Expert Systems with Applications 40 (2013) 6521–6530

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Segmentation of neuronal nuclei based on clump splitting and a two-step binarization of images

A. LaTorre [a,*], L. Alonso-Nanclares [a,b], S. Muelas [c], J.M. Peña [c], J. DeFelipe [a,b]

[a] Instituto Cajal, Consejo Superior de Investigaciones Científicas (CSIC), Avenida Doctor Arce 37, 28002 Madrid, Spain
[b] Laboratorio Cajal de Circuitos Corticales, Centro de Tecnología Biomédica, Universidad Politécnica de Madrid, Campus Montegancedo S/N, Pozuelo de Alarcón, 28223 Madrid, Spain
[c] Facultad de Informática, Universidad Politécnica de Madrid, Campus Montegancedo S/N, Boadilla del Monte, 28660 Madrid, Spain

ARTICLE INFO

Keywords:
Segmentation
Clump splitting
Concavity analysis
Overlapping objects
Binarization

ABSTRACT

In this paper we present an algorithm to segment the nuclei of neuronal cells in confocal microscopy images, a key technical problem in many experimental studies in the field of neuroscience. We describe the whole procedure, from the original images to the segmented individual nuclei, paying particular attention to the binarization of the images, which is not straightforward due to the technical difficulties related to the visualization of nuclei as individual objects and incomplete and irregular staining. We have focused on the division of clusters of nuclei that appear frequently in these images. Thus we have developed a clump-splitting algorithm to separate touching or overlapping nuclei allowing us to accurate account for both the number and size of the nuclei. The results presented in the paper show that the proposed algorithm performs well on different sets of images from different layers of the cerebral cortex.

1. Introduction

A major challenge in neuroscience is to determine the structural organization of the cerebral cortex (hippocampus and neocortex) (reviewed in DeFelipe, 2010). In the present study, we have focused on the neocortex which is a multi-laminated, highly organized structure that contains different neuronal cell types and a diverse range of glial cells. Neuroanatomists have dedicated considerable efforts to define methods that accurately estimate the number of cells (neurons and glia) in the cerebral cortex. Among them, stereological methods have been developed to estimate the number of cells in a given volume of tissue. However, there are often discrepancies in the results from different laboratories due to the diverse methodologies and mathematical approaches used to obtain the estimates. Thus the development of methods to determine the actual number of cells in the brain is a major aim in neuroscience.

In this paper we present an algorithm to segment neurons' nuclei in confocal microscopy images (a widely used technique to analyze the microstructure of the brain) from the rat somatosensory cortex as a model to count cells. This type of image constitutes a challenging problem for several reasons. First, due to the limitations inherent to the acquisition process, the binarization of original images must be conducted carefully in order to minimize the amount of noise present in the binary images. Second, even when image noise has been reduced to a minimum in the post-binarization images, nuclei are still difficult to segment, as they may overlap with other nuclei and these nuclei may belong to different cell types (i.e., neurons, astrocytes, oligodendrocytes, pericytes, etc.). To overcome these problems, we have used two different sets of images taken from histological sections from the rat somatosensory cortex (for simplicity, neocortex unless otherwise specified). One set shows the nuclei of all types of cells (neurons and non-neuronal cells). The other set of images was obtained from the same sections that were double labeled to selectively visualize neurons. In the latter case, neurons are very difficult to segment due to large irregularities in their contours since not only the soma is stained but also their proximal processes. However, we have used this information to automatically remove the nuclei belonging to other cells types from the images in the first set, thereby greatly facilitating and accelerating the process of counting neurons (as described in Section 2).

The proposed algorithm is divided into two sub-algorithms. The first one focuses in binarizing the original images in such a way that nuclei are separated from background. Binarization has been an important task in Image Processing since almost its beginning. Otsu (1979) proposed a method to automatically reducing a gray-level image into a binary image by computing the optimum thresholding value, i.e., minimizing the intra-class variance. Niblack (1985) proposed a local thresholding algorithm that computes the thresholding value according to the average and standard deviation of the neighborhood under evaluation. Such neighborhood

* Corresponding author.
E-mail addresses: antonio.latorre@csic.es, atorre@fi.upm.es (A. LaTorre), aidil@cajal.csic.es (L. Alonso-Nanclares), smuelas@fi.upm.es (S. Muelas), jmpena@fi.upm.es (J.M. Peña), defelipe@cajal.csic.es (J. DeFelipe).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.06.010

must be small enough to preserve local details and large enough to suppress noise. Kittler and Illingworth (1986) assume that an image is characterized by a mixture distribution and try to estimate its density function based on the histogram of the image. Other approaches, such as Level Set Methods (LSM), use signed functions to track the contours of the objects, where its zero level corresponds with the actual contour of the objects (Osher & Fedkiw, 2003).

