Tutorial 2 : More Image Processing Methods and FRAP Analysis

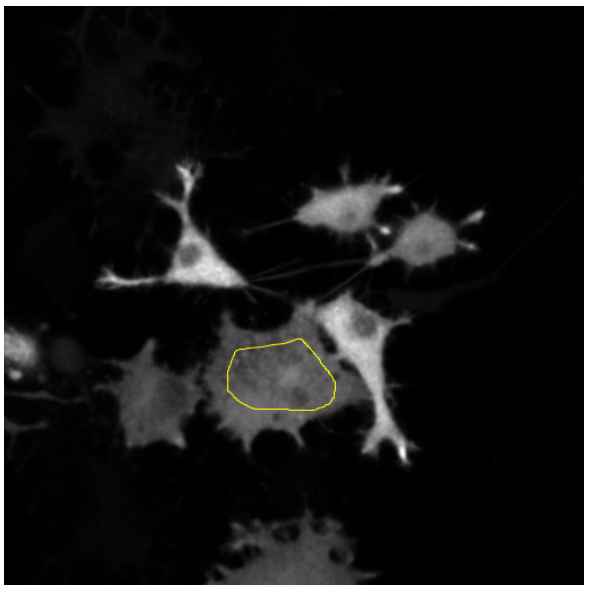
Course : Image Processing and Quantitative Data Analysis
Instructor : Marten Postma
Teaching assistants : Aaron Lin, Aoming Sun, Catherine Chia
Date: 7th June, 2023

Table of content

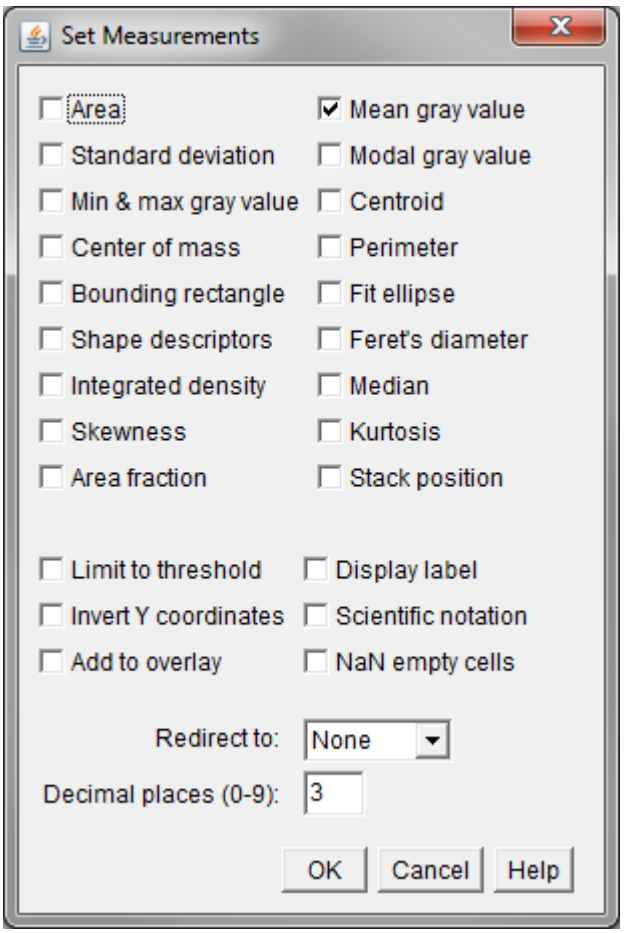
1.0 Basic Image Processing in ImageJ

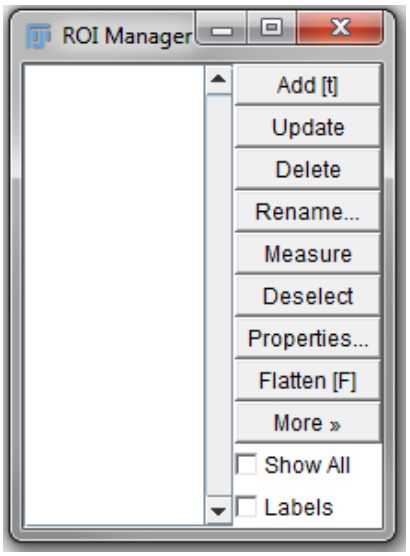
1.1 Measuring in regions of interest (ROIs)

