Electronic Supplementary Material (ESI) for Nanoscale.
This journal is © The Royal Society of Chemistry 2019

Supplementary Information

Brilliant Blue, Green, Yellow, and Red Fluorescent Diamond Particles: Synthesis, Characterization, and Multiplex Imaging Demonstrations

Nicholas Nunn[1*], Neeraj Prabhakar[2], Phillip Reineck[3], Valentin Magidson[4], Erina Kamiya[5], William F. Heinz[4], Marco D. Torelli[1] Jessica Rosenholm[2], Alexander Zaitsev[6], Olga Shenderova[1*]

[1] Adámas Nanotechnologies, Inc. 8100 Brownleigh Drive, Suite 120 Raleigh, NC 27617, USA

[2]Pharmaceutical Sciences Laboratory, Faculty of Science and Engineering, Åbo Akademi University, Tykistokatu 6A, Turku 20520, Finland

[3]ARC Centre of Excellence for Nanoscale BioPhotonics & School of Science, RMIT University, Melbourne, VIC 3001, Australia

[4]Optical Microscopy and Analysis Laboratory, Cancer Research Technology Program, Frederick National Laboratory for Cancer Research, Frederick, MD 21702, USA

[5]Optical Microscopy and Analysis Laboratory, Office of Science and Technology Resources, Center for Cancer Research, National Cancer Institute, National Institutes of Health, Frederick, MD 21702, USA

[6]College of Staten Island (CUNY), 2800 Victory Blvd., Staten Island, NY 10312, USA

**Corresponding Authors:*

N. Nunn, nnunn@adamasnano.com;

O. Shenderova, oshenderova@adamasnano.com

Table S1. Parameters used for brightness comparison shown in Figure 1 of main text.									
Name	**concentration [mg/mL]**	**concentration [µM]**	**Assumed particle diameter [nm]**	**MW [g/mol]**	**bandpass**	**longpass**	**excitation [nm]**	**ex int [mW]**	**integration time**
green FND	0.1	5.40E-05	120	n.a.	450	500	450	44.7	5s
FITC	0.00033	1	n.a.	332.31	450	-	450	44.7	5s
140 nm NV	0.1	3.40E-05	140	n.a.	550	600	550	59.6	5s
AF647	0.001	0.769	n.a.	1300	600	-	600	29.1	5s

Table S2. Optical filters and spectrometers used for imaging demonstrations.				
Part No.	**Manufacturer**	**Type**	**Window (FWHM width)**	**Use (Methods Section 2)**
FF01-470/28	Semrock	Band Pass	**456-484**	Blue Exc. Filter for (wide field)
FF01-517/20	Semrock	Band Pass	**507-527**	Green Exc. Filter for (wide field)
BLP01-488R	Semrock	Long Pass	**488-**	Long Pass Emission filter for blue excitation (wide field)
BLP02-561R	Semrock	Long Pass	**561-**	Long Pass Emission filter for green excitation (wide field)
HQ535/50	Chroma	Band Pass	**510-560**	Green Band Pass Emission for blue excitation (wide field)
HQ610/75	Chroma	Band Pass	**572-648**	Red Band Pass Emission for Green Excitation (wide field)

Table S3. Excitation filters and spectroscopy setups for emission spectra in **Figure 2.**			
Microscope	**Filter**	**Spectrometer**	**Use**
Nikon Eclipse Ti-S Microscope	UV-2A Filter	Princeton Instruments Acton SP2150 Spectrometer	UV excitation Figure 2a
IX-71 Inverted Epifluorescent Microscope (Olympus)	470/28 Excitation, with 488 LP Filter (emission)	Ocean Optics HR2000 Spectrometer	Blue excitation Figure 2b

Table S4. Microscope objectives used for wide field fluorescence imaging of labeled polymer microbeads (Olympus IX-71 Microscope)			
Objective	**Magnification**	**Numerical Aperture**	**Type**
UPlan SApo	100x (160x with 1.6x multiplier lens)	1.4	Oil
LUC PlanFL N	40x (64x with 1.6x multiplier lens)	0.6	Air
PlanC N	10x (16x with 1.6x multiplier lens)	0.25	Air

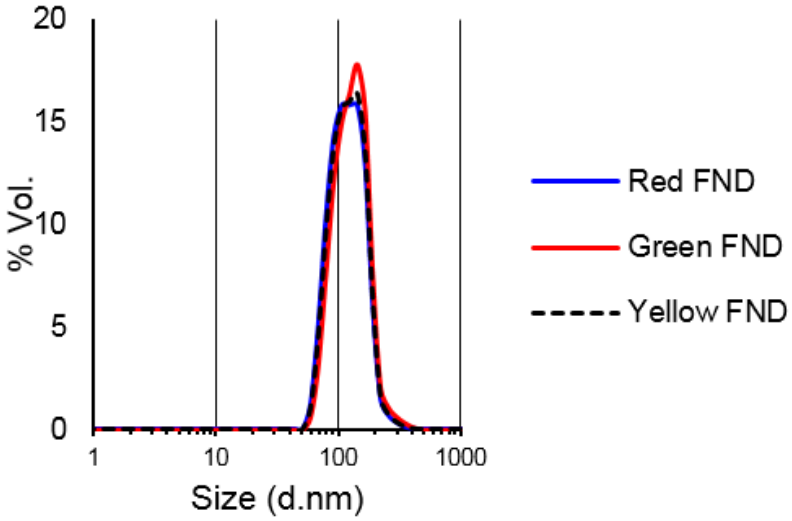


Figure S1. Volumetric particle size distributions measured in deionized water using dynamic light scattering (Malvern Zetasizer Nano ZS).

