Chapter 10

Computer Vision-Based Image Analysis of Bacteria

Jonas Danielsen and Pontus Nordenfelt

Abstract

Microscopy is an essential tool for studying bacteria, but is today mostly used in a qualitative or possibly semi-quantitative manner often involving time-consuming manual analysis. It also makes it difficult to assess the importance of individual bacterial phenotypes, especially when there are only subtle differences in features such as shape, size, or signal intensity, which is typically very difficult for the human eye to discern. With computer vision-based image analysis — where computer algorithms interpret image data — it is possible to achieve an objective and reproducible quantification of images in an automated fashion. Besides being a much more efficient and consistent way to analyze images, this can also reveal important information that was previously hard to extract with traditional methods. Here, we present basic concepts of automated image processing, segmentation and analysis that can be relatively easy implemented for use with bacterial research.

Key words Image segmentation, Object recognition, Region properties, MATLAB, ImageJ

1 Introduction

In bacterial research, images of bacteria are still mostly analyzed and interpreted through methods reliant on continuous manual input. The main disadvantages of this are related to our human nature, with the risk of subjectivity and reproducibility — especially after long sessions—as well as the fact that it takes up a significant amount of time. These issues are especially apparent when there are hundreds, if not thousands, of images to analyze [1]. The recent revolution in fluorescent microscopy with computer-controlled microscopes that can acquire images at high speed in multiple dimensions, including channels, time, and space only adds to the complexity of image analysis — in many cases making quantitative manual analysis an impossible task. An alternative way of identifying bacteria and analyzing them is to use computer algorithms instead; this subfield of artificial intelligence is called computer vision. Computer vision has the advantage of being automated, objective, and completely reproducible; this approach also has the added benefit of "seeing" data that is typically not detectable by a human operator [1]. While this method is

Pontus Nordenfelt and Mattias Collin (eds.), *Bacterial Pathogenesis: Methods and Protocols*, Methods in Molecular Biology, vol. 1535, DOI 10.1007/978-1-4939-6673-8_10,

currently commonly used in other fields of application, such as facial recognition [2], optical character recognition for scanning documents [3], medical imaging [4], and automatic cell counters [5–7], there is a distinct lack of general approaches within bacterial pathogenesis research. Examples found [8–11] pertaining computer vision are generally restricted to one specific study or a group of studies and in overall computer vision seems currently to be an under-developed field in bacterial pathogenesis research. Besides the obvious benefits for accurate quantification and detailed analysis, one of the most interesting applications of automatic image analysis is the possibility to identify a single bacterium and track its movements reliably over time in information dense movies. Here, we present the basic concepts of computer vision with a focus on explaining principal methods and their uses, in an effort to give a starting point for interested scientists.

2 Materials

ImageJ (NIH) and MATLAB (Mathworks) are two of the most commonly used software packages for computer vision analysis. ImageJ is open source and much of its functionality comes from user-designed plugins. MATLAB has several toolboxes that expand the functionality and include many optimized algorithms for computer vision. We have listed some useful ImageJ plugins and MATLAB toolboxes.

2.1 ImageJ (See Note 1)

1. Otsu Thresholding.
2. Watershed Transformation.
3. Multi Thresholder.
4. Cell Counter.
5. Hough Circles.
6. Bio-Formats.

2.2 MATLAB (See Note 2)

1. Image Processing Toolbox.
2. Computer Vision System Toolbox.
3. Fuzzy Logic Toolbox.
4. Signal Processing Toolbox.
5. Bio-Formats Toolbox (*see* **Note 3**).

