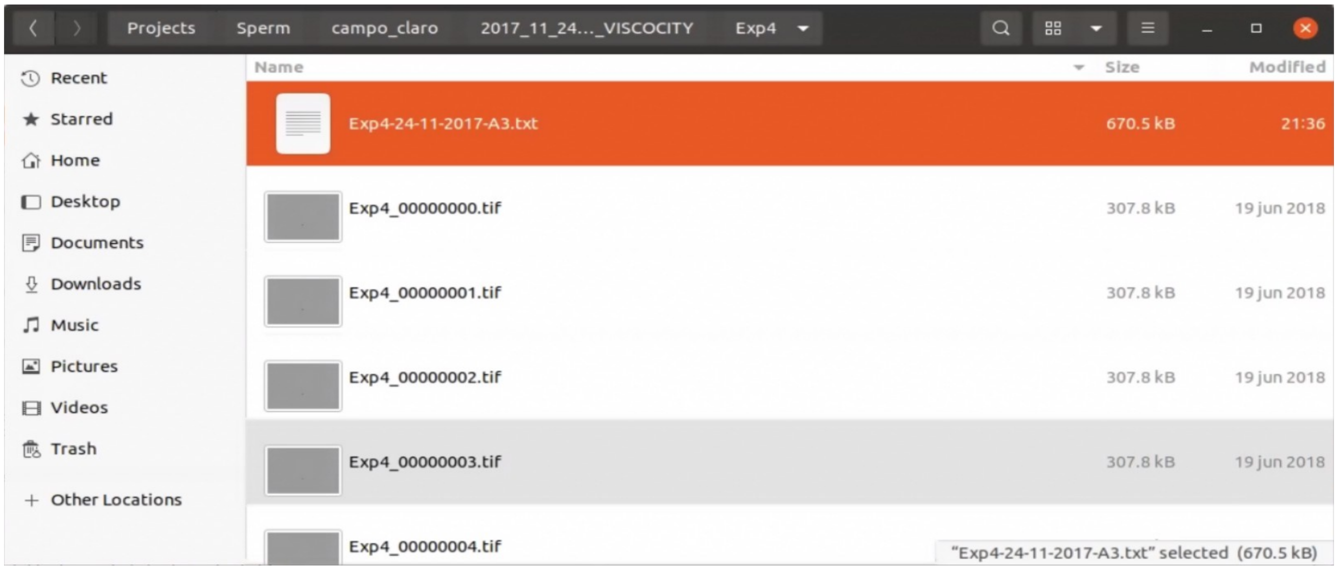
Sperm centerline tracing --- Images from Corkidi microscope

0. Required input images

Corkidi's microscope can acquire thousand of images per experiment using a piezoelectric device which allows to obtain images at different heights. The output of the microscope is a series of 2D images with a single txt with the height at which the images were acquired. Figure 1 display an example of images acquired with the microscope. The folder containing the images must also include the txt file containing the heights (orange rectangle). **Make sure that a single txt file is included in the folder, sometimes an additional file with microscopy information can be included and need to be removed.**



Name	Size	Modified
Exp4-24-11-2017-A3.txt	670.5 kB	21:36
Exp4_00000000.tif	307.8 kB	19 jun 2018
Exp4_00000001.tif	307.8 kB	19 jun 2018
Exp4_00000002.tif	307.8 kB	19 jun 2018
Exp4_00000003.tif	307.8 kB	19 jun 2018
Exp4_00000004.tif		

"Exp4-24-11-2017-A3.txt" selected (670.5 kB)

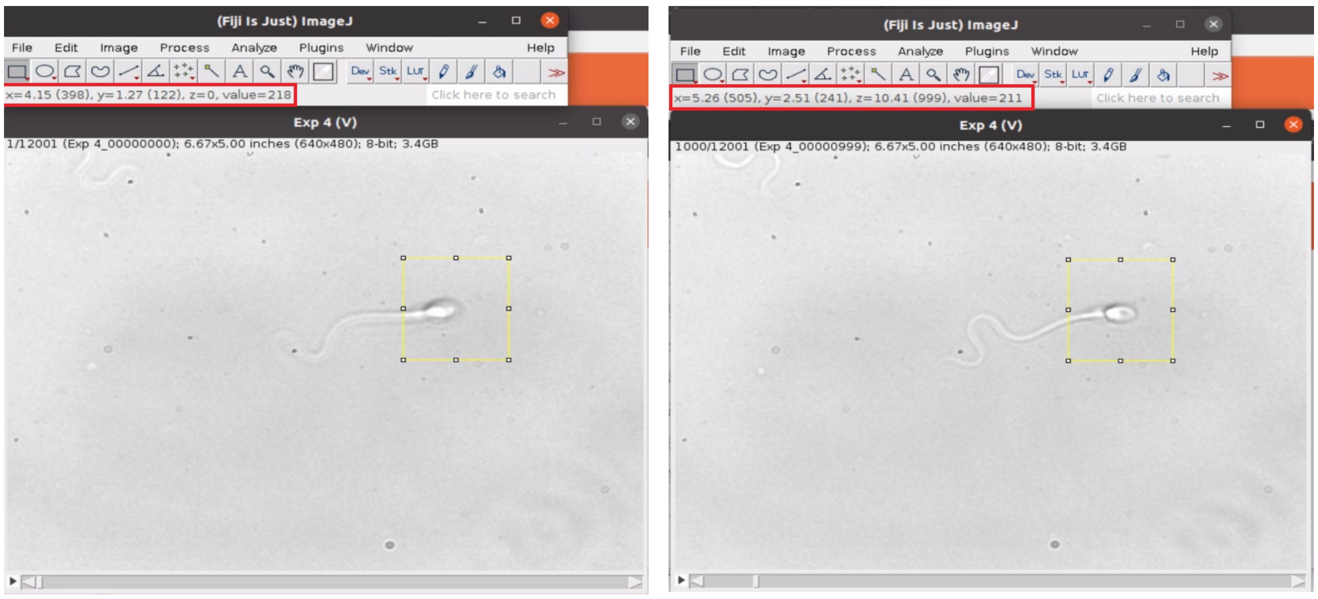
Figure 1. Example of images acquired with the microscope saved in the folder Exp4 and the corresponding txt file with the heights for each image.

1. Correcting shift

We expect that first image acquired would correspond to the height number 1, the second image acquired would correspond to the height number 2, however this is not the case and sometimes there is not synchronization between the images acquired and the height signal. Then, we have a shift in the mapping between images and height (usually a drift of 1 image). This shift needs to be corrected.

The function "**f01_get_shift_values_forCreatingStacks.m**" allows to find the drift value between images and the heights. It requires some inputs:
folder_path → The path to the folder containing the images
image_prefix → prefix used to identify the images. Image file name should be [image_prefix + 8 id numbers + .tif]
crop → crop containing the sperm head in the firsts 1,000 images

Figure 2 depicts an example of computing the crop coordinates where a rectangle is draw in the image (yellow rectangle). Figure 2a depicts that the first image is inside this rectangle and its left-upper coordinate is (398, 122) see red rectangle. Figure 2b depicts that the image 1000 is inside this rectangle and its right-lower coordinate is (505, 241) see red rectangle.

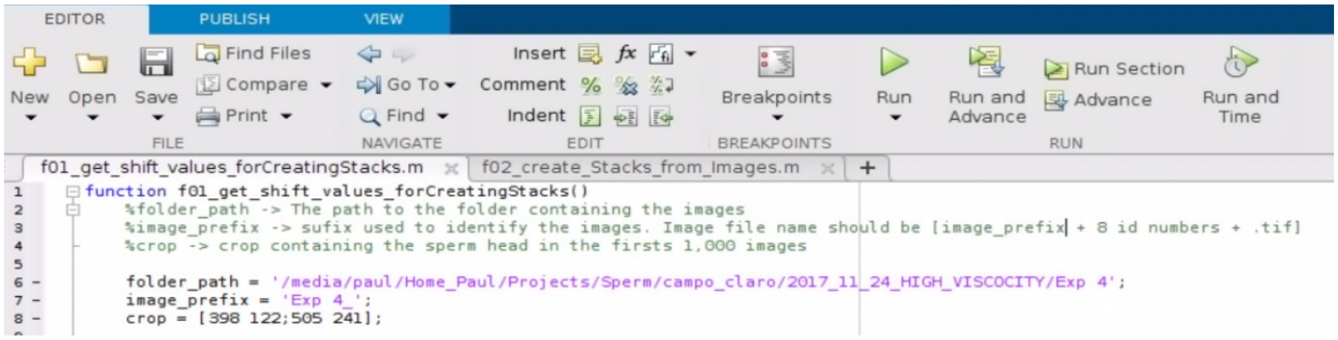


```
x=4.15 (398), y=1.27 (122), z=0, value=218
1/12001 (Exp 4_00000000); 6.67x5.00 inches (640x480); 8-bit; 3.4GB
```

```
x=5.26 (505), y=2.51 (241), z=10.41 (999), value=211
1000/12001 (Exp 4_00000999); 6.67x5.00 inches (640x480); 8-bit; 3.4GB
```

Figure 2. Example of drawing a rectangle containing the sperm head.

Figure 3 shows an example of setting the correct parameters for detecting the shift. After setting the parameters correctly, the code can be run.

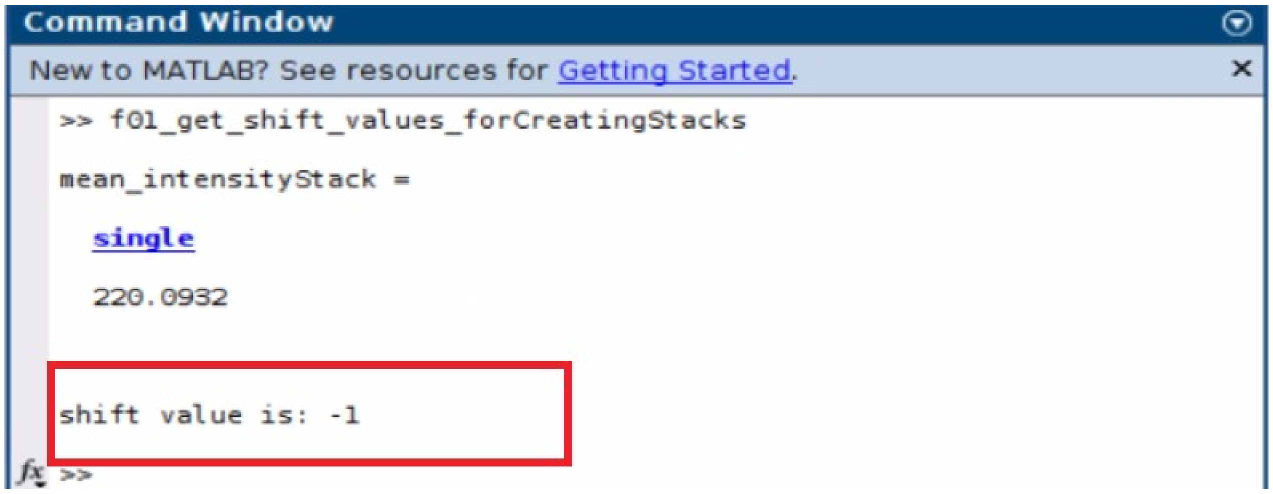


```
function f01_get_shift_values_forCreatingStacks()
    %folder_path -> The path to the folder containing the images
    %image_prefix -> sufix used to identify the images. Image file name should be [image_prefix + 8 id numbers + .tif]
    %crop -> crop containing the sperm head in the firsts 1,000 images

    folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp 4';
    image_prefix = 'Exp 4_';
    crop = [398 122;505 241];
```

Figure 3. Example of setting the correct parameters for the función “**f01_get_shift_values_forCreatingStacks.m**”.

The output is a value “**shift value is:** ” that indicates the shift between images and file txt.

NOTE: if the shift value is large then there is not a good synchronization between the piezoelectric devise and the signal of heights.



