Atom trapping simulation

February 20, 2017

Theoretical

Calculation of field intensity was done following Kimble A state-insensitive, compensated nanofiber trap, where they got the following equations for the field of a fiber with radius a.

$$E_x\left(r,\phi,z,t\right)=A_{lin}\frac{\beta_{11}J_1\left(h_{11}a\right)}{2q_{11}K_1\left(q_{11}a\right)}\left[\left(1-s_{11}\right)K_0\left(q_{11}r\right)\cos\left(\phi_0\right)+\left(1+s_{11}\right)K_2\left(q_{11}r\right)\cos\left(2\phi-\phi_0\right)\right]e^{i\left(\omega t-\beta_{11}z\right)} \quad (1)$$

$$E_y\left(r,\phi,z,t\right)=A_{lin}\frac{\beta_{11}J_1\left(h_{11}a\right)}{2q_{11}K_1\left(q_{11}a\right)}\left[\left(1-s_{11}\right)K_0\left(q_{11}r\right)\sin\left(\phi_0\right)+\left(1+s_{11}\right)K_2\left(q_{11}r\right)\sin\left(2\phi-\phi_0\right)\right]e^{i\left(\omega t-\beta_{11}z\right)} \quad (2)$$

$$E_z\left(r,\phi,z,t\right)=iA_{lin}\frac{J_1\left(h_{11}a\right)}{K_1\left(q_{11}a\right)}K_1\left(q_{11}r\right)\cos\left(\phi-\phi_0\right)e^{i\left(\omega t-\beta_{11}z\right)} \quad (3)$$

$$s_{11}=\left[\frac{1}{\left(h_{11}a\right)^2}+\frac{1}{\left(q_{11}a\right)^2}\right]\left[\frac{J_1^{'}\left(h_{11}a\right)}{h_{11}aJ_1\left(h_{11}a\right)}+\frac{K_1^{'}\left(q_{11}a\right)}{q_{11}aK_1\left(q_{11}a\right)}\right] \quad (4)$$

$$h_{11}=\sqrt{k_0^2n_1^2-\beta_{11}^2} \quad (5)$$

$$q_{11}=\sqrt{\beta_{11}^2-k_0^2n_2^2} \quad (6)$$

$$I\left(r;\phi=0,z=0,t=0\right)\equiv I\left(r\right)=\sqrt{\left|E_x\right|^2+\left|E_y\right|^2+\left|E_z\right|^2} \quad (7)$$

Here, ϕ denotes the azimuthal position in the transverse plane, ϕ_0 indicates the polarization axis for the input polarization relative to the x axis, n_1 and n_2 are the indices of refraction inside and outside the waveguide, β_{11} is the mode propagation constant, $1/h_{11}$ is the characteristic decay length for the guided mode inside the fiber, $1/q_{11}$

is the characteristic decay length for the guided mode outside the fiber, A_{lin} is the real-valued amplitude for the linearly polarized input, J_l is the lth Bessel function of the first kind and K_l is the lth modified Bessel function of the second kind.

After calculating the field intensity, we want to get the trap potential the ^{87}Rb atoms see. For this we look at a resonance frequencies of the atom and compute the dipole potential a field with certain intensity will cause (Following Optical dipole traps for neutral atoms)

$$U_{dipole}\left(r\right)=\frac{\pi c^2\Gamma}{2\omega_0^3}\left(\frac{2+\mathcal{P}g_Fm_F}{\Delta_{2,F}}+\frac{1-\mathcal{P}g_Fm_F}{\Delta_{1,F}}\right)I\left(r\right) \tag{8}$$

Here g_F is the well-known Landé factor and $\mathcal{P}$ characterizes the laser polarization ($\mathcal{P}=0,\pm1$ for linearly and circularly $\sigma^{\pm}$ polarized light). The detunings $\Delta_{2,F}$ and $\Delta_{1,F}$ refer to the energy splitting between the particular ground state $^2S_{1/2}$, F and the center of the hyperfinesplit $^2P_{3/2}$ and $^2P_{1/2}$ excited states, respectively. The two terms in brackets represent the contributions of the $D2$ and the $D1$ line to the total dipole potential. ω_0 is the optical transition frequency of the $D1$ line, and Γ is the natural line width of this line.