3 Methods

3.1 Basic Program Design

A flowchart for a basic bacterial image analysis software (*see* **Note 4**) is shown in Fig. 1. The software in question is broken up into major sections, which in turn consists of smaller and more specific modules. The initial step, input, is composed of simply loading the raw

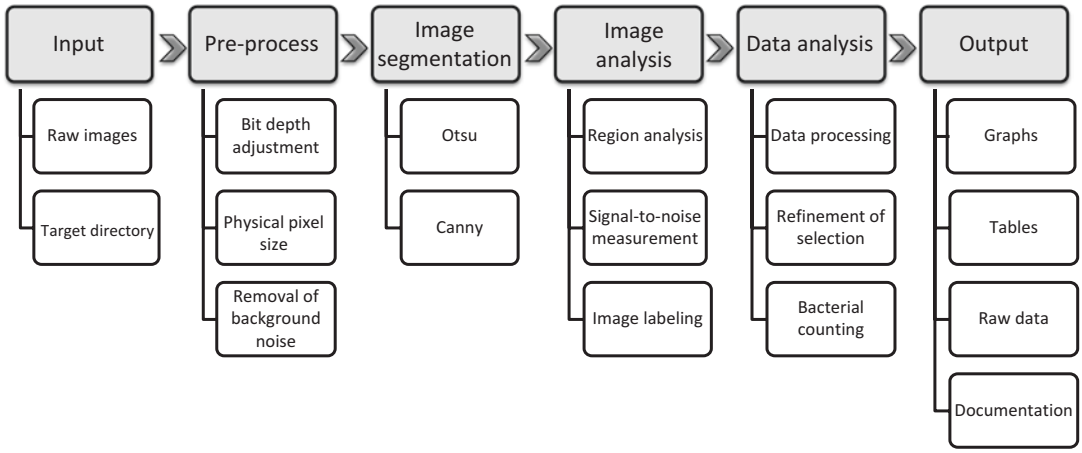

Fig. 1 Flowchart for a basic bacterial image analysis software

image into the software, and defining the directory the results will be deposited in. The preprocessing step is used for preparing the image for segmentation by cleaning up noise, reducing background and extracting important information about the raw image (often designated metadata) that can be used at later stages. It is also here that any global adjustments to the data is done (*see* **Notes 5** and **6**). The third step, image segmentation, is where the preprocessed image is turned into a binary mask. In relation to this chapter, a mask is considered a binary image consisting of only 1s and 0s, and the 1s correspond to regions defined as objects and the 0s as regions defined as background. Image segmentation is also the step that often needs to be optimized the most to achieve good results from the image analysis. The generated mask works by functioning as a filter in the image analysis step and the analysis is performed on the same preprocessed image as the mask was generated from. This means that the software can divide analyzed areas of the preprocessed image that corresponds to the regions that are composed of 0s and 1s in the mask. Different analysis modules can be present in the same software as can be seen in Fig. 1 and each will generate its own data output. The generated data in the previous step is then analyzed, structured, and refined in the data analysis step. Examples of analyses are counting the numbers of bacteria found, calculating bacterial area based on μm/pixel, and calculating the eccentricity of bacteria. Refinement of selection is essential to ensure that only relevant data is analyzed, that different groups of bacteria are separated, and outliers can be identified. The goal of the data analysis step is to present the relevant data in an understandable and useable form. The final step, output, involves converting the refined data into tables and graphs to present the results of the analysis. Another important function of the output is to generate all the data from each step and store it in a structured system. This allows the data and the whole process to be reexamined without having to rerun

the entire process. Most importantly, together with the program code, this provides documentation on exactly what was done, on what it was done, when it was done, and how it was done.

3.2 Principal Methods

Segmentation is a common step in image analysis and consists of separating objects from the background in an image. This is based on identifying features, such as pixel intensity, and using that to divide the pixels into non-overlapping regions [12]. For the purpose of this chapter the aim of the segmentation is to acquire a binary image that is typically called a mask. The mask is then used for defining the regions the software should analyze, and this step is therefore critical for performing image analysis with any reliability. The four major groups of grayscale segmentation methods that will be examined here are the following; thresholding, edge detection, active contour, and watershed transformation.

Threshold-based segmentation functions by using a threshold value to separate objects from the background. Bi-level thresholding results in pixels being divided into two classes, objects and background, and requires the use of only one threshold value. Using multiple thresholds allows multiple classes or groups of pixels to be identified, and this method goes under the name of multilevel thresholding [12].

Edge detection defines edges based on discontinuities in intensity between pixels [13]. This can be done due to the general assumption that it will correspond to discontinuities in depth, properties, illumination, or surface orientation [14]. One common problem with edge detection is fragmentation, which is where the edges are not completely connected to each other, but instead consists of multiple fragments. A second issue is the presence of false edges which can be considered the equivalent of false positives concerning detection [14].