The second sub-algorithm deals with the problem of dividing groups of overlapping cells into binary images, usually referred to as clump splitting in the literature. The simplest approach for this problem is to use uniform erosion, as previously described (Arcelli & Sanniti, 1984; Di Ruberto, Dempster, Khan, & Jarra, 2002; Suzuki & Abe, 1985; Thompson, Bartels, Haddad, & Bartels, 1990). However, this technique will only work when the convexities of the clump are deep enough, and this is usually not the case (Yeo, Jin, Ong, Jayasooriah, & Sinniah, 1994). Some alternatives try to overcome this problem by selectively applying erosion to certain parts of the cell, although these approaches are limited due to the sensitivity to the values of some of their parameters and being also very computationally intensive (Ong, Jayasooriah, Yeow, & Sinniah, 1992). Other algorithms attempt to exploit some prior knowledge about the images, mainly the shape of the cells and the concavities of their contours (Jin, Yeo, Ong, Jayasooriah, & Sinniah, 1994; Kumar, Ong, Ranganath, Ong, & Chew, 2006; Makkapati & Naik, 2009; Schmitt & Reetz, 2009; Wang, 1998; Yeo et al., 1994). Some algorithms assume that cells have an ellipse-like shape and use different techniques to fit ellipses in the binary images (Bai, Sun, & Zhou, 2009; Evans, Berman, & Talbot, 2004; Talbot & Appleton, 2002; Yang & Parvin, 2003; Yu, Pham, Zhou, & Wong, 2007), where each of these potentially overlapping ellipses is an individual cell. The main problem for these algorithms is how to divide the contour into segments and assign these segments to the different fitted ellipses. For example, Bai et al. (2009) propose an interesting algorithm based on ellipse-fitting that divides the contour of the cells into different segments. The points dividing these segments are called concave points. Additionally, for each of these concave points, one extra point is added to deal with those cases in which only one concavity is appreciable or the cells are incomplete. However, the authors suggest placing the new concave point at "the half index of the point sequence". This could be a good approximation when there are only two overlapping cells and they are of the same size. Nevertheless, this is not always true, at least for the problem under consideration in this paper. Other algorithms assume circular shapes (LaTorre et al., 2011) and use optimization techniques to search circular structures in the image that maximize the contrast between the inner and outer parts of the cells. Finally, Watershed algorithms have also been used for clump splitting of adjacent cells (Battenberg & Bischofs-Pfeifer, 2006; Beucher & Lantuéjoul, 1979; Serra, 1982). Although they can be successfully used with some images (Bhagwat, Krishna, & Pise, 2010; Mao, Zhao, & Tan, 2006; Tai et al., 2007; Vincent & Soille, 1991), with cells that largely overlap, cells of different size or cells with poor gradient at the concavities, they can exhibit the well-known problems of over and under-segmentation (Bai et al., 2009).

In our algorithm, we would like to highlight two main contributions. First, we propose an appropriate binarization process tailored to the particularities of the images under consideration, as state-of-the-art algorithms do not seem to provide satisfactory results. The second contribution is the extension of the clumpl splitting algorithm already proposed by Yeo et al. (1994), improving the generated output with specific techniques (highlighted in Section 2). These two contributions, when combined, allow us to obtain remarkable results in a fully automated way, as shown in Section 3.

The remainder of the paper is organized as follows: In Section 2 the proposed algorithm is introduced and described in detail, while Section 3 depicts the results obtained with our algorithm and compares its performance with two state-of-the-art algorithm. Section 4 presents the final conclusions of this work and outlines some directions for future research. Finally, although the present study has focused on the cerebral cortex, the algorithm developed in this article can be applied to other regions of the brain.

2. Presentation of the algorithm

The algorithm presented in this paper has been divided into three main steps or sub-algorithms that will be described in Sections 2.1–2.3, respectively.