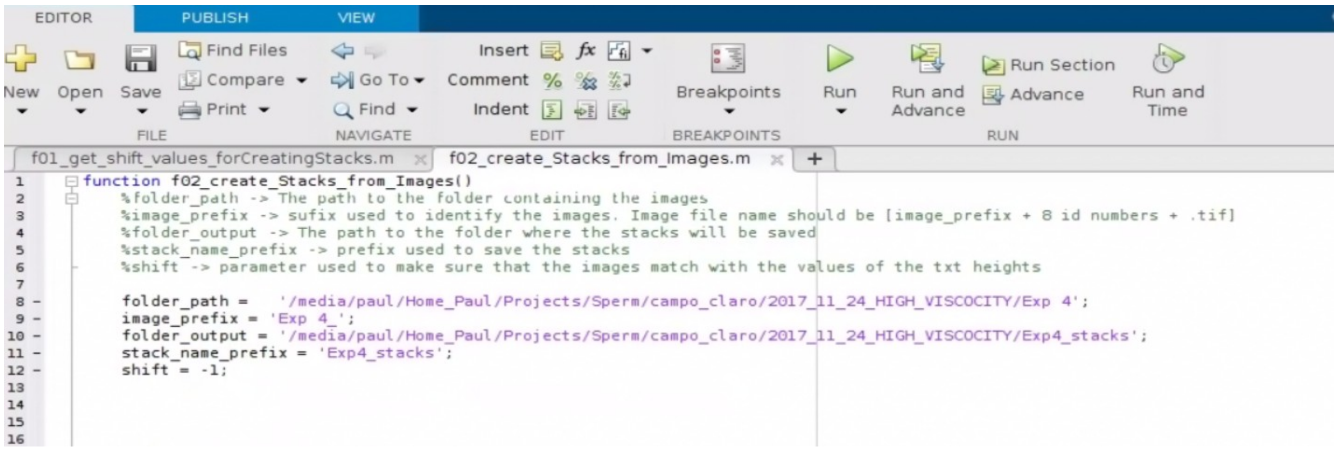
```
>> f01_get_shift_values_forCreatingStacks

mean_intensityStack =

  single

  220.0932


shift value is: -1
>>
```

Figure 4. Output from the function "**f01_get_shift_values_forCreatingStacks.m**".

2. Creating stacks

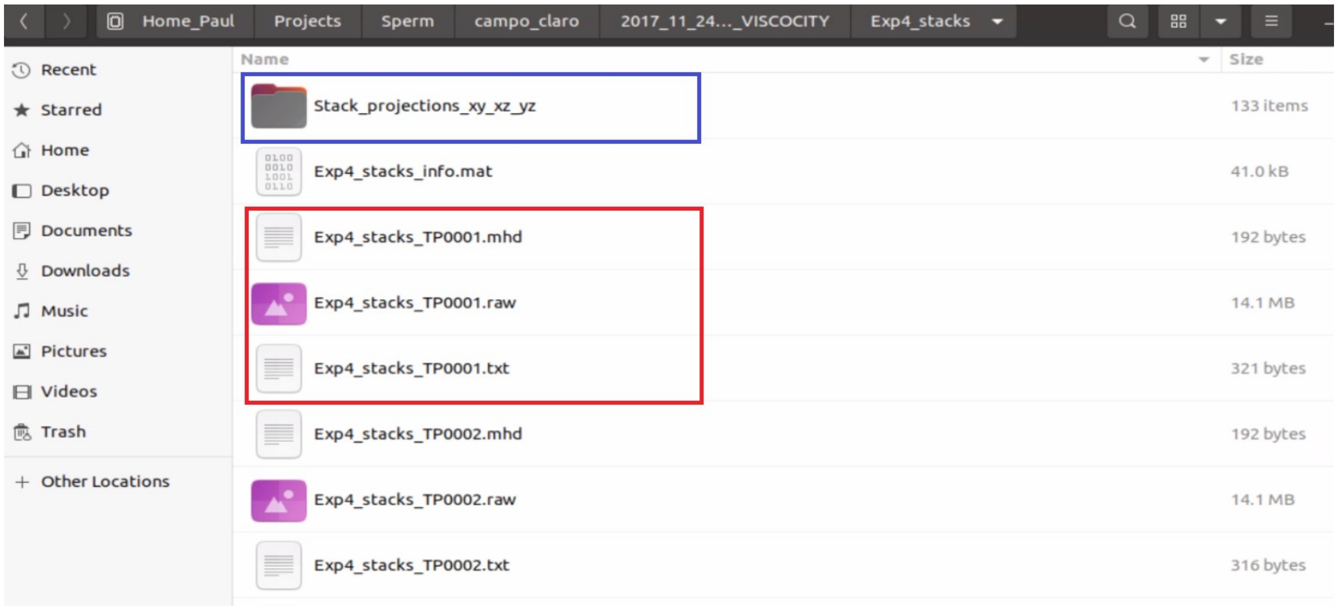
The function "**f02_create_Stacks_from_Images.m**" allows to create the 3D stacks as a raw file. It require the following inputs:
folder_path -> The path to the folder containing the images
image_prefix -> prefix used to identify the images. Image file name should be [image_prefix + 8 id numbers + .tif]
folder_output -> The path to the folder where the stacks will be saved
stack_name_prefix -> prefix used to save the stacks
shift -> parameter used to make sure that the images match with the values of the txt heights

Figure 5 depicts an example of setting the parameters for the function correctly. Next, the user can run the code which generates the 3d raw stack with an associated txt file of heights. Figure 6 depicts the output of the function "**f02_create_Stacks_from_Images.m**", three files are generated for each stack with extension "mhd", "raw" and txt (Fig. 6 red rectangle) and a folder "Stack_projections_xy_xz_yz" (Fig. 6 blue rectangle) containing the stack projection along the x, y and z axis which allows to identify the stacks that the sperm flagellum's is inside the 3D stack. The file mhd have information related to the size of the stack, number of dimentions, type of variable, etc, the file raw contain all the data, while the file txt have the z value for each slice. The folder "Stack_projections_xy_xz_yz" must be open (image by image) or dragging the folder to Fiji to visualize the images. Verify image to image that the flagellum's is inside the projections, if the flagellum is not inside the images then the tracing algorithm will not work. Figure 7 display an example of a sperm with the flagellum inside the stack, for this example the initial time point t=1 satisfy that the flagellum is inside the projections and, from t=1 to t=133 (last time point) all the projections satisfy that the flagellum is inside the projections. Thus, we can trace the flagellum's centerline for time point 1 to time point 133. Let's assume that the experiment only satisfy that the flagellum's centerline is inside the projections from time point 10 to time point 60, thus the algorithm can trace only 51 time points, this are few time points and the experiment must me disregard. I select experiments with 90 time points (1 second experiment) or more time points to trace.



```
function f02_create_Stacks_from_Images()
    %folder_path -> The path to the folder containing the images
    %image_prefix -> sufix used to identify the images. Image file name should be [image_prefix + 8 id numbers + .tif]
    %folder_output -> The path to the folder where the stacks will be saved
    %stack_name_prefix -> prefix used to save the stacks
    %shift -> parameter used to make sure that the images match with the values of the txt heights

    folder_path =   '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp 4';
    image_prefix = 'Exp 4_';
    folder_output = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks';
    stack_name_prefix = 'Exp4_stacks';
    shift = -1;
```

Figure 5. Example of setting the parameters for the function “**f02_create_Stacks_from_Images.m**” correctly.



Name	Size
Stack_projections_xy_xz_yz	133 items
Exp4_stacks_info.mat	41.0 kB
Exp4_stacks_TP0001.mhd	192 bytes
Exp4_stacks_TP0001.raw	14.1 MB
Exp4_stacks_TP0001.txt	321 bytes
Exp4_stacks_TP0002.mhd	192 bytes
Exp4_stacks_TP0002.raw	14.1 MB
Exp4_stacks_TP0002.txt	316 bytes

Figure 6. Output of the function “**f02_create_Stacks_from_Images.m**”.

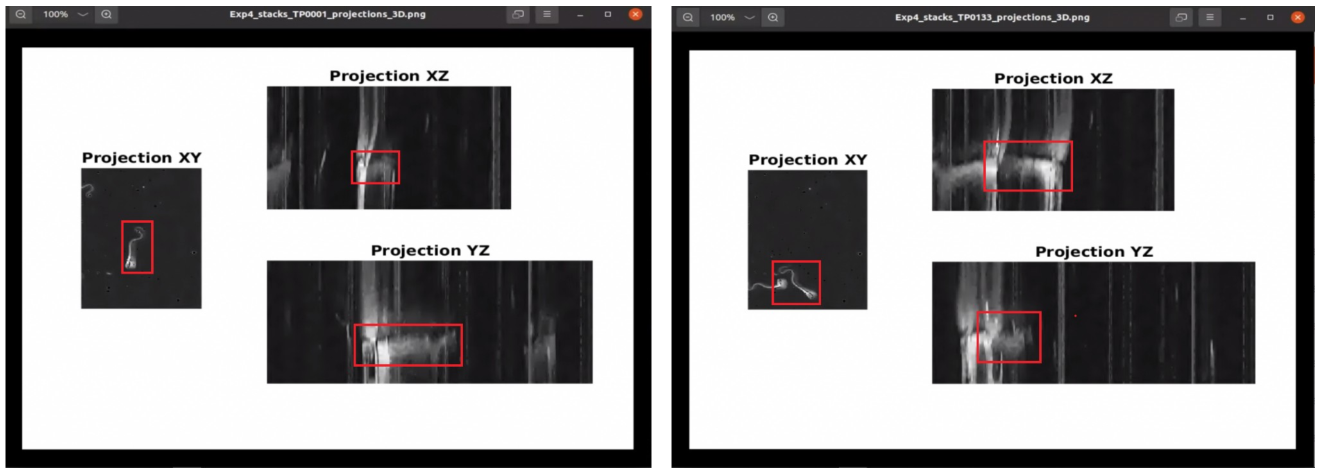
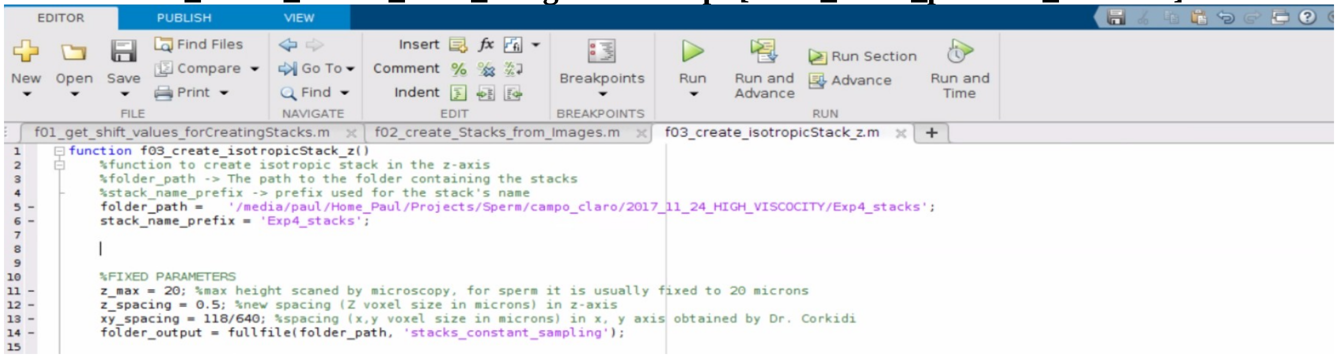


Figure 7. Example of a sperm with the flagellum inside the projections (time point 1 and time point 133)

3. Creating stack with uniform spacing in Z

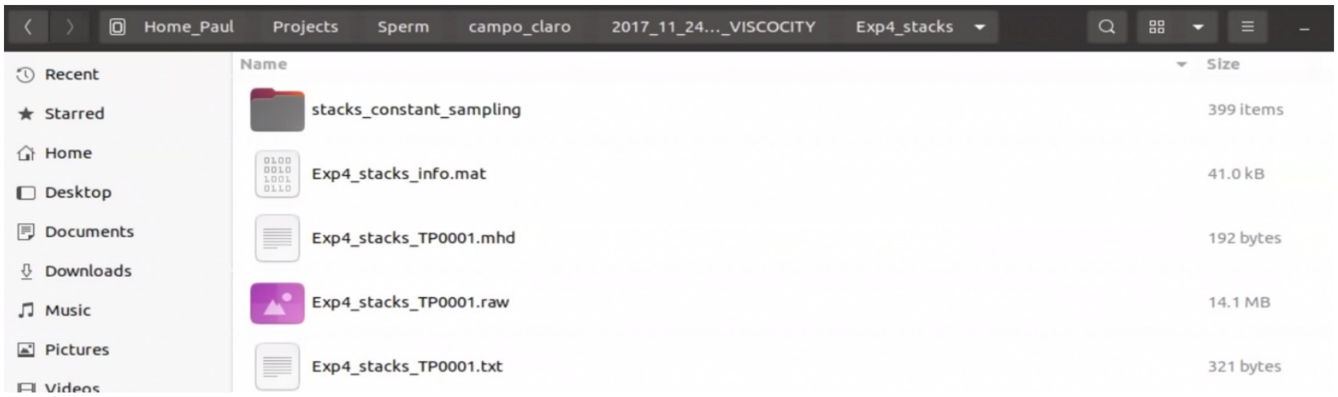
The previous stack does not have uniform spacing in z-axis (due to the piezoelectric devise moving at different speed usually slower at the button and top), you can verify this opening the txt file associated with the heights for each stack. The file to create isotropic stacks is "**f03_create_isotropicStack_z.m**", it requires the following inputs:
folder_path -> The path to the folder containing the stacks created with function "f02_create_Stacks_from_Images.m"
stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + "extension"]

