SUPPORTING INFORMATIONS TO:

Continuous-Flow Flat Jet Setup for Uniform Pulsed Laser Postprocessing of Colloids

Swen Zerebecki[1,2], Sven Reichenberger*[1,2], Stephan Barcikowski[1,2]

[1]Technical Chemistry I and Center for Nanointegration Duisburg-Essen (CENIDE), University of Duisburg-Essen, 45141 Essen, Germany

[2]NanoEnergieTechnikZentrum (NETZ), University of Duisburg-Essen, 47057 Duisburg, Germany

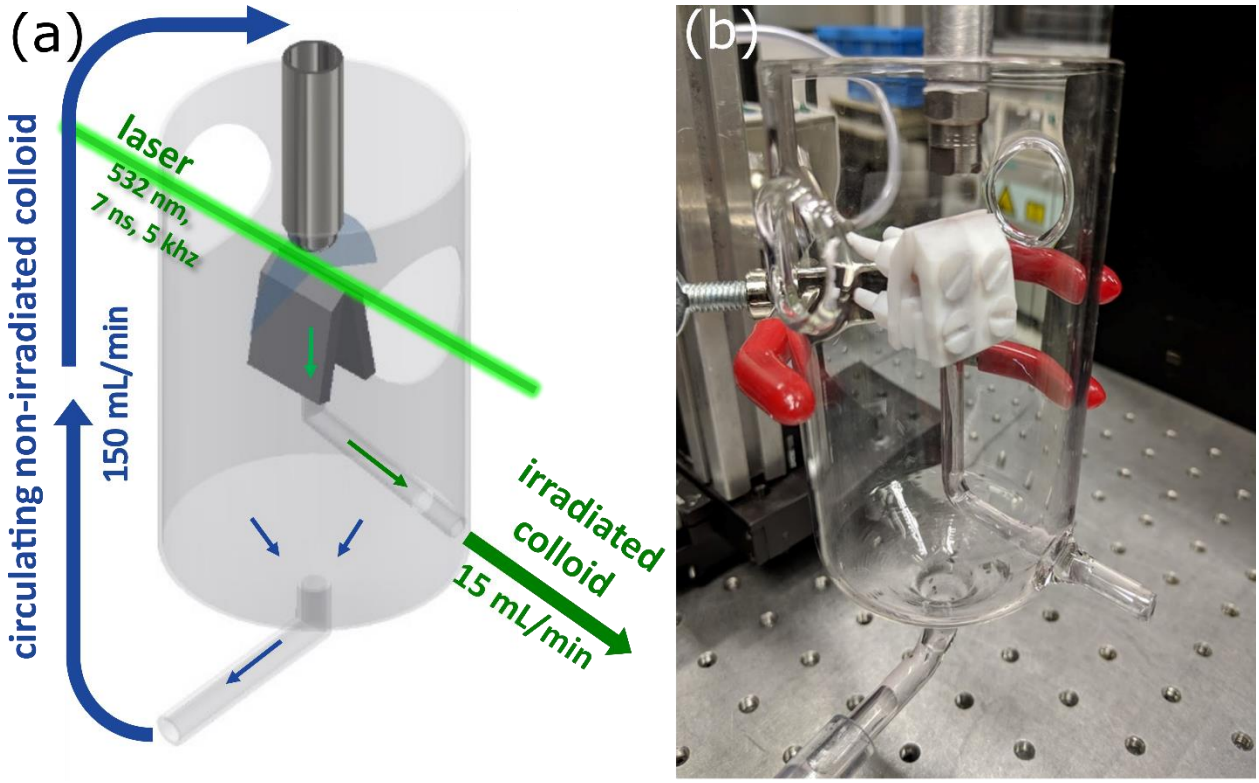

Fig. S1. (a) Schematic drawing of the separation funnel for the flat jet setup that allows the separation of the irradiated liquid (form the homogeneous part of the flat jet) and the unirradiated liquid. The unirradiated liquid can be recycled back through the nozzle. (b) Photograph of the home-made glass separation funnel. The separation funnel consists of a 6 cm in diameter beaker with an outlet at the bottom. Inside the beaker, an additional glass tube (4 mm inner diameter)

leads from the middle of the beaker sideways outside. On top of this middle glass tube, sit two sharpened 3 mm thick PTFE sheets (held together by PTFE screws). These PTFE-knifes are positioned directly below the laser beam and cut out the irradiated 1 mm wide section of the flat jet. The irradiated liquid flows through the additional middle tube and gets removed from the system, while the unirradiated liquid is circulation from the bottom of the beaker through a pump back to the top and again through the flat jet nozzle.