2.1. Alg. 1: Two-step binarization

This algorithm processes original images obtained with the confocal microscope and transforms them into binary images in which cells are separated from noise and background. Both channels' images (DAPI and NeuN) undergo the first step of this algorithm, whereas only DAPI channel images undergo the second step. The reason for this is that we do not need accurate cell boundaries for the NeuN channel images as the whole clump splitting process will be carried out on the DAPI images and the NeuN images will only be used to filter out nuclei of cells that do not belong to neurons. Therefore, conducting the second step on the NeuN images would be unnecessary.

There are several well-known binarization approaches in the literature (Kittler & Illingworth, 1986; Niblack, 1985; Otsu, 1979), which offer satisfactory results in many different applications. However, during the course of this study we observed that the available binarization algorithms did not provide satisfactory results when applied to our images. Even after the use of noise

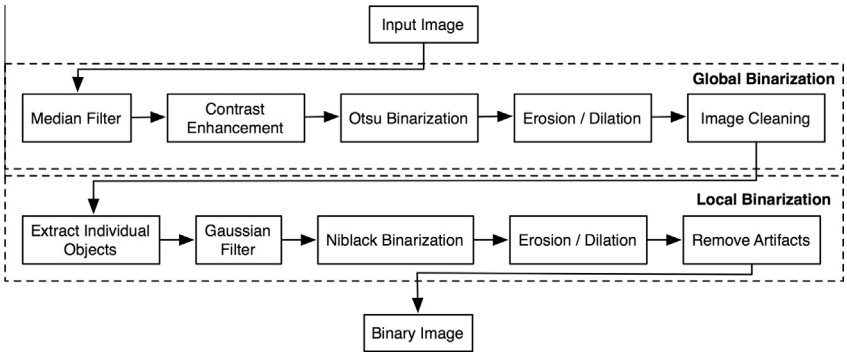


Fig. 1. Algorithm 1: Two-step binarization. In the first step, the input image is processed as a whole, whereas in the second step each of the binary blobs identified in the first step is processed individually. Both channels, DAPI and NeuN, undergo the first step of the algorithm, whereas only DAPI images undergo the second step.

reduction and contrast enhancement filters (see below), the images still exhibited non-uniform brightness that make it impossible for a global thresholding algorithm to offer accurate binarization of cells with sharp and well-defined contours. For this reason, we propose the two-step binarization algorithm depicted in Fig. 1. The first step of the algorithm aims to obtain a first rough binarization of the image, trying to remove as much noise as possible without taking into account the contours of the cells. The second step then processes each clump coming from the previous one attempting to provide a soft reconstruction of the contours of the cell. Both steps are described in detail in the following paragraph.

In the first step (Global Binarization), the image is processed as a whole, with a pipeline of filters that are applied globally to the image. The objective of this first step is to obtain a first approximation of the binary image, identifying the cells present in the image that will be refined in the second step. Prior to the application of any binarization algorithm, the image is processed with a median filter to reduce noise while trying to preserve edges (Lim, 1990) and its contrast is enhanced with the CLAHE algorithm (Zuiderveld, 1994). Then, we use Otsu's method (Otsu, 1979) to obtain a first binary image, which is subject to the typical morphological transformations (erosion/ dilation) (Di Ruberto et al., 2002) and a cleaning process aiming to remove small structures that come from the noise still present in the image. Table 1 summarizes the parameter values used for each of the aforementioned filters. It should be noted that all these values are the default values used by the Matlab implementation of all the filters, except for the erosion/dilation mask. In this case, we found the depicted mask to generate images with better contours.

The second step (Local Binarization) attempts to improve the accuracy of the binarization process by using the binary image obtained in the first step to identify individual binary blobs. Each of these blobs is then used to compute its bounding box in the original image. These bounding boxes are then subject to a Gaussian low-pass filter, which gave us better results than the median filter in this second step, and binarized again but, in this case, with Niblack's algorithm (Niblack, 1985), which obtained more accurate results in our experiments than other well-known binarization algorithms such as Otsu's method (Otsu, 1979) or Kittler's Minimum Error Thresholding algorithm (Kittler & Illingworth, 1986). The new binary blobs are subject to the same morphological transformations used in the global binarization step and the final image is also cleaned to remove small objects as in the previous case. Fig. 2 show examples of the problems of using global thresholding and how the two-step binarization deals with them. The specific parameter values for the algorithms used in the Local Binarization step are depicted in Table 2.