Figure 8 depicts an example of setting the parameters for the function correctly. Next, the user can run the function, a folder with name "stacks_constant_sampling" is created containing the stacks with uniform sampling in z axis (see Fig. 9). If the folder is empty then there must be an error with the parameters.
Note: Generating these files can occupy a large amount of space, you can remove the files obtained from the microscopy and the files created (extension mhd, raw and txt) with the function "f02_create_Stacks_from_Images.m" except [stack_name_prefix + _info.mat]



```
function f03_create_isotropicStack_z()
    %function to create isotropic stack in the z-axis
    %folder_path -> The path to the folder containing the stacks
    %stack_name_prefix -> prefix used for the stack's name
    folder_path =   '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks';
    stack_name_prefix = 'Exp4_stacks';

    

    %FIXED PARAMETERS
    z_max = 20; %max height scaned by microscopy, for sperm it is usually fixed to 20 microns
    z_spacing = 0.5; %new spacing (Z voxel size in microns) in z-axis
    xy_spacing = 118/640; %spacing (x,y voxel size in microns) in x, y axis obtained by Dr. Corkidi
    folder_output = fullfile(folder_path, 'stacks_constant_sampling');

```

Figure 8. Example of setting the parameters for the function "**f03_create_isotropicStack_z.m**" correctly.



Name	Size
stacks_constant_sampling	399 items
Exp4_stacks_info.mat	41.0 kB
Exp4_stacks_TP0001.mhd	192 bytes
Exp4_stacks_TP0001.raw	14.1 MB
Exp4_stacks_TP0001.txt	321 bytes

Figure 9. Output of function “**f03_create_isotropicStack_z.m**”. A folder with name “stacks_constant_sampling” is created containing the stacks with uniform sampling in z axis.

4. Correcting drift in the XY plane

The images are acquired at different heights using the piezoelectric device, this generates a force that moves slightly the XY plane. This can be easily seen if there is garbage in the images, the center for the garbage must be fixed across the different planes (z-slices) however it moves (see Fig. 10). We use FIJI (https://imagej.net/software/fiji/) to correct the drift in the 3D image stack.

1. Download fiji (https://imagej.net/software/fiji/downloads) and uncompressed it (see Fig. 11)
2. Download StackReg (http://bigwww.epfl.ch/thevenaz/stackreg/) and uncompressed it. Copy the files StackReg_.jar and StackReg_.java to the folder “**Fiji.app/plugins**”
3. Download TurboReg (http://bigwww.epfl.ch/thevenaz/turboreg/) and uncompressed it. Copy the files TurboReg_.jar and TurboReg_.java to the folder “**Fiji.app/plugins**”
4. Download MIJI (http://bigwww.epfl.ch/sage/soft/mij/) jars files. We need mij.jar and ij.jar, then copy the files to the folder folder “Fiji.app/jars” and move the jars file to that folder.

If the files has been set correctly (the previous steps only has to be done once), then the user can run the function to correct the drift “**f04_correct_xy_drift.m**”, it requires the following inputs:
folder_path -> The path to the folder containing the stacks with **uniform spacing** created with function “f03_create_isotropicStack_z.m”
stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + “extension”]
threshold_head -> a threshold value that segments the head but removed the flagellum.
seed_point_head -> 3D position of the sperm’s head center
TP_initial -> the initial time point containing a sperm head (usually time point equal to 1)

Figure 12 depicts an example of how to calculate the **threshold_head, seed_point_head and TP_initial** parameters. The first stack with uniform spacing is open, and the sperm head is identified. Next, the cursor (red dot) is located at the center of the sperm head and the correct z position is identified (z=9). The Fiji toolbar (red rectangle) provides information of cursor location and position intensity in the red square. The intensity at the head is 254, then we can chose thresholdHead using a lower value, in this case 250 works fine, the seed_point position is x= 434, y = 186 and z = 9. Finally, the initial_timePoint is equal to 1 because the first stack contains this sperm.
Figure 13 depicts an example of setting this parameters correctly in the function “**f04_correct_xy_drift.m**”. Next, the user can run the function, a new stack with the drift corrected is created for each stack of the folder “**folder_path**”. The new stacks file name should be

[stack_name_prefix + 8 id numbers + "_DC" +"extension"]. Figure 14 depicts an example of correcting the drift for the images depicted in Fig. 10.

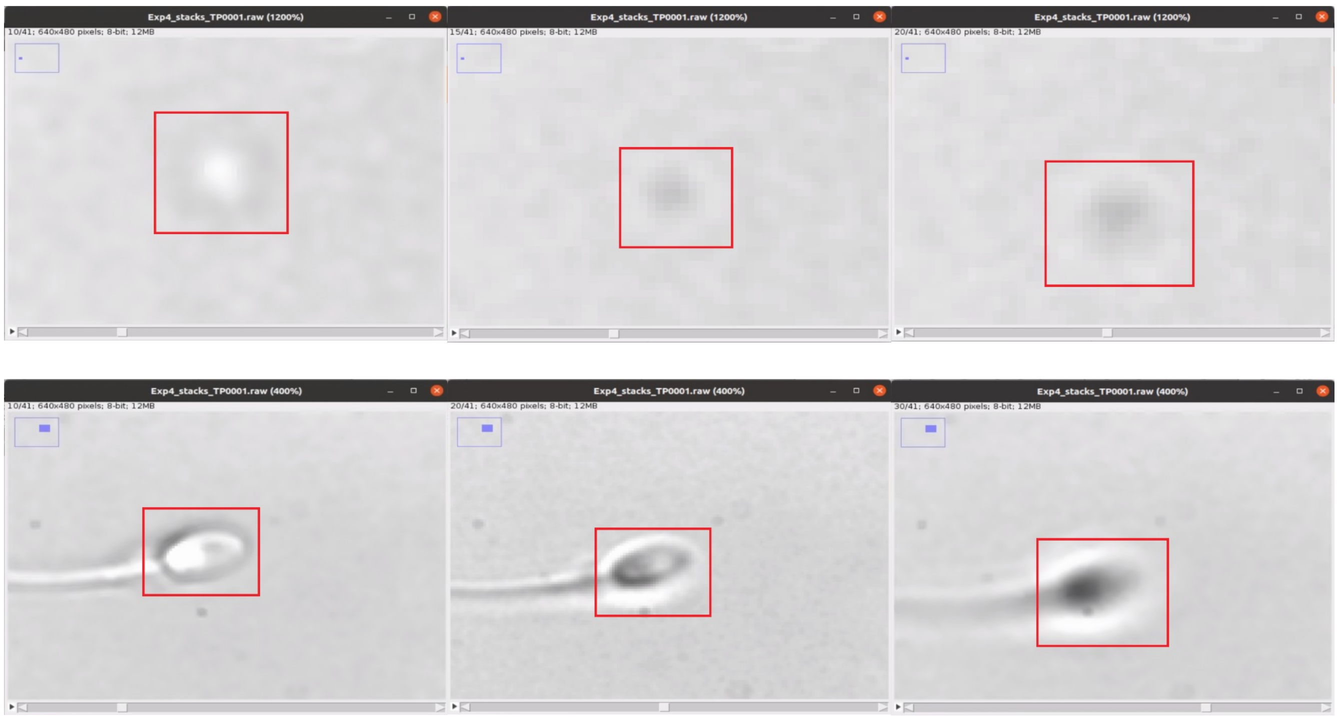

Exp4_stacks_TP0001.raw (1200%)
10/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001.raw (1200%)
15/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001.raw (1200%)
20/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001.raw (400%)
10/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001.raw (400%)
20/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001.raw (400%)
30/41; 640x480 pixels; 8-bit; 12MB

Figure 10. Example of drift in the image. First row, fixed spot (a) at the slice z=10, (b) at z= 15 and (c) at z= 20, we can easily observe that in 10 images the center has moved. Second row, sperm head (a) at the slice z=10, (b) at z= 20 and (c) at z= 30

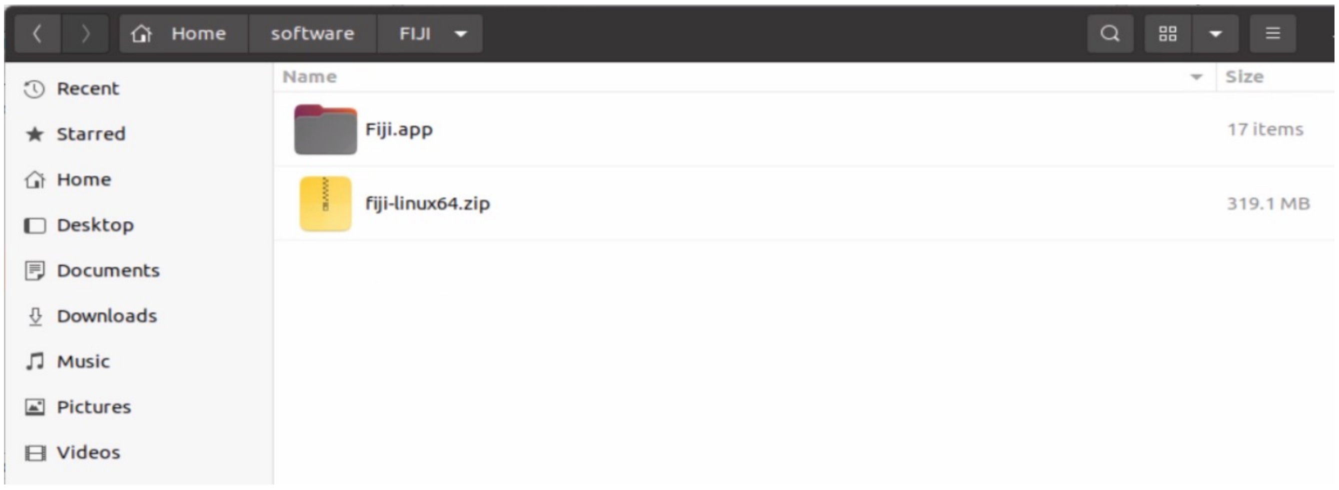
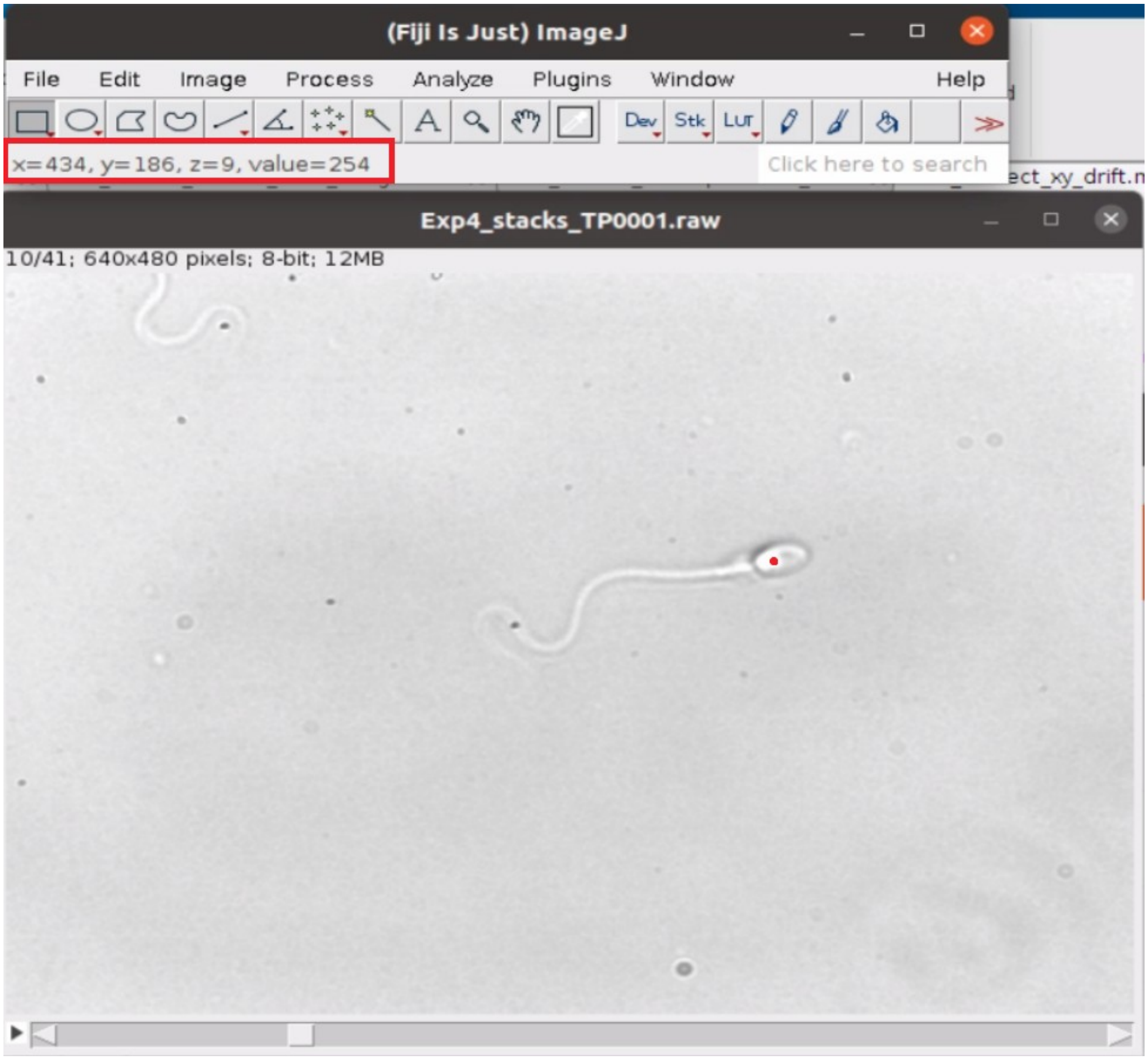
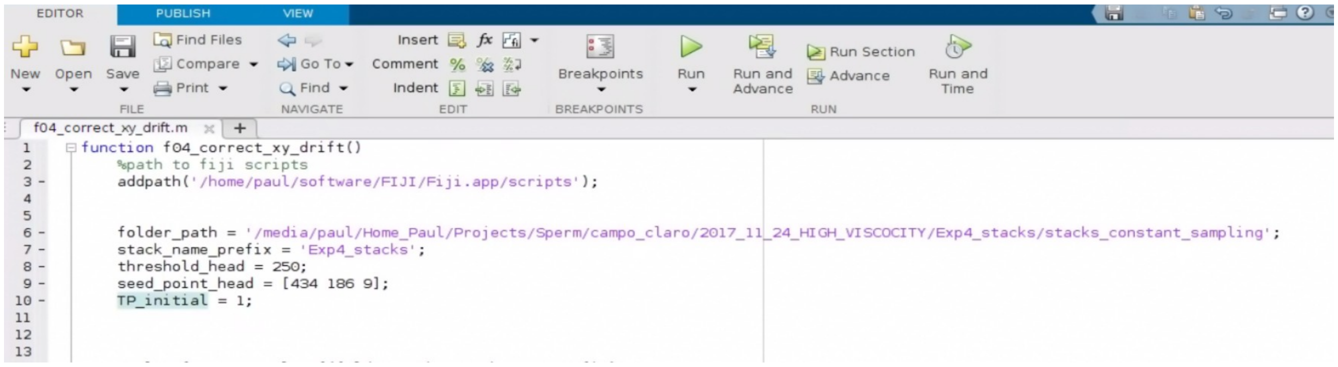