In many cases analysis of microscopy images starts by choosing specific regions of interest from which we would like to measure/quantify certain parameters, e.g. the mean or median value. ImageJ has built-in four basic types of regions that you can draw by using the mouse. These include the rectangular, ellipse, polygon and freehand tools. Some of these tools have multiple types, which can be selected by using the right mouse button.



a. Open the image N1E.tif and draw regions of interest using these four types. Instead of using the mouse for shaping the ROI you can also use **Edit -> Selections -> Specify** to change certain properties (dimensions) of the ROI accurately. When drawing a rectangle or ellipse you can constrain the shape by holding the shift button while drawing, try this yourself.
b. Draw a region of interest and go to **Analyze -> Measure** or CTRL-M. A Results window should open and you should see the mean intensity value for all the pixels within the last drawn ROI. Change the position of the ROI and repeat measure. Try to copy the values to clipboard by using the Edit menu in the Results window (or CTRL-C) and copy these to excel (or another spreadsheet program). Now try to clear specific values in the list and clear the whole list (check **analyse** menu, https://imagej.nih.gov/ij/docs/menus/analyze.htm).
c. In the previous case only the mean value was calculated in the ROI, you can however also calculate other statistical parameters, go to **Analyze -> set measurements**. Check the boxes for area, median, standard deviation, center of mass and integrated density and repeat measurement. What is the difference between IntDen and RawIntDen (see https://imagej.nih.gov/ij/docs/menus/analyze.html#set)?





1.2 ROI manager and multi measure

Drawing of ROIs and subsequent measuring at multiple places is very useful for quickly inspecting your data to see if there are interesting features or to check. However, when you would like to perform a more systematic analysis using ROIs ImageJ has a tool called ROI manager, which allows you to draw multiple separate ROIs and do multiple measurements on these ROIs. You can also keep track, save and organize your ROIs.

a. Open the ROI manger by going to **Analyze -> Tools -> ROI Manger,** ImageJ should open a dialogue as shown on the right hand side. Open the N1E.tif image and draw a freehand ROI on a cell and add it to the ROI Manager. The ROI should appear in the list on the left hand side. The name of the ROI is a unique identifier, you can change the name by selecting the ROI in the list and use rename (e.g. cell1). You can also delete a ROI or change its position and then use update to update the new position. After drawing you can use the 't' key to quickly add a ROI to the list without using the menu in the ROI manager.

b. Now create ROIs for all the cells you can find in the image, and name them cell1, cell2, cell3, … Use the Brightness and contrast tool for finding the dim cells as well. Also draw a ROI in the background region of the image and name it bg. Select different ROIs in the list, you will see that the active ROI is visible, check the show all tick box to see them all. Did you miss or

double any cells? Go to **More -> Options** and check the Use ROI names as labels to show the ROI names as well.

c. Now you have selected many ROIs, save the ROIs using **More -> Save**, make sure you selected all ROIs in the list or none. A zip file was created containing all the ROIs, check what is in the zip file. Close and reopen the image, and drag the zip file with ROIs on the ImageJ toolbar. By using this approach you can store and reuse your ROIs, you can also drag a single .roi file on ImageJ.

d. Go to Set Measurements and check the Display label tick box and choose some parameters you would like to measure in all ROIs. Now click the measure button in the ROI manager. Copy the complete list to an excel sheet (or another spreadsheet program). In excel you can now collect your measurements and do further processing, stats and plotting.

e. The N1E.tif image contains background fluorescence, you can subtract the background after measurement, you also subtract the background from the image first, go to Process -> Math -> Subtract, use the value from the background ROI (make sure no ROIs are still active on the image, otherwise the value will be subtracted from the ROI region)

1.3 Line Profiles

Apart from regions of interest it is also possible to measure the intensity along a line segment with ImageJ. This is often useful to measure the size of objects or look at co-localization of different proteins.

