B)
Single trial simulation that gives a 3D MATLAB visual of the cylinder and the gamma trajectory.

```
% Same as in the original code.


r=1.0000;       % inner radius of the beaker. Has to be 1 for this code.
thickness=3;    % outer radius of the beaker

% Specific source point.
z1=0.2599;      % for this simulation the height of the cylinder is always 1.
x1=0;
y1=1.5;

% getting random trajectory
    R2=5;                       % the length of the trajectory
    theta= acosd(1-2*rand());   % uniform theta distribution in degrees. REMEBER this is in FLOATING
POINT
    Phi= 2*180*rand();          % uniform phi distribution in degrees. REMEBER this is in FLOATING POINT.

    x2p= R2*sind(theta)*cosd(Phi); % measured in the source coordinate
    y2p= R2*sind(theta)*sind(Phi); % measured in the source coordinate
    z2p= R2*cosd(theta);        % measured in the source coordinate
    x2=x2p+x1;                  % measured in the cylinder coordinate
    y2=y2p+y1;                  % measured in the cylinder coordinate
    z2=z2p+z1;                  % measured in the cylinder coordinate

% plotting point
plot3([x1,x2],[y1,y2],[z1,z2]);
hold on

% plots a cylinder in 3d which can be rotated in 3d
cylinder(r);                    % a cylinder with radius 1
colormap([1,1,1]);
hold on
daspect([1 1 1])                % stretches the sphere
axis ([-3 3 -3 3])
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
grid on
rotate3d on

Condition=0;                    % initializing the condition
```

```
% counting hits
% checking for exception conditions where Phi blows up

if (ceil(Phi*10000)/10000== 180.00001/2) && (ceil(theta*10000)/10000 == 180.00001/2 ) % comparing up to four
sig-fig
    xaa=x1;
    xSource=xaa-x1;               % always zero

    % check for the absolute value of x
    if abs(x1)== r
        % tangent condition so does not matter which direction the
        % trajectory goes. But we are not going to count this as a hit.

        Condition=0;
    elseif abs(x1)<r
        % intersects the circle at two points

        Condition=1;
        %since xaa = x1 the y of the source and the cylinder coordinate is
        %the same

        yaa1=(r^2-(xaa)^2)^.5;
        yaa2=-(r^2-(xaa)^2)^.5;

        ySourceCoa1=yaa1-y1;        % converting into source coordinate
        ySourceCoa2=yaa2-y1;

        % check for direction
        % For this condition the aPhiPrime1 and aPhiPrime2 will be positive
        % 90 no matter what.

        aPhiPrime1= atand(ySourceCoa1/xSource);    % finds Phi in the source coordinate
        aPhiPrime2= atand(ySourceCoa2/xSource);

        % for aPhiPrime1
        if xaa>0 && ySourceCoa1>0
            PhiPrime1=aPhiPrime1;
        elseif xaa<0 && ySourceCoa1>0
            PhiPrime1=aPhiPrime1 +180;
        elseif xaa<0 && ySourceCoa1<0
            PhiPrime1=aPhiPrime1+180;
        else
            PhiPrime1=aPhiPrime1+360;
        end

        % for aPhiPrime2
        if xaa>=0 && ySourceCoa2>0
```

```
            PhiPrime2=aPhiPrime2;
        elseif xaa<0 && ySourceCoa2>0
            PhiPrime2=aPhiPrime2+180;
        elseif xaa<=0 && ySourceCoa2<0
            PhiPrime2=aPhiPrime2+180;
        else
            PhiPrime2=aPhiPrime2+360;
        end

        % rounding to four sig digits
        Phi=ceil(Phi*10000)/10000;
        PhiPrime1=ceil(PhiPrime1*10000)/10000;
        PhiPrime2=ceil(PhiPrime2*10000)/10000;
        fprintf('PhiPrime1= %f PhiPrime2=%f PhiPrime =%f',PhiPrime1,PhiPrime2,Phi)

        % check if the PhiPrime == Phi
        if (PhiPrime1==Phi) || (PhiPrime2==Phi)
            Condition=3;
        end

    else
        % It does not intersect the circle at all
        Condition=0;
    end


elseif(Phi== (90))&& (theta==0)

    % never touches the detector
    Condition=0;
else
    % getting x and y projection
    % r= radius of the cylinder

    a=1+(tand(Phi))^2;
    b=2*tand(Phi)*(y1-x1*tand(Phi));
    c=(x1^2)*(tand(Phi))^2-2*x1*y1*tand(Phi)+(y1^2)-r^2;

    xa= (-b+((b^2)-4*a*c)^.5)/(2*a);
    xb= (-b-((b^2)-4*a*c)^.5)/(2*a);

    if (isreal(xa)||isreal(xb))==1
        Condition=1;

        % Checking for condition 2 (if it hits the detector)
        xSourceCo1=xa-x1;          % finding new value of x prime in the source coordinate
        xSourceCo2=xb-x1;