Results

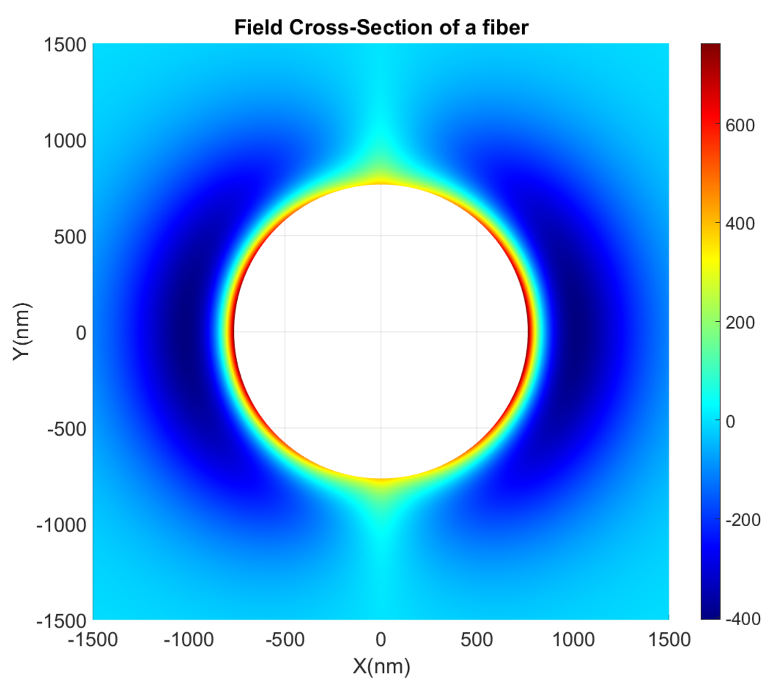

Figure 1: Fiber Red + Blue Field Intensity Cross-Section

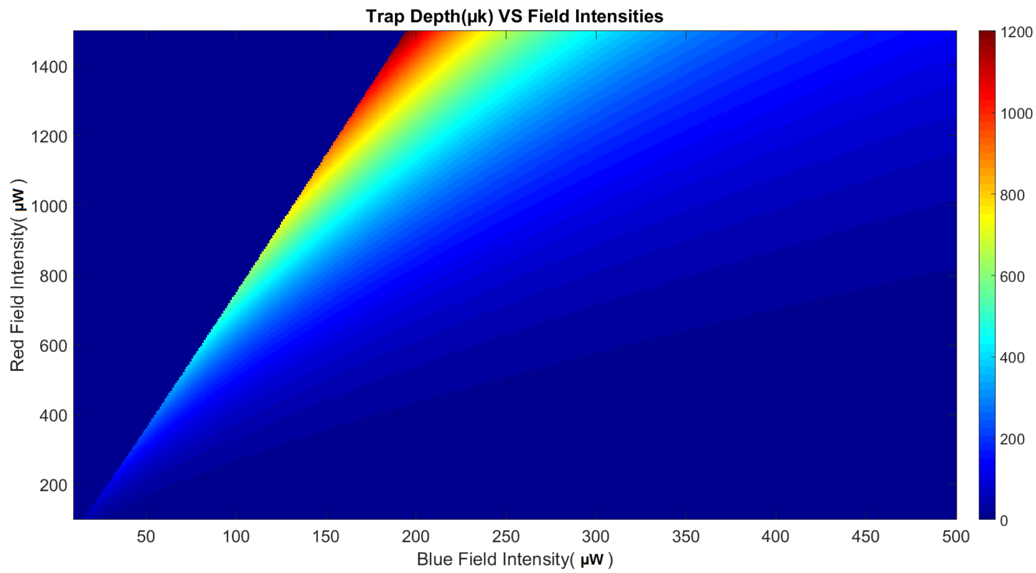

Figure 2: Trap depth for various field intensities

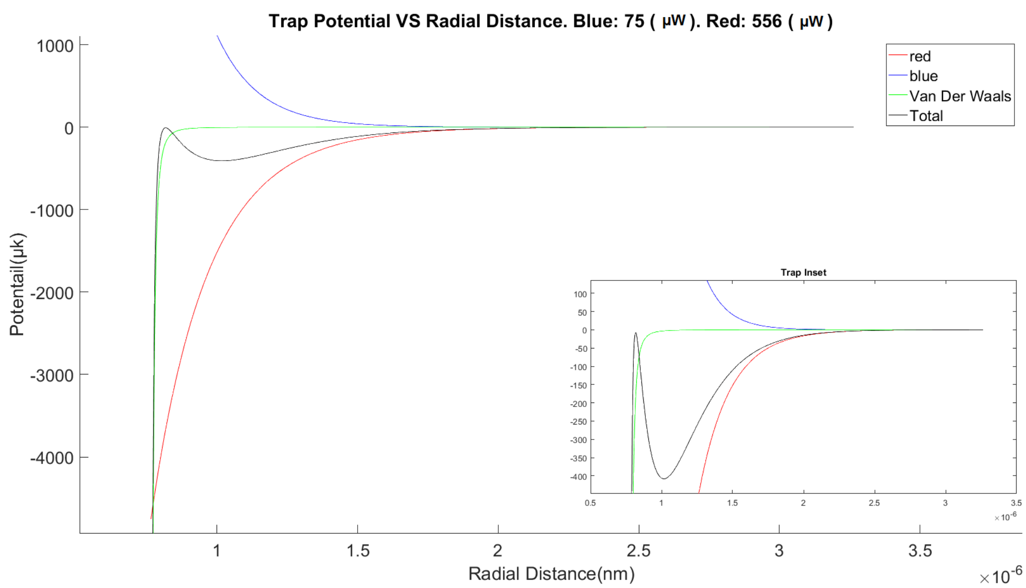

Figure 3: A slice of the total potential

This figure shows the case where the trap depth is 400(μk). Notice how the peak of the barrier that keeping the atom from falling on the toroid is just above zero. There is an interplay here between the height of this barrier and the depth of the trap, so if we want to make extra sure that the atom will not exit the trap to the left (through tunneling), we can raise the barrier, but then we will lose some of the trap depth. Playing with the ratio between red and blue light can also move the trap location farther or closer to the toroid.