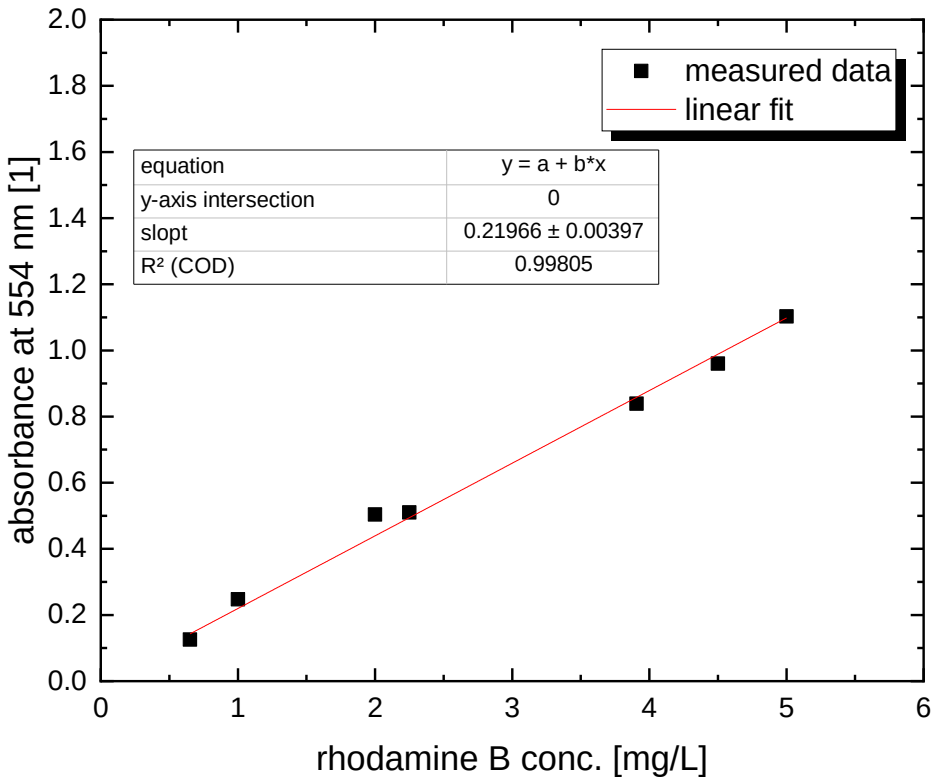

Fig. S2. Calibration curve of rhodamine B solutions

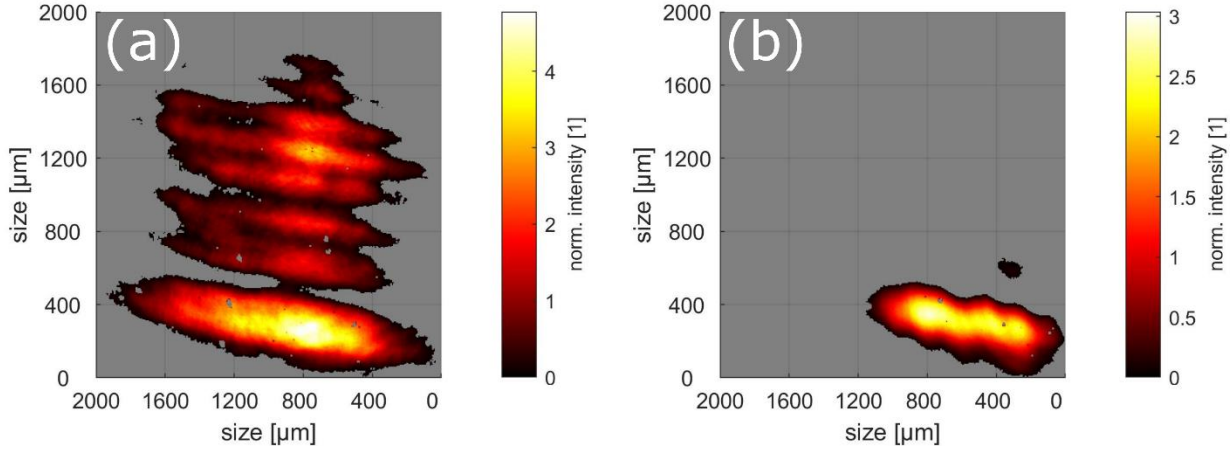

Figure S3. (a) Raw ($1/e^2$) beam profile of the utilized Nd-YAG slab laser. (b) Cutted beam profile after optimization with two mechanical slits.

$$PPV = \frac{\tau_S}{f_{rep}} = \frac{S_H\ \sigma_{liq}}{\dot{V}\ f_{rep}} \qquad \text{(eq. S1)}$$

(eq. S1). Calculation of the applied pulses per volume element (PPV)

Tab. S1. Measured and adjusted parameters for single-pulse-per-volume-element-conditions (PPV-conditions).