a. Open the image HeLa-Actin.tif and select the straight line drawing tool by clicking the icon. Draw a line across a structure in the cell and go to **Analyze -> Plot Profile** or use CTRL-K. A new window will open that displays the intensity along the line you have drawn. If you have set the resolution of the image (CTRL-SHIFT-P) then the x-axis will be displayed with a real distance or else it will be displayed in pixels. Use the List button to show the plotted values, try to copy the results to excel.

b. Now hit the **Live** button and move the line around or change its dimensions and look how your line profiles change. For some applications it is preferable to increase the thickness of the line, you can do this by double-clicking the line icon, change the width (in pixels) and see what the effect is on your profile. For which cases would it be useful to use thicker lines?

1.4 Kymograph and z-slice

Another type of line profile that is often used is the so-called 'Kymograph', which allows you to plot line-profiles over time or another data axis. ImageJ has a standard built- in Kymograph tool, but you can also find more elaborate plugins that allow more complex analyses and interaction.

a. Open the RapaMycin_LMB1_SnailA.tif image, this dataset shows a nuclear envelope anchor and a nucleoplasmic protein that translocates to the nuclear envelope, check the data set for both channels, what happens over time?. Draw a line perpendicular to the edge of the nuclear envelope and go to **Analyze -> Multi Kymograph -> Multi Kymograph.** A new image is created, what are the lines representing in this image and is represented by the columns?

b. What are the pixel counts in this image representing? Can you save this window as a separate tiff image? By using the built- in Kymograph, the two channels are interleaved, we

would however like to have a Kymograph of each channel and combine them to a new hyperstack. Try to solve this by looking for the right commands in the Image menu (hint : make sure you close all images and reload the RapaMycin_LMB1_SnailA.tif before trying to solve this)

c. For data sets that consist of z-slices it is also possible to create a slice along a line segment. Open the confocal-series.tif image and draw a line across a structure in the image. Now you can reslice by going to **Image –> Stacks -> Reslice**, or use the '/' key. After clicking OK ImageJ will create a new image, what are the rows and columns representing in this case? Is this indeed slicing?

1.5 Z-projections

If we work with z-stacks or time-series it is often useful to simply average or add-up all the images or perform some kind of mathematical operation along the z- or t-axis. ImageJ has a built- in operation called z-project, which can just do that both on z-stacks and time-series.

a. Open the time-series image mTq2_RFP_Maturation.tif and play the movie and inspect the two channels. This 48 hr data set shows live maturation of a cyan and red fluorescent protein, which takes a while. Now change the image type to 32-bit and go to **Image -> Stacks -> Z-project** and choose Average Intensity. What kind of information can we extract immediately from this image? Why did we ask to change to 32-bit before applying this operation? Also try some other z-project options.

b. Open the image TGAT_wt_009.nd2, this data set contains a time series of three channels, first mTq2 (cyan) membrane marker, second mVenus (yellow) TGAT protein and third free mCherry (red) a cytoplasm marker. The data looks quite noisy for the individual frames, apply the averaging z-projection and inspect the noise in both the individual frames and the average.

c. It is also possible to make special z-projections where you can label z-depth or time information using a LUT. Open LifeAct-3DEmbryo dataset, this file contains a z-stack of an early cleaving embryo, which was labelled with LifeAct-mTq2 a marker for F-actin. First make z-projection of this image in the way described above, next make a colour code projection using **Image -> Hyperstacks -> Temporal-Color Code**. Try a few different LUTs, which one do you think is the best to use? Why could this way of visualization be useful for some applications?

d. Now open the time series 210501_OptoG011_04_OptoTIAM.tif, this data set shows ruffling of cell edges, make a Temporal color projection to show cell movement in one image.