        zSourceCo1=(xSourceCo1 * cosd(theta))/(sind(theta)*cosd(Phi)); % finding value of z prime in the
```

```
source coordinate
        zSourceCo2=(xSourceCo2 * cosd(theta))/(sind(theta)*cosd(Phi));
        zCylinderCo1=zSourceCo1+z1;        % getting the value of z prime in the cylinder coordinate
        zCylinderCo2=zSourceCo2+z1;

        if  (zCylinderCo1>0 && zCylinderCo1<1) || (zCylinderCo2>0 && zCylinderCo2<1)  %checking if the z
value
            Condition=2;

            % checking for condition 3 (direction of the trajectory)
            % getting all the values of y
            ya1=-((r^2)-(xa^2))^.5;        % finds the y when the trajectory intersects the cylinder
            ya2=-ya1;
            yb1=((r^2)-(xb^2))^.5;
            yb2=-yb1;

            ySourceCo11=ya1-y1;            % converting into source coordinate
            ySourceCo12=ya2-y1;
            ySourceCo21=yb1-y1;
            ySourceCo22=yb2-y1;

            % getting all the values of PhiPrime

            PhiPrime11= atand(ySourceCo11/xSourceCo1);   % finds the Phi in the source coordinate
            PhiPrime12= atand(ySourceCo12/xSourceCo1);
            PhiPrime21= atand(ySourceCo21/xSourceCo2);
            PhiPrime22= atand(ySourceCo22/xSourceCo2);

            % checking for the different quadrant condition
            % for PhiPrime 11

            if xSourceCo1>0 && ySourceCo11>0
                PhiPrime1=PhiPrime11;
            elseif xSourceCo1<0 && ySourceCo11>0
                PhiPrime1=PhiPrime11+180;
            elseif xSourceCo1<0 && ySourceCo11<0
                PhiPrime1=PhiPrime11+180;
            else
                PhiPrime1=PhiPrime11+360;
            end

            % for PhiPrime 12

            if xSourceCo1>0 && ySourceCo12>0
                PhiPrime2=PhiPrime12;
            elseif xSourceCo1<0 && ySourceCo12>0
                PhiPrime2=PhiPrime12+180;
            elseif xSourceCo1<0 && ySourceCo12<0
```

```
    PhiPrime2=PhiPrime12+180;
else
    PhiPrime2=PhiPrime12+360;
end

% for PhiPrime 21

if xSourceCo2>0 && ySourceCo21>0
    PhiPrime3=PhiPrime21;
elseif xSourceCo2<0 && ySourceCo21>0
    PhiPrime3=PhiPrime21+180;
elseif xSourceCo2<0 && ySourceCo21<0
    PhiPrime3=PhiPrime21+180;
else
    PhiPrime3=PhiPrime21+360;
end

% for PhiPrime 22

if xSourceCo2>0 && ySourceCo22>0
    PhiPrime4=PhiPrime22;
elseif xSourceCo2<0 && ySourceCo22>0
    PhiPrime4=PhiPrime22+180;
elseif xSourceCo2<0 && ySourceCo22<0
    PhiPrime4=PhiPrime22+180;
else
    PhiPrime4=PhiPrime22+360;
end

% round off to four sig-fig
Phi=ceil(Phi*10000)/10000;
PhiPrime1=ceil(PhiPrime1*10000)/10000;
PhiPrime2=ceil(PhiPrime2*10000)/10000;
PhiPrime3=ceil(PhiPrime3*10000)/10000;
PhiPrime4=ceil(PhiPrime4*10000)/10000;

 % finding the two right PhiPrime to get the values of y and x
 % there is two correct value for PhiPrime. One for xa and other for xb

    if PhiPrime1==Phi
        CorrectPhiPrime1=PhiPrime1;
        yf1=ya1;
        x=xa;
        Condition=3;
    end
    if PhiPrime2==Phi
        CorrectPhiPrime1=PhiPrime2;
        yf1=ya2;
```

```
                    x=xa;
                    Condition=3;
                end
                if PhiPrime3==Phi
                    CorrectPhiPrime2=PhiPrime3;
                    yf2=yb1;
                    x=xb;
                    Condition=3;
                end
                if PhiPrime4==Phi
                    CorrectPhiPrime2=PhiPrime4;
                    yf2=yb2;
                    x=xb;
                    Condition=3;
                end

                % finding the distance travelled by the trajectory in the
                % detector if it hits it

                if Condition==3
                    distance =((xa-xb)^2 + (yf1-yf2)^2)^.5;
                    Si=-(log(1-rand()))*8.3793;
                    if Si<distance
                        Condition =4;
                    end
                end
        end

    end
end



% Checking if the ray satisfied both conditions

if Condition ==4
    Hits=1;
    fprintf('Yeah!! it hit the detector. The two true PhiPrime are % f and %f \n', CorrectPhiPrime1,
CorrectPhiPrime2);
elseif Condition ==3
    frpintf(' the trajectory passes through \n')

else
    fprintf('The gamma ray does not hit the detector :( It satisfied up to condition: %d\n',Condition);
end
```