Laser parameters	
Laser spot hight	360 µm
Repetition rate	4500 Hz
Flat jet setup	
Thickness flat jet	130 µm
Width flat jet (PTFE-knifes, receiver of the irradiated liquid)	1 mm
Volumetric flow rate (flat jet)	12 mL min
PPV	1.05
Cylindrical jet setup	
Cylindrical jet diameter	1.03
Volumetric flow rate (cyl. Jet)	77 mL/min
PPV	1.05

$$U = \frac{1}{\bar{E}} \sqrt{\frac{1}{A} \iint [E(x,y) - \bar{E}]^2 dxdy} \qquad \text{(eq. S2)}$$

(eq. S2). Calculation of the intensity deviation according to ISO13694:2000

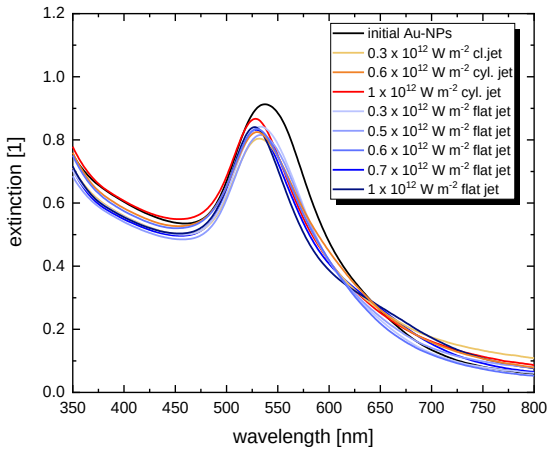

Fig. S4. Measured UV/vis spectra of Au-NPs after PLFL, with the two different liquid jet setups at the different laser intensities.

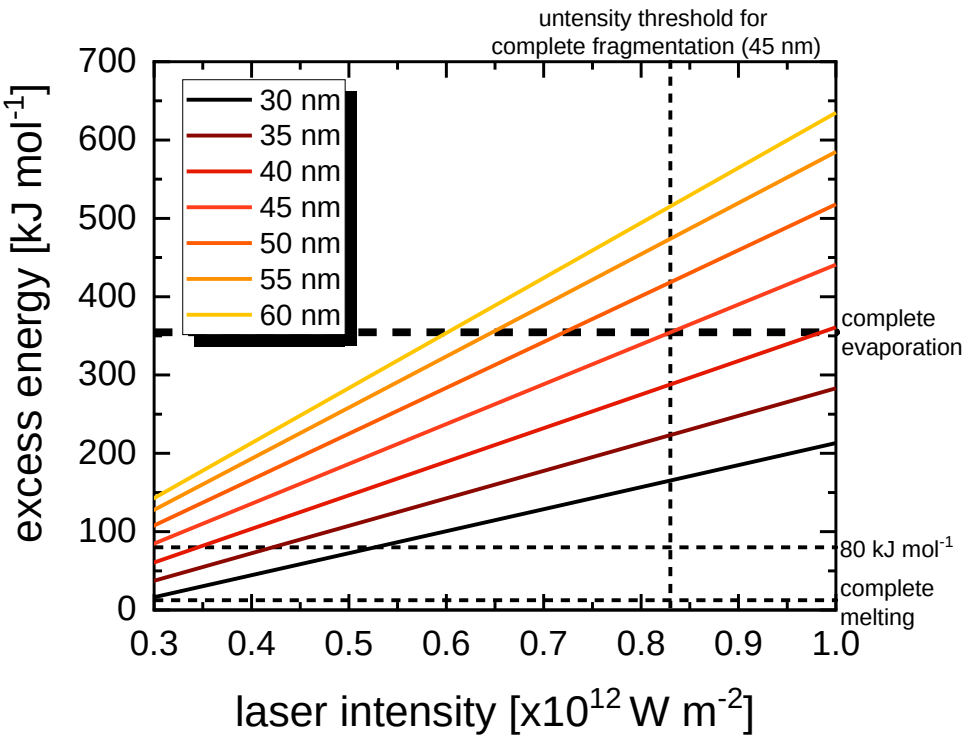

Fig. S5. Calculated maximal excess energy for the phase transitions of Au-particles after reaching the boiling temperature versus the applied laser intensity for a single 7-ns, 532 nm laser pulse. Based on the calculation of the core temperature of Au-nanoparticles in accordance with Baffou et al. [15,16]