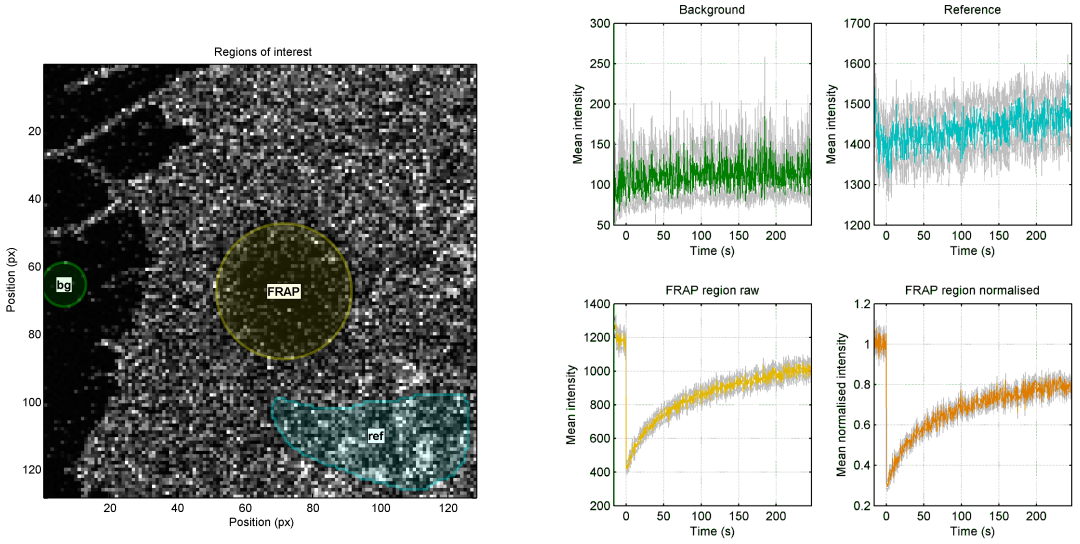
1.6 Z-profile and multi-measure

When dealing with time series it is often of interest to analyze time traces of regions of interest. ImageJ has a build in tool that allows you to calculate the mean intensity in a region of interest over time.

a. Open the mTq2_RFP_Maturation.tif and go to **Image -> Stacks -> Plot Z-axis** profile. Now click the live button and use the slider to slide through the different images. There should be a cursor in your time profile that indicates the time point in the stack.
b. The profile shows the mean pixel intensity of the whole image, however we want to determine the time profile for each cell in the image. Because the cells are appearing over time and even moving a bit it is useful to make an average z-projection first to locate the cells. Now draw a freehand ROI around one cell and add it to the ROI manager, now switch to the stack and check the z-axis profile (make sure you still have live switched on). What happens when you move the ROI around? Did you manage to capture the cell for the whole time trace? Also use the slider to see if this was the case.
c. Draw and add two more ROIs around the other cells as well. Now split the channels using **Image -> Color -> Split Channels** and extract the time traces using the multi measure tool. Save the data and open it in excel and make a plot of both channels. Because the ImageJ does not include the time axis when using multi measure, make your own axis in excel. Do you see a difference between the traces in the CFP and RFP channel?
d. Open the dataset RhoA Sensor HeLa and apply the average Z-projection twice. Now draw around each cell a ROI and add it to the ROI manager, also draw an ROI in the background and add it to the ROI manager. Now determine the time traces of the first two channels for all the ROIs and load both data sets in excel. In these cells a RhoA sensor is expressed, the ratio of the intensity of the second channel and first channel is a quantitative measure for sensor activity. Before calculating the ratio subtract the background trace from each channel. Make a plot of the response curves and also add the average response curve. Also make a plot of curves that are normalized to the first 50 seconds. When were the cells activated with Histamine and inhibited by Mepyramine?

1.7 Simple Fluorescence recovery after photobleaching (FRAP) analysis

Open the stack 120915 12_AnxA4_FRAP.tif, this stack contains a Fluorescence Recovery After Photobleaching (FRAP) measurement of AnnexinA4 fused to GFP. After some time the circular region in the middle is bleached and the recovery of the fluorescence can be monitored.



a. Set the pixel size and time step to 207 nm and 0.33 s. Hit the play button and see what happens, what do you see by eye?
b. Load the three provided ROIs and extract the mean intensity over time, copy these to excel and plot the curves, what do you notice about the different curves? When was the photobleaching applied?
c. Correct the FRAP and reference curve for background fluorescence and normalize the background corrected frap curve with the background corrected reference curve. What is the effect of these corrections? Why are these two corrections applied?
d. Normalize the corrected FRAP curve with respect to the pre-FRAP intensities. Why is recovery of fluorescence occurring?