Finally, we try to clean the binary blobs and remove bright elements that appear in the original image tightly coupled to neurons' nuclei (nuclei of other non-neuronal cells). This step can be considered as an extra optimization of the algorithm, as this scenario does not happen that frequently. However, as its application to clumps not containing bright cells is safe (they will not be modified) it is always a good idea to include this step in the overall pipeline. For this purpose, we follow the procedure depicted in Fig. 3. We start the process with the bounding box of the original image and its binarized counterpart (Fig. 3(1) and (2)). In the original image, the background pixels, those equal to zero, are inverted (Fig. 3(3)). Thereafter, the image with the inverted background is binarized again (Fig. 3(4)). As can be seen, with this process we have exploited the fact that the cells that we want to remove are quite bright and, by inverting the background, we have increased the average brightness level of the image and the thresholding binarization algorithm used in this step actually discards the neurons' nuclei. Finally, we need only to invert the background again and subtract the (cleaned and filled) remaining binary image from the original binary image (Fig. 3(5) and (6), respectively).

Once all the binary blobs have been processed, the whole image is recomposed by placing each blob at the appropriate place in the whole image.

Table 1
Parameter values of algorithms used in the global binarization step.

Median filter neighborhood	3×3
CLAHE number of tiles	8×8
CLAHE contrast enhancement limit	0.01
CLAHE number of bins	256
CLAHE range	Full range
CLAHE distribution	Uniform
CLAHE alpha	0.4
Erosion/dilation mask	[0 1 0; 1 1 1; 0 1 0]

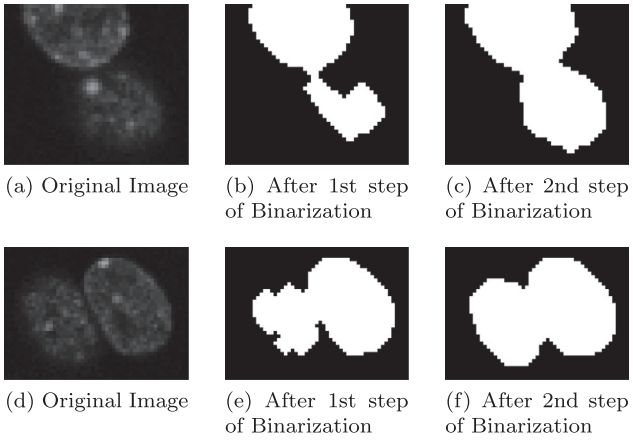


Fig. 2. Examples of two clumps after the first and the second step of the binarization.

Table 2
Parameter values of algorithms used in the Local Binarization step.

Gaussian filter size	3×3
Gaussian filter sigma	0.5
Niblack's filter k constant	-0.2

2.2. Alg. 2: Clump splitting of binary image

The second algorithm is depicted in Fig. 4. In this algorithm, each binary blob coming from the DAPI channel is processed individually to find concavities that could indicate the existence of a clump of nuclei. Due to the limitations of the binarization process and the irregularities in the contours of the nuclei, we must impose some conditions for such irregularities to be considered concavities. Different authors propose the use of different measures (or combinations) and thresholds (Bai et al., 2009; Yeo et al., 1994). For our problem, we have found a combination of three measures to be very effective to identify whether a segment of the contour is a concavity or not.

The first of these measures is the size, in pixels, of the candidate concavity (the pixels laying between the contour of the neuron and the convex hull). We establish a minimum threshold and all concavities with a size smaller than that value are discarded. At this point, we introduce a second threshold, which is termed "Minimum Join Concavity Area" in Table 3, to improve the quality of the clump splitting. This threshold is used to rescue all those candidate concavities with an area below the first threshold that are

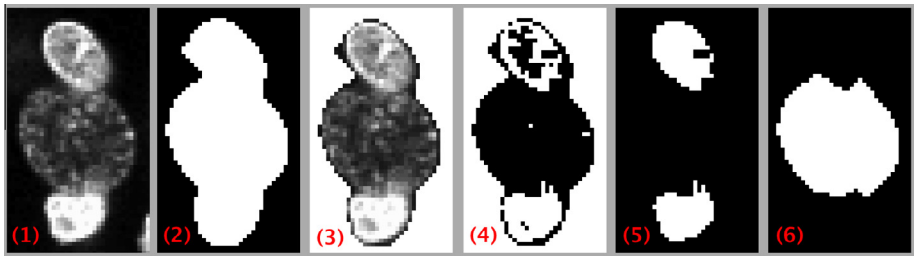


Fig. 3. Removal of bright artifacts from the image: (1) original image; (2) binary image; (3) inversion of background; (4) binarization of inverted image; (5) substraction of inverted background; (6) substraction of image 5 from image 2.

