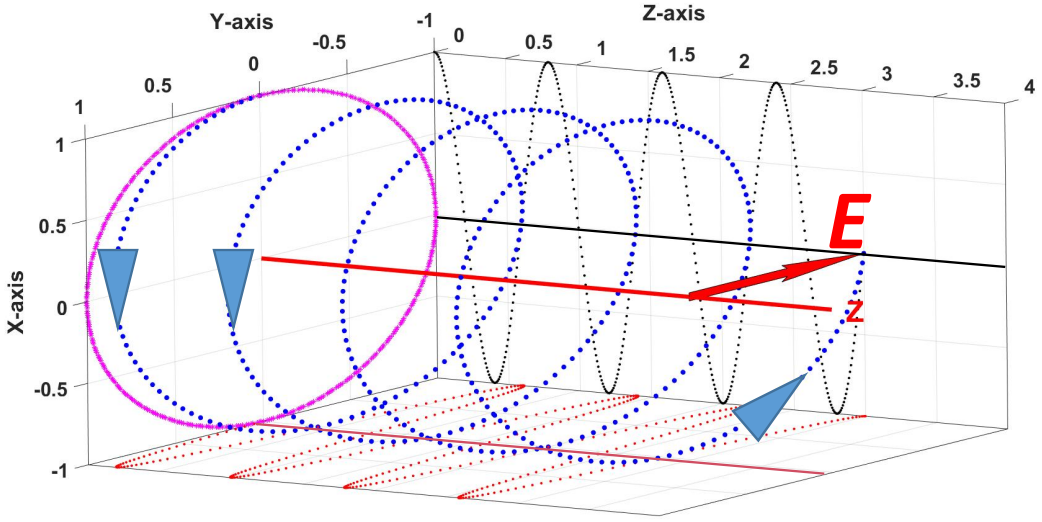
To boost the flavor of electromagnetics, we accompanied many problems with short Matlab code acting as *a calculator and result illustrator*. We completely ignored the description of algorithms behind the codes because they are often quite complicated and required ample, very sophisticated as well as abstract mathematical specifics poorly helping to understand the physical picture. Such approach lets us shift the focus from the often emasculated and practically fruitless problems primarily based on math transformations to more complicated but close to practical tasks. We hope that it helps to visualize typically invisible EM fields and stimulate in-depth analysis and the following discussion of the fundamental principles. Projecting images onto a big screen, the lectures may organize whole class conversation making the audience to be significantly engaged. These scripts are for people who have never programmed in Matlab or CST before.
We cheer our readers to look through Matlab scripts and CST models to discover and sophisticate the algorithm embedded in them. We encourage to use the student edition of MATLAB and CST STUDIO SUITE® to get enhanced problem understanding.

Attention. Regrettably, copy and paste into Matlab Command Window saves appropriate Matlab format just in Chrome Web Browser. You have to restart https://emfieldbook.com/ in https://www.google.com/ if your browser is different.

EM Wave Polarization.



Refresh the material of Section 5.1 in Chapter 5. Copy and paste Matlab script bellow into **Matlab Command Window,** enter the input parameters a and ψ according to expression (5.2) in Chapter 5, and run it. The full-screen Figure 1 animates the space behavior of E-vector matching three consecutive wavefronts. The blue vector corresponds to the wavefront at $t = 0$, $z = 0$, red one follows behind and reaches the cross section $z = 0$ later with delay of Δt, while the green one comes to the same cross section $z = 0$ at $t = 2\Delta t$. The matching color thick dots trace the movement of each vector tip along the helical path forwards the z-axis. The magenta, red and black dots in $\theta\varphi-, \theta Z-$ and $\varphi Z-$ plane demonstrate correspondingly the behavior of the total E-vector (polarization ellipse) and its harmonic components $E_\theta(z)$ and $E_\varphi(\mathrm{z})$. The larger diameter dots of the corresponding color depicts the components' magnitude at $z = 0$. Finally, the image in Figure 1 is rotated to be comparable to Figure 5.1.4 in Chapter 5[1]. Please check and understand all the cases mentioned in Section 5.1.1