2.0 Basic Image Processing in Python

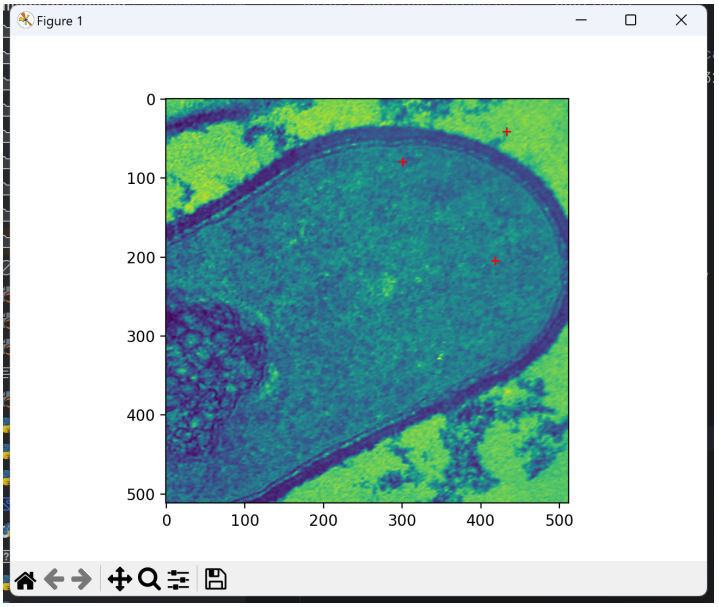
2.1 Regions of interest (ROI) with OpenCV

Code

```
#Libraries
#Use these 3 lines in this order
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt

#Other libraries
import cv2
```

To define regions of interest (ROIs) using OpenCV and the `ginput` function, you can follow these steps:

```
#2. Load the image using OpenCV:
mage = cv2.imread('image.jpg')

#3. Convert the image from BGR to RGB format (required for `ginput`)(if needed):
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

#4. Display the image using `matplotlib.pyplot`:
plt.imshow(image_rgb)
plt.show()

#5. Use `ginput` to select points on the displayed image:
points = plt.ginput(n=3)
```



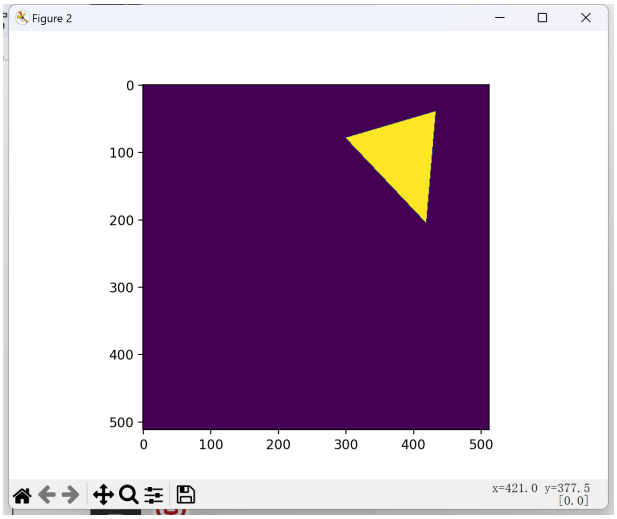
#This will allow you to select three points on the image by clicking on them. You can adjust the #value of `n` to select a different number of points if needed.