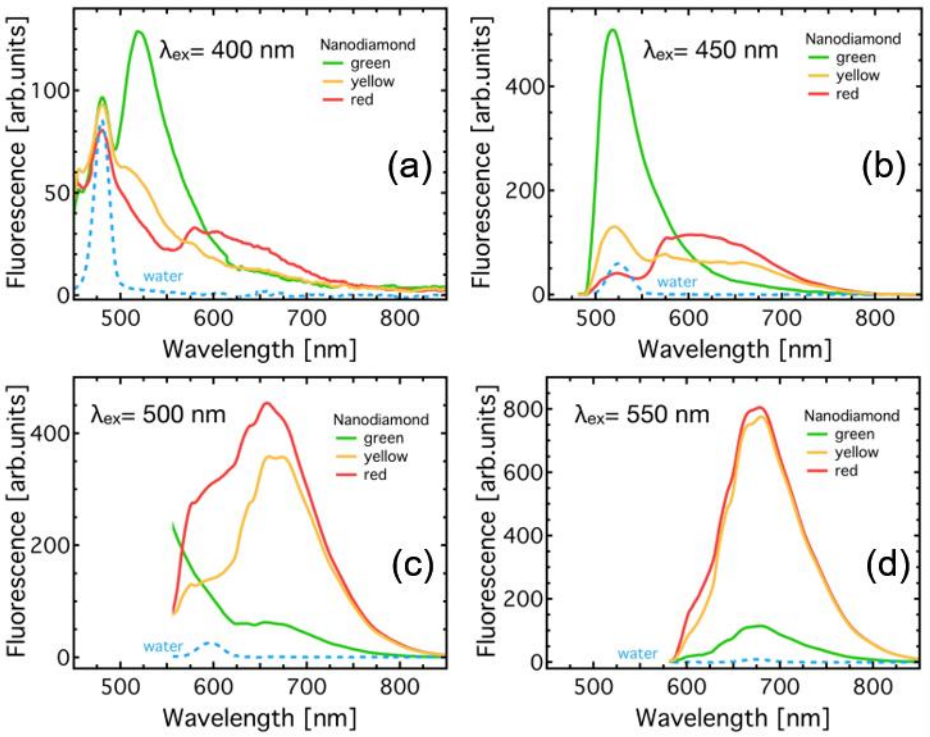


Figure S2. Laser excitation emission spectra for green, yellow, and red fluorescent diamonds in deionized water suspension at 400 nm (a), 450 nm (b), 500 nm (c), and 550 nm (d). Significant water Raman scattering intensity leads to artificial skewing of the emission spectra.

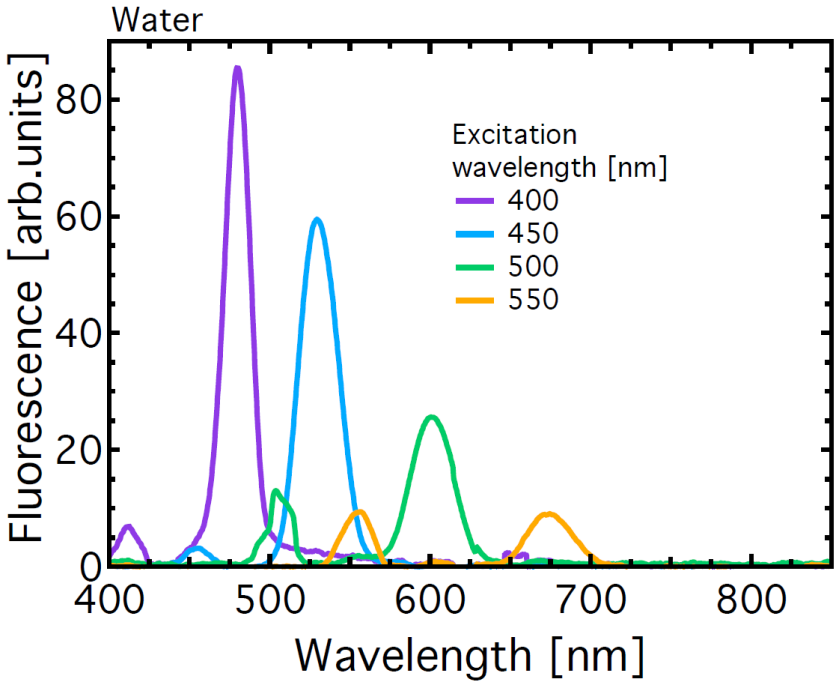


Figure S3. Laser excitation wavelength dependent water Raman scattering intensity.