Figure 11. Downloading Fiji and unzipping it.



```
x=434, y=186, z=9, value=254
```

Figure 12. Getting the sperm parameters for the function, first we select the z-plane that is at the middle of the head (z=9), the we position the mouse at the center of the head (red dot). The red rectangle show the head's center position (x= 434, y = 186 and z = 9) and the intensity (254)



```
function f04_correct_xy_drift()
    %path to fiji scripts
    addpath('/home/paul/software/FIJI/Fiji.app/scripts');


    folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks/stacks_constant_sampling';
    stack_name_prefix = 'Exp4_stacks';
    threshold_head = 250;
    seed_point_head = [434 186 9];
    TP_initial = 1;
```

Figure 13. Example of setting the parameters for the function “**f04_correct_xy_drift.m**” correctly.

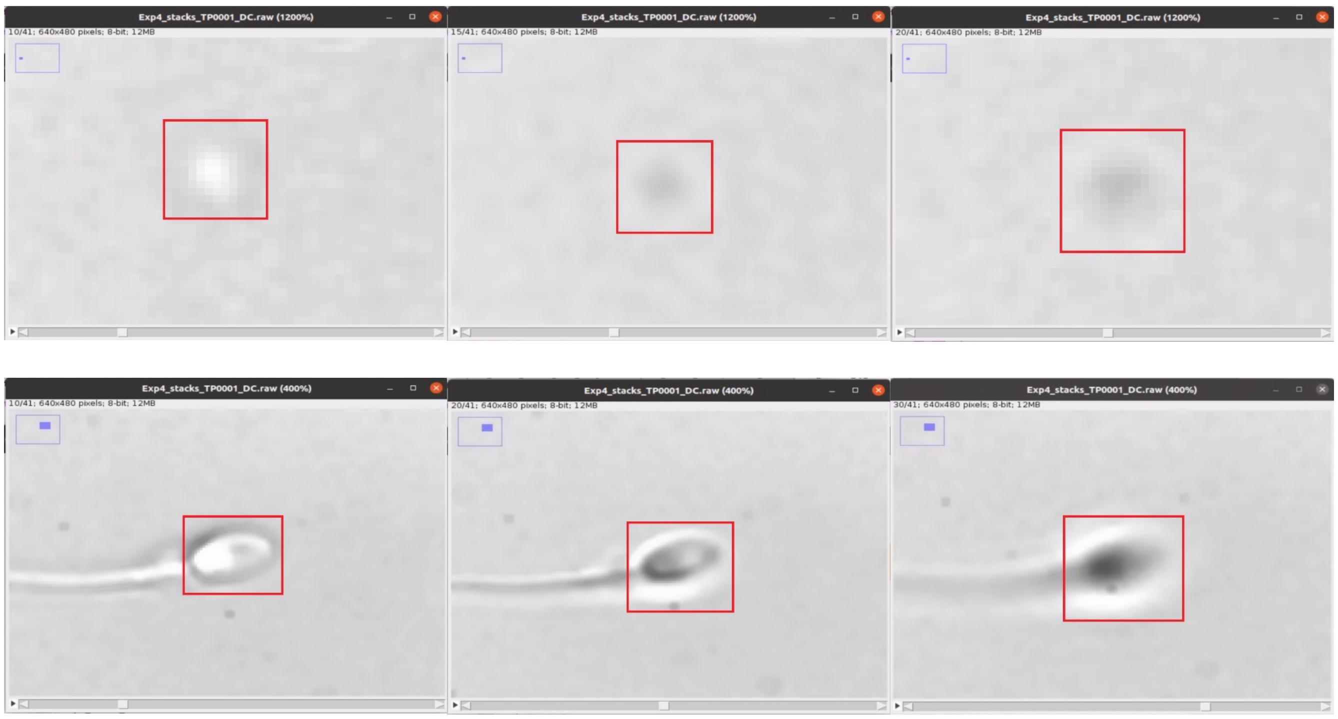

Exp4_stacks_TP0001_DC.raw (1200%)
10/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001_DC.raw (1200%)
15/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001_DC.raw (1200%)
20/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001_DC.raw (400%)
10/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001_DC.raw (400%)
20/41; 640x480 pixels; 8-bit; 12MB
Exp4_stacks_TP0001_DC.raw (400%)
30/41; 640x480 pixels; 8-bit; 12MB

Figure 14. Example of correcting the drift for the images in Fig. 10.

5. Detecting the flagellum tip for each sperm

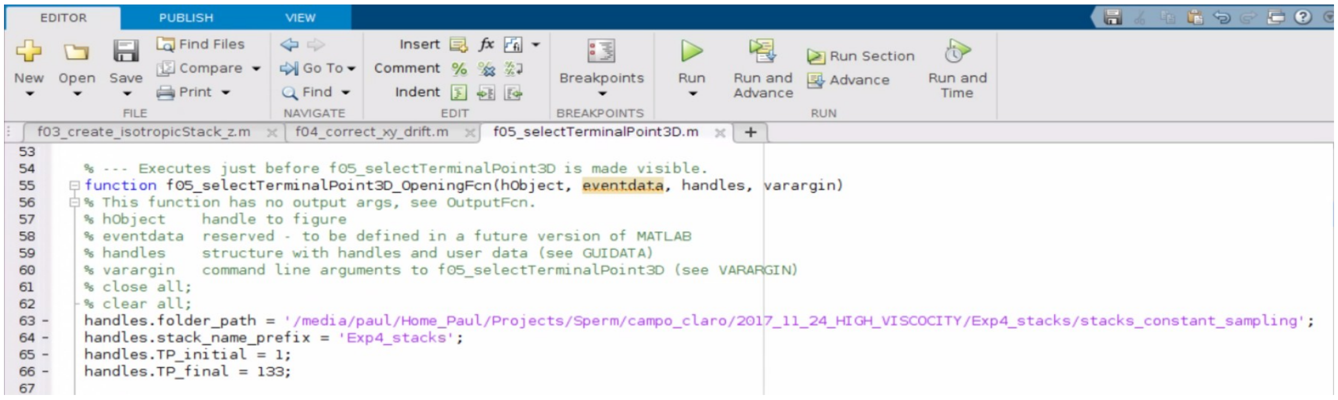
The current version to trace the flagellum’s centerline for bright field images requires the user to manually detect the flagellum’s terminal point for each sperm. We have created a graphic user interface (GUI) that allows to easily track the terminal point for each experiment. The function “**f05_selectTerminalPoint3D.m**” allows to open the GUI, it requires the following inputs:

folder_path -> The path to the folder containing the stacks with **uniform spacing** created with function “f03_create_isotropicStack_z.m”
stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + “extension”]
TP_initial -> initial time point to trace the sperm centerline
TP_final -> last time point to trace the sperm centerline

TP_initial and **TP_final** parameters correspond to the initial time point and final time point that the flagellum is inside the 3D stack (see Section 2).

These parameters should be set at line 63-66 from the file “**f05_selectTerminalPoint3D.m**”. Figure 15 depicts an example of setting this parameters correctly. Note that, we are selecting the TP_initial = 1 and TP_final = 133 this means that the flagellum is inside the stack from time point 1 up to time point 133. Next, the user can run the script “**f05_selectTerminalPoint3D.m**”, if the parameters are correctly the a Graphic User Interface (GUI) will be showed as depicted in Figure 16. The GUI has 3 images displaying the maximum intensity projection of the 3D current stack along the z-axis (xy view), x-axis (yz view) and y-axis (xz view), two buttons to navigate across different time points (**Next** and **Previous**) and Windows size that allows to choose the neighborhood size to display the image. Usually, the user only has to use the xy view to select the terminal point by clicking the flagellum’s tail tip, this is because the last part of the flagellum’s tail is aligned with the xy plane (tha tracing algorithm works fine in these cases). There are some cases were the last part of the flagellum’s tail is aligned with the z-axis (seen as a vertical bright line in xz and yz), in these cases the tracing algorithm has difficulties and the user must also select the terminal point in the xz or yz plane. Figure 17 depicts an example of tracking flagellum’s tip point. The current track point is displayed as a white asterisk, while the previous 10 tracked points are depicted as a red asterisks. The user must track the terminal point from TP_initial to TP_final without closing the GUI. Only after finishing tracking all the time points, the user can close the windows. The previous requirement is necessary because a file is generated every-time that the script “**f05_selectTerminalPoint3D.m**” is run with the positions of the points that were tracked.

