Automated Segmentation and Single Cell Analysis for Bacterial Cells

-Description of the workflow and a step-by-step protocol-

Authors:
Michael Weigert and Rolf Kümmerli

1 General Information

This document presents a new workflow for automated segmentation of bacterial cells, and the subsequent analysis of single-cell features (e.g. cell size, relative fluorescence values). The described methods require the use of three freely available open source software packages. These are:

1. ilastik: `http://ilastik.org/` [1]
2. Fiji: `https://fiji.sc/` [2]
3. R: `https://www.r-project.org/` [4]

Ilastik is an interactive, machine learning based, supervised object classification and segmentation toolkit, which we use to automatically segment bacterial cells from microscopy images. Segmentation is the process of dividing an image into objects and background, a bottleneck in many of the current approaches for single cell image analysis. The advantage of ilastik is that the segmentation process does not require priors (e.g. information on cell shape), and can thus be used to analyze any type of objects. Furthermore, ilastik also works for low-resolution images. Ilastik involves a user supervised training process, during which the software is trained to reliably recognize the objects of interest. This process creates an "Object Prediction Map" that is used to identify objects in images that are not part of the training process. The training process is computationally intensive. We thus recommend to use of a computer with a multi-core CPU (ilastik supports hyper threading) and at least 16GB of RAM-memory. Alternatively, a computing center or cloud computing service could be used to speed up the training process. The training process, although time consuming, has to be carried out only once. Afterwards, the trained classifier is applicable to all experiments with the same type of objectives.

Segmentation is then followed by high throughput extraction of cell parameters in Fiji. For that purpose, we developed specific macro scripts (see Fiji_scripts part1 - 5). We will explain the use of these scripts in detail in the "Walkthrough" section (4) below. The current version of the scripts are adapted to work with the attached example files. It might be necessary to adjust them to the user's specific needs.

In a final step, the extracted cell parameter data will be important into a R-based graphical interface called ShinyApp [5], which we specifically programmed for this workflow. This step is required because the information on cell parameters need to be connected to the descriptive variables (e.g. date, time, image_ID, treatment, channel, etc.) of the experiment, which are at this stage only encrypted in the file title. The ShinyApp is a user friendly interface and does not require knowledge of R programming. Once this step is completed, a data set containing all analyzed images can be extracted as a spread-sheet in csv-format, which can then be fed into any standard statistical software package for in-depth analysis.

2 Methods Part I: Training of ilastik and Automated Segmentation

It is beyond the scope of this document to provide detailed instructions on how to use ilastik. We refer readers to the detailed instruction manuals available on `http://ilastik.org`. The documentations contain an in-depth descriptions of all the required steps (in ilastik), we briefly outline below.

2.1 Getting Started

In this step, we feed training pictures to ilastik and choose the starting conditions for the training phase. To begin this process, we choose the "Pixel Classification + Object Classification" workflow. Next, ilastik asks for training images, which should cover the full spectrum of variation observed within an experiment (e.g. variation in cell numbers, contrast, cell size, etc.). We typically perform segmentation based on phase contrast images, but it also works with bright field or fluorescence images. In the next applet, we then select features that will be used to classify pixels (e.g. Colors, Edges, Textures, etc.). It is advisable to start with a wide range (or even all) features, which can subsequently be reduced if necessary.
All steps below should be repeated for all training images until object recognition is satisfactory *(it might be necessary to go back and forth between the images, especially if cell numbers differ)*. Now, we are ready to launch the training process.

2.2 Training Procedure

In the training process we supervise the object classification in ilastik. Based on this manual classification, ilastik will create a so-called "Object Prediction Map" that is later used for segmentation. *The training process should be performed in the "Off-mode" , since the "Live-Mode" is computationally very expensive. To do so, toggle the button "Live Update". Training is in the "Off-mode" when red circles with a black crosses appear behind the label names.*

1. Create two labels, one for the background and one for the objects.
2. Mark the background and the objects on the training images with the respective labels (Figure 1A & 2A: red = background, green = object).
3. To control the result, toggle to "Live Update". *This option will compute the "Object Prediction Map". If the object identification requires further improvement, toggle back to the "Off-mode" and repeat step 2.*
4. Only applicable for low resolution/contrast images: To better distinguish the objects from the background we can only mark the part of the cell with the highest contrast (Figure 1 B), which is usually the center of the cell. This will improve object classification but also introduce a bias in cell size. This bias can be corrected at a later stage (step 3.2; Figure 1C).