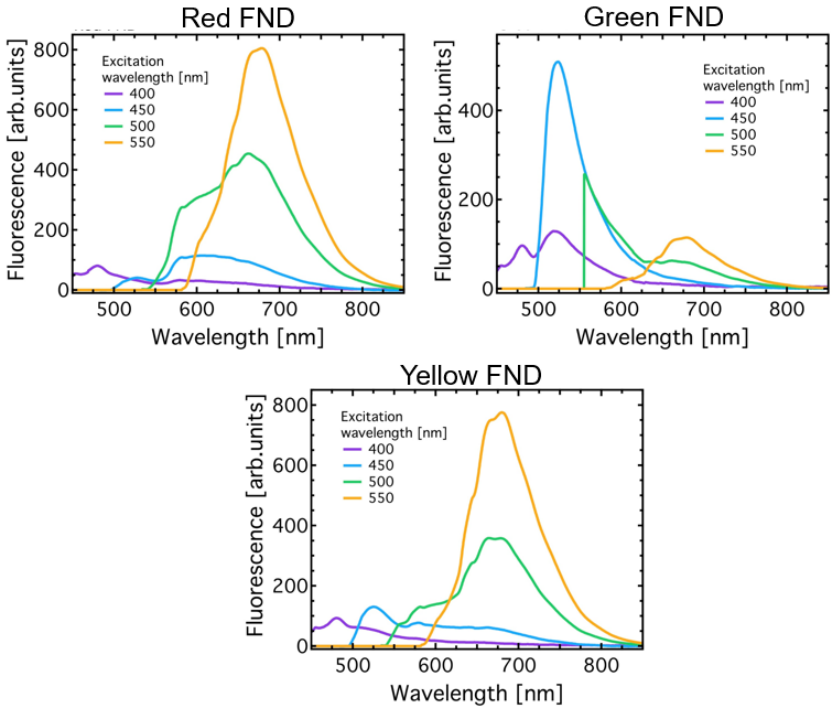
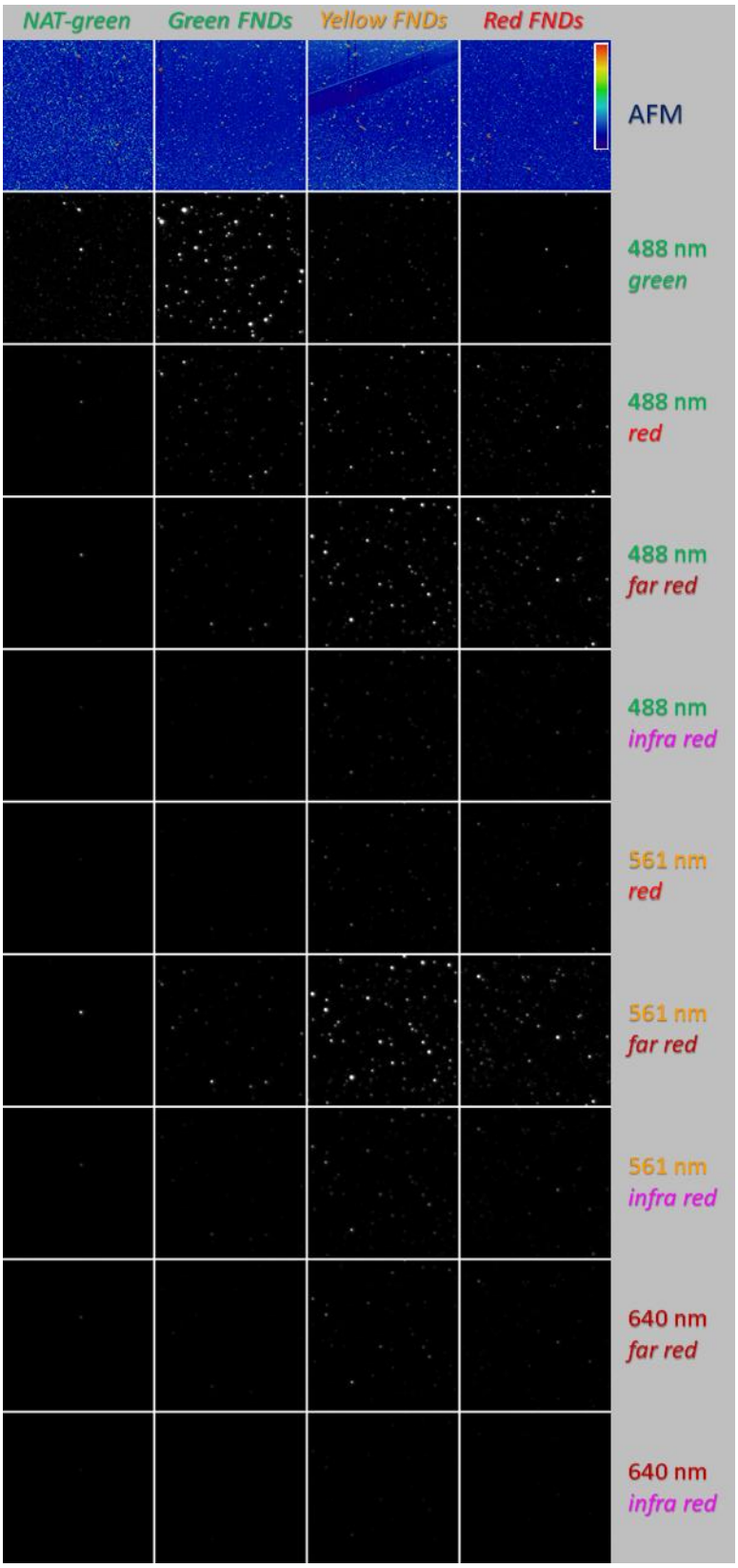
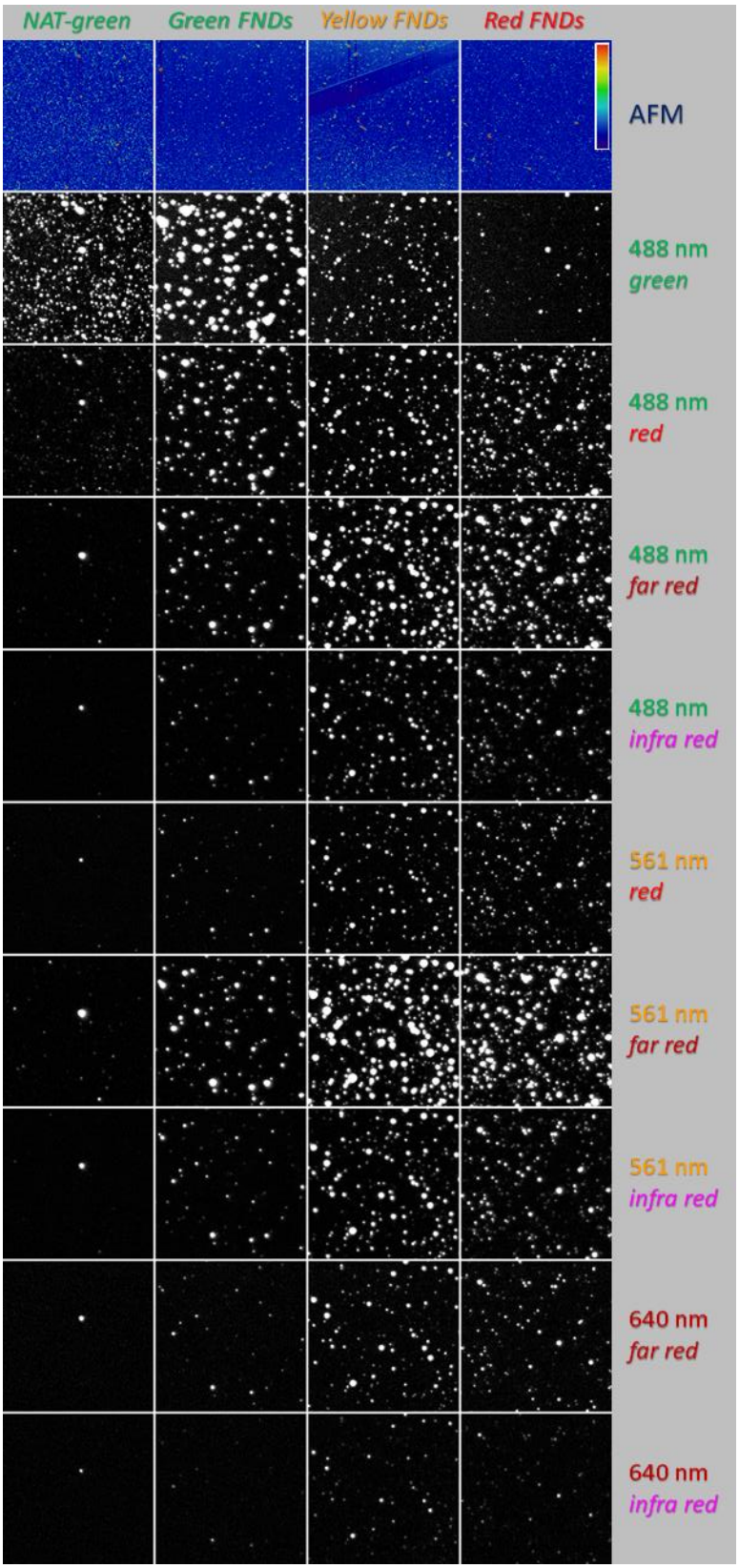