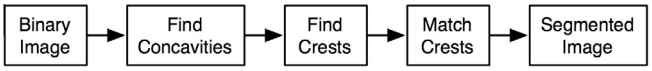


Fig. 4. Algorithm 2: Clump splitting of binary image. This algorithm takes the binary image returned by Algorithm 1 and processes each binary clump, dividing them into individual nuclei when needed. Only the DAPI channel image undergoes this algorithm.

still over this second threshold. The purpose of this "rescue" is to keep some large irregularities stored to be (possibly) used when matching concavities in the event that it is necessary to match a concavity against a point in the contour (see below).

The second of these measures is the maximum-minimum distance of any point of the concavity to the convex hull of the clump. This measure aims to avoid selecting concavities that are not deep cuts on the clump. When the area is large enough this will not usually happen. However, we continue using this filter to edge on the side of caution.

The third measure used is the concavity degree ($D(S_i)$) described in Yeo et al. (1994). This measure is defined as the ratio between the length of the concavity and the length of the corresponding part of the convex hull (see Yeo et al., 1994 for a graphic depiction of this measure). The objective of this measure is similar to the previous one, but is expressed in relative terms rather than as an absolute value.

Only those candidate concavities satisfying these three conditions are considered for further processing (and those with an area greater than the second area threshold are kept stored). Fig. 5 provides a graphical depiction of how these three measures are obtained.

The following steps are extensions to the work by Yeo et al. (1994). Once all the concavities of a clump have been identified, each concavity is processed in order to determine the number of crests present in that concavity. A crest can be defined as a local maximum of the line made up of the points of a concavity when it is rotated to lay on the line joining the first and the last point of the concavity (Fig. 6(5)). Since the considered images are raster images and the concavities are represented as polylines, before searching for the local maximum the concavity is smoothed with a moving average lowpass filter. The rationale behind this analysis is that sometimes multiple nuclei form a clump in which more than two nuclei are involved in a concavity. In these cases there will not be a single crest in the concavity but multiple ones. With this processing we are able to detect such cases and provide a more accurate clump splitting by matching crests and not concavities directly. Fig. 6 depicts the crest extraction process.

Table 3
Parameter values.

Min. concavity area	20 px.
Min. join concavity area	10 px.
Min. distance to bounding box	5 px.
D (S_i)	10%
Min. area between crests	35 px.
Local search angle	0.50 rad

Fig. 5. The area between the magenta and the yellow lines represents a concavity (and the number of pixels between them indicate its size, the first of the measures considered), whereas the area between the blue and the yellow lines represents a join concavity. The maximum-minimum distance (the second measure) of a concavity is the length of the green line. Finally, DS_i (the third measure) is computed by dividing the length of the magenta line and the length of its yellow counterpart. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Finally, we should join crests among them or with the contour to create the splitting lines that will separate the clumps into individual nuclei. To create such lines, the algorithm generates all the possible combinations of crests among them and with the contour. However, not all the combinations are considered, as some of them will be invalid. In particular, the following four situations will lead to an invalid combination:

1. Two crests belong to the same concavity (Fig. 7(a)).
2. The area between the line joining two crests and the contour of the clump is smaller than the "minimum area between crests" (Fig. 7(a)).
3. Two of the splitting lines of a combination intersect (Fig. 7(b)).
4. Two lines matching crests A and B with points in the contour are matched with points too close to B and A, respectively (i.e., this would have the same effect as matching A and B) (Fig. 7(c)).

Additionally, the special case of matching two crests that are already touching and thus a splitting line is not possible between them should be considered on an individual basis (Fig. 7(d)).

When processing a match between crests or a crest and the contour, we allow the algorithm to explore not only a single point on the "opposite" side of the nucleus (the one for which the normal vector to the concavity intersects the contour) but rather a neighborhood delimited by an angle ("local search angle" in Table 3). From this exploration we will keep the shorter line or, if one of the explored points lays within a join concavity (those with an area smaller than 20 pixels but larger than 10 pixels), the line joining the crest and the join concavity will be kept (see Fig. 8).

