**Instructions for using RNA spot counting programs for defining cell boundaries and detecting RNA spots within them**

LaTashaa C.R. Fraser and Sanjay Tyagia,b,c

aPublic Health Research Institute, bDepartment of Medicine, New Jersey Medical School, Rutgers University, Newark, New Jersey

cTo whom correspondence should be sent (sanjay.tyagi@rutgers.edu)

**Software**

You will need MATLAB, illastik, and ImageJ to accomplish these tasks. Download and install them on your system. They are available for both MAC and windows based operating systems and the programs that are provided here will work on both. We are providing four MATLAB scripts: counting\_spots\_two\_channel.m, counting\_spots\_one\_channel.m, select\_cells\_freehand.m, refining\_illastik\_cell\_segmentation.m and accessory MATLAB functions in a folder named as accessory functions. Place the four scripts and the accessory functions folder in your main MATLAB folder. Set path in MATLAB to the accessory functions folder.

**Image Folders**

1. Move your image files to a folder of your choice. This set may contain DIC, DAPI, and RNA spot images in various other channels. The RNA spot images would be z-stacks. The program assumes that the file names consist of four parts - as in descripter\_wavelength\_position.tif, for example, cs1-1\_w1DIC Camera\_s1.tif. In this example, the string left to the first underscore is a descriptor for your experiment, between the two underscores is the wavelength, and to the right of the second underscore but before the suffix .tif is the position of the field. The last element is the file type, which is tif. If the fine names depart from this scheme rename them.
2. Avoid having unnecessary files and subfolders within this folder.
3. Create four additional subfolders within this folder with the following names: compressed z-stacks, processed files, saved matlab figures, cell boundary files. The results of your analysis will be saved here.

**Segmenting cells in combined DIC-DAPI images using select\_cells\_freehand.m**

1. This is simpler of the two cell segmentation approaches that are presented here. This program uses DIC and DAP images, combines them in a single colored image, and then asks the user to draw lines around cells using the MATLAB freehand tool. Like refining\_illastik\_cell\_segmentation.m that is described later, this program saves cell boundaries in the cell boundary files folder which will then be accessible to the two spots counting programs.
2. Navigate to the file that you want to process using the following two commands in the MTATLAB command line: [file,path] = uigetfile; file\_location=cat(2,path,file)
3. Copy the file and the path that is returned and paste in the first line, replacing the existing path. Also, change other paths and namesin the next three lines of the program.
4. Run select\_cells\_freehand.m
5. When the DIC/DAPI image appears, and the program asks you to indicate the number of cells. Estimate the number of cells from the image and provide the answer.
6. To draw the cell boundaries, left click and hold the mouse then draw a curve around each cell in the image. You can let the left button go when you are near or at the starting point.
7. If cells are bordering each other, draw a complete circle around one cell. When defining the border of the neighboring cell, begin inside the boundary of the first cell, draw around the second cell ending inside the first cell’s boundary. It is important to clearly cross the boundary of the first cell both in the beginning and in the end.
8. After all the cell borders are drawn for the image, the data will be saved in the cell boundary files folder as .mat files.
9. If you want to process multiple files one after another, it is possible to specify them all using a wildcard character in the file name in line 1. The program will loop until all the files in the directory are processed.

**Cell segmentation using illastik in a batch mode**

For this approach we use z-stacks acquired in tetramethyl rhodamine (TMR) channel for one of the RNA target to segment the cells. When compressed into single image, these z-stacks exhibit significant level of background fluorescence in addition to RNA spots, which can be used to detect cell boundaries using high contrast. An alternative approach that yields images with better contrast for cell body is to stain them with Viafluor 405 (Biotium, Fremont, CA) before fixation. With this second method a single in-focus image will be sufficient for cell boundary determination. In contrast with the TMR based approach, which starts with a z-stack, with Viafluor 405 staining a single in-focus layer is sufficient and there no need to do z-stack compression in the next step.

**Compress TMR z-stacks into a single layer of maximum intensity projection in preparation of cell segmentation by ilastik**

1. Go to FIJI
2. Process>Batch>Macro
3. Specify the input and output directories. The input directory is where your z-stacks are located. Output directory can be the compressed z-stack folder.
4. In the macro window that appears, paste the macro code listed below:

run("Z Project...", "projection=[Max Intensity]");

//run("Brightness/Contrast...");

run("Enhance Contrast", "saturated=0.35");

1. Hit Process. The new merged files will appear in the compressed z-stacks folder.
2. This step is necessary only for those files that are z-stacks and only for those that you want to use for cell boundary determination. Don’t perform this step on files that you will use for RNA spot counting.