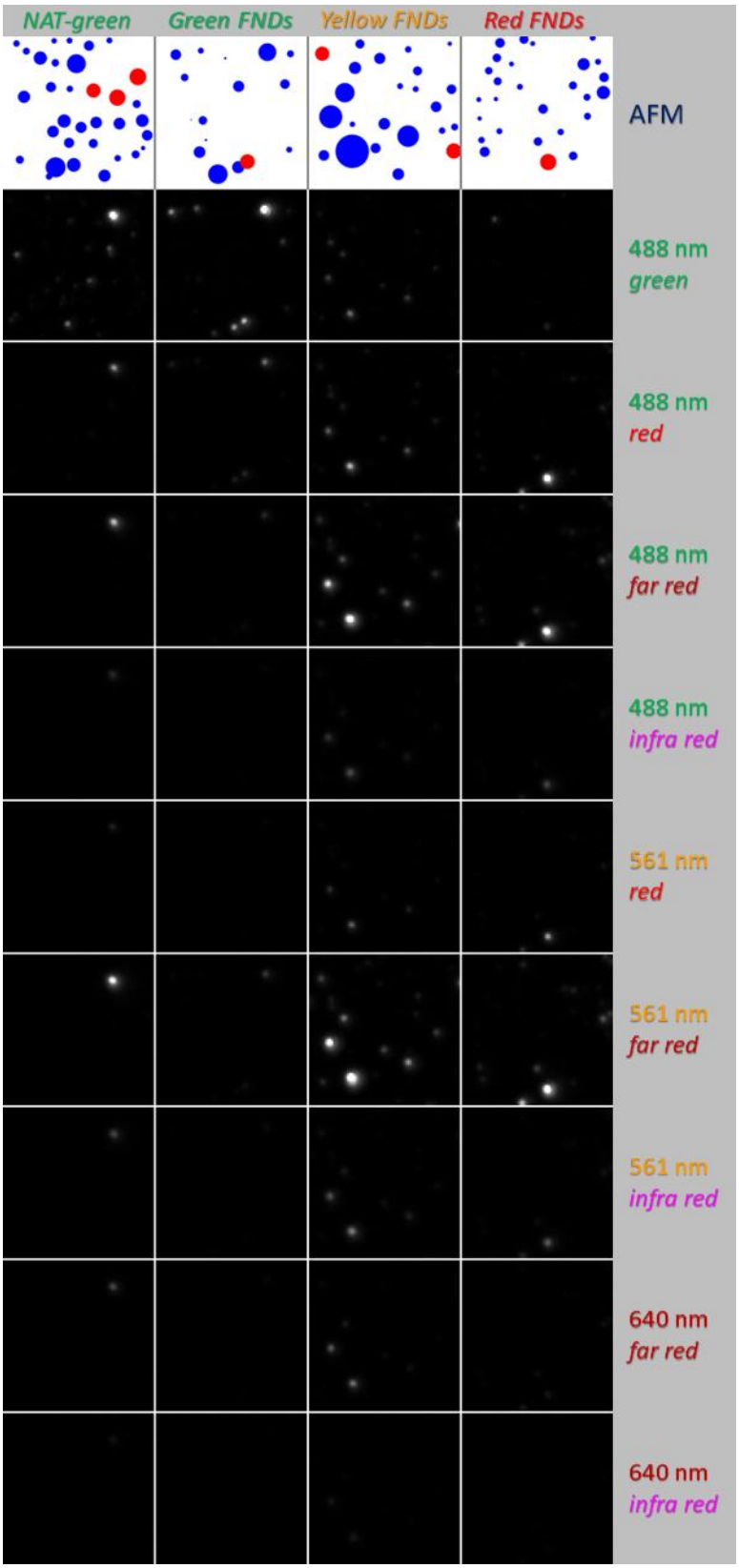
Figure S4. Excitation wavelength dependent emission spectra for red, green and yellow FNDs.


NAT-green
Green FNDs
Yellow FNDs
Red FNDs
AFM
488 nm
green
488 nm
red
488 nm
far red
488 nm
infra red
561 nm
red
561 nm
far red
561 nm
infra red
640 nm
far red
640 nm
infra red

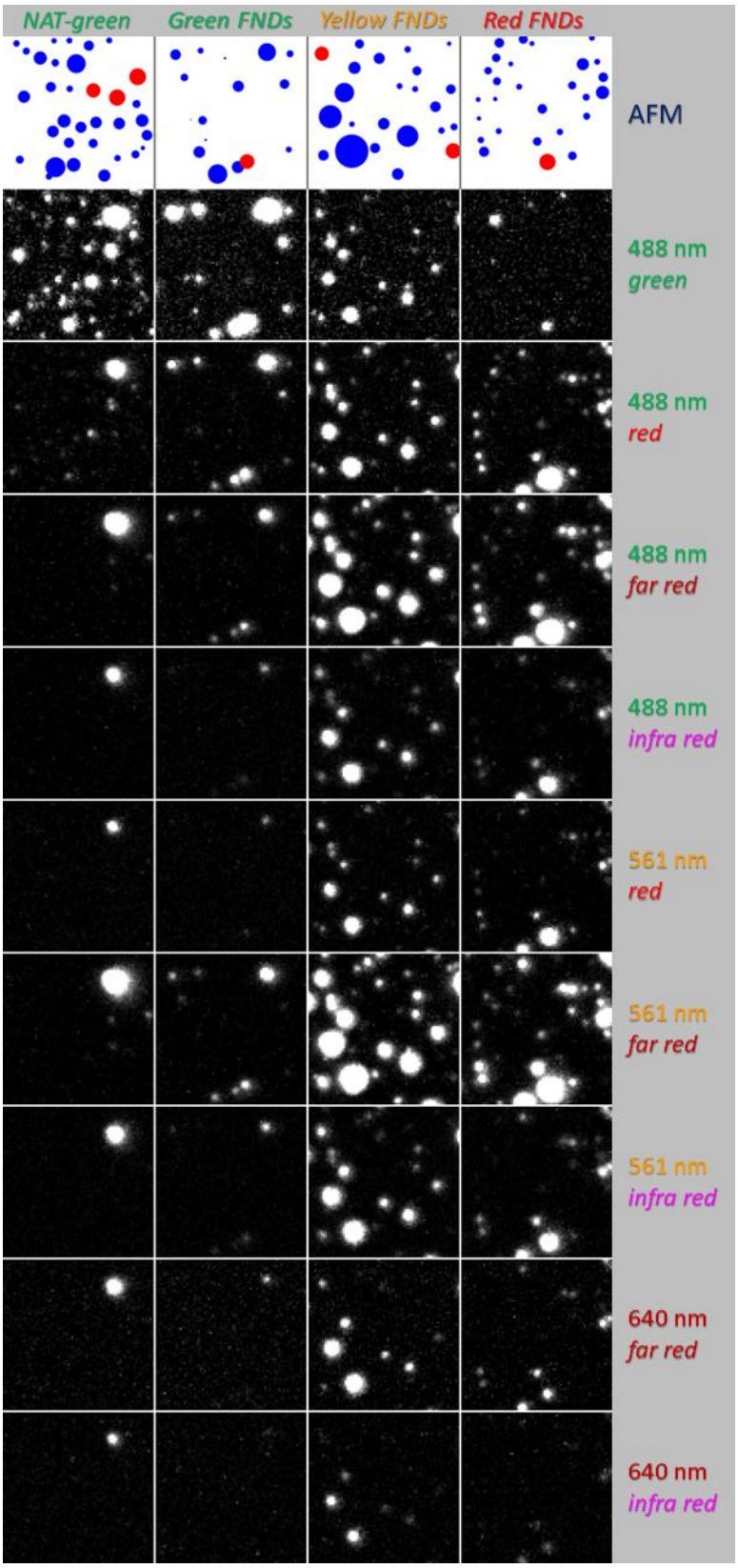
(a)


NAT-green
Green FNDs
Yellow FNDs
Red FNDs
AFM
488 nm
green
488 nm
red
488 nm
far red
488 nm
infra red
561 nm
red
561 nm
far red
561 nm
infra red
640 nm
far red
640 nm
infra red

(b)


NAT-green
Green FNDs
Yellow FNDs
Red FNDs
AFM
488 nm
green
488 nm
red
488 nm
far red
488 nm
infra red
561 nm
red
561 nm
far red
561 nm
infra red
640 nm
far red
640 nm
infra red