The generated file is located in the folder where the stacks are saved (**folder_path**) with the file name given be [stack_name_prefix + “_endPoints_” + N + “.mat”] where N is the number of times that the script “**f05_selectTerminalPoint3D.m**” has been ran.



```
% --- Executes just before f05_selectTerminalPoint3D is made visible.
function f05_selectTerminalPoint3D_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to f05_selectTerminalPoint3D (see VARARGIN)
% close all;
% clear all;
handles.folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks/stacks_constant_sampling';
handles.stack_name_prefix = 'Exp4_stacks';
handles.TP_initial = 1;
handles.TP_final = 133;
```

Figure 15. Example of setting the parameters for the function “**f05_selectTerminalPoint3D.m**” correctly.

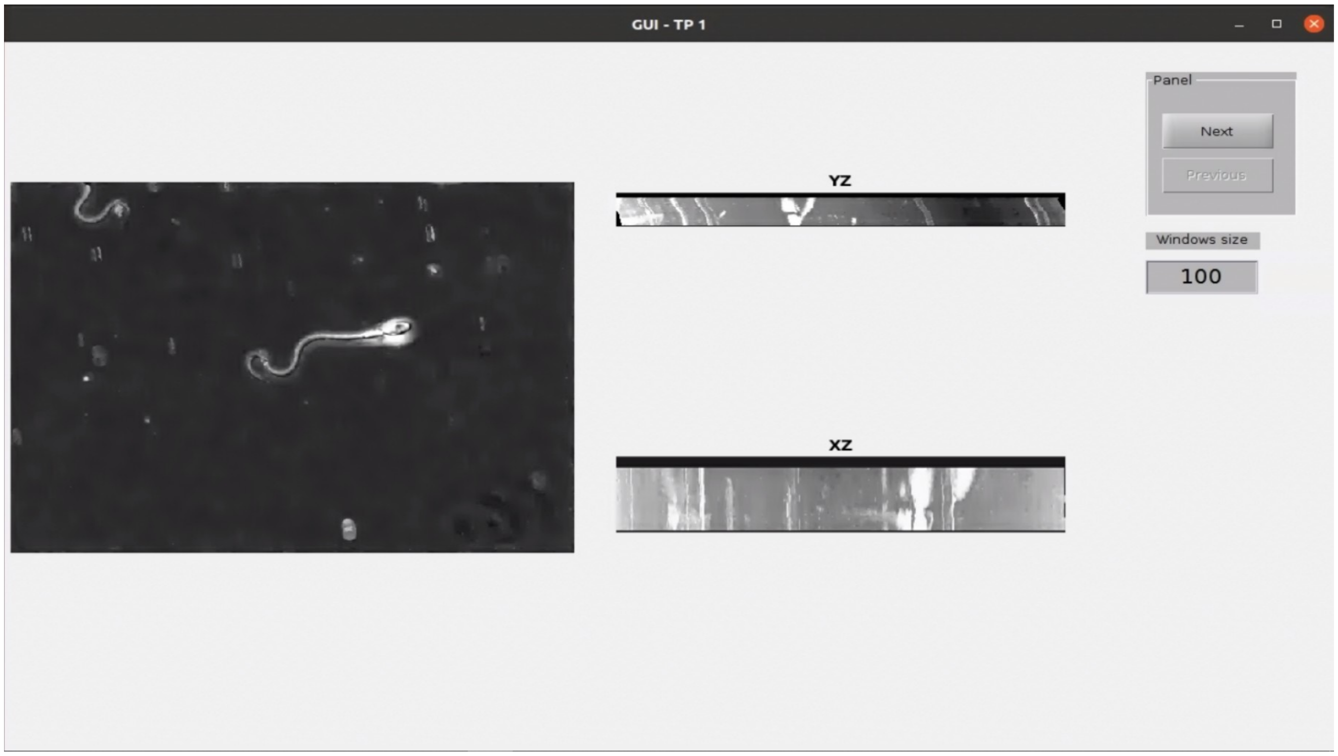


Fig. 16 Graphic User Interface to track sperm’s tail tip

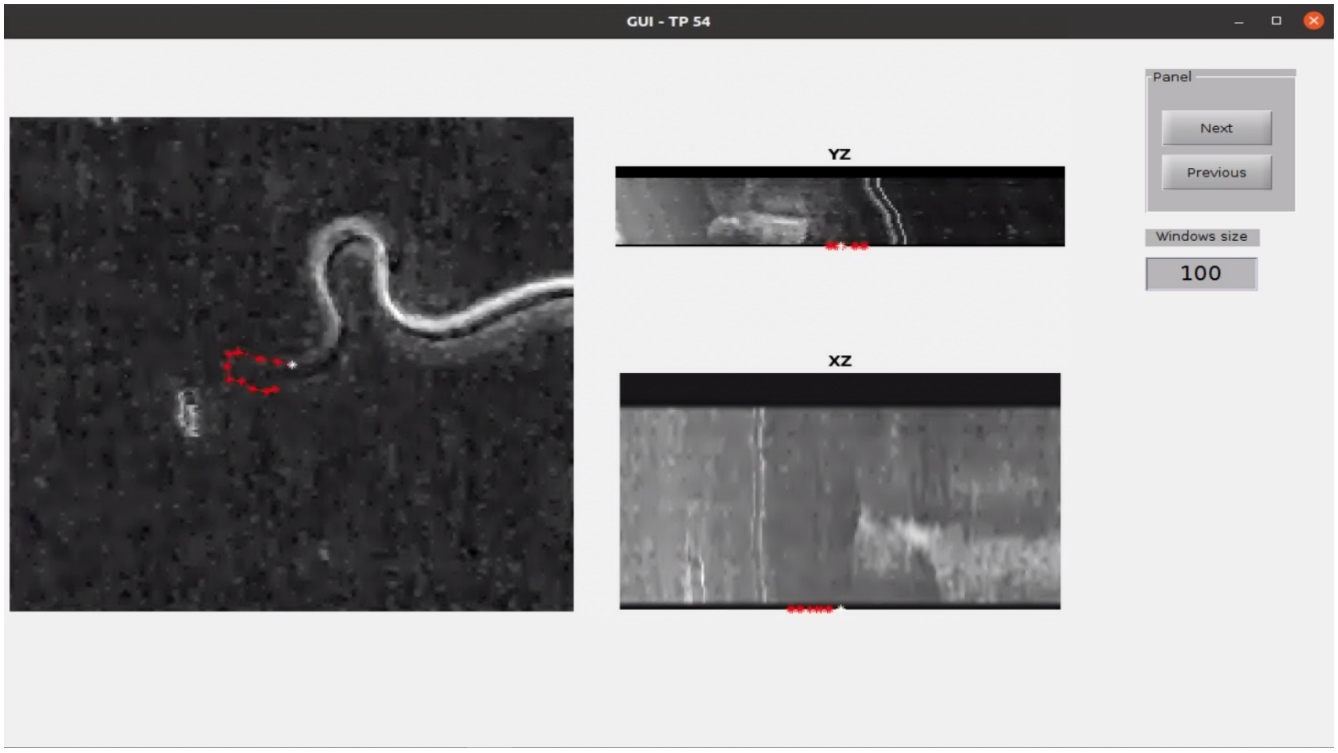
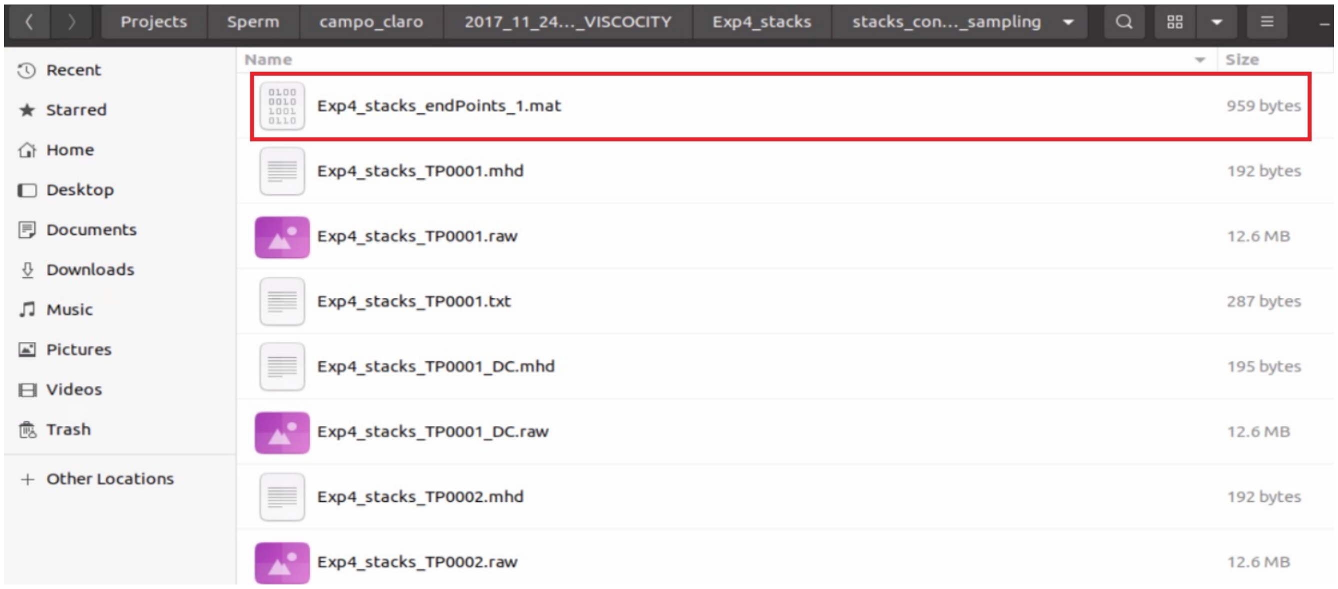


Figure 17 Tracking flagellum’s tip point. The current track point is displayed as a white asterisk, while

the last 10 tracked points are depicted as a red asterisk.



Name	Size
Exp4_stacks_endPoints_1.mat	959 bytes
Exp4_stacks_TP0001.mhd	192 bytes
Exp4_stacks_TP0001.raw	12.6 MB
Exp4_stacks_TP0001.txt	287 bytes
Exp4_stacks_TP0001_DC.mhd	195 bytes
Exp4_stacks_TP0001_DC.raw	12.6 MB
Exp4_stacks_TP0002.mhd	192 bytes
Exp4_stacks_TP0002.raw	12.6 MB

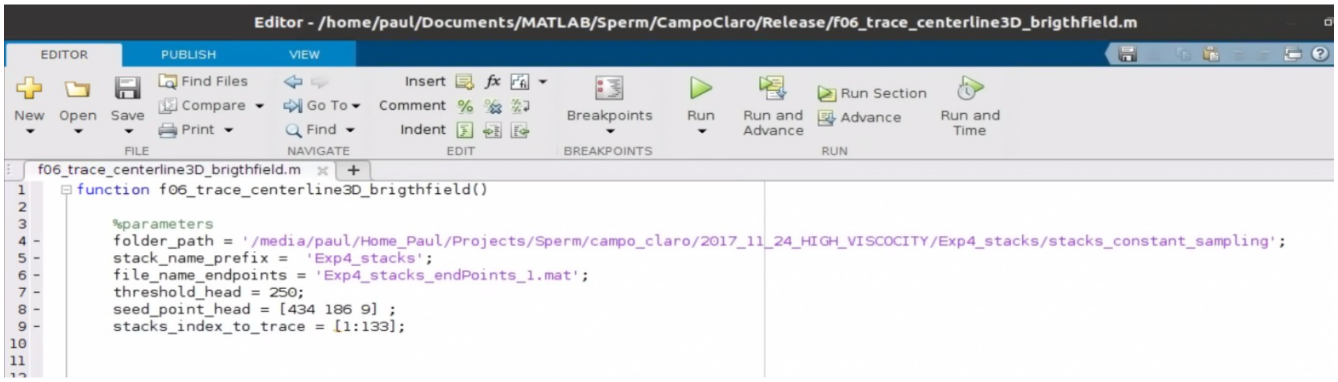
Figure 18. Example of file generated using the GUI in red rectangle.

6. Running the tracing algorithm

The previous steps allows to setup the 3D stack and the terminal points. The script "**f06_trace_centerline3D_brigthfield.m**" allows to trace the flagellum's centerline from the 3D image stacks, it requires the following inputs:
folder_path -> The path to the folder containing the stacks with **uniform spacing** created with function "f03_create_isotropicStack_z.m"
stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + "extension"]
file_name_endpoints -> the mat file containing the information of terminal points and created using the script (**f05_selectTerminalPoint3D.m**) described in the previous step.
threshold_head -> a threshold value that segments the head but removed the flagellum.
seed_point_head -> 3D position of the sperm's head center
stacks_index_to_trace = [**TP_initial**:**TP_final**] -> TP_initial is the initial time point to trace while TP_final is the final time point to trace.