**Batch segmentation of cells in maximum intensity projection files created as above using ilastik**

1. Open ilastik and double click on the first option Pixel Classification
2. Create new name and save in cell boundary files folder
3. Click “+ Add New”; then “Add separate Images”; then select a few compressed z-stackimages from compressed z-stacks folder then click Open. All the selected files will open.
4. Click select feature selection on the left panel, then Select Features, and highlight only all sigma 3.5 and 5 in the list of features for color/intensity, edge, and texture
5. Click on Training.
6. If the image appear dim, change contrast of the image by first clicking on the window leveling icon (small grey square with a gradient) and then left click and hold anywhere on the image and move the mouse left, right, up, and down until the desired contrast is obtained.
7. Click on + Add Label. Use label 1 for cells and label 2 for background.
8. Change the size of the brush for the labels to 3.
9. Click and highlight label 1. While label 1 is highlighted, draw few lines on the cells, which tells the program what classifies as a cell.
10. Highlight label 2. While label 2 is highlighted, draw a few lines in the background areas, and make sure to include background regions between cells.
11. Click Suggest Features in the left panel followed by Run Feature Selection followed by Select Features Set.
12. Click live update. Wait until the calculations in the bottom right are finished. Active requests should go to 0.
13. Fill in cells and backgrounds while keeping the live segmentation clicked. Optimize segmentation by defining cells and background using labels iteratively. Unwanted drawn lines can be erased using the erase tool in the left panel. You can perform this training on all of the files that are included in the list. These can be made current via a tab on the left panel. Training lines can be drawn in these images as well to obtain good segmentation.
14. After calculations are over, click segmentation label in the group visibility window on the left. The probability and segmentations views can be visualized alternatively.
15. When satisfied with the segmentation, save segmentation by Project > save project. If there are some objects much smaller than cells in the segmentation image, ignore them. They will be removed during refine segmentation step later.
16. In the prediction export; select simple segmentation from the source drop down and choose export image settings as tif in the format section of the output file info section of the popup window. Click OK.
17. Click batch processing > select raw files. Open the cell boundary files folder and select all of the files to be processed. Click open. The files will be added to the window on the right.
18. Any unwanted files can be deleted at this stage.
19. Click process all files, segmented files will be added to the folder where the original files were present.
20. Save project and exit illastik.

**Fixing the segmented images created by illastik using refining\_illastik\_cell\_segmentation.m and creating .mat files specifying cell boundaries**

1. Set the directories for the MATLAB.
2. Change the paths for the cell boundary files folder, save\_location\_cell\_boundary\_files, save\_location\_matlab\_figures in the first three lines of refining\_illastik\_cell\_segmentation.m
3. If you wish to process multiple files, specify them using the wildcard character, for example (cs\*Simple Segmentation\*) in line 2.
4. Run the program. If all the directories were set correctly, an image of cell boundaries overlaid on the cells will appear and you will be asked how many walls you want to draw. The key task is to survey the image for neighboring cells that are aberrantly merged together and draw lines between them which will separate them. Pick a number and then slowly draw lines with mouse. If you have picked more lines then necessary, draw in empty space.
5. If no cells were found to be merged, then input 0 in answer to the question.
6. The program will come up with an image of colored cells. If satisfied with the segmentation, input y. Otherwise say n, in that case you can run the program again. If multiple files were specified in the beginning the program will process them all one by one.
7. The program will save the segmentation results (cell boundaries) as .mat files in cell boundary files folder. These files will be utilized by spot two RNA spot counting programs.

**Counting RNA Spots using counting\_spots\_one\_channel.m**

1. Set the directories for the MATLAB.
2. Navigate to the file that you want to process using the following two commands in the MTATLAB command line: [file,path] = uigetfile; file\_location=cat(2,path,file)
3. Copy the path that is produced and paste it in place of the existing path in line 1 of counting\_spots\_one\_channel.m
4. Also change the paths for save\_location\_matlabfigs, and save\_location\_csv in the next two lines.
5. As mentioned earlier, you should have a folder called saved matlab figures
6. Run counting\_spots\_one\_channel.m
7. If all the directories were set correctly, figure 1 with spots will appear. Draw a rectangle around an area with clearly separated spots. A 3D plot showing the intensities of the area you selected will be presented. You can rotate this plot in 3D. From this plot choose the lower and the upper threshold for the intensity of the single spots, while ignoring the background intensity. The spots must be above the lower threshold to be counted. The upper threshold is used for an adaptive thresholding process which helps to detect closely placed spots. The upper threshold should be a little lower than then the highest peak you want to detect. Avoid the including the transcription sites in the collections of spots that you choose in this step.
8. An image with circles overlaid with the spots will appear and you will be asked if you want to save the results.
9. If you are satisfied with the quality of spot detection, type ‘y’ to save the counts, which will be saved in a .csv file in the main directory.
10. The figures with segmented and counted spots will be saved in the saved matlab figures folder.
11. If you wish to process multiple files iteratively, specify them in the file name in the first line using wildcard characters at appropriate locations.

**Counting RNA Spots using counting\_spots\_two\_channel.m**

1. Set the directories for the MATLAB.
2. Navigate to the file that you want to process using the following two commands in the MTATLAB command line: [file,path] = uigetfile; file\_location=cat(2,path,file)
3. Copy the path that is produced and paste it in place of the existing path in line 1 of counting\_spots\_two\_channel.m specifying file\_location1.
4. Do the same for the file for the second channel specifying file\_location2 in line 2.
5. Also change the paths for save\_location\_matlab\_figures, and save\_location\_csv in the next two lines.
6. As mentioned earlier, you should have a folder called saved matlab figures
7. Run counting\_spots\_two\_channel.m
8. If all the directories were set correctly, figure 1 with spots will appear. Draw a rectangle around an area with clearly separated spots. In contrast to the counting\_spots\_one\_channel program, this program uses a single threshold for each channel. Avoid the including the transcription sites in the collections of spots that you choose in this step. Pick a threshold that is clearly above the noise but includes all the spots that you wish to detect.
9. You will similarly be asked to pick a threshold for the second channel. Pick the second threshold as well.
10. An image with circles overlaid with the spots in two colors will appear and you will be asked if you want to save the results.
11. If you are satisfied with the quality of spot detection, type ‘y’ to save the counts, which will be saved in a .csv file in the main directory.
12. The figures with segmented and counted spots will be saved in the saved matlab figures folder.
13. This program is designed to process just one file at time. Chane the name of the files in line one and two and run again if you want to process another file.