Active contour segmentation uses a malleable spline, or snake, to adhere to object contours. Snakes utilize algorithms that minimizes the energy present when the snake has localized and molded itself to a feature [15]. The strengths of active contouring are: tracking objects in motion, it is autonomous and adaptive, sensitivity is scalable through Gaussian smoothing, and that the snake's behavior is intuitive. A disadvantage of this method is that there is a risk that matching the major feature will result in a lower energy compared to matching minor details, and therefore it may miss those. Further issues are that the accuracy of the method is directly proportional to the computational time and there exists a tendency to remain stuck in local minima states [15].

Watershed transformation is easiest illustrated by imagining a topographical relief, with pixel intensity seen as depth. This means that the higher the pixel's intensity is, the deeper it is located. The first step is the placement of markers in objects and the background. The second and final step is to "flood" the image, using the markers as the sources of "water." A "dam" is created every time distinct catchment basins' water would come into contact

with each other. This results in a segmented image where each marker should be in its own separate region. The major issue with this method is the high risk of over-segmentation (*see* **Note 7**) [16]. Watershed transformation is useful as a secondary segmentation method, where it can be used to separate merged regions from each other and in that way achieve a better mask.

Otsu's method is a threshold segmentation variant. The algorithm calculates the threshold by minimizing the intra-class variance and maximizing the inter-class variance between the pixel classes. This generates a so called "optimal threshold," which both removes the need to manually enter the threshold value but more importantly maximizing the separability of bacteria and background. However, this method requires the pixel intensity distribution to be following a bimodal histogram and therefore cannot reliably be used on multi-class images [17]. Fortunately, Otsu's method is easily adapted to be able to calculate multiple thresholds which allow it to segment these multi-class images. This modified method generally goes under the name of Multi Otsu [18]. However, it is rather inefficient and time-consuming [19] (*see* **Note 8**).

Canny edge detector is another commonly used segmentation method (*see* **Note 9**), which as an edge detector functions by identifying edges based on difference in intensity between pixels. This is done through five different steps of computation. First a Gaussian filter is applied to negate background noise followed by identifying the gradients of intensity, suppressing non-maxima, finding potential edges by identifying the gradients of intensity, and finally by suppressing weak edges through hysteresis [13]. One of the advantages with Canny is how sensitive the method is at detecting edges, but it is less robust than Otsu and has a tendency to over-segment, especially when exposed to Gaussian background noise.

Otsu's optimal threshold can replace Canny's own calculated high threshold value (*see* **Note 10**) and this improves the segmentation results from the Canny edge detector [20].

3.2.1 Refinement of Bacterial Selection

One of the most essential aspects of image analysis is to properly screen and select, based on the research aim, the data that should be analyzed. Enhancing data selection is important due to the fact that the software will be non-discriminatory without explicit directions, for example it would treat artifact signals as equally important as bacterial signals. This enhancement of data can be based on multiple parameters, some which are detailed in Table 1 along with in what scenarios certain parameters would be preferred. This has the additional advantage of allowing the analysis of different signals from a single image. For instance, separating two different species of bacteria can be achieved by comparing their eccentricity if there is a significant difference in shape between the bacterial species. In comparison, an area based separation would not perform nearly as well if the species had a similar size, despite any differences in shape. The opposite is seen in removal

Table 1
Bacterial selection measurements easily attainable with MATLAB and their potential applications for refinement

Measurement	Good for
Signal-to-noise	Eliminating weakly fluorescent bacteria and artifacts. Evaluating image and segmentation quality (*see* **Note 13**)
Area	Eliminating artifacts that significantly differ in size Calculating number of bacteria based on average size, when working with clustering bacteria
Eccentricity	Separating different bacterial species
Mean intensity	Eliminating weakly fluorescent bacteria and artifacts
Centroid	Labeling and identifying bacteria
Weighted Centroid	Combine with Centroid to generate a vector for spatial correlation of molecular activity (*see* **Note 14**)
STD/SEM pixel intensity	Separating homogenously fluorescent bacteria from heterogeneously fluorescent bacteria
Perimeter	Component for calculating the length of the medial axis of rod shaped bacteria (*see* **Note 15**)
Color	Separating bacteria based on different fluorescence markers

of artifacts—a task area-based separation excels at—while eccentricity separation will be lacking (*see* **Notes 11** and **12**). This is due to artifacts having a random shape, and there is a smaller potential span of measurement compared to area. Another way of isolating bacteria behaving differently would be to utilize the standard deviation of pixel intensity, and separate the population into groups depending on how homogenous the fluorescence appears.