2.3 Thresholding Images Based on Object Classification

Step 2.2 creates an "Object Prediction Map" that is now used to segment the image (result of segmentation: Figure 2B). It might be necessary to switch back and forth multiple times between the training and the thresholding mode to improve segmentation.

1. Set "Input Channel" to the label you chose for the objects (e.g. Label 2 $\rightarrow$ Green $\rightarrow$ "Input Channel" = 1).
2. We set "Sigma" to small values (e.g. 0.1). Sigma uses a Gaussian to smooth the "Object Prediction Map", which is not necessarily needed. *For details, please refer to the ilastik-documentation.*
3. We choose an intermediate "Threshold" (e.g. 0.5), in order to reliably segment the different objects from one another. *The threshold value allows us to change the size of the objects, which are recognized as objects. Higher values indicate smaller objects an vice versa.*

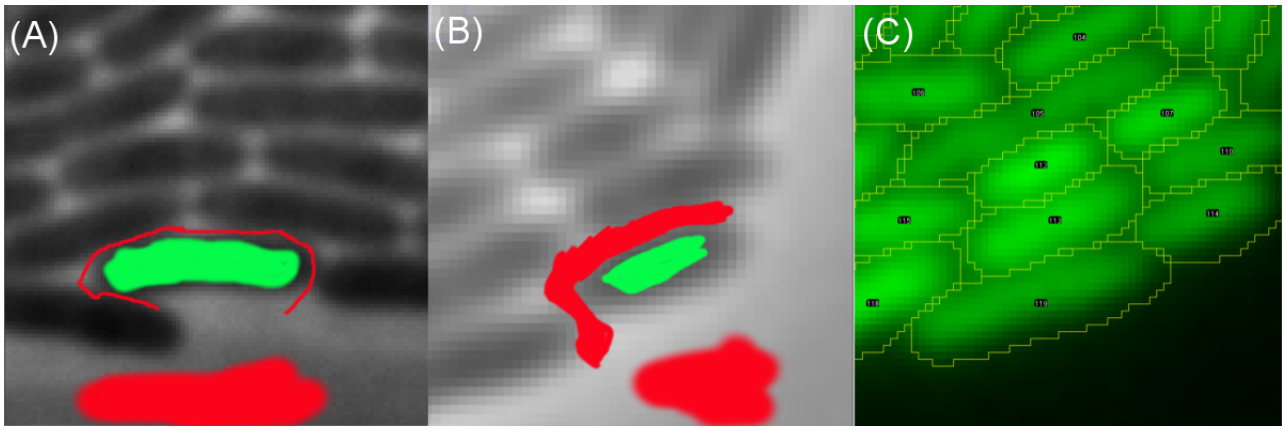

Figure 1: **(A)** High resolution/contrast images. In the training process 2.2 we can mark the whole cell with the ”Object Label” (green) since resolution is high enough to separate cells. **(B)** Low resolution/contrast. Only the center of the cell with high contrast level is marked with the ”Object Label” (green). Resolution is not high enough to reliable separate cells, therefore we need to clearly separate the cell with the ”background label” (thick red line). By only marking areas with high contrast level, we introduces a bias in cell size, which can be corrected later. **(C)** Here we show a gfp-fluorescence image, where the bias, introduced in (B), is already corrected (step 3.2). Outlines of the cells now perfectly match the boarder of the cells. The possibility to correct for size bias allows us to reliably segment images, even if the used camera resolution is poor (e.g. high sensitivity cameras)

4. We choose and appropriate ”Size Range” (e.g.10-1000000). This parameter is useful to exclude non-biological objects such as dust particles.

5. By pressing ”Apply”, the threshold-settings are applied to the image (Figure 2B). Every object should now be appear in a different color.

6. If the segmentation is satisfactorily the move to step 2.4. If an error is spotted, try to change the threshold or go back to training mode.