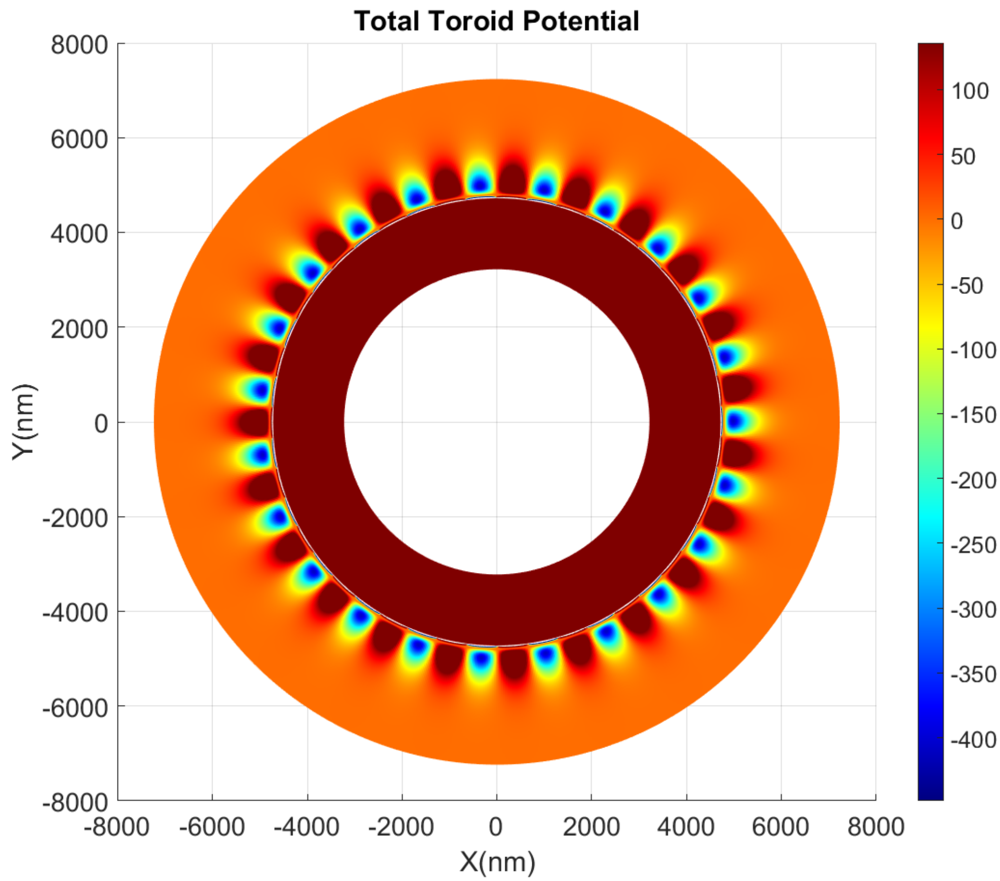

Figure 4: Top view of trap potential

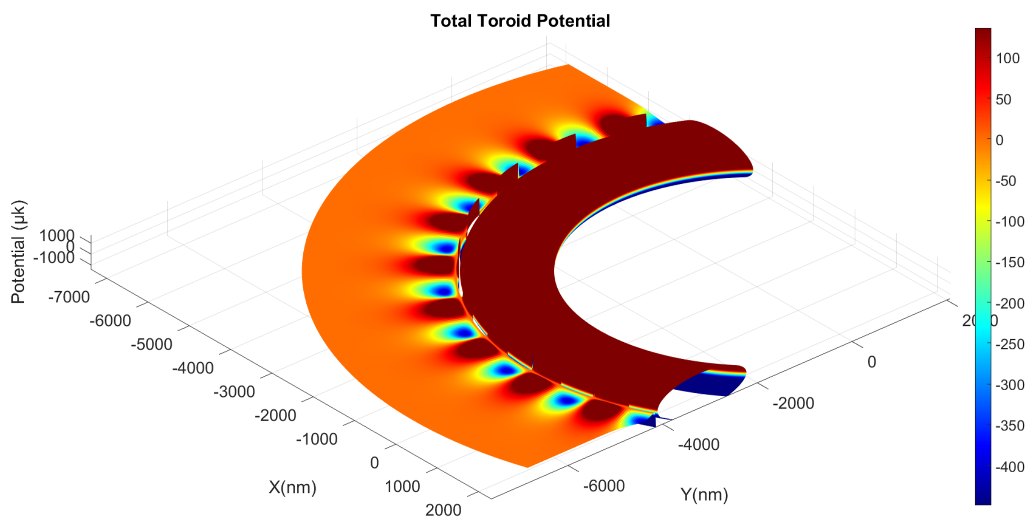

Figure 5: Perspective view of trap potential

Matlab Code

CalculateEvanescentElectricField.m

```matlab
function [ElectricFieldEnergy,r,phi] = CalculateEvanescentElectricField(lambda,A_in,phi_0,FiberRadius,mode_area,RadialVector,AzimuthalVector)
% Returs the energy of the electric field (microK) and the polar coordinates used to plot it correctly.
% The electric field calculations are taken from kimble article - http://iopscience.iop.org/article/10.1088/1367-2630/14/2/023056/pdf
%
% Input parameters :
% lambda          - Input field wavelength
% A_in            - Input field intensity
% phi_0           - polarization angle relative to the fiber
% FiberRadius     - Radius of waveguide
% beta            - Mode propagation constant
% mode_area       - Effective area (cm^2) occupied by the mode
% RadialVector    - A vector that holds the points in the radial direction
% AzimuthalVector - A vecotr that holds the angles for the field calculation. Can also be a scalar.

    n1 = 1.447;                         % Refractive Index Inside Waveguide
    n2 = 1.0;                           % Refractive Index Outside Waveguide
    k0 = 2*pi/lambda;                   % Wave number
    beta = (n1+n2)/2*k0;
    h11 = sqrt(k0^2*n1^2-beta^2);      % Characteristic Decay Length Inside Fiber
    q11 = sqrt(beta^2-k0^2*n2^2);      % Characteristic Decay Length Outside Fiber
    a =FiberRadius;                    % Fiber Radius nanometer (changing letters for bervity)
    Kb = 1.38e-23;                     % Boltzmann constant Joule/Kelvin
    Gamma = 5.746e6;                   % [Hz] in the 1*gamma formulations (books, not ofer)