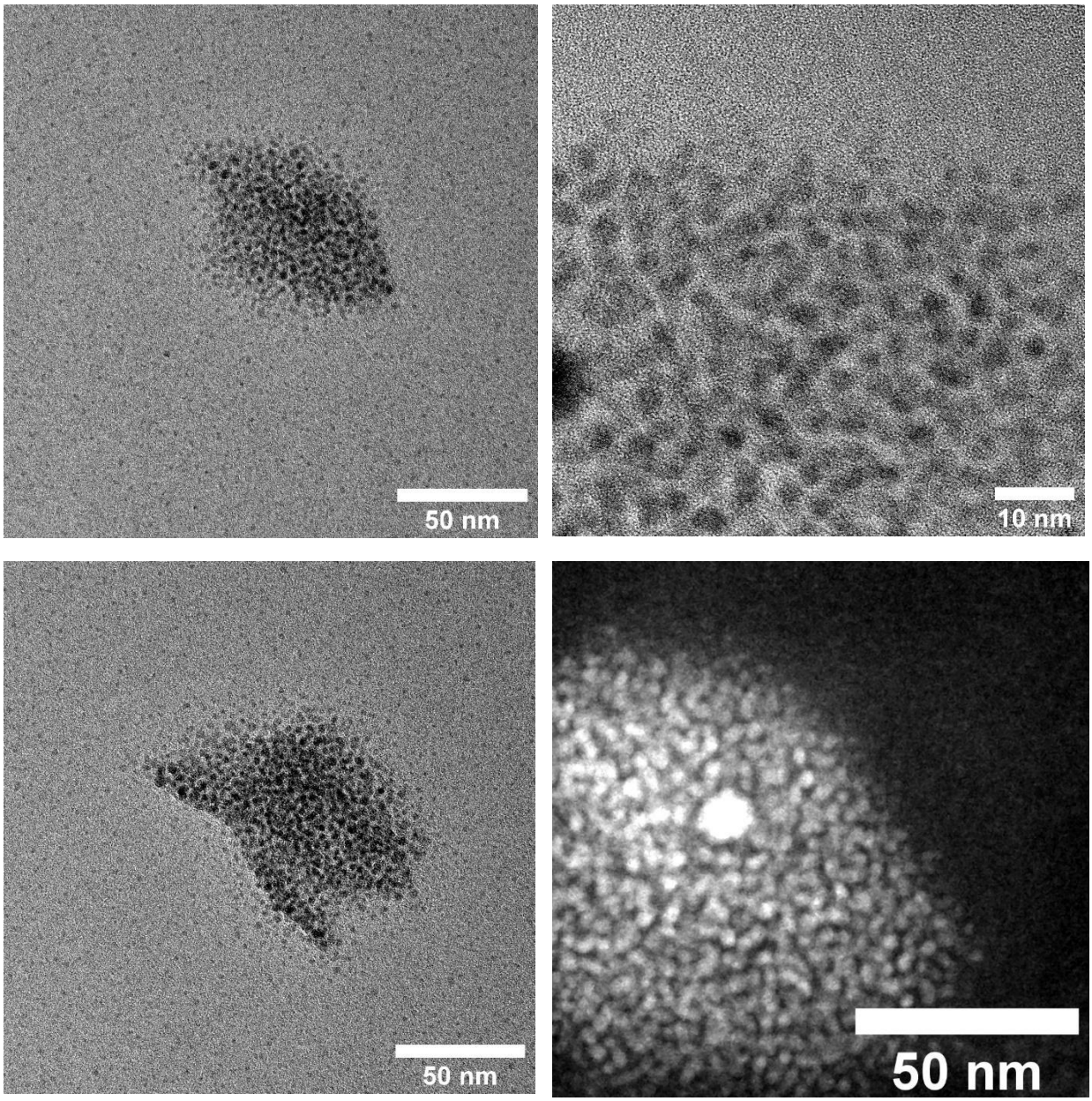

Fig. S6. Additional TEM-pictures of Au nanoclusters from LFL at $1 \cdot 10^{12}$ W m^{-2}

