Image Processing and Quantitative Image Analysis
Assignment B

Nick Bounatsos (2768686)

Universiteit van Amsterdam

1 Image data inspection

1.1 Question 1

The shape of the original image was obtained by using the shape attribute of the numpy array and was (70, 2, 260, 348). More precisely:

- timepoints: 70.
- channels: 2.
- pixel height: 260.
- pixel width: 348.

After splitting the image to donor and acceptor, the shape of the two images is the same, excluding the channel attribute.

1.2 Question 2

The FRET ration won't be reliable if the donor and acceptor channel are not aligned. This process is distance dependant and it requires both channels to be in close range, with the rule of thumb indicating a distance of 10nm or less apart. In cases like this, we should align the channels before calculating the FRET ratio.

1.3 Question 3

The Figure 1 shows the complete workflow of the process. In more detail:

1. Import data: We first import the image using the skimage package. In order to get a better idea of the data, we have to see it's shape, which indicates that there are two channels in the image.
2. Split the image: We then split the image, extracting the donor and the acceptor channel and saving them to seperate variables. Donor is assigned to channel 1 and acceptor to channel 2.
3. Calculate the z-projections: The z-projections were calculated by getting the mean intensity of each pixel across all z-planes in each channel.

4. Remove the background using napari: In this case, we used the napari package to remove the background. After calling the napari viewer, we added the z-projected images to the viewer. We also created a new shape layer in which we would save the ROI (Region Of Interest). The "add path" tool was used to draw the ROI manually and after that, we parsed the new image from the viewer to our program. The ROI drawing was saved under napari_fx.csv for future use.
5. Remove noise data using Gaussian smoothing: We used the Gaussian smoothing technique, again using the skimage package, to filter both channels from noisy data. After that, created a threshold so that pixels with low signal were assigned NaN values.
6. Calculate FRET ratios: In order to calculate the FRET ratios, we divided the acceptor to the donor frames.
7. Export as GIF: Finally, we exported the GIF, using the imageio package. A conversion to 8-bit for each frame was done before that.

2 Image processing workflow

2.1 Question 4

The logic behind the calculation of the mean of the z-projection of each image stack was to reduce the dimensionality of the image from three down to two. This way we can visualize the 3D image as 2D, using the intensities of the pixels as the third dimension.

2.2 Question 5

The threshold used to assign NaN values to the noisy data is:

$$I^{(i)}_{threshold} = \mu^{(i)} + 2\sigma^{(i)}$$

Where:

- $I^{(i)}_{threshold}$ is the intensity threshold for the ith time point.
- $\mu^{(i)}$ is the mean intensity of the background at the ith time point.
- $\sigma^{(i)}$ is the standard deviation of the background intensity at the ith time point.

The reason behind this is

2.3 Question 6

The AffineTransform function from the skimage package applies these operations in the following order [1]:

1. Scale
2. Shear
3. Rotate
4. Translate

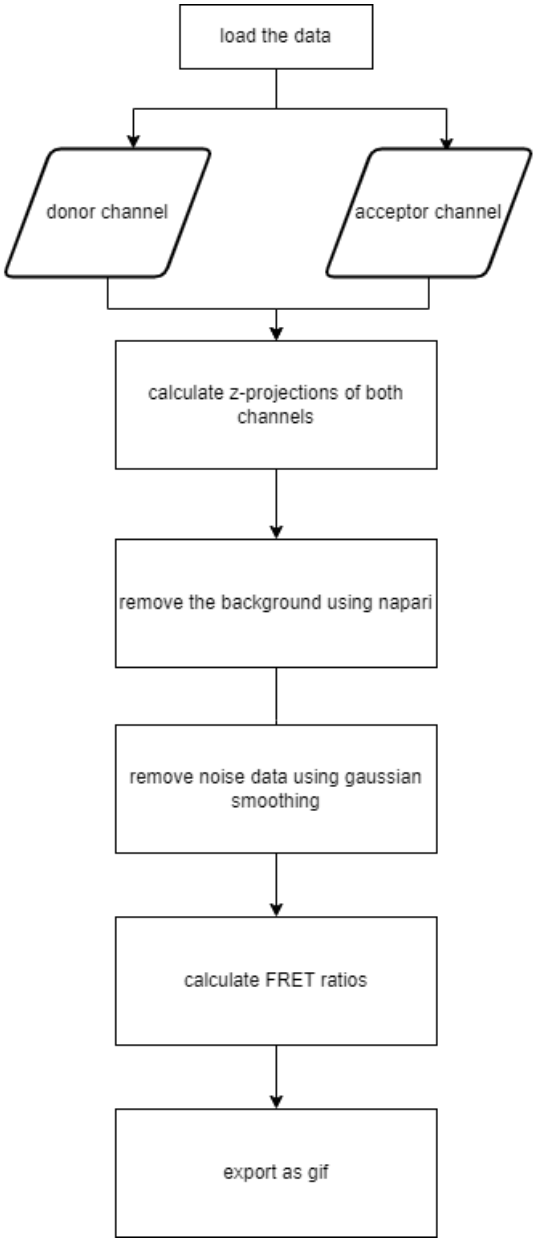

Fig. 1: The workflow of the python script used to analyse the image.

3 Data analysis and discussion

3.1 Question 7

When histamin or mepyramine is administered, there can be changes in the FRET ratios because of the interaction of the substances with the cell activity. There are also timepoint delays in the FRET changes because it takes time for the cells to react to these substances.

3.2 Bonus question

If the interest is to look at cell-to-cell differences, we would segment the image for each cell, and then we would calculate the FRET ratios for each of the segmented region.

- Follow previous workflow.
- Segment the image to isolate each cell.
- For each segmented region, calculate the FRET ratios.

References

1. skimage.transform.affinetransform. `https://scikit-image.org/docs/stable/api/skimage.transform.html#skimage.transform.AffineTransform`. Accessed: 2023-06-07.