    c = 3e8;                           % Speed of light  [m/sec]
    finesse = 2.5e4;                   % finnes of the cavity
    lambda_res1 = 795e-9;              % rubidium87 resonance D1 [meter]
    lambda_res2 = 780e-9;              % rubidium87 resonance D2 [meter]
    omega0 = 2*pi*c/lambda_res1;       % Optical transition frequency of D1
    omega1 = 2*pi*c/lambda_res2;       % Optical transition frequency of D2
    omegaIn = 2*pi*c/lambda;

```

```
    D_besselk = @(nu,z) 0.5*(besselk(nu-1,z)-besselk(nu+1,z));  % Auxillary function for the derivative of besselk function
    D_besselj = @(nu,z) 0.5*(besselj(nu-1,z)-besselj(nu+1,z));  % Auxillary function for the derivative of besselj function

    %% Calculate filed intensity I(r)
    s11 = (1/(h11*a)^2+1/(q11*a)^2)*(D_besselj(1,h11*a)/(h11*a*besselj(1,h11*a))+D_besselk(1,q11*a)/(q11*a*besselk(1,q11*a)));
    A = A_in*beta*besselj(1,h11*a)/(2*q11*besselk(1,q11*a)); B = 1i*A_in*besselj(1,h11*a)/besselk(1,q11*a);

    if length(AzimuthalVector) > 1
        [r, phi]  = ndgrid(RadialVector,AzimuthalVector);
    else
        phi = AzimuthalVector;
        r = RadialVector;
    end

    Ex = A*((1-s11)*besselk(0,q11*r).*cos(phi_0)+(1+s11)*besselk(2,q11*r).*cos(2*phi-phi_0));
    Ey = A*((1-s11)*besselk(0,q11*r)*sin(phi_0)+(1+s11)*besselk(2,q11*r).*sin(2*phi-phi_0));
    Ez = B*besselk(1,q11*r).*cos(phi-phi_0);
    ElectricFieldIntensity = sqrt(abs(Ex).^2+abs(Ey).^2+abs(Ez).^2);

    %% Calculate trap energy U(r)
    I = ElectricFieldIntensity/mode_area*finesse;
    delta0 = omegaIn - omega0;
    delta1 = omegaIn - omega1;
    U = (pi*c^2*Gamma/(2*omega0^3))*(2/delta1+1/delta0)*I;
    ElectricFieldEnergy = 1e6*U/Kb; % [microK]
end
```