Change n to -1 to find something interesting ;)

#6. Next, convert the selected points to integer coordinates:
roi_points =points .reshape((-1, 1, 2))

#7. Draw the ROI on the image using OpenCV:
cv2.polylines(image,roi_points , isClosed=True, color=(0, 255, 0), thickness=2)

#8. Display the image with the ROI:
plt.figure()
plt.imshow(image)
plt.show()



2.2 Regions of interest (ROI) with Napari

- **Create a simple rectangular mask with predefined coordinate using skimage**

  Code

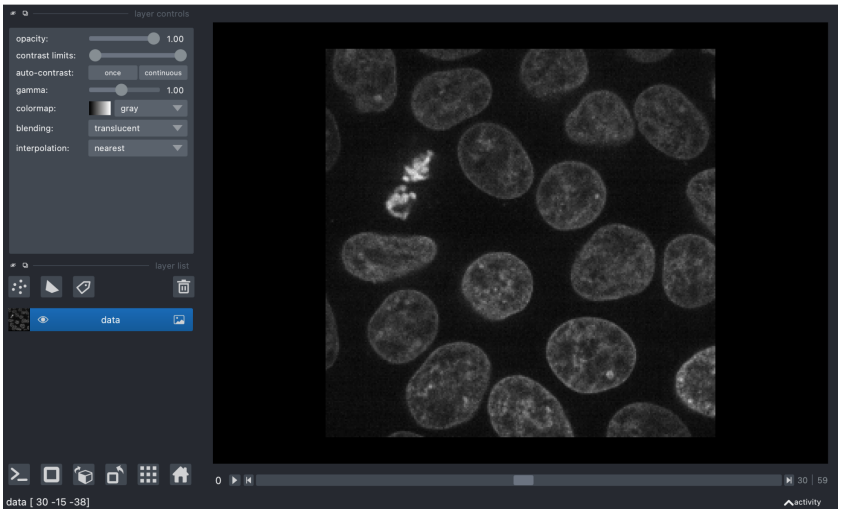
```
import skimage
from skimage import util
from skimage.data import cells3d

#Import data
data = util.img_as_float(cells3d()[:, 1, :, :])  # grab just the nuclei

# Create a basic mask
mask = np.ones(shape=data.shape[0:2], dtype="bool")

# Draw a filled rectangle on the mask image
```

```
rr, cc = skimage.draw.rectangle(start=(357, 44), end=(740, 720))
mask[rr, cc] = False

# Apply the mask
image[mask] = 0
```

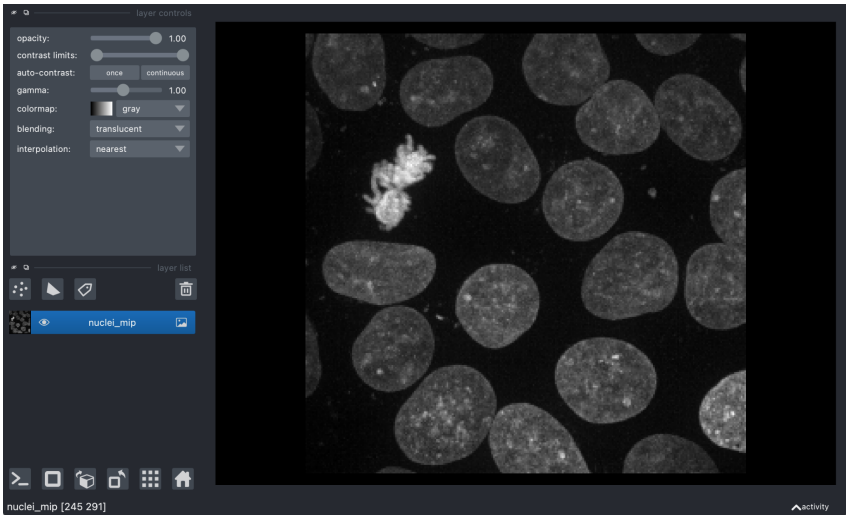
- **Extract region of interest (ROI) as a mask and export this mask as an image file using Napari**

  Code and images

```
from skimage import util
from skimage.data import cells3d
import napari

#Import data
data = util.img_as_float(cells3d()[:, 1, :, :])  # grab just the nuclei

viewer = napari.Viewer()
viewer = napari.view_image(data)
```