Raytracing Matlab program:

```
clearvars -except data_nonnan %Clear everything exept Beamprofile data


n = 1.31; %Index of refraction of water

dz = 1; %Step size for computation purposes
        %dy = 1; %Separation between rays

aperture = 500;
number_rays = 1000;
number_rays=number_rays;
dy = (2*aperture)/number_rays;
y = -aperture+dy/2:dy:aperture-dy/2; %Field of view


%% Ray matrix
[raymatrix,x_optaxis,zmax,l_ray_f_air,l_ray_lens] = flatjet(n,dz,y); %Select
flatjet of sphere


%Figures
figure(1)
plot(x_optaxis,raymatrix','b') %Rays
hold on

plot(x_optaxis,zeros(1,length(x_optaxis)),'k--') %Optical axis
hold off

%% Energydensity

fluence=zeros(1,number_rays)+100;

%% Taiking the intensity distribution of a crosssection of the beam profile
        %%%%%%%%%First Run AnalysisofBeamprofile!
% Intensitys = rmmissing(data_nonnan(:,50));
% span = length(Intensitys)/20;
% Intensitys = smooth(Intensitys,span(1,1));
% dti=(length(Intensitys)/number_rays);
% ti=(dti/2:dti:length(Intensitys)-dti/2); %dti 9 in test and 12 in test raw
for 2000 rays
% Interpol_intensitys=interp1(Intensitys,ti);
% Interpol_intensitys(isnan(Interpol_intensitys))=0;
% Interpol_intensitys=Interpol_intensitys/max(Interpol_intensitys);
% figure (3)
% plot(Interpol_intensitys)
% set(gcf,'PaperUnits','inches','PaperPosition',[0 0 4 3])
% print -djpeg Beamprofile.jpg -r300
%
% fluence=transpose(fluence(1)*Interpol_intensitys);
```

```
%% Lambert Beer Atenuation
%E_1=E_0*10^(-Exconst*conc*distance)
%
%extconst=1/(100 mg/L * 10 mm)=0.001 mg*mm/L
%
%dz=1 µm
extconst=0.001;
conc=2000; %mg/L

[LBatenuation]=LambertBeer(number_rays,dz,raymatrix,fluencematrix,l_ray_f_air
,l_ray_lens,extconst,conc);

fluencematrix=fluencematrix.*LBatenuation;
%% Varianze Calculation
%Only takes data inside the liquid/lense
[A,B]=size(fluencematrix);
lens_fluencematrix=zeros(A,B);
for i = 1:number_rays
for j = l_ray_f_air(i):l_ray_f_air(i)+l_ray_lens(i)
lens_fluencematrix(i,j)=lens_fluencematrix(i,j)+fluencematrix(i,j);
end
end

lens_fluencematrix(lens_fluencematrix == 0) = NaN;
col = sum(sum(~isnan(lens_fluencematrix)));
fluencematrix_nonnan =
lens_fluencematrix/(nansum(lens_fluencematrix(:)/(col)));
total_mean = nansum(fluencematrix_nonnan(:))/(col);
total_diff = (fluencematrix_nonnan-total_mean).^2;
total_var = nansum(total_diff(:))/((col)-1);
norm_total_var=sqrt(total_var)/(total_mean);




%% Plotting

skip = 1;  % Only to improve display of 3D-Graphics --> skip these many
number rows and columns, change it accordingly if equal to 1 no data gets
skipped
fluencematrix_reduced = fluencematrix(1:skip:end,1:skip:end) ;  % matrix
reduced
fluencematrix_reduced(fluencematrix_reduced < 0.1) = 0.1;

a=[0:0.05:1];
c=zeros(1,length(a));
%a=exp([log(0.0001):0.01:log(1)]);
%a=fliplr(a);
b=[c;a;c];
mymap=transpose(b);
```

```
Color3=[0 0 1;0 1 0;1 0 0];


figure (4)
surf(fluencematrix_reduced,'FaceColor','interp','EdgeColor','none');
set(gca,'XTick',[0 0.2*zmax/dz 0.4*zmax/dz 0.6*zmax/dz 0.8*zmax/dz zmax/dz])
set(gca,'XTickLabel',[0 0.2 0.4 0.6 0.8 1] )
set(gca,'YTick',[0 0.2*number_rays 0.4*number_rays 0.6*number_rays
0.8*number_rays number_rays])
set(gca,'YTickLabel',[0 0.2 0.4 0.6 0.8 1] )
set(gca,'ZScale','log');
set(gca,'ColorScale','log')
caxis([1 1000])

view(0,90)
colormap(mymap);
xlabel('x [mm]')
ylabel('y [mm]')
h=colorbar
set(get(h,'label'),'string','laser intensity [%]')
set(gcf,'PaperUnits','inches','PaperPosition',[0 0 4 3])
print -djpeg bw.jpg -r300
```