EvanescentFieldOutsideMicrotoroid.m

```
close all
clear all
%% constants
Phi_0 = 0.0;                              % Input Polariztion Angle
LambdaRed = 1085e-9;                      % Red WaveLength [meter]
LambdaBlue = 729e-9;                      % Blue WaveLength [meter]
ToroidSmallRadius = 750e-9;               % Fiber Radius [meter]
Kb = 1.38e-23;                            % Joule/Kelvin
A_in_Red = 556e-3;                        % mW
A_in_Blue = 75e-3;                        % mW
h_bar = 1.054e-34;                        % meter
ToroidLargeRadius = 3971.7e-9;            % meter
c3 = 1e6*(1.6e-19*4.9*(1e-10)^3)*(1e9)^3/Kb; % [microKelvin*nm^3]
mode_area = 300e-8/(40*pi);               % Area occupied by the mode [cm^2]
ROILength = 2500e-9;                      % length of Region Of Interest away from the fiber
Margin = 15e-9;                           % Too close to the surface the plot explode because of vdw
RadialVector = linspace(ToroidSmallRadius+Margin,ToroidSmallRadius+Margin+ROILength,1000);
Optimization = false;

%% Find optimal values for the intensity of the blue and red fields
if Optimization
    OptimizationN = 501;
    TrapDepth = zeros(OptimizationN,OptimizationN);
    TrapSpatialWidth = zeros(OptimizationN,OptimizationN);
    Power1 = linspace(10,500,OptimizationN) *1e-3 / A_in_Blue;
    Power2 = linspace(100,1500,OptimizationN) *1e-3 / A_in_Red;
    VanDerWaals=-c3./(((RadialVector-ToroidSmallRadius)*1e9).^3);
    AzimuthalVector=0;
    RedFieldBase=CalculateEvanescentElectricField(LambdaRed,A_in_Red,Phi_0,ToroidSmallRadius,
      mode_area,RadialVector,0);
    [BlueFieldBase,r,phi] = CalculateEvanescentElectricField(LambdaBlue,A_in_Blue,Phi_0,
      ToroidSmallRadius,mode_area*0.67,RadialVector,0);
    for i = 1:OptimizationN
        for j = 1:OptimizationN
            RedField = RedFieldBase * Power2(j);
            BlueField = BlueFieldBase * Power1(i);
```

```
            TotalField = RedField+BlueField+VanDerWaals;
            TrapDepth(i,j) = GetTrapDepth(TotalField);
        end
    end
    %% plot optimization process results
    Power1 = Power1 * A_in_Blue;
    Power2 = Power2 * A_in_Red;
    imagesc(Power1*1000,Power2*1000, TrapDepth')
    title('Trap Depth(microK) VS Field Intensities');
    xlabel('Blue Field Intensity(microW)');
    ylabel('Red Field Intensity(microW)');
    set(gca,'Ydir','Normal');
    set(gca,'Xdir','Normal');
    set(gca,'FontSize',20)
    colormap jet
    [C,I] = max(TrapDepth(:));
    [blueInd,redInd] = ind2sub(size(TrapDepth),I);
else

    %% calculate for a specific intensity
    AzimuthalVector = linspace(0,2*pi,1000);
    RedField = CalculateEvanescentElectricField(LambdaRed,A_in_Red,Phi_0,ToroidSmallRadius,
      mode_area,RadialVector,AzimuthalVector);
    [BlueField,r,phi] = CalculateEvanescentElectricField(LambdaBlue,A_in_Blue,Phi_0,
      ToroidSmallRadius,mode_area*0.67,RadialVector,AzimuthalVector);
    VanDerWaals = -c3./(((r-ToroidSmallRadius)*1e9).^3);
    TotalField = BlueField+RedField+VanDerWaals;
    [x,y] = pol2cart(phi,r);

    %% plot fiber cross section     figure(1)
    surf(x*1e9,y*1e9,BlueField+RedField,'EdgeColor','none','LineStyle','none','FaceLighting','
      phong');
    view(0,90)
    colormap(jet(1024))
    xlim([-ToroidSmallRadius*2e9 ToroidSmallRadius*2e9])
    ylim([-ToroidSmallRadius*2e9 ToroidSmallRadius*2e9])
    daspect([max(daspect)*[1 1] 1]);
    colorbar();
    title('Field Cross-Section of a fiber');
    xlabel('X(nm)');
```

```matlab
    ylabel('Y(nm)');
    set(gca,'Ydir','Normal');