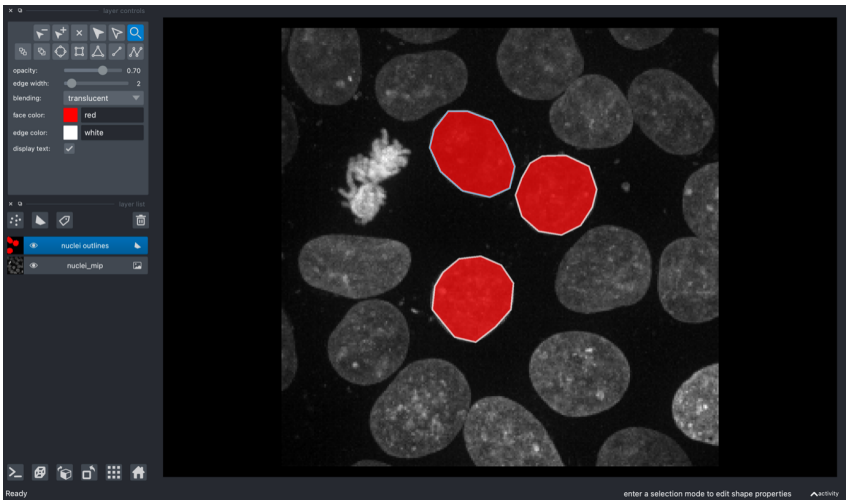

```
#For this example we'll work with a 2D maximum intensity projection of our cells in order to keep things simple.
# Take the maximum intensity projection of the cells
nuclei_mip = viewer.layers['data'].data.max(axis=0)

# Remove select and remove all the current layers from the viewer
viewer.layers.select_all()
viewer.layers.remove_selected()

# Add in the maximum intensity projection
viewer.add_image(nuclei_mip);
```



# Add new empty shape layer for the mask to the viewer before the next step
viewer.add_shapes(name='nuclei outlines', face_color='red', edge_color='white', opacity=0.7)

#(Interactive step) Notice now in the top left corner of the viewer we have a new controls panel corresponding to the shapes layer with buttons for creating and editing shapes. They include a select mode for dragging and resizing shapes, a vertex selection mode for dragging vertices, tools for adding and subtracting vertices from existing shapes, buttons for reordering shapes, and tools for drawing lines, ellipses, rectangles, paths, and polygons.

#(Interactive step) We will draw some shapes with the polygon tool around a couple of different nuclei.



The vertices for these shapes can be obtained from the shapes layer as follows:
viewer.layers['nuclei outlines'].data

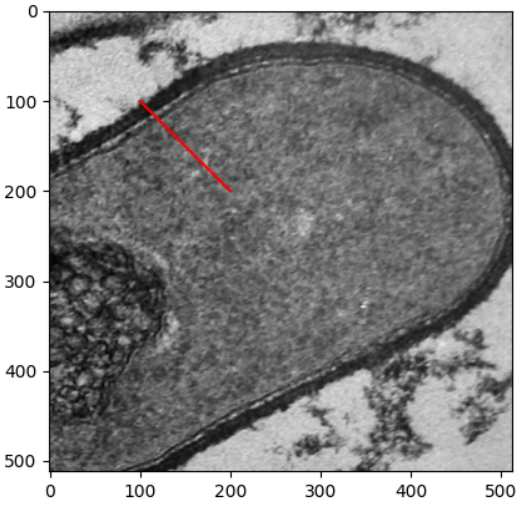
#To export the nuclei mask as an image file (e.g. svg)
viewer.layers.save('nuclei-outlines.svg', plugin='svg')