3.3 Results: Segmentation Example

Figure 2 illustrates the process of a bacterial analysis using a software whose structure is based on the flowchart in Fig. 1. From the initial raw dual channel image of GFP (channel 1) and mCherry (channel 2) marked fluorescent bacteria we analyzed the 20% brightest and dimmest bacteria in channel 1. The original microscopy image with both channels is shown in Fig. 2a. Figure 2b, c also contains the segmented masks of channel 1 by Otsu segmentation and Canny segmentation using Otsu threshold value respectively. The difference in segmentation between Canny and Otsu is illustrated through visually comparing their masks, and by automatically counting the amount of bacteria each method found. In addition, the numbers of segmented bacteria are compared to the manually counted bacteria to test the validity of the segmentation methods and the results of this test are shown in the table *All Objects* in Fig. 2d. The results are presented as the percent of all

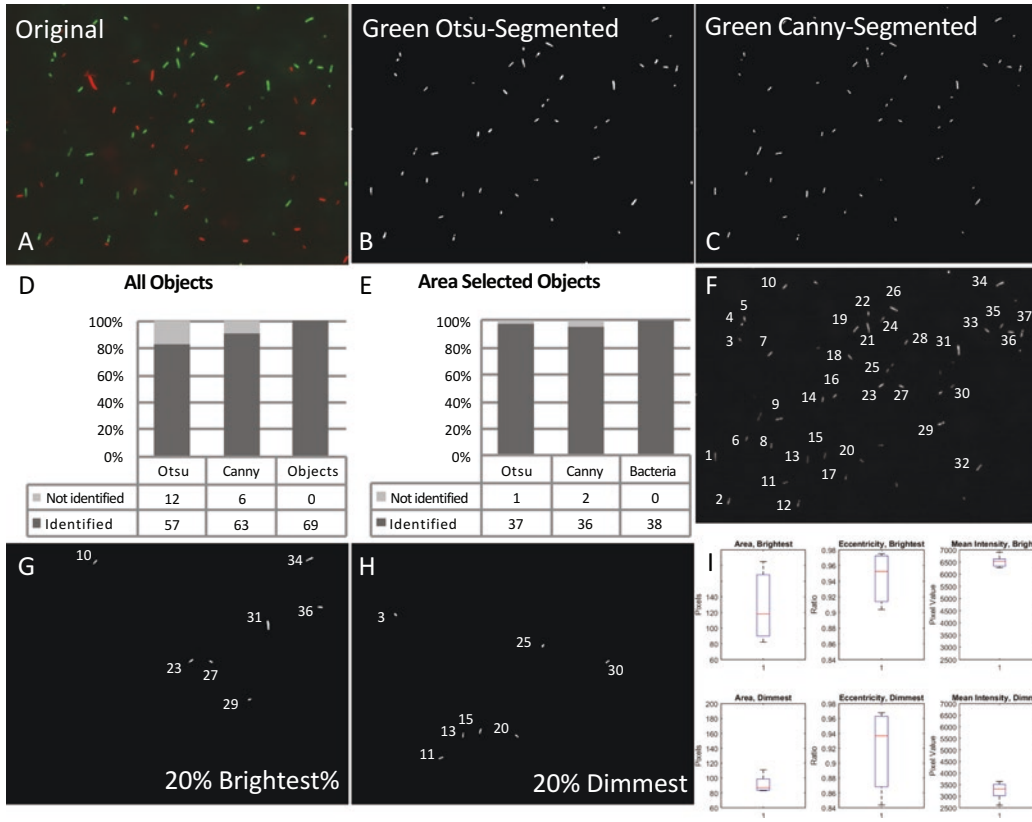

Fig. 2 Application of automated image analysis. (**a**) Raw dual channel microscopy image. (**b**) Mask generated from channel 1/green channel in (**a**) using Otsu segmentation. (**c**) Mask generated from channel 1/green channel in (**a**) using Canny segmentation. (**d**) Validity testing of Otsu and Canny segmentation by comparing the ratio of identified objects, in (**b**) and (**c**) respectively, with the total amount of objects present in channel 1/green channel from (**a**). (**e**) Validity testing of the area based selection refinement method for identifying bacteria of interest applied on the Otsu and Canny segmented masks compared with the manually selected bacteria in channel 1/green channel from (**a**). (**f**) Channel 1/green channel raw image where each bacterium identified using an area based selection on (**b**) has been individually labeled. (**g**) Mask containing the 20 % brightest bacteria isolated from the area based bacterial selection in (**f**). (**h**) Mask containing the 20 % dimmest bacteria isolated from the area based bacterial selection in (**f**). (**i**) Comparison between the 20 % brightest (**g**) and 20 % dimmest bacteria (**h**) in area, eccentricity, and mean intensity

the manually counted objects found by each method. As can be seen Otsu identifies 57 out of 69 (83 %) objects while Canny manages to identify 63 out of 69 (91 %) objects present in the image (*see* **Note 16**). This does not necessarily mean that every object we found actually corresponds to bacteria, nor does it mean that this is the type of bacteria we are interested in studying (*see* **Note 17**). The next step is the refinement of bacterial selection and in this case an area based selection (*see* **Notes 18** and **19**) was applied to enhance the identification of bacteria generated from the both the Otsu and the Canny segmentation. As before, the amount of