[1] Excellent animations can be found at Prof. Ian Copper's website http://www.physics.usyd.edu.au/teach_res/mp/doc/cemPol1.htm, and these two https://www.youtube.com/watch?v=Q0qrU4nprB0 and https://www.youtube.com/watch?v=8YkfEft4p-w.

```
close all, clc; clear; a=input('Enter Magnitude of Current Ratio a = ')
psi=input('Enter Phase (|psi|<= pi/2) of Current Ratio [rad] psi = ')
H=6*pi; omega_t=linspace(0,pi,5e0); gamma_z=linspace(0,H,1.5e2);
x = ([-1, 1/8, 1/16, 1, 1/16, 1/8,-1]+1)/2; y = [-1/20,-1/20,-1/10, 0, 1/10, 1/20, 1/20]/3;
for i=1:length(omega_t); for j=1:length(gamma_z)
        Exx=x*cos(omega_t(i)-gamma_z(j)); Exy=y*sin(omega_t(i)-gamma_z(j)+psi);
        Eyx=y*sin(omega_t(i)-gamma_z(j)); Eyy=x*cos(omega_t(i)-gamma_z(j)+psi);
        Ex(i,j,:)=Exx+Eyx; Ey(i,j,:)=a*(Exy+Eyy); end; end
figure('units','normalized','outerposition',[0 0 1 1]); hold on; box on;
grid on; g1 = hgtransform; g2 = hgtransform; g3 = hgtransform;
text(0,0,1.25*H,'\bfZ','FontSize',45,'Color','k'); xlabel('\bf\Theta-axis'); ylabel('\bf\phi-axis');
ht(1)=text(1,-1,1,'\bfE_\theta vs. Z-coordinate'); set(ht(1),'FontSize',25,'Rotation',90)
ht(2)=text(-1,1,1,'\bfE_\phi vs. Z-coordinate'); set(ht(2),'FontSize',25,'Rotation',90,'Color','r')
v1=[0,0,0]; v2=[0,0,1.1*H]; v=[v2;v1]; hp=plot3(v(:,1),v(:,2),v(:,3),'k'); set(hp,'LineWidth',8);
r=-linspace(0,pi,10); th=linspace(0,2*pi,20); [R,T]=meshgrid(r,th);
X=R.*cos(T)/25; Y=R.*sin(T)/25; Z=R+H*1.25; s=surf(X,Y,Z); set(s,'FaceColor',[0 0 0]);
   for j=1:length(gamma_z)
        xlim([-1 1]); ylim([-1 1]); zlim([0 1.25]*H); view(144,33); axis square
        hpt1=patch('XData',Ex(1,j,:),'YData',Ey(1,j,:),'FaceColor','b','Parent',g1);
        hpt2=patch('XData',Ex(2,j,:),'YData',Ey(2,j,:),'FaceColor','r','Parent',g2);
        hpt3=patch('XData',Ex(3,j,:),'YData',Ey(3,j,:),'FaceColor','g','Parent',g3);
        g1.Matrix = makehgtform('translate',[0,0,gamma_z(j)]);
        g2.Matrix = makehgtform('translate',[0,0,gamma_z(j)]);
        g3.Matrix = makehgtform('translate',[0,0,gamma_z(j)]);
        h1=g1.Matrix; h2=g2.Matrix; h3=g3.Matrix;
        scatter3(hpt1.XData(4),hpt1.YData(4),h1(3,4),140,'b','filled');
        scatter3(hpt2.XData(4),hpt2.YData(4),h2(3,4),140,'r','filled');
        scatter3(hpt3.XData(4),hpt3.YData(4),h3(3,4),140,'g','filled');
        if j==1; M=100; else; M=20; end;
        scatter3(hpt1.XData(4),hpt1.YData(4),0,10,'m*');
        scatter3(-1,hpt1.YData(4),h1(3,4),M,'r','filled');
        scatter3(hpt1.XData(4),-1,h1(3,4),M,'k','filled');
        scatter3(hpt2.XData(4),hpt2.YData(4),0,10,'m*');
        scatter3(-1,hpt2.YData(4),h2(3,4),M,'r','filled');
        scatter3(hpt2.XData(4),-1,h2(3,4),M,'k','filled');
        scatter3(hpt3.XData(4),hpt3.YData(4),0,10,'m*');
        scatter3(-1,hpt3.YData(4),h3(3,4),M,'r','filled');
        scatter3(hpt3.XData(4),-1,h3(3,4),M,'k','filled');   drawnow
        if j~=length(gamma_z) delete(hpt1); delete(hpt2); delete(hpt3); end
   end; delete(ht); camroll(-130);
```