2.4 From Segmentation to Objects

This step (”Object Feature Selection”) calculates the features of objects (Figure 2C). *For details see ilastik documentation.*

1. Select ”All features”. *The amount of calculated features can be reduced any stage.*

2. Switch to the applet ”Object Classification”.

3. Create two labels, choose the ”Object Label” (green) and click on one cell to mark it.

4. Toggle ”Live Preview” to identify all objects.

2.5 Export Objects

These objects (from step 2.4) can now be exported (Figure 2D), as a binary black and white image.

1. Switch to the applet ”Object Information Export”

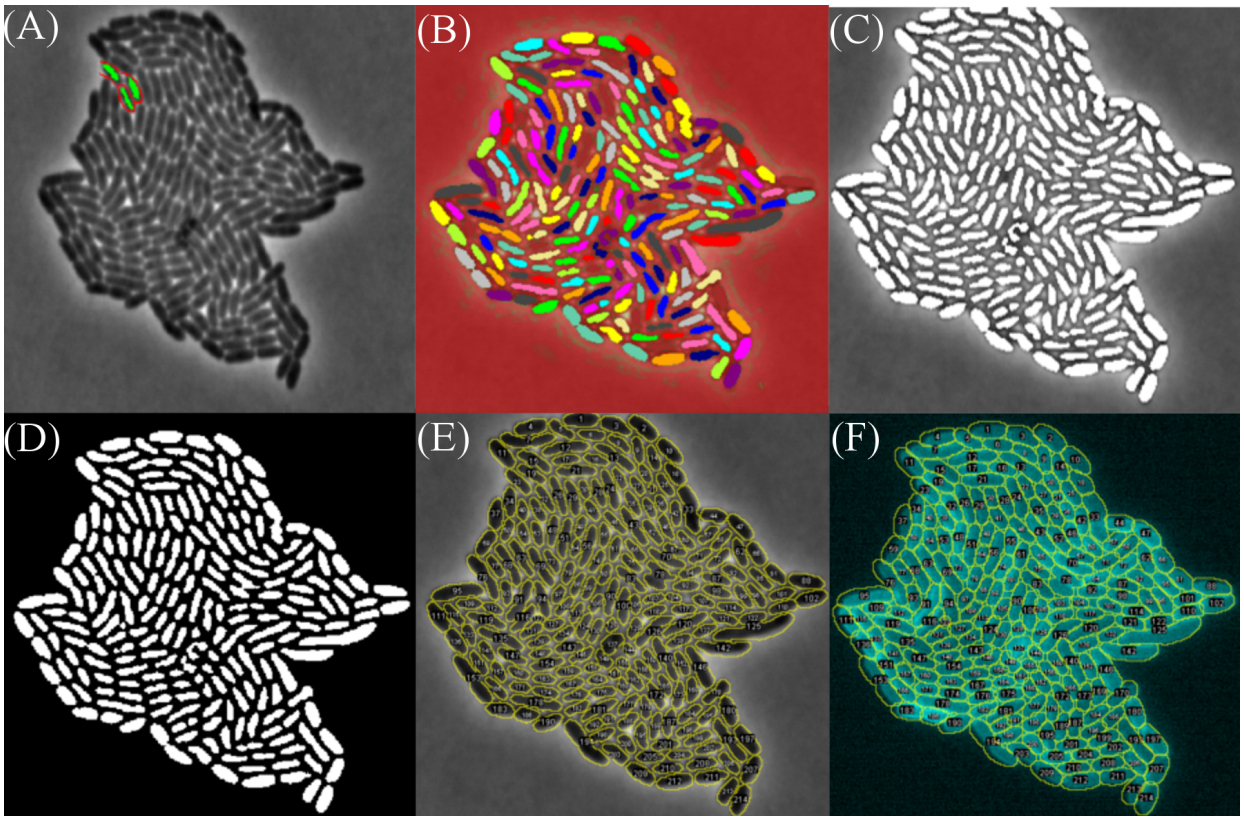

Figure 2: **(A)** Objects (green) and background (red) were marked with the corresponding label (step 2.2 in ilastik). **(B)** Visualization of segmentation, based on the training procedure (step 2.3 in ilastik). Every cell will appear with an individual color. If results are not satisfactory, either the threshold-value has to be changed or cells have to be manually separated in the training step (2.2) **(C)** Based on training and segmentation, ilastik predicts objects (step 2.4). **(D)** Predictions are exported as a black and white image (here opened in Fiji; step 2.5). **(E)** This segmentation image can be used to create regions of interest (ROIs; yellow outlines) in Fiji. **(F)** ROIs can than be used to measure cell features in different images e.g. RFU in different channels.