Usually **threshold_head, seed_point_head** and **TP_initial** are the same values as those used for the script "**f04_correct_xy_drift.m**" and the values **TP_initial**, **TP_final** must be equal to the time points for which the flagellum's tip point was detected and should match the values used for the script "**f05_selectTerminalPoint3D.m**". Figure 19 depicts an example of setting the correct parameters for tracing the flagellum's centerline, then the script can be run. The algorithm took approximately 16 minutes to trace the 133 stacks (7.2s per stack), three new files are generated per stack
[stack_name_prefix + 8 id numbers + "DC_trace.swc"] -> have the flagellum's centerline coordinates in the swc file format (http://www.neuronland.org/NLMorphologyConverter/MorphologyFormats/SWC/Spec.html).
[stack_name_prefix + 8 id numbers + "DC_trace.swc.vtk"] -> paraview vtk file to visualize the trace in paraview

[stack_name_prefix + 8 id numbers + "DC_rec.png"] -> maximum intensity projection along the z-axis for the stack with an overlay of the flagellum's centerline tracing in green. The "png" files are saved in the folder "**trace_projections_xy**" located inside the folder "**folder_path**".



```
function f06_trace_centerline3D_brigthfield()

    %parameters
    folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks/stacks_constant_sampling';
    stack_name_prefix =  'Exp4_stacks';
    file_name_endpoints = 'Exp4_stacks_endPoints_1.mat';
    threshold_head = 250;
    seed_point_head = [434 186 9] ;
    stacks_index_to_trace = [1:133];
```

Figure 19. Example of setting the parameters for the function "**f06_trace_centerline3D_brigthfield.m**" correctly.

7. Correcting errors in tracing

The folder containing "**trace_projections_xy**" containing the created images **[stack_name_prefix + 8 id numbers + "DC_rec.png"]** in the previous step is very important to easily verify the output of the tracing algorithm in the xy projection. Usually, the tracing algorithm works fine for all the cases, however the tracing may fail in some cases (due to a bright structure near the centerline, a lot of bending in the flagellum, another flagellum near, etc). In these difficult cases, the tracing algorithm needs addition information to trace correctly the flagellum. First, the user needs to identify a stack incorrectly trace, to this end visually inspect any of the images and identify an image with a possible error due to wrong tracing or incorrect location of the sperm's head center position. Figure 20 display an example where the tracing algorithm incorrectly traces the flagellum's tail (red rectangle) due to a bright-structure near the flagellum. Once that an incorrect trace has been identified (in this case it correspond to time point 16, you can identified the time point in the name of the image), the user needs to correct it. To this end, we have developed an GUI with the features to correct the trace. Figure 21 depicts the GUI, it has the following features:

1. **Display projection**: it display three different views corresponding to the maximum intensity projection along the z-axes (**xy projection**), y-axes (**xz projection**) and x-axes (**yz projection**) to visually inspect the image stack.
2. **Display trace**: every projection can also be overlay with the centerline traces by the algorithm, displayed as a blue line. The user can hide the trace by deactivating the "**Display trace**" checkbox.
3. **Zoom in/Zoom out**: The user can zoom in/zoom out by pressing in the keyboard "+" (addition) or "-" (subtraction) in the xy view. Also using the keyboard letter "a" and "s". Note this option only works if the user is not using any of the other options (if all the buttons are in gray).
4. **Set sperm's head center point**: The user can modify the sperm's head center point used by the algorithm by pressing the button "**Set head center**", the user then needs to select the correct position in the **xy, yz or xz projections** to have the coordinate 3D of the sperm seed point, a red point indicating the current position is displayed every time the user selects a new position. The button's "**Set head center**" color is changed to green to indicate that is currently active, the user can deactivate this option by pressing again the button "**Set head center**" (the button's color

must change to gray). Note: this option must be used only if the sperm's head center position is incorrectly identified by the algorithm.

5. **Set sperm's flagellum's tip**: The user can modify the flagellum's tip used by the algorithm by pressing the button "**Set flagellum's tip**", the user then needs to select the correct position in the **xy, yz or xz projections** to have the coordinate 3D of the flagellum's tip, a green point indicating the current position is displayed every time the user selects a new position. The button's "**Set flagellum's tip**" color is changed to green to indicate that is currently active, the user can deactivate this option by pressing again the button "**Set flagellum's tip**" (the button's color must change to gray). Note: this option must be used only if the flagellum's tip point was incorrectly identified by the algorithm.
6. **Remove objects**: A possible source of errors for the tracing algorithm is having bright-structures near the flagellum, and possible taking shortcut's for a flagellum with a large amount of bending. The user can use the buttons "**draw polygon xy**", "**draw polygon xz**" and "**draw polygon yz**" to set region were the algorithm should not search for the flagellum's centerline. Therefore, avoiding errors. Press the button corresponding to the plane (the button's color must change to green to indicate that it is currently in use) where the user wants to remove a region (usually the xy plane), the button should change to color green (see Fig. 22a) and the user can left-click in the plane to define the region (blue region with transparency) to remove . Finally, to finish the region the user can double click (for a detailed explanation on how to draw the region see https://www.mathworks.com/help/images/ref/drawpolygon.html#mw_c843532b-20b1-45e2-a60b-4ec12390b674). If the user finished a region, then it can press again the button (the button's color must change to gray to indicate that it is currently deactivated the option) to remove the region (see Fig. 22b). The user must repeat this process for the regions where the tracing algorithm has errors (usually one region per stack).
7. **Undo region**: if for some reason the user incorrectly removed a region, then it can be undo be pressing the button "**Undo region**".
8. **Run tracing**: The user can run the tracing algorithm using the previous defined features by pressing the button "**Run tracing**", the button's color must change to red indicating that it is currently tracing the centerline (it takes around 7 seconds to trace). Once the tracing algorithm has finished the button's color must change to gray and the display is automatically updated with the new trace (see Fig. 23).

The script "**f07_correct_tracing.m**" allows to display the GUI, it requires the following inputs:
folder_path -> The path to the folder containing the stacks with **uniform spacing** created with function "f03_create_isotropicStack_z.m"
stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + "extension"]
file_name_endpoints -> the mat file containing the information of terminal points and created using the script (**f05_selectTerminalPoint3D.m**) described in the previous step.
threshold_head -> a threshold value that segments the head but removed the flagellum.
seed_point_head -> 3D position of the sperm's head center
stacks_index_to_trace -> Time point to correct the trace.

These parameters should be set at line 57-62 from the file "**f07_correct_tracing.m**". Note that this parameters are exactly the same as those defined for the function "**f06_trace_centerline3D_brigthfield.m**", except that in this case the parameter "**stacks_index_to_trace**" should be set only to the time point to be corrected. Figure 24 depicts an example of setting the correct parameters for tracing the flagellum's centerline (script "**f07_correct_tracing.m**"), then the script can be run.

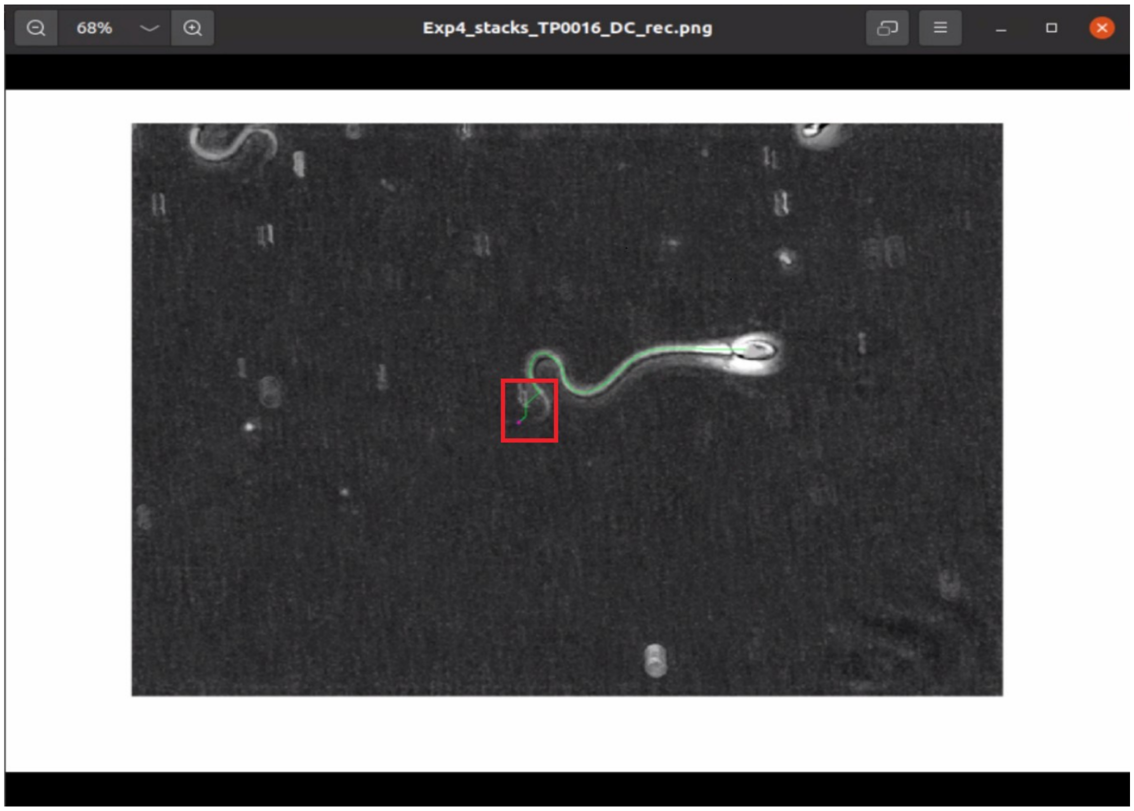


Figure 20. Identifying an incorrect flagellum's centerline tracing in red rectangle.

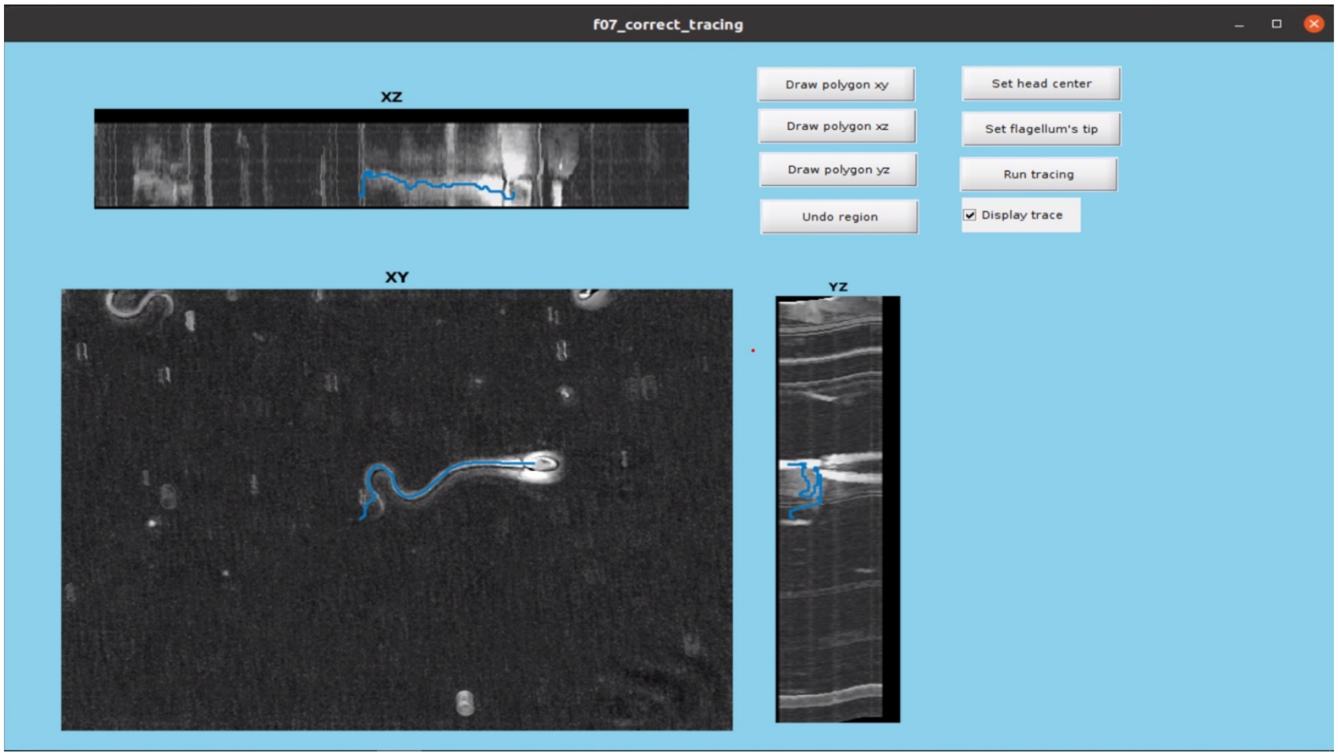
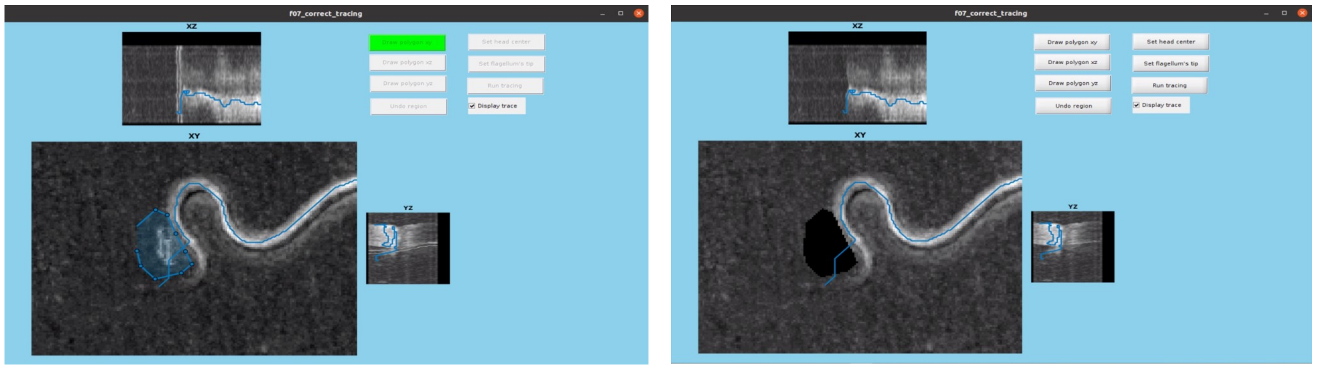