(c)


NAT-green
Green FNDs
Yellow FNDs
Red FNDs
AFM
488 nm
green
488 nm
red
488 nm
far red
488 nm
infra red
561 nm
red
561 nm
far red
561 nm
infra red
640 nm
far red
640 nm
infra red

(d)

***Figure S5**: Correlative fluorescence microscopy and AFM of individual FNDs (red, yellow, green, NAT-green). Each field of FNDs was imaged with a spinning disk confocal microscope at different combinations of excitation laser and emission collection channels. For each excitation/emission combination, gray-levels are scaled up to present estimates of relative brightness per unit laser power per unit of emission bandwidth. Emission bands are typical spectral ranges used on commercial microscopes to collect "green" (e.g., GFP), "red" (e.g., RFP), "far red" (e.g., AF647) and "infra-red" (e.g., AF750) fluorescence signals (see MATLAB source code in Supplementary Information for details of grayscale adjustment).*

(a) Full field of correlative AFM and fluorescence microscopy. Each square is 30 x 30 µm. Color range (violet to red) in AFM field corresponds to height (0–100 nm). Light microscopy contrast is adjusted to present brightest particles.

(b) Same as (a), but contrast of light microscopy image is increased 25-fold, to present low intensity particles.

(c) A crop within the fields presented in (a) and (b). Each square is 8.2 x 8.2 µm. On the left, circles represent particles detected on AFM and qualified in MATLAB analysis, with diameter proportional to height. For reference, all particles between 80 nm and 100 nm are presented in red; particles outside of this diameter range are presented in blue. Light microscopy is presented with the same normalization as in (a).

(d) Same as (c), but with the same normalization as in (b).

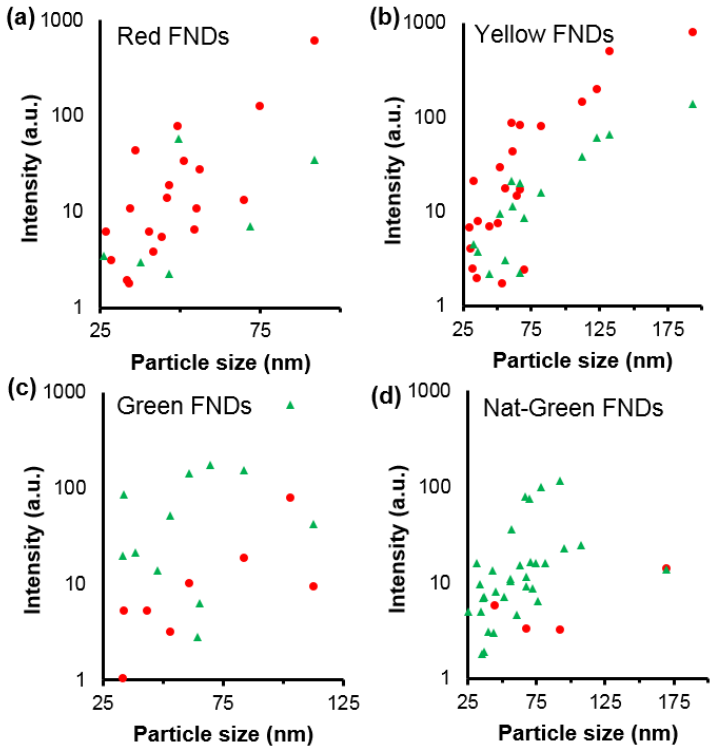


Figure S6*: Scatter plots of particle intensity versus particle size for only fluorescent particles of each diamond type: (a) red FNDs, (b) yellow FNDs, (c) green FNDs, and (d) natural green FNDs. In each plot, the red circles represent fluorescent particles under 561 nm excitation with emission in the far-red channel, and the green triangles represent fluorescent particles under 488 nm excitation with emission in the green channel. For these plots, the green emission window under 488 nm excitation and the far-red emission window under 561 nm excitation were used. Only particles assigned a non-zero level of fluorescence by MATLAB were evaluated (see MATLAB source code in Supplementary Information). The fluorescence intensity generally increases with particle size, though there are some outliers. Analysis of many more particles correlated by size would improve the data quality; however, the general trend can still be observed in most cases. Red FNDs exhibit brighter fluorescence in the far-red channel as opposed to the green channel (a). Yellow FNDs exhibit a fluorescence brightness in the far-red channel that is similar to the red FNDs, but also exhibit higher brightness in the green channel (b). The fluorescence intensity of the RTA synthesized green FNDs is generally higher than that of the natural green FNDs in both the green and far-red channels (c-d). It should be noted that the particle sizes determined here as the AFM z-height of particles are generally much lower than sizes measured by DLS. This may be caused by a 'plate'-like shape of particles of, which often exhibit wider lateral dimension (in the x-y plane of the solid substrate) and significantly smaller z-heights as determined by AFM.*[19, 32]

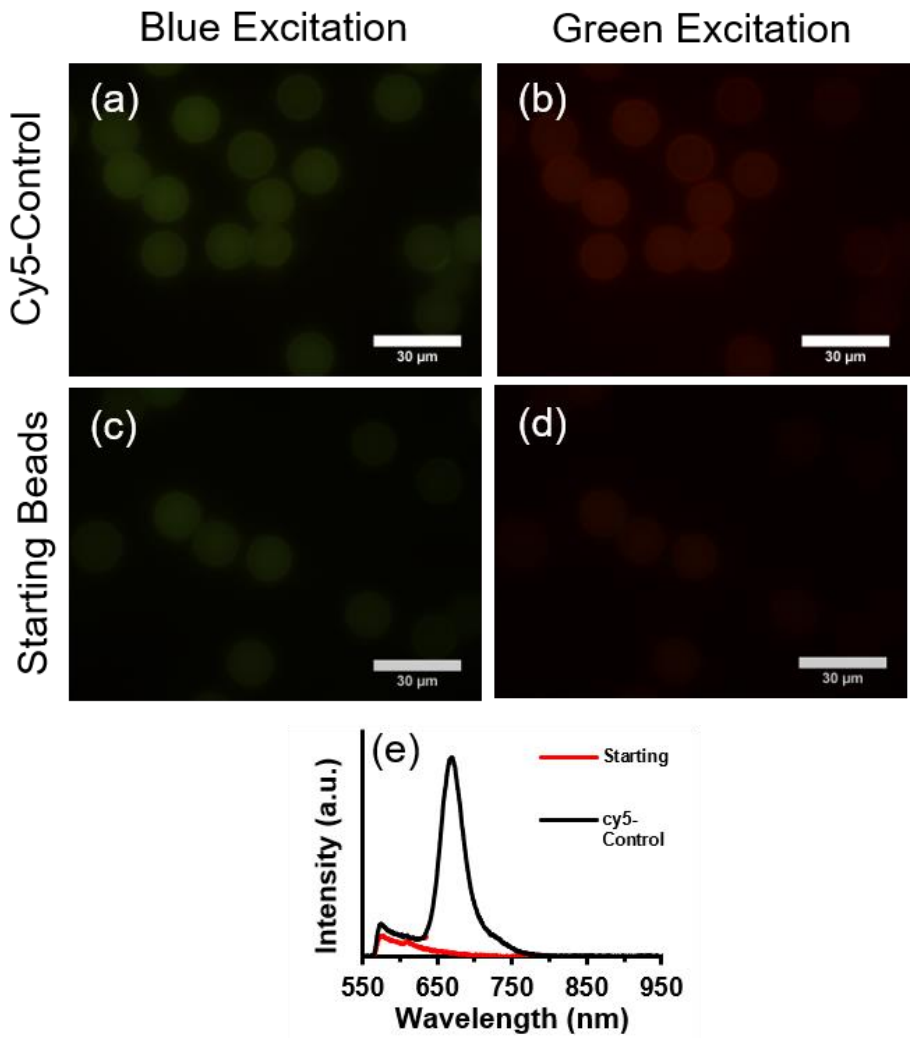


Figure S7. Validation of SA-Bead activity and uniformity using cy5-biotin control.