For each of the valid combinations we compute the overall length of all the splitting lines and select the combination that

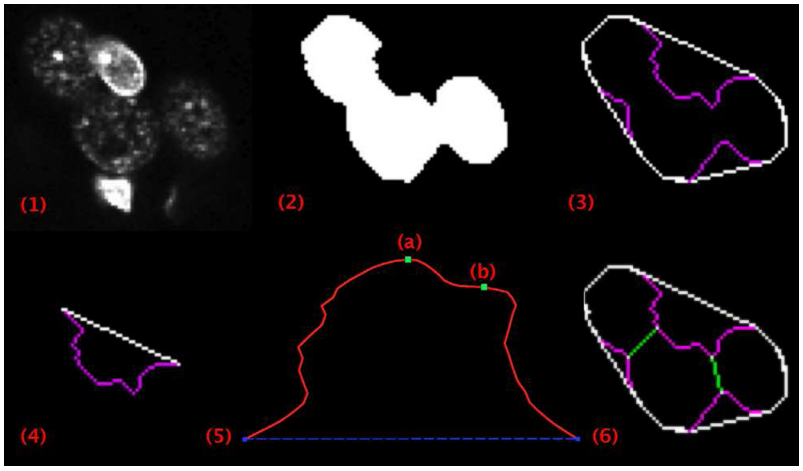
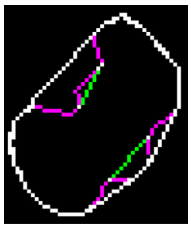
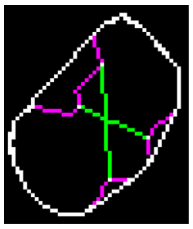


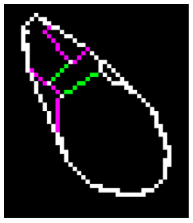
Fig. 6. Example of a multi-crest concavity analysis. (1) and (2) represent the original and the binarized image, respectively. In (3) we show all the concavities, while (4) represents the concavity selected for analysis (the upper right-most one). In (5) the (smoothed) concavity can be seen with the two local maxima ((a) and (b)) marked in green. Finally, (6) represents the division of the clump into three nuclei by matching the two identified crests of the selected concavity with those of the other concavities. It should be noted that one of the crests has been moved due to the effect of the local search described in this section.



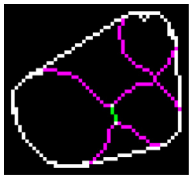
(a) Invalid combinations 1 and 2



(b) Invalid combination 3

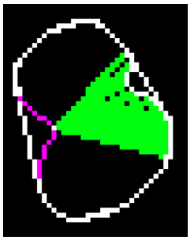


(c) Invalid combination 4

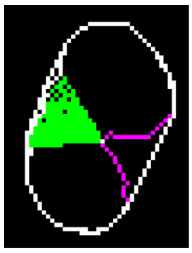


(d) Concavities already touching

Fig. 7. Examples of invalid combinations of crests and of the special case of two touching concavities that must be considered individually.



(a) Local Search matching a join concavity



(b) Local Search matching the contour

Fig. 8. Examples of how the local search is conducted when matching a crest against a point in the contour with and without a join concavity ((a) and (b), respectively). Multiple join lines are depicted simultaneously (in green) to show the search angle employed. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

minimizes this value. This combination will be used to divide the original clump into individual nuclei.

All the parameters used in this algorithm and their corresponding values are summarized in Table 3. These values were obtained by tuning them through several executions.

One final point to note is that, during the experimentation described in this paper we have observed a special case of clumps in which two nuclei were so tightly coupled that they presented two concavities of sizes ranging from 10 to 20 pixels. With the algorithm described in the previous paragraphs, these nuclei would remain together, as we only search for crests in those concavities with a size larger than 20 pixels. For this reason, when the algorithm finds a clump with two concavities with sizes ranging from 10 to 20 pixels, and only in this case, these two concavities are used to split the clump into two nuclei.

Finally, although this is a specific case of the images under analysis that could be disregarded in other situations, it does not affect the generality of the algorithm.

2.3. Alg. 3: Merging of both channels

The third and last algorithm combines the information coming from the DAPI channel with that available in the NeuN channel. The objective of this step is to filter out those nuclei appearing in the DAPI channel that are not present in the other channel (i.e., nuclei from non-neuronal cells). For this purpose, each of the individual nuclei identified by Algorithm 2 in the DAPI channel is taken into account and its overlapping with the neurons in the NeuN channel is computed. As the DAPI channel contains the neurons' nuclei and the NeuN channel contains the soma plus the nuclei of the neuron, we can expect this overlapping value to be high in spite of the fact that the nuclei are not always present in the labeled neurons due to the plane of section (i.e., when only a portion of the neuron is included in the section). In our experiments we have found that a suitable value for this threshold is 70%, to allow for small errors in the binarization of the NeuN channel or the clump splitting in the DAPI channel. However, the algorithm is not particularly sensitive to variations at this threshold and values ranging from 60% to 80% should provide similar results (see Fig. 9).