Figure 21. GUI displayed when correcting tracing errors.



(a) (b)

Figure 22. (a) GUI setting a region where the algorithm should not search for the flagellum’s centerline. (b) region already created in black.

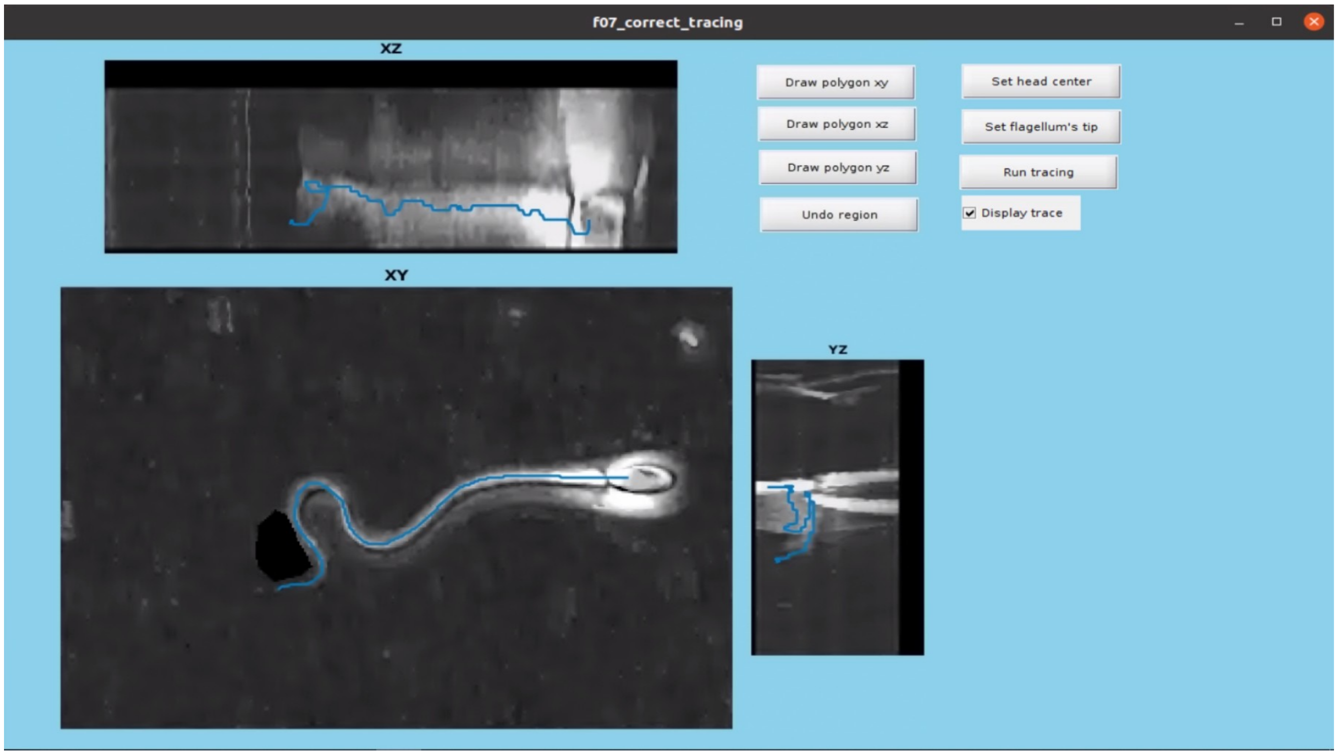
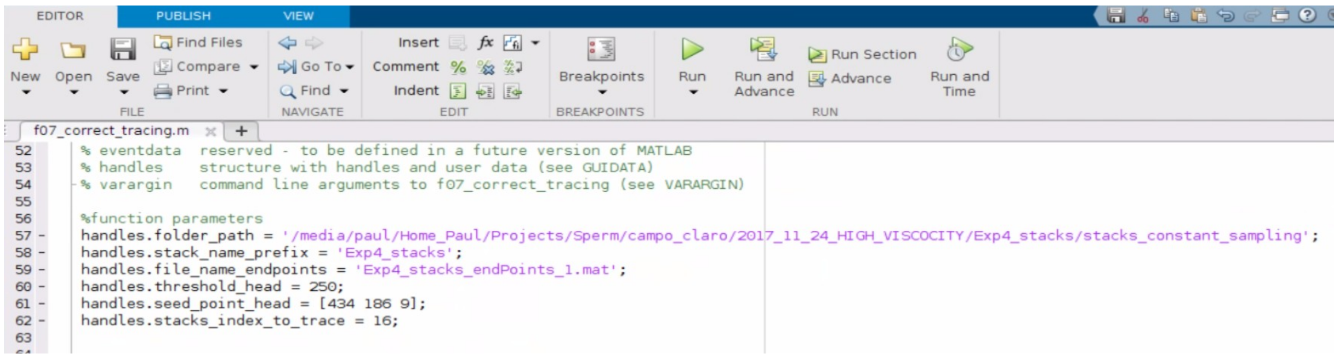


Figure 23. Example of correcting the trace



```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to f07_correct_tracing (see VARARGIN)

%function parameters
handles.folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks/stacks_constant_sampling';
handles.stack_name_prefix = 'Exp4_stacks';
handles.file_name_endpoints = 'Exp4_stacks_endPoints_1.mat';
handles.threshold_head = 250;
handles.seed_point_head = [434 186 9];
handles.stacks_index_to_trace = 16;
```

Figure 24: Example of setting the parameters for the function “**f07_correct_tracing.m**” correctly.

8. Visual verification of trace in the three views (xy, yz, xz)

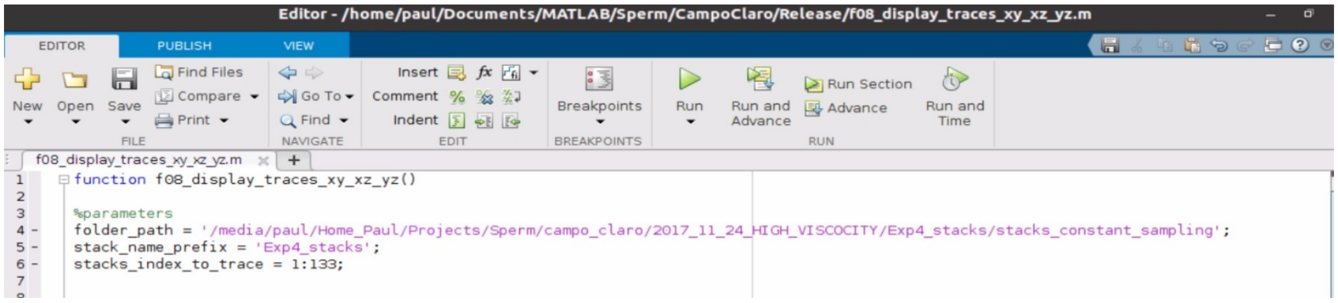
So far, we have already verified the trace in the xy view. To make sure that there are not errors in the trace, we created a function to display the trace in the three different views. The script “**f08_display_traces_xy_xz_yz.m**” allows to create visualization in the views xy, yz and xz, it requires the following parameters:

folder_path -> The path to the folder containing the stacks with **uniform spacing** created with function “f03_create_isotropicStack_z.m”

stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + “extension”]

stacks_index_to_trace = [**TP_initial**:**TP_final**] -> **TP_initial** is the initial time point to trace while **TP_final** is the final time point to trace.

Note that this parameters are exactly the same as those defined for the function “**f06_trace_centerline3D_brigthfield.m**”. Figure 25 depicts an example of setting this parameters correctly, then the user can run the script. A new folder named “**trace_projections_xy_xz_yz**” inside “**folder_path**” is created with the projections for each stack. The user can verify that each trace is correctly trace in the three planes, if the user finds an error in the trace then it should run the script “**f07_correct_tracing.m**” to correct the trace. Figure 26 depicts an example of verification, the left column correspond to the maximum intensity projection (MIP), while the right column has the overlay of the trace and the MIP.



```
function f08_display_traces_xy_xz_yz()

%parameters
folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks/stacks_constant_sampling';
stack_name_prefix = 'Exp4_stacks';
stacks_index_to_trace = 1:133;
```

Figure 25: Example of setting the parameters for the function “**f08_display_traces_xy_xz_yz .m**” correctly.

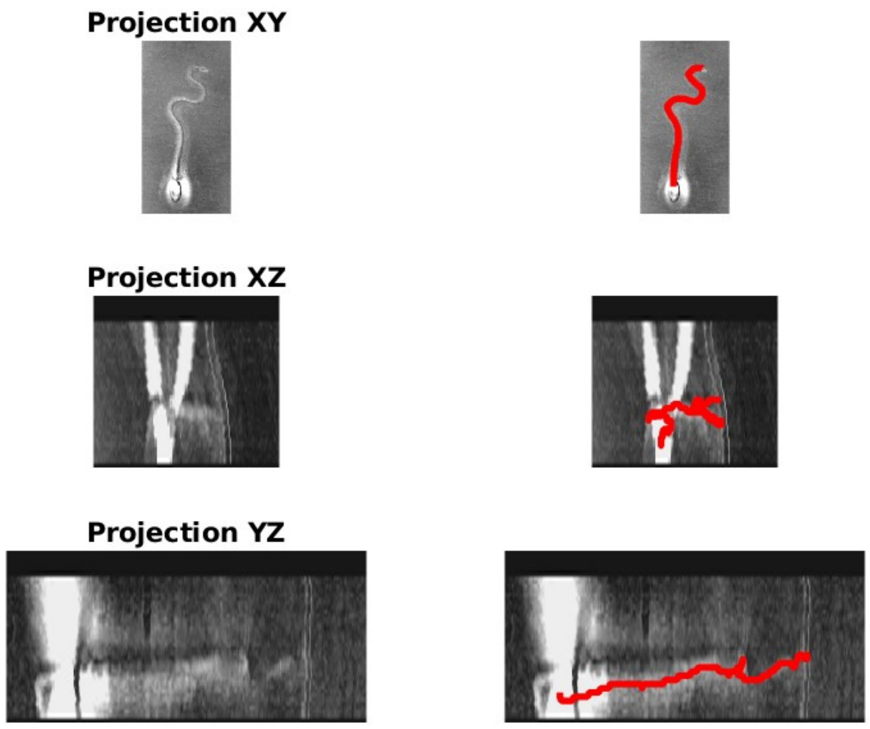


Figure 26. Verification of correct trace in the three view, the left column correspond to the maximum intensity projection (MIP), while the right column has the overlay of the trace and the MIP.