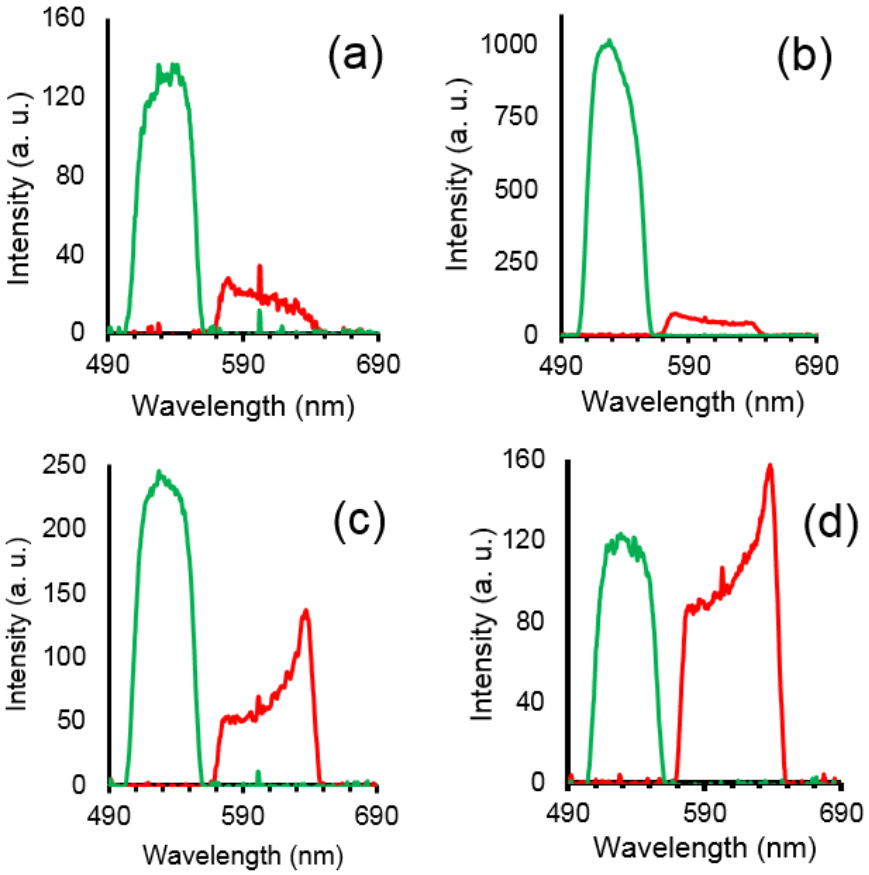


Figure S8. Emission spectra of (a) starting beads, (b) green labeled beads, (c) yellow labeled beads, and (d) red labeled beads under blue and green excitation with band pass emission filters specified in table S2.