2. Open export configuration by clicking on ”Choose Export Images Settings...”
3. ”Change Output File Info” to ”png” and save by clicking on ”OK”.
4. Press ”Export” to export one image. *The exported Image will be saved in the folder, where the example images are located. Ilastik will extend the original filename by ”_Object Prediction”.*
5. As a quality control, we can now load a representative image into Fiji. If necessary adjust contrast levels by applying ”Auto” (Image→Adjust→Threshold).
6. Check whether segmentation was successful (image should look like figure 2 D).

2.6 Automated Segmentation

If the ”Object Prediction” is successful and satisfactory, we can use the ”Batch Input”-function to automatically apply the segmentation to all images in a folder.

3 Methods Part II: Extracting Parameters from Single Cells

Now we switch to Fiji. Fiji is an image processing package to facilitate scientific image analysis, based in ImageJ [3]. It is freely available under `https://fiji.sc/`. A step-by-step wlakthrough for the following protocol can be found in section 4 (including example images).
Each of the following sections includes a short description of the implemented process, the Fiji-commands needed to carry out the process on a single image, and information on how to automatically apply the process to all images.

3.1 Thresholding Images in Fiji and Creating Uncorrected ROIS

In this step we will use the binary images received form ilastik to create regions of interests (ROIs), which can later be use to extract cell parameters (e.g. length, RFU, etc.) from any type of image (e.g. fluorescent images). To create ROIs we first have to threshold the binary images in Fiji. *Thresholding: Choosing a cuttoff value, so that every pixel below or above the threshold will be considered as background or object, respectively.*

1. Open a binary image in Fiji (Figure 2D).
2. Threshold images: Image→Adjust→Threshold; *Setting: Upper slider 1, Lowe slider 2, Rest: Default;B&W*
3. Now we can create ROIs with "Analyze Particles" (Figure 2E). Analyze→Analyze Particles; *Settings: Clear results, Add to Manager, Exclude on Edges. With the "Size" and "Circularity" argument we can excluded objects which are no cells.*
4. The generated ROIs can now be saved by selecting More→Save in the ROI Manager. *ROIs will be saved in a .zip-container.*

Script for batch process: part1_creating threshold images

Remarks for skript usage: It can be desirable to change size and circularity of objects for batch processing. In this case, the size=0-Infinity and circularity=0.00-1.00 arguments can be changed in the following command-line: run("Analyze Particles...", "size=0-Infinity circularity=0.00-1.00 exclude clear add").

3.2 Correcting for Segmentation Biases and x,y-Position of ROIs

In this step we can correct for segmentation biases we might have introduced at the segmentation step (see step 2.2/4). Moreover, we can also correct for drift in the x, y position of the ROIs, which can occur during filter cube change. Once we have defined the correction values they can automatically be applied to all images using the following two scripts. Script:

- part2_manually_shift roi
- part3_manually_size_adjust

Remarks for script usage: The script applies the same corrections to all images. Please verify whether corrections generate reliable ROIs.

Creating Corrected ROIs (only for batch processing)

We wrote a script that allows automated adjusting of all ROIs:

PART4_CREATING ADJUSTED ROIS

3.3 Extracting Cell Parameters and RFU

Here we can extract information from the ROIs we have created above. First we have to set the parameters we want to measure (e.g. fluorescence intensity, cell size, cell shape). A full list of parameters that Fiji can measure (incl. description) can be found here: `https://imagej.nih.gov/ij/docs/menus/analyze.html`.

1. Choose the parameters you want to measure ANALYZE→SET MEASUREMENTS; *Make sure that the option for "Display Label" is checked.*
2. Open an image and the corresponding ROI. In the ROI-Manager check the option: "Show All".
3. Measure parameters with: ANALYZE→MEASURE.
4. Copy the results into an Excel-file and save as .csv (separated by comma, semicolon or tab).

We wrote a script that allows automated batch processing of all images: PART5_MEASURING

Remark for skript usage:
Image names and names of the ROIs have to correspond exactly, for the automated script to work. We have included an option in the script that allows to change the file name (of the file loaded), if names of images and ROIs do not correspond (e.g. CH1 and CH2; see section 4 for details).