2.3 Interpolation along a line (Line scan)

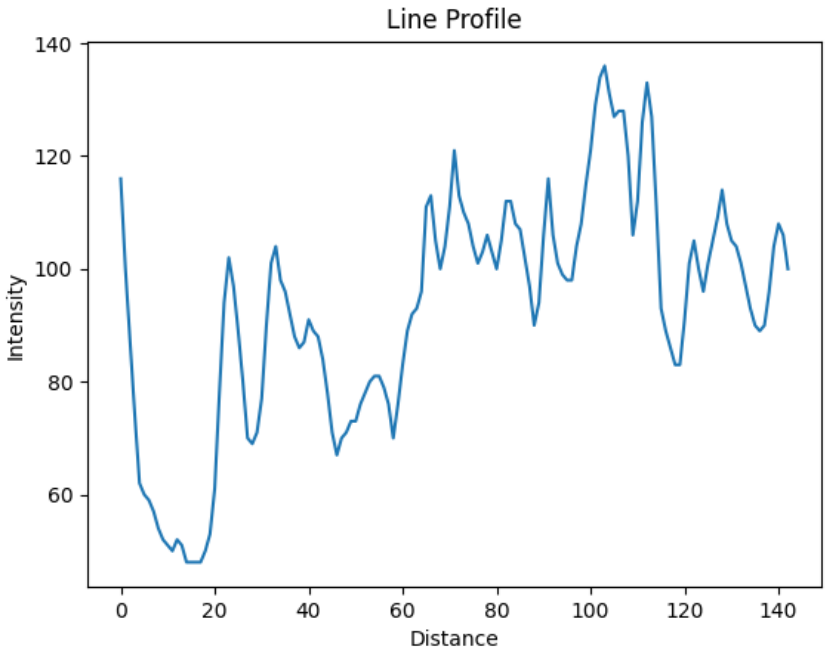
https://scikit-image.org/docs/stable/api/skimage.measure.html#skimage.measure.profile_line

Code

```
#Libraries
import matplotlib.pyplot as plt
from skimage import measure
from skimage import io

#Import image
filepath = 'Coli-EM-512.tif'

image = io.imread(filepath)

# Define the start and end coordinates of the line profile
start = (100, 100)
end = (200, 200)

# Extract the line profile
profile = measure.profile_line(image, start, end)

# Plot the image
plt.imshow(image, cmap='gray')

# Plot the line on the image
plt.plot([start[1], end[1]], [start[0], end[0]], color='red')
plt.show()
```




```
# Plot the line profile
plt.plot(profile)
plt.xlabel('Distance')
plt.ylabel('Intensity')
plt.title('Line Profile')
plt.show()
```



2.4 Interpolation in affine transformation (translation)

Affine transformation (e.g. translation, rotation, shears) is an important basic image processing method especially in the application of image registration. Shown below is an example snippet for translating one image based on another image.

- https://en.wikipedia.org/wiki/Affine_transformation#Image_transformation
- https://scikit-image.org/docs/stable/auto_examples/registration/plot_register_translation.html
- https://scikit-image.org/docs/stable/api/skimage.transform.html#skimage.transform.AffineTransform

Code

```
#Libraries
from skimage.registration import phase_cross_correlation
from skimage.transform import AffineTransform, warp

# Calculate the shift between the two images
shift, error, diffphase = phase_cross_correlation(image1 ,image2)

# Create an affine transform object with the shift
tform = AffineTransform(translation=(-shift[0], -shift[1]))

# Apply the transformation to the image
corrected_image = warp(image2, tform.inverse)

# Replace the original channel with the corrected channel
image2 = corrected_image
```

- *phase_cross_correlation* is used to calculate the shift (translation) between the two images (image1 and image2). This function returns the shift, error, and phase difference
- The *warp* function is used to apply the inverse of the affine transformation to the image2. This aligns the image2 to image1.
- The corrected image replaces the original (misaligned) image with the corrected position.

But don't stop here! There are many more affine transformation types that might be useful in your application. Read out more about them:

- https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_geometric_transformations/py_geometric_transformations.html
- (Matlab) https://nl.mathworks.com/discovery/affine-transformation.html

2.5 Z-projection

Z-projection is a method to analyze a stack image by applying a projection method (maximum, mean, sum slices, etc.) to pixels along the z-axis (e.g. thickness or time). It is important to understand the effect of each z-projection method on the final image as it may distort the analysis. This site gives a good summary about z-projection:

https://medium.com/@damiandn/an-intoduction-to-biological-image-processing-in-imagej-part-3-stacks-and-stack-projections-942aa789420f

Below is a short snippet to generate maximum and mean z-projections from an image stack.

```
#Libraries
import numpy as np
import matplotlib.pyplot as plt
from skimage import io

#Import image
path = "test.tif"
image = io.imread(path)

#Maximum projection
image_max= np.max(image, axis=0)
plt.imshow(image_max)

#Mean projection
image_mean =  np.mean(image, axis=0)
plt.imshow(image_mean)
```