3. Results and discussion

In this section we present and discuss the results obtained with the application of the proposed algorithm to a set of 4 images

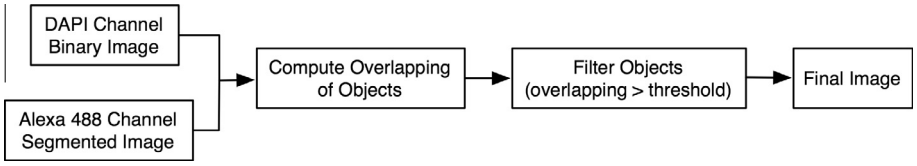


Fig. 9. Algorithm 3: Merging of DAPI and NeuN channels' information. The images returned by Algorithm 1 (NeuN channel) and Algorithm 2 (DAPI channel) to remove nuclei from non-neuronal cells present in DAPI channel.

Table 4
Computer configuration and programming language.

PC	Intel Core i7–2600 K 4 cores 3.4 Ghz CPU
Operating system	Ubuntu Linux 12.10
Prog. language	Fiji 1.47c and Matlab R2011b

(from layers 2, 3, 5A and 6A) of the somatosensory neocortex from 14-day-old rats. These images were selected as representative in terms of density and distribution of neurons along the cortical layers. They also constitute a good benchmark as the number of neurons at each stack of images is relatively high, ranging from approximately 1,200 to 1,700 neurons.

The results reported for this work have been obtained with the computer configuration and programming language displayed in Table 4.

3.1. Acquisition of the images

Postnatal 14-days-old rats ($n = 6$) were perfused with 4% paraformaldehyde and the fixed brain was sliced coronally into sections (50 μm) that were collected serially. All animals were handled in accordance with the guidelines for animal research set out in the European Community Directive 86/609/EEC and all the procedures were approved by the local ethics committee of the Spanish National Research Council (CSIC). Sections containing the hindlimb region of the somatosensory cortex (S1HL; by Paxinos & Watson, 2007) were stained using immunofluorescence with a mouse anti-neuron specific nuclear protein (NeuN, 1: 2000, Chemicon, Temecula, CA, USA) and Alexa Fluor 488 goat-anti mouse (1: 1000, in blocking solution; Molecular Probes). Thereafter, the sections were stained with a solution containing 105 mol/L of the fluorescent dye 4,6-diamidino-2-phenylindole (DAPI; Sigma D9542, St Louis, USA). After staining, the sections were mounted with ProLong Gold Antifade Reagent (Invitrogen) and examined on the Zeiss 710 confocal laser scanning system. NeuN (for visualizing neurons) and DAPI (for visualizing nuclei of all types of cells) fluorescence was recorded through separate channels. Image stacks of 40–50 planes were obtained with an EC PL NEO 40× immersion lens (N.A. 1.3), using a z-step of 1 μm and a scanning resolution of 512 × 512 pixels (pixel size 0.5 μm). An example of both types of images is shown in Fig. 10.

3.2. Validation of the proposed algorithm

To validate the accuracy of the proposed algorithm, the segmented images obtained from its application have been reviewed by an expert in the field of neuroanatomy. Table 5 summarizes the results obtained in terms of properly segmented nuclei and incorrect segmentations, according to the manual segmentation conducted by the expert. In order to provide more insight into the behavior of the algorithm, we have reported the possible different types of error separately. Type-1 error refers to cells divided that have been divided into multiple parts when they should have not. Type-2 error represents the opposite case: two or more adjacent cells that should have been divided into several cells have been kept together. Type-3 error occurs when background noise is detected as a cell in the binarization process. Finally, Type-4 error happens when a correct cell has not been detected by the binarization step of the algorithm.