3.4 Assigning Factor Levels to Data

From the generated file, we have extract information that are encoded in the factor name (see table 1 and table 2). We can do this in R-based ShinyApps (this step is simple to execute and will be briefly explained in 4).

Name	Area	RFU
24122016_TreatmentA_Factor1_..._CH1.tif	1	1000

Table 1: The exported Excel-file from Fiji lists important image information (e.g. time, treatment, factor level, channel) in a concatenated form under the label "Name".

Date	Treatment	Factor	...	Chanel	Area	RFU
24122016	TreatmentA	Factor1	...	CH1	1	1000

Table 2: The ShinyApp allows to split up the relevant information into different columns for subsequent analysis.

4 Walkthrough with Example Dataset

This walkthrough explains the application of the Fiji scripts using representative example images. It starts with the binary images received from ilastik.

Comments before we can start the procedure:

- Name of example image received from ilastik:
  3h_control_Image002_CH1_Object Predictions.png
- "_Object Predictions.png" is added to the original filename by ilastik and is essential for the script to work. It is possible to remove/change this extra label by changing change the "Object Predictions.png"-argument in script: PART1_CREATING THRESHOLD IMAGES; filename=replace(title,"Object Predictions.png","");;

Starting the Procedure

1. Start Fiji and open script part1_creating_threshold_images and run it. Follow the instructions.
2. Go into the folder EXAMPLE DATA\GFP) and load one of the gfp-images.
3. Load the corresponding ROI.zip-file.
4. Select the gfp image, and in the ROI-Manager check the option "Show all".
5. Zoom in and check alignment (size, and x/y position). To correct alignment, run script PART2_MANUALLY_SHIFT ROI and/or PART3_MANUALLY_SIZE_ADJUST. To undo the changes, "re-load" the ROIs or rerun the script with inverted signs (e.g $-\rightarrow+$) using the same values. Please remember the implemented correction values, as they will be needed in the next step. *Correction factors are sensitive to scale, it might be necessary to set the scale in your image.* (`http://imagej.net/SpatialCalibration`).
6. To create corrected ROIs, open and run script PART4_CREATING_ADJUSTED_ROIs and enter the correction values from the previous step.
7. To measure cell features (e.g. size, RFU,...), follow the instructions from step 3.3/1 first ("Set measurement").
8. To extract object parameters of interest, open and run script part5_measuring. *Please note, that ilastik did the segmentation based on phase contrast images, which corresponds to channel name "CH1". GFG-images were, however, recorded in channel "CH2". If we want to measure RFU in GFP-images we thus have to change the name of the file, since the script requires the names to correspond exactly. We have implemented an automated renaming option in the script. When running the script, it first asks for the name of the channel used for segmentation (here "CH1") and then the name of the channel to be analyzed (here "CH2"). This option will simply adjust the file names while being processed by Fiji, and it will not change the actual filename in the folder.*
9. Copy results from the output table into an excel file and save as .csv (with column names) in the folder.
   (a) Download R or use a portable version.

(b) Start R and run our script SPLIT_DATA_GUI.R. In a first step, the script First it will automatically download and install all the necessary packages[1]

(c) A new tab/window in your browser will open with the graphical ShinyApp interface.

(d) Follow the instructions on the left-hand side of the interface, from top to bottom. In a first step, upload the csv-file, then you can assign, split and rename factors. Finally, the modified table can be exported as csv-file and used for in-depth statistical analysis and plotting.

References

[1] Sommer C. et al., *ilastik: Interactive Learning and Segmentation Toolkit*, Eighth IEEE International Symposium on Biomedical Imaging (ISBI). Proceedings, (2011), 230-233

[2] Schindelin J. et al., *Fiji: an open-source platform for biological-image analysis*, Nature Methods 2012

[3] Abramoff, M.D. et al., *Image Processing with ImageJ*. Biophotonics International, 2004

[4] R Development Core Team, *A language and environment for statistical computing*, R Foundation for Statistical Computing 2015

[5] Chang W. et al., *Shiny: Web Application Framework for R*, (2017)

[1] Once the packages are installed, following two lines are not longer needed: install.packages("shiny",repos='http://cran.us.r-project.org') and install.packages("splitstackshape",repos='http://cran.us.r-project.org')