    set(gca,'FontSize',20)

    %% plot radial dependence at maximum
    figure(2)
    hold on
    if length(AzimuthalVector) > 1
        TotalFieldSlice = TotalField(:,1);
        plot(r(:,1)',RedField(:,1)','r')
        plot(r(:,1)',BlueField(:,1)','b')
        plot(r(:,1)',VanDerWaals(:,1)','g')
        plot(r(:,1)', TotalField(:,1)','k');
        legend('red','blue','Van Der Waals','Total');
        y1 = ylim;
        ylim([y1(1)/2 max(TotalFieldSlice)*1.1]);
        title(sprintf('Trap Potential VS Radial Distance. Blue: %d (mW). Red: %d (mW)',
          A_in_Blue*1000,A_in_Red*1000));
        xlabel('Radial Distance(nm)');
        ylabel('Potentail(microK)');
        set(gca,'Ydir','Normal');
        set(gca,'FontSize',20)
        axes('position',[.55 .175 .35 .35])
        box on
        hold on
        title('Trap Inset')
        plot(r(:,1)',RedField(:,1)','r')
        plot(r(:,1)',BlueField(:,1)','b')
        plot(r(:,1)',VanDerWaals(:,1)','g')
        plot(r(:,1)', TotalField(:,1)','k');
        peak = findpeaks(-TotalFieldSlice(40:end));
        ylim([-peak*1.1 peak/3]);
    else
        plot(r,RedField,'r')
        plot(r,BlueField,'b')
        plot(r,VanDerWaals,'g')
        plot(r, TotalField,'k');
        legend('red','blue','Van Der Waals','Total');
        y1 = ylim;
        ylim([y1(1)/2 max(TotalField)*1.1])
```

```
        title(sprintf('Trap Potential VS Radial Distance. Blue: %d (mW). Red: %d (mW)',
          A_in_Blue*1000,A_in_Red*1000));
        xlabel('Radial Distance(nm)');
        ylabel('Potentail(microK)');
        set(gca,'Ydir','Normal');
        set(gca,'FontSize',20)
        axes('position',[.55 .175 .35 .35])
        box on
        hold on
        title('Trap Inset')
        plot(r,RedField,'r')
        plot(r,BlueField,'b')
        plot(r,VanDerWaals,'g')
        plot(r, TotalField,'k');
        peak = findpeaks(-TotalField(40:end));
        ylim([-peak*1.1 peak/3])
    end

    %% plot 3d potential
    figure(3)
    NumOscillationsInToroid = round(2*pi*ToroidLargeRadius/LambdaRed);
    ToroidField = repmat(TotalField(:,1),1,size(phi,2)) .* cos(NumOscillationsInToroid*phi);
    [x,y] = pol2cart(phi,r+ToroidLargeRadius);
    surf(x*1e9,y*1e9,ToroidField,'EdgeColor','none','LineStyle','none','FaceLighting','phong')
      ;
    colormap(jet(1024))
    title('Total Toroid Potential');
    xlabel('X(nm)');
    ylabel('Y(nm)');
    zlabel('Potential (microK)');
    set(gca,'FontSize',20)
    daspect([max(daspect)*[0.01 0.01] 0.1])
    caxis([-peak*1.1 peak/3])
    zlim([-peak*6 peak*7])

    %% plot toroid along with potentail
    hold on
    ToroidColormap = gray(256);
    ToroidColormap = ToroidColormap(1:200,:);
    ToroidColormap = cat(1,ToroidColormap,flip(ToroidColormap,1));
```

```
    [ToroidBodyX,ToroidBodyY,ToroidBodyZ] = Torus(ToroidLargeRadius*1e9,ToroidSmallRadius*1e9,360);
    ToroidBodyZ = ToroidBodyZ/max(max(ToroidBodyZ))*max(max(ToroidBodyX))*ToroidSmallRadius/ToroidLargeRadius*2;
    surf(ToroidBodyX,ToroidBodyY,ToroidBodyZ,'EdgeColor','none','LineStyle','none','FaceLighting','phong');
end
```

GetTrapDepth.m

```
function [TrapDepth,TrapLocation] = GetTrapDepth(Field)
    if max(Field) < 0
        TrapDepth = 0;
        return;
    end
    dField = diff(Field(50:round(length(Field)/2)));
    [~,TrapLocation] = min(abs(dField));
    if TrapLocation == length(dField) || dField(TrapLocation-1)*dField(TrapLocation+1) > 0
        TrapDepth = 0;
    else
        TrapDepth = abs(Field(TrapLocation+49));
    end
end
```