MATLAB Source Code for Image Processing

```
% This script is finding "particles" from a table exported from ThunderSTORM (last 2016 "dev" version)
% The algorithm excludes from consideration any spots which have more than one spot in a vicinity (defined by variable "Radius") in any spectral channel.
% Therefore, only particles which happened to be well separated are analyzed.
% Previous steps:
% a. Z stack (250nm Z step, 4um Z range) was acquired for each of 10 spectral channels, listed below.
% b. Maximum intensity projection for each stack was created in ImageJ.
% c. All 10 projections were assembled into a stack in ImageJ, to be analyzed in ThunderSTORM.
% d. Thunderstorm was configured with "Camera Settings" according to manufacture supplied datasheet, for translation of pixel intensities to photon counts.
% e. Analysis parameters in ThunderSTORM were set to defaults (2D, MLH fitting), except for threshold, which was manually set to 5.
% f. Following filter was applied, to remove high noise peaks and peaks
%     with width far from expected: (sigma > 50 & sigma < 200) & (intensity  > 80) . Here sigma is in nanometers, and intensity - in photons.
% g. The table was exported from ThunderSTORM as .csv file, with all fields checked in the saving dialogue.
%

% Steps after this script
% a. Results files (.mat and .txt) can be integrated into data structures for further analysis
% b. .csv "for ThunderSTORM" files can be "Imported" into ThunderSTORM,
% when original data Z stack is opened. This way "qualified" localizations
% from this script are overlaid  as red crosses on the original data, allowing visual inspection.

% Order of channels in this experiment:

% 1: 640-730
% 2: 640-640

% 3: 561-730
% 4: 561-640
% 5: 561-561

% 6: 488-730
% 7: 488-640
% 8: 488-561
% 9: 488-488
```

```
%10: 405-561

clc; clear all; close all

Path_for_reading_and_writing_files =
'C:\Data\_2019_02_19_Diamonds\LM_with_AFM_aligned_by_Erina\Same_stacks_with
out_AFM';

Name_of_CVS_table_saved_from_Thunderstom_without_extension =
'ND_1_red_glue_AFM';

filename = [ Path_for_reading_and_writing_files filesep
Name_of_CVS_table_saved_from_Thunderstom_without_extension '.csv' ];

Save_Path_And_Name_Without_Extension = ...
    [ Path_for_reading_and_writing_files filesep
Name_of_CVS_table_saved_from_Thunderstom_without_extension '_spectra' ];

Show_only_NDs_with_signal_in_all_seven_488_561_excitations = 0;

Skip_every_N_particle_in_plotting = false;
N_to_skip = 20;
percent_saturation = 90; % This percentage of particles will be presented with total
    %intensity linearly incoded by color in the color bar. The rest, top intensity will be
shown with max color

Radius = 500; %nm, max displacement from the reference particle

tic
% Auto-generated by MATLAB on 2019/02/21 18:13:53
% Import file from ThunderSTORM
delimiter = ','; startRow = 2; formatSpec = '%f%f%f%f%f%f%f%f%f%[^\n\r]';
fileID = fopen(filename,'r');
dataArray = textscan(fileID, formatSpec, 'Delimiter', delimiter, 'TextType', 'string',
'EmptyValue', NaN, 'HeaderLines' ,startRow-1, 'ReturnOnError', false, 'EndOfLine',
'\r\n');
fclose(fileID);
Table = table(dataArray{1:end-1}, 'VariableNames',
{'id','frame','xnm','ynm','sigmanm','intensityphoton','offsetphoton','bkgstdphoton','uncertai
nty_xynm'});
clearvars delimiter startRow formatSpec fileID dataArray ans;

% Our code

ID_Frame_X_Y_Intensity = [ Table{:,1:4} , Table{:,6}]; % First column is "id", then frame,
x, y and in #6 in intensity
```

```
% Create holders for results
Results                = []; % May be changed for pre-allocation for speed if needed
Accepted_localizations  = [];

ID_X_Y_Int = cell(10,1); % to hold separate table for each spectral option

Reference_channels_order = [7 9 1:6 8 10]; % This order should not matter,
% but putting hight polulation channel first should speed up the process.

for Reference_slice_number = Reference_channels_order
    for i=1:10
        ID_X_Y_Int{i} = ID_Frame_X_Y_Intensity( ID_Frame_X_Y_Intensity(:,2)== i, [1 3 4
5] );
    end

    Reference = ID_X_Y_Int{Reference_slice_number};

    % Create holders for results
    Results_per_ref               = []; % May be changed for pre-allocation for speed if
needed
    Accepted_localization_IDs      = [];
    Rejected_localization_IDs      = [];
    List_of_rejected_multiples_IDs = [];

    % Loop via all particles in reference color image
    for i = 1:size(Reference,1)

        Xref = Reference(i,2);
        Yref = Reference(i,3);

        Intensity_vector = zeros(1,10);
        IDs_vector       = zeros(1,10);

        More_then_one_spot = 0;

        % Loop via rest of colors (spectral combinations) for one point
        for j = [1:(Reference_slice_number-1) (Reference_slice_number+1):10]

            L = ID_X_Y_Int{j}; % Current spectral combination data
            Distances = ( (L(:,2)-Xref).^2 + (L(:,3)-Yref).^2 ).^0.5; % distances to all
localizations
            ind = find(Distances < Radius);
            N_hits = length(ind);

            if N_hits >= 2
```

```
            % Two or more localization within radius that only supposed to
            % have one particle. Data for this particle will be discarded
            More_then_one_spot = 1;
            List_of_rejected_multiples_IDs = [List_of_rejected_multiples_IDs ; L(ind,1)];
%#ok<AGROW>
         else
            if N_hits == 0
               Intensity_vector(j) = 0; % intensity in this channel is below detection limit
               IDs_vector(j)      = -1; % Non-existing ID number for situation where no
peak was found
            else % meaning N_hits == 1
               Intensity_vector(j) = L(ind,4);
               IDs_vector(j)       = L(ind,1);
            end
         end
      end
      % Filling in seed localization ID from the reference layers, which need to be
excluded in runs with other reference layers
      IDs_vector(Reference_slice_number) = Reference(i,1);

      % Adding "valid" sequence to the list, which later will be checked again to remove
sequences which share a localization
      if ~More_then_one_spot
         Intensity_vector(Reference_slice_number) = Reference(i,4);
         Results_per_ref                 = [Results_per_ref                  ; [ Intensity_vector
Xref Yref ] ]; %#ok<AGROW>
         Accepted_localization_IDs = [Accepted_localization_IDs ;   IDs_vector
]; %#ok<AGROW>
      else
         % our sequence of localization will not be added to results, but we have to
exclude considered localization from further analysis, at different refernce channels
         Rejected_localization_IDs = [Rejected_localization_IDs ;   IDs_vector
]; %#ok<AGROW>
         % This one will need to be later combined with regected multiples
      end
   end

   Accepted_localization_vector_with_negative_ones = Accepted_localization_IDs(:); %
Make a linear array with all elements of the matrix included
   % Remove all "-1" values
   Accepted_localization_vector = Accepted_localization_vector_with_negative_ones(
Accepted_localization_vector_with_negative_ones ~= -1 );
   sorted_Accepted_localizations_vector = sort(Accepted_localization_vector);  %
Softing all frames from smaller to larger.
   check_repetitions = diff(sorted_Accepted_localizations_vector); % It will produce
array of length N-1, with 0 in case same number is used twice.
```

```
    % It means that same point is used more then ones, and all "particles" involved in this
should be removed from results
    List_of_Duplicating_IDs_with_repetitions     =
sorted_Accepted_localizations_vector(~check_repetitions);
    if ~isempty(List_of_Duplicating_IDs_with_repetitions)
        List_of_Duplicating_IDs_without_repetitions  =
List_of_Duplicating_IDs_with_repetitions([true;
logical(diff(List_of_Duplicating_IDs_with_repetitions))]);

        % Now we have a list of all problematic localization, and need to remove