Function: sphere

```
function [raymatrix,z_optaxis,zmax,l_ray_f_air,l_ray_lens] = sphere(n,dz,y)

 radius = 500; %Radius of spherical surface
 zmax = 2*radius;
 z_front = 0:dz:2*radius; %x-axis back of plane surface
 z_optaxis = [z_front]; %total optical axis
 y(y==0)=10^-10;
 raymatrix = zeros(length(y),length(z_optaxis));

 l_ray_lens=[];
 l_ray_f_air=[];

 %Ray tracing
 for i = 1:length(y)

 %Refraction at the full sphere
 [ray_lens, ray_air, x_lens] = sphere_refract_full(y(i),radius,n,dz,zmax);
 l_ray_lens = [l_ray_lens; length(ray_lens)];
 %Ray comming in
 x_front_air = 0:dz:x_lens(1);
 ray_front_air = y(i)*ones(1,length(x_front_air));
 l_ray_f_air = [l_ray_f_air; length(ray_front_air)];
 %Create matrix of rays (adjust length if necessary)

 if length(ray_lens)+length(ray_air)+length(x_front_air) <= length(z_optaxis)
 raymatrix(i,:) = [ray_front_air, ray_lens, ray_air];
 else
 raymatrix(i,:) = [ray_front_air, ray_lens(1:length(ray_lens)-1), ray_air];
 end
 %raymatrix_lens(i,:)=[setdiff(raymatrix(i,:),ray_front_air), ray_lens]

 end
```

Function: flatjet

```
function [raymatrix,z_optaxis,zmax,l_ray_f_air,l_ray_lens] = flatjet(n,dz,y)
 zmax=1000;
 thickness=130;
 z_optaxis = 0:dz:zmax-dz;
 y(y==0)=10^-10;
 raymatrix = zeros(length(y),length(z_optaxis));

 l_ray_lens=[];
 l_ray_f_air=[];

 %Ray tracing
 for i = 1:length(y)


 %Ray comming in
 x_front_air = 0:dz:0.5*(zmax-thickness-dz);
 ray= y(i)*ones(1,length(zmax));
```

```
 %Create matrix of rays (adjust length if necessary)
 l_ray_f_air = [l_ray_f_air; length(x_front_air)];
 l_ray_lens = [l_ray_lens; thickness];
 raymatrix(i,:) = [ray];


 end
```

Function: energydensity

```
function [fluencematrix] = energydensity(raymatrix,fluence)

Nall = [];
binall = [];
[n,m] = size(raymatrix);
[N,edges,bin]= histcounts(raymatrix(:,1),n);

for i = 2:m

    Nall = [Nall;N];
    binall = [binall bin];
    [N,trash,bin] = histcounts(raymatrix(:,i),edges);

end



fluencematrix=zeros(n,m);

for i = 1:n
for j = 1:m-1

fluencematrix(binall(i,j),j) = fluencematrix(binall(i,j),j) + fluence(i);

end
end

end
```

Function: LambertBeer

```
function[LBatenuation]=LambertBeer(number_rays,dz,raymatrix,fluencematrix,l_r
ay_f_air,l_ray_lens,extconst,conc)

atenuationconst=-extconst*(conc/1000);

[A,B]=size(fluencematrix);
LBatenuation=ones(A,B);
```

```
h_ray_lens=raymatrix(1:number_rays,l_ray_f_air+l_ray_lens)-
raymatrix(1:number_rays,l_ray_f_air); %fix -1
h_ray_lens=h_ray_lens(:,1);
d_ray_lens=sqrt(l_ray_lens.^2 - h_ray_lens.^2);

for i = 1:number_rays
for j = l_ray_f_air(i):l_ray_f_air(i)+l_ray_lens(i)

 LBatenuation(i,j)=LBatenuation(i,j)*10^(atenuationconst*((j-
l_ray_f_air(i))/(l_ray_lens(i)))*d_ray_lens(i)); %fix -1

end
end
for i = 1:number_rays
for j = l_ray_f_air(i)+l_ray_lens(i)+1:B

 LBatenuation(i,j)=LBatenuation(i,j)*10^(atenuationconst*d_ray_lens(i));

end
end
```