bacteria the method found was counted and compared to the manually counted bacteria of interest, and this is shown in the table *Area Selected Objects* of Fig. 2e. In this scenario it allows us to avoid analyzing outliers, artifacts, vertically positioned bacteria, and the initial formation of new dividing bacteria**.** The image (Fig. 2f) which numbers selected bacteria in the original image is the result of labeling the bacteria identified by the area based selection applied on the Otsu segmentation from Fig. 2b. Further data selection can be used to specify targets even more and in this scenario we compare the most fluorescent and the least fluorescent bacteria based on mean intensity (*see* **Note 20**). Figure 2g, h shows the resulting selection of the 20% brightest and the 20% dimmest from the area selected Otsu mask. Finally, the regions we identified have been analyzed and compared and the resulting graph is shown in Fig. 2i (*see* **Note 21**). As can be seen in Fig. 2i the areas are quite different between the brightest and the dimmest bacteria, while the eccentricity does not differ nearly as much. Such differences would be very hard to find by manual analysis, and can provide a source for both generating new hypothesis as well as testing existing ones.

4 Notes

1. Fiji (Fiji is just ImageJ), is an alternative version of ImageJ that comes bundled with many useful plugins for image processing and computer vision-based methods.
2. Many universities and research institutions (especially those that have Engineering or Science faculties) have group licenses for MATLAB.
3. Bio-Formats is a standardized and open format for reading and writing microscopy image data and metadata. It is developed by Open Microscopy Environment (OME).
4. A good software is built to fulfill one clearly defined goal or function. It should also be separated into manageable modules or minor software, to facility debugging, adding minor functions, modifications, and understanding.
5. One typical problem that can occur is that the bit depth of the image is misclassified by the computer. This is when the bit depth is considered to be higher or lower than it actually is. The result of this is that all intensity values will be too low or too high. One example of this happening is taking images with a 12 bit camera and opening them in MATLAB where they will be considered 16 bit images. This makes the image either extremely dark or bright, and segmentation is unlikely to succeed without adjusting the bit depth. In MATLAB this can be done by mul-

tiplying every pixel value with **VALUE** = $2^{(\text{wrong bit depth-real bit depth})}$, or as in the example, $2^{(16-12)} = 2^4$.

```
image=floor (image.*/VALUE);
```

A second possible method is by performing a bit shift on the image with difference in bit depth used as value. In this example, 12 – 16 = –4 = VALUE.

```
image=bitshift (image,VALUE);
```