9. Convert traces to micrometers

The last step convert the voxel coordinates to micrometers and we also smooth the traces with penalized cubic splines (https://www.mathworks.com/help/curvefit/csaps.html). The script "**f09_convert_traces_to_micron_and_smooth.m**" allows to do this, it requires the following parameters:

folder_path -> The path to the folder containing the stacks with **uniform spacing** created with function "f03_create_isotropicStack_z.m"

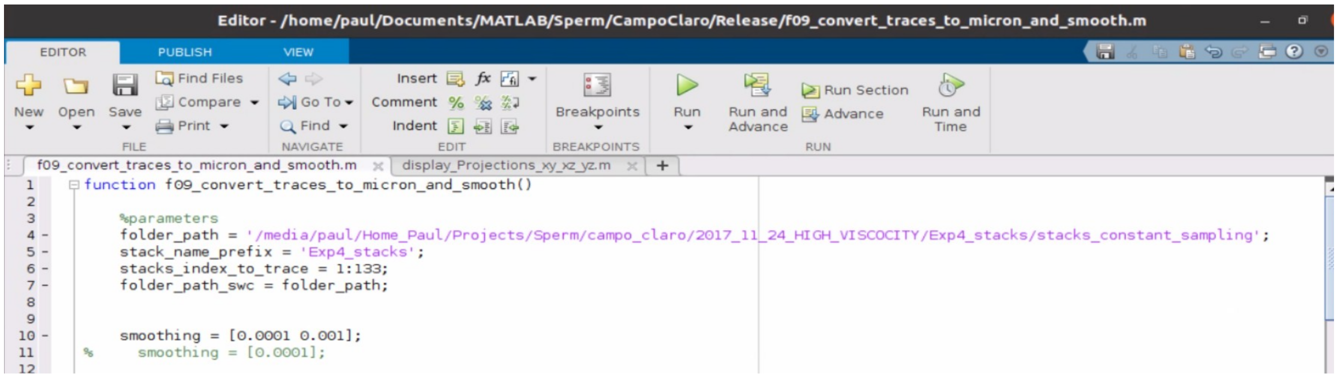
stack_name_prefix -> prefix used to identify the stack. Stacks file name should be [stack_name_prefix + 8 id numbers + "extension"]

stacks_index_to_trace = [**TP_initial**:**TP_final**] -> **TP_initial** is the initial time point to trace while **TP_final** is the final time point to trace.

folder_path_swc -> The path to the folder containing the traces created with function "**f06_trace_centerline3D_brigthfield.m**". Note that this folder is usually the same that **folder_path**, however in some cases if the user moves the traces to a different folder (for organization purposes) then this path must be changed.

smoothing -> the amount of smoothing to apply in the range (0, 1], the value 1 corresponds to not smoothing while near to 0 correspond to maximum smoothing. I usually use a value of 0.0001 but the user can use other values such as 0.001.

Figure 27 depicts an example of correctly setting the parameters for the script "**f09_convert_traces_to_micron_and_smooth.m**". Next, the user can run the script, it will generate the folder "**trace_micras**" inside the folder "**folder_path**", this folder contain all the data in micrometers, it has the following folders:
1) **raw** → have the swc file with the coordinates in micrometers, without applying smoothing.
2) **trace_smooth** → have a folder for each value of smoothing applied. Each folder has the swc file with the coordinates in micrometers and applying smoothing.
3) **images_traces_interpolated_xy** → images displaying the raw traces, with an overlay for the different values of smoothing utilized (only the xy plane). Figure 28 depict an example of this overlay, note that a smoothing of 0.001 is very similar to the raw trace, while the value 0.0001 creates an even smoother version of the trace.
Note: the traces created have only 100 points per trace as required by Hermes.
Additinally, mat files are created in the folder "**trace_micras**" which have the X, Y, and Z coordinates of the traces to be easily open in Matlab. Figure 29 display an example of the generated folders and mat files.


Editor - /home/paul/Documents/MATLAB/Sperm/CampoClaro/Release/f09_convert_traces_to_micron_and_smooth.m
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
Go To
Find
NAVIGATE
Insert
Comment
Indent
EDIT
Breakpoints
BREAKPOINTS
Run
Run and Advance
Run Section
Advance
Run and Time
RUN
f09_convert_traces_to_micron_and_smooth.m
display_Projections_xy_xz_yz.m

```
function f09_convert_traces_to_micron_and_smooth()

    %parameters
    folder_path = '/media/paul/Home_Paul/Projects/Sperm/campo_claro/2017_11_24_HIGH_VISCOCITY/Exp4_stacks/stacks_constant_sampling';
    stack_name_prefix = 'Exp4_stacks';
    stacks_index_to_trace = 1:133;
    folder_path_swc = folder_path;


    smoothing = [0.0001 0.001];
%     smoothing = [0.0001];
```

Figure 27. Example of setting the parameters for the function "**f09_convert_traces_to_micron_and_smooth.m**" correctly.

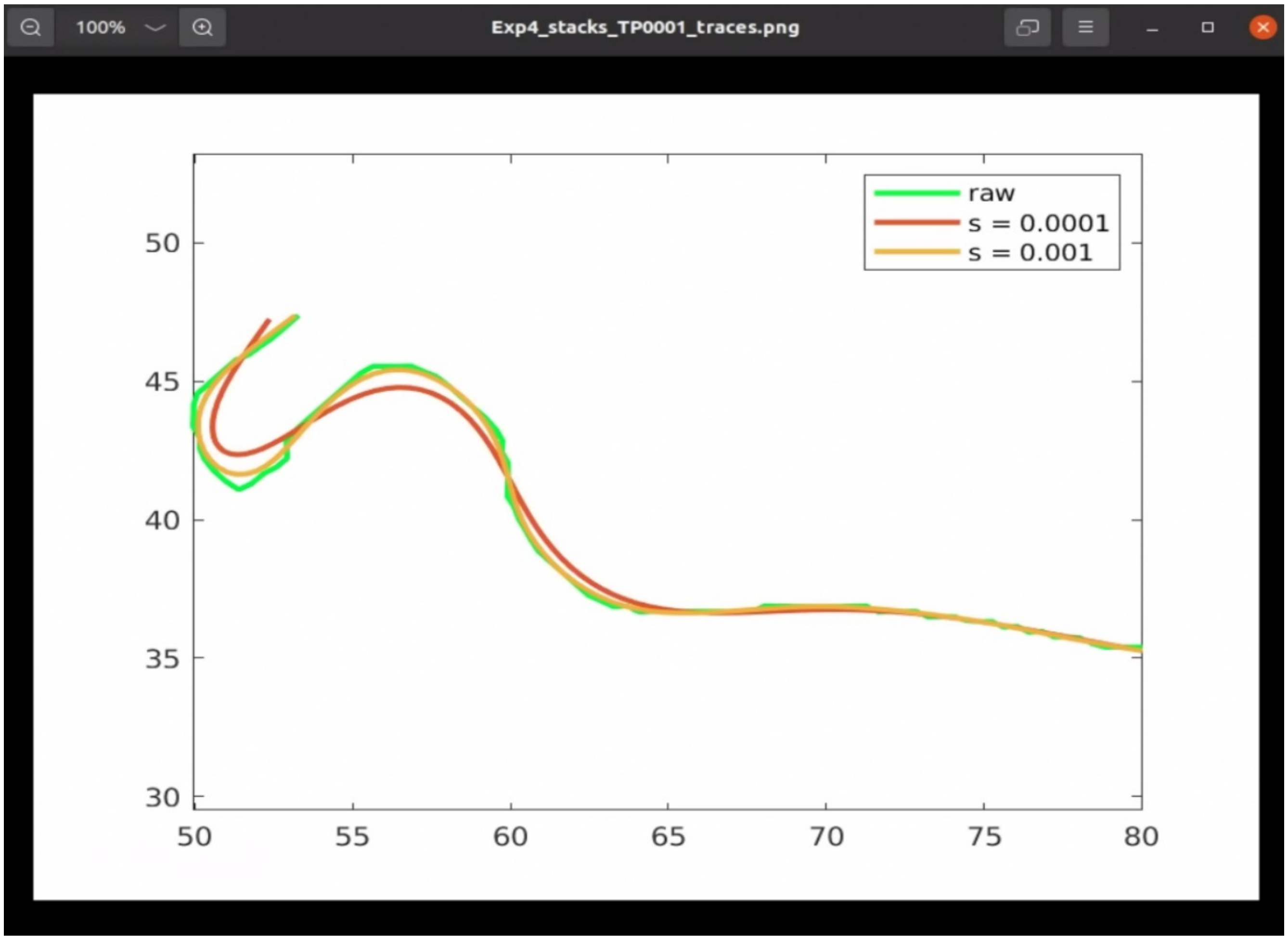
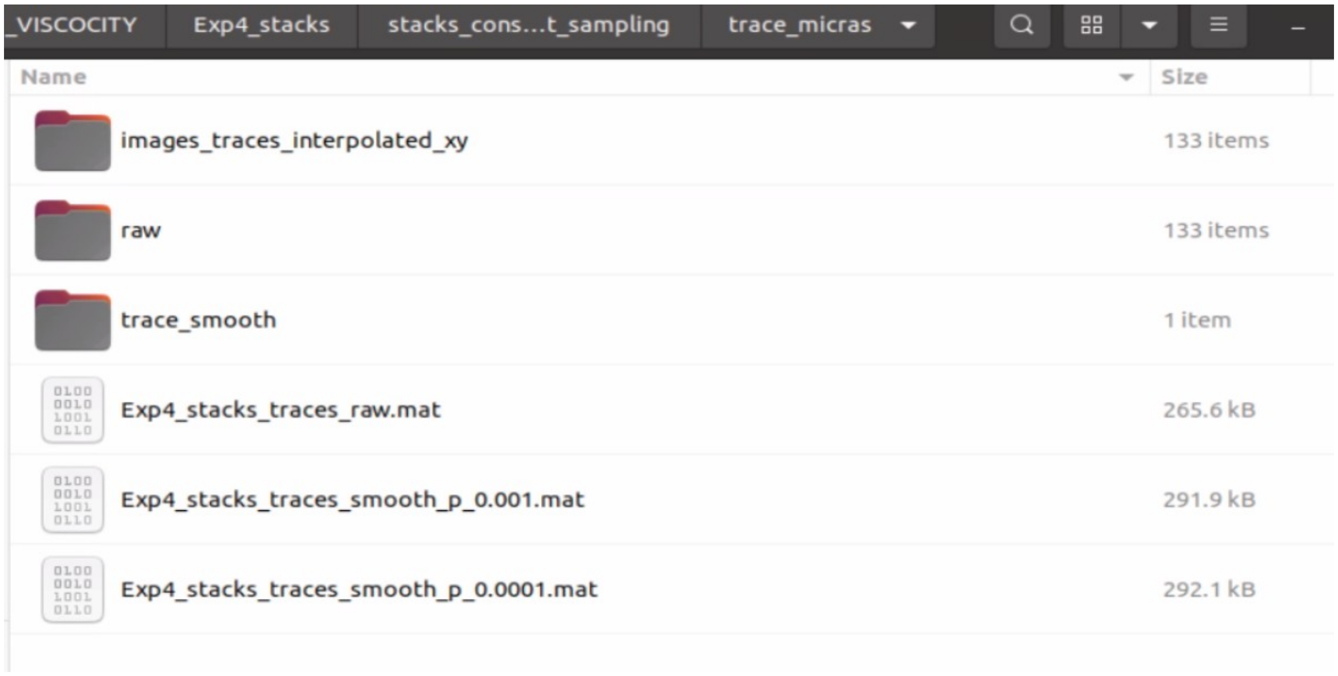


Figure 28. Example of overlay raw trace and smoothed traces.



Name	Size
images_traces_interpolated_xy	133 items
raw	133 items
trace_smooth	1 item
Exp4_stacks_traces_raw.mat	265.6 kB
Exp4_stacks_traces_smooth_p_0.001.mat	291.9 kB
Exp4_stacks_traces_smooth_p_0.0001.mat	292.1 kB

Figure 29. Example of generated files by the function "**f09_convert_traces_to_micron_and_smooth.m**".