As can be seen in Table 5, the percentage of correctly segmented nuclei of neurons is very high for all the images considered (always over 91%). Moreover, for the image from layer 6A the percentage of correctly segmented neuronal nuclei reaches 98%. Regarding the

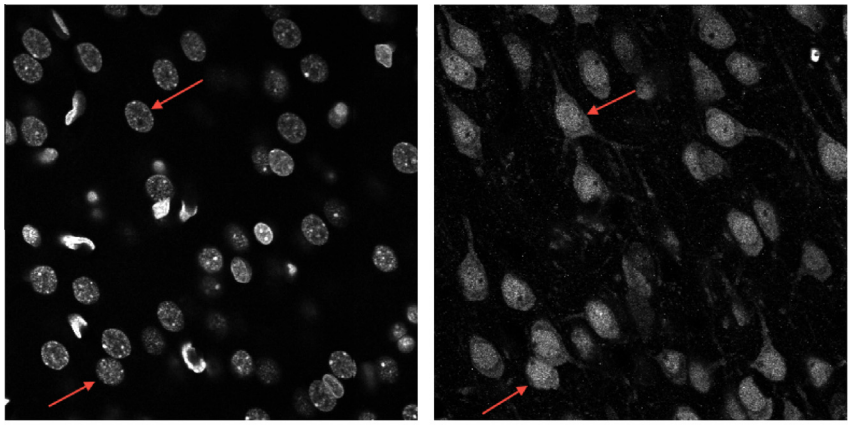


Fig. 10. Image samples from a single slice of the rat cerebral cortex. These two images were taken from the same field and plane of a section double labeled for DAPI and NeuN. (a) DAPI channel image showing nuclei from all cell types. (b) NeuN channel image showing the nuclei, the soma and proximal processes of neurons. Arrows indicate examples of the same neurons in both images. Note the difficulty involved in directly working with the NeuN channel due to the irregular shape of the neurons, whereas in the DAPI channel, numerous labeled nuclei from cells which are not neurons can be observed (that do not appear in the NeuN channel). These non-neuronal nuclei were discarded to obtain an estimation of the number of neurons based on the visualization of their nuclei.

Table 5
Validation results.

	Layer 2	Layer 3	Layer 5A	Layer 6A
Number of manually detected cells	1394	1714	1124	1555
Correct	92.72%	93.45%	91.71%	98.21%
Type-1 error: Over-segmentation	1.13%	0.41%	4.15%	0.26%
Type-2 error: Under-segmentation	2.48%	3.48%	0.51%	0.51%
Type-3 error: Noise detected as cell	0.28%	0.29%	0.76%	0.13%
Type-4 error: Undetected cell	3.39%	2.38%	2.88%	0.90%

different types of errors, it should be noted that the algorithm performs very well in removing noise; in all cases it reported less than 1% of the detected structures incorrectly as neuronal nuclei, i.e., noise in the image mistakenly binarized as a nucleus. Furthermore, the number of missing neuronal nuclei remains moderate, with values ranging from 1% to 3.5%. Regarding the errors obtained by the clump splitting we can observe that these errors also remain quite low. For type-1 errors (over-segmentation), except for layer 5A, the percentage of over-segmented neuronal nuclei remains below (or very close to) 1%. For type-2 errors, we have very low percentages for layers 5A and 6A (below 1%), whereas it remains moderate for the other two images (always below 3.5%).

Fig. 11 shows the results obtained after applying the described algorithm to the sample image depicted in Fig. 10(a). Fig. 11(a) is the result of applying the two-step binarization algorithm to the input image, whereas Fig. 11(b) presents the results after applying the proposed clump-splitting algorithm and merging the information of both channels. Nuclei that did not require splitting appear completely in white, whereas clumps of cells that have been split exhibit highlighted concavities (in magenta) and the splitting lines in green. The convex hull is also depicted, in white, to show the size of the different concavities. Finally, in this image we can also see an example of the special case described at the end of Section 2.2. This special case is represented with an orange splitting line.

3.3. Comparison with Level Set Methods

Level Set Methods are popular numeric techniques for tracking and segmenting shapes and contours (Osher & Fedkiw, 2003). In the recent years, these methods have gained a lot of attention and have been widely used in the field of image segmentation (Li, Xu, Gui, & Fox, 2010; Tai et al., 2007). For this reason, in order to assess the convenience of using the two-steps binarization mechanism presented here, we have conducted the same experimentation, replacing the proposed binarization by the Level Set Method. In particular, we have used the Distance Regularized Level Set Evolution (DRLSE) algorithm, which is a generalization of Level Set Methods that incorporates an intrinsic mechanism to keep the Level Set Function (LSF) in a good condition (i.e., it is smooth and not too steep or too flat) (Li et al., 2010).