6. Occasionally the image metadata does not contain the μm/pixel information, or the data could be wrong. This can be remedied when using the Bio-Formats toolbox in MATLAB by assigning the correct values.

```
pixelSize=ome.units.quantity.Length(java.
lang.Double(.VALUE), ome.units.UNITS.
MICROM);
YOURMETADATA.setPixelsPhysicalSizeX(pixel
Size, 0); (Exchange X for Y for setting Y
value)
```

7. Over-segmentation is a common problem when using watershed transformation, but it can sometimes be solved by defining a height threshold in the segmentation code. Done in MATLAB by adding the "imhim" function directly before the watershed function:

```
Image2=imhim (Image1, VALUE);
L=watershed (Image2);
```

A value of 20 is considered to be the threshold for suppressing shallow minima. Higher values will result in suppression of less shallow minima.

8. Two more efficient multilevel Otsu thresholding methods are: NM-PSO-Otsu [21], and Two-Stage Multithreshold Otsu [22].
9. Edge detection/Canny cannot be used on its own to segment an image since it only results in lines, and not a complete mask. It requires the assistance of morphological closing, filling, and opening operations to arrive at a complete mask. The structuring element chosen for these morphological operations should be as similar in shape as possible to the bacteria you are analyzing. Adjusting the structuring elements and their size through trial and error is often required to get good results.
10. The lower threshold should be set as the Otsu threshold value divided by 2. This is possible since the low threshold will always be half that of the high threshold in the original automatic Canny calculation [20].
11. Objects 1/10 of the mean size can generally be considered artifacts and can therefore in many cases be discarded [7].
12. Eccentricity is measured from 0 to 1, with 0 being a circle and 1 corresponding to a line.

13. Signal-to-noise is calculated by first removing the initial mask generated from the original image. Then the mean of the resulting image — which ideally should only consist of background pixels — is calculated and converted to double precision. The standard deviation of the background is calculated as well. Lastly the background mean is subtracted from every region's mean intensity value before being divided by the standard deviation. The following code has been adapted for use in MATLAB.

```
backgroundValues=ORIGINAL(~MASK);
backgroundMean=mean (backgroundValues(:));
backgroundValues=double (backgroundValues);
backgroundSTD=std (backgroundValues);
Signal-to-noise=(Region -backgroundValues)/
backgroundSTD;
```

14. Weighted centroid calculates the center of the region based on location and intensity values. In comparison the standard centroid calculation results in a center of mass. These two disparate markers can be used to calculate a vector between them in the bacteria. This vector can then be used for spatial correlation of molecular activity.
15. The medial axis length can be calculated with the following formula (Eq. 1):

$$\text{Length} = \frac{\text{Perimeter} + \sqrt{\left(\text{Perimeter}^2 - 16 \times \text{Area}\right)}}{4} \qquad (1)$$

16. As can be seen here, different segmentation methods will not generate the exact same results, due to approaching the information in the image differently. This makes choosing your segmentation method based on your images extremely important, and this will often require a trial and error approach. One advantage of using self-written software for this is the ability to run any sort of different segmentation methods in parallel and then comparing the results.
17. All objects are not bacteria, or the type of bacteria that we are interested in studying. This is a typical problem when you have more than one bacterial strain in your culture. A second scenario would be when you have an uneven fluorescence marker production and you only want to study bacteria over a certain fluorescence threshold.
18. In this case the MATLAB function bwareaopen was used for the area based selection. For this function you simply specify the pixel area and everything below that threshold will be removed.

19. Generally, you cannot use the same selection values on different segmentation methods and expect equal results, due to the difference in the mask generated. In this case different pixel areas were used for selection depending on the segmentation method.
20. The MATLAB function `bwpropfilt` was used on the "MeanIntensity" value parameter of `regionprops` to select the 20 % brightest and dimmest regions. This works by specifying which value it should select from, the number of objects, and if it should select the largest or smallest objects.
21. The MATLAB functions `regionprops` and `boxplot` was used for analyzing the regions and generating the graphs respectively.