corresponding "spectral particles" from Result table.
        % Accepted_localization_IDs matrix was built at the same pattern as
        % Results_per_ref, so that we need to look which lines there contains problematic
localizations

        n_of_lines = size(Accepted_localization_IDs, 1);
        Accepted_Particles_Lines = true(n_of_lines , 1);
        Additional_Localization_To_Reject = [];

        for line_n = 1: n_of_lines
            if ~isempty(intersect (Accepted_localization_IDs(line_n,:) ,
List_of_Duplicating_IDs_without_repetitions) )
                Accepted_Particles_Lines(line_n) = false;
                Additional_Localization_To_Reject = [Additional_Localization_To_Reject
Accepted_localization_IDs(line_n,:)]; %#ok<AGROW>
            end
        end

        % Remove lines with duplication from Results_per_ref table
        Results_per_ref = Results_per_ref(Accepted_Particles_Lines,:);

        sorted_Accepted_localizations_vector = setdiff(
sorted_Accepted_localizations_vector, Additional_Localization_To_Reject);

    end

    Accepted_localizations = [Accepted_localizations ;
sorted_Accepted_localizations_vector]; %#ok<AGROW>
    Results = [Results; Results_per_ref]; %#ok<AGROW>

    Considered_IDs_with_negative_ones = [Accepted_localization_IDs(:);
Rejected_localization_IDs(:);  List_of_rejected_multiples_IDs(:)];
    Considered_IDs = Considered_IDs_with_negative_ones(
Considered_IDs_with_negative_ones ~= -1 );
```

```
    [ ~ , index_of_not_considered_localizations] = setdiff( ID_Frame_X_Y_Intensity(:,1) ,
Considered_IDs );

    ID_Frame_X_Y_Intensity = ID_Frame_X_Y_Intensity(
index_of_not_considered_localizations , : );
 disp(Reference_slice_number);
 whos Res*
end

All_IDs_from_original_table = Table{:,1};

[~,index_of_lines_included_in_results, ~] = intersect( All_IDs_from_original_table,
Accepted_localizations );


Full_data = Table{:,:}; % all data loaded from TS
% Only take lines with IDs which were used in the results
M = Full_data( index_of_lines_included_in_results ,:);

% "Order of magnitude" estimated correction for detection optical path
%    channel 488  561  640  732nm regular excitation for this band
BandWidth = [30       50   75   90  ];
Cental_WL = [525  600  700  810 ];
QE        = [0.8  0.81 0.71    0.47];
QE_used_in_TS_settings = 0.8;
QE_relative = QE / QE_used_in_TS_settings;
% Relative laser power estimated from the power measured above objective
% lens of dry objectives
%                    405nm   488nm   561nm   640nm
Relative_laser_powers = [0.37   1.5     1.25    1   ];

Intensity_correction_coefficient = zeros(1,10);
% #1   640-732
Intensity_correction_coefficient(1) =   QE_relative(4) * BandWidth(4)*
Relative_laser_powers(4);
% #2    640-640
Intensity_correction_coefficient(2) =   QE_relative(3) * BandWidth(3)*
Relative_laser_powers(4);
% #3    561-732
Intensity_correction_coefficient(3) =   QE_relative(4) * BandWidth(4)*
Relative_laser_powers(3);
% #4    561-640
Intensity_correction_coefficient(4) =   QE_relative(3) * BandWidth(3)*
Relative_laser_powers(3);
% #5    561-561
```

```
Intensity_correction_coefficient(5) =   QE_relative(2) * BandWidth(2)*
Relative_laser_powers(3);
% #6    488-732
Intensity_correction_coefficient(6) =   QE_relative(4) * BandWidth(4)*
Relative_laser_powers(2);
% #7    488-640
Intensity_correction_coefficient(7) =   QE_relative(3) * BandWidth(3)*
Relative_laser_powers(2);
% #8    488-561
Intensity_correction_coefficient(8) =   QE_relative(2) * BandWidth(2)*
Relative_laser_powers(2);
% #9    488-488
Intensity_correction_coefficient(9) =   QE_relative(1) * BandWidth(1)*
Relative_laser_powers(2);
% #10   405-561
Intensity_correction_coefficient(10)=   QE_relative(2) * BandWidth(2)*
Relative_laser_powers(1);

% Adjust photon counts for collection efficiency
Results_Corr = Results;
Results_Norm = Results;
% Last two columns with X and Y values will not be altered
for i=1:size(Results,1)
   Results_Corr(i,1:10) = Results(i,1:10)     ./ Intensity_correction_coefficient;
   Results_Norm(i,1:10) = Results_Corr(i,1:10) / sum(Results_Corr(i,1:10)) * 100;
end
disp('Time of data processing:');
toc

% Saving result table as both mat file and text file .csv

save( [ Save_Path_And_Name_Without_Extension '.txt' ], 'Results_Corr', '-ascii','-tabs');
save( [ Save_Path_And_Name_Without_Extension '.mat' ], 'Results_Corr',
'Results_Norm', 'filename','Radius','M');

% Export TS table which will only have localizations accepted by MATLAB

header_string = '"id","frame","x [nm]","y [nm]","sigma [nm]","intensity [photon]","offset
[photon]","bkgstd [photon]","uncertainty_xy [nm]"';
formatSpec = '%.1f,%.1f,%.10f,%.10f,%.10f,%.10f,%.10f,%.10f,%.10f,%s%s';

fileID = fopen([Save_Path_And_Name_Without_Extension '_for_TS_import.csv'],'w');
fprintf(fileID, '%s%s%s', header_string, char(13), newline );

n_lines = size(M,1);
```

```
for line_n = 1:n_lines
   fprintf(fileID, formatSpec,...
      M( line_n,  1), M( line_n,  2), M( line_n,  3), M( line_n,  4), M( line_n,  5), ...
      M( line_n,  6), M( line_n,  7), M( line_n,  8), M( line_n,  9), char(13) , newline     );
end
fclose(fileID);

if Show_only_NDs_with_signal_in_all_seven_488_561_excitations
   %remove lines with zeros in columns 3 to 9
   index = prod(Results_Corr(:,3:9), 2) ~=0; % here prod will multiply all numbers in
each line
   Results_Corr   = Results_Corr(   index, :);  % Saved results will not be changed
   Results_Norm = Results_Norm( index, :);
end

% Define colors for each particle plot, based on its total intensity
figure(1);
c_table = colormap('jet');
colorbar;
figure(2);
colormap('jet');
colorbar;
TI = sum(Results_Corr(:,1:10),2); % Total intensity for each channel
sTI = sort(TI);
ind = round(length(TI)*percent_saturation/100);
colormap_index = ceil(TI/sTI(ind)*64); % Round does not work here, because index
cannot be zero in MATLAB.
      % This array has to have numbers from 1 to 64. No elemen of sTI
      % should be zero, so 1 should me min, and max in made 64 below:
colormap_index(colormap_index > 64) = 64;

% Plotting Results structure
% Cental_WL = [525  600  700  810 ];
x_1_2 =     Cental_WL([4 3    ]);    % For 640 excitation
x_3_4_5 =   Cental_WL([4 3 2  ]);    % For 561 excitation
x_6_7_8_9 = Cental_WL([4 3 2 1]);    % For 488 excitation
x_10 =      Cental_WL( 2       );    % For 405 excitation and 561nm band emission

if ~Skip_every_N_particle_in_plotting
   disp('All data will be ploted');
   N_to_skip = 1; % to plot every line
   h1 = zeros(size(Results,1) , 4);
   h2 = zeros(size(Results,1) , 4);
else
   disp('Only subset of the data will be plotted')
   disp(['N_to_skip variable is currently ' num2str(N_to_skip) ]);
```

```
    N_point_to_be_plotted = length(1:N_to_skip:size(Results_Corr,1)); % Array used
below in the FOR loop
    h1 = zeros(N_point_to_be_plotted , 4);
    h2 = zeros(N_point_to_be_plotted , 4);
end

figure(1); hold on
counter = 1;
for i=1:N_to_skip:size(Results_Corr,1)
    RGB = c_table( colormap_index( i ), :);
    % Get single particle data
    y= Results_Corr(i,1:10);
    h1(counter,1) = plot(x_1_2,     y(1:2),'-', 'Color', RGB);
    h1(counter,2) = plot(x_3_4_5,   y(3:5),'-', 'Color', RGB);
    h1(counter,3) = plot(x_6_7_8_9, y(6:9),'-', 'Color', RGB);
    h1(counter,4) = plot(x_10,      y(10) ,'*', 'Color', RGB);
    counter = counter + 1;
end

figure(2); hold on
counter = 1;
for i=1:N_to_skip:size(Results_Norm,1)
    RGB = c_table( colormap_index( i ), :);
    % Get single particle data
    y= Results_Norm(i,1:10);
    h2(counter,1) = plot(x_1_2,     y(1:2),'-', 'Color', RGB);
    h2(counter,2) = plot(x_3_4_5,   y(3:5),'-', 'Color', RGB);
    h2(counter,3) = plot(x_6_7_8_9, y(6:9),'-', 'Color', RGB);
    h2(counter,4) = plot(x_10,      y(10) ,'*', 'Color', RGB);
    counter = counter + 1;
end
```