Acknowledgments

This work was supported by the Crafoord Foundation, the Swedish Research Council, the Swedish Society of Medicine, and the O.E. and Edla Johansson Foundation.

References

1. Danuser G (2011) Computer vision in cell biology. Cell 147(5):973–978. doi:10.1016/j.cell.2011.11.001
2. Sinha P, Balas B, Otrovsky Y, Russel R (2006) Face recognition by humans: nineteen results all computer vision researchers should know about. Proc IEEE 94(11):1948–1962. doi:10.1109/JPROC.2006.884093
3. Smith R (2009) Hybrid page layout analysis via tab-stop detection. Paper presented at the Proceedings of the 10th international conference on document analysis and recognition, Barcelona
4. Computer Vision in Medical Imaging (2014), vol 2. Computer Vision
5. Lojk J, Sajn L, Cibej U, Pavlin M (2014) Automatic cell counter for cell viability estimation. Paper presented at the 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija
6. Farnoush A (1977) The application of an image analyzing computer (Quantimet 720) for quantitation of biological structures—the automatic counting of mast cells. Microsc Acta 80(1):43–47
7. Selinummi J, Seppälä J, Yli-Harja O, Puhakka JA (2005) Software for quantification of labeled bacteria from digital microscope images by automated image analysis. BioTechniques 39(6):859–863. doi:10.2144/000112018
8. Forero MG, Crisóbal G, Sroubek F (2004) Identification of tuberculosis bacteria based on shape and color. Real-Time Imaging 10(4):251–262. doi:10.1016/j.rti.2004.05.007
9. Forero MG, Crisóbal G, Alvarez-Borrego J (2006) Automatic identification of *Mycobacterium tuberculosis* by Gaussian mixture models. J Microsc 223(2):120–132. doi:10.1111/j.1365-2818.2006.01610.x
10. Vallotton P, Sun C, Wang D, Turnbull L, Ranganathan P (2009) Segmentation and tracking individual *Pseudomonas aeruginosa* bacteria in dense populations of motile cells. Paper presented at the Image and vision computing New Zealand, 24th International Conference, Wellington
11. Jung CR, Scharcanski J (2005) Robust watershed segmentation using wavelets. Image Vis Comput 23(7):661–669. doi:10.1016/j.imavis.2005.03.001
12. Mahmoudi L, El Zaart A (2012) A survey of entropy image thresholding techniques. Paper presented at the 2012 2nd International conference on advances in computational tools for engineering applications (ACTEA), Beirut
13. Canny J (1986) A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell 8(6):679–698. doi:10.1109/TPAMI.1986.4767851
14. Lindeberg T (1998) Edge detection and ridge detection with automatic scale selection. Int J Comput Vis 30(1):117–154. doi:10.1023/A:1008097225773

15. Kass M, Witkin A, Tersopoulos D (1988) Snakes: active contour models. Int J Comput Vis 1(4):321–331. doi:10.1007/BF00133570
16. Meyer F (1994) Topographic distance and watershed lines. Signal Process 38(1):113–125. doi:10.1016/0165-1684(94)90060-4
17. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Syst Man Cybern 9(1):62–66. doi:10.1109/TSMC.1979.4310076
18. Ping-Sung L, Tse-Sheng C, Pau-Choo C (2001) A fast algorithm for multilevel thresholding. J Inf Sci Eng 17:713–727
19. Zhang Y, Wu L (2011) Optimal multi-level thresholding based on maximum Tsallis entropy via an artificial bee colony approach. Entropy 13(4):841–859. doi:10.3390/e13040841
20. Fang M, Yue G, QingCang Y The study on an application of Otsu method in Canny operator. In: Proceedings of the 2009 International Symposium on Information Processing, Huangshan, P. R. China, 21–23 August 2009. p 109–112
21. Zahara E, Fan S-KS, Tsai D-M (2005) Optimal multi-thresholding using a hybrid optimization approach. Pattern Recognit Lett 26:1082–1095. doi:10.1016/j.patrec.2004.10.003
22. Huang D-Y, Wang C-H (2009) Optimal multi-level thresholding using a two-stage Otsu optimization approach. Pattern Recognit Lett 30:275–284. doi:10.1016/j.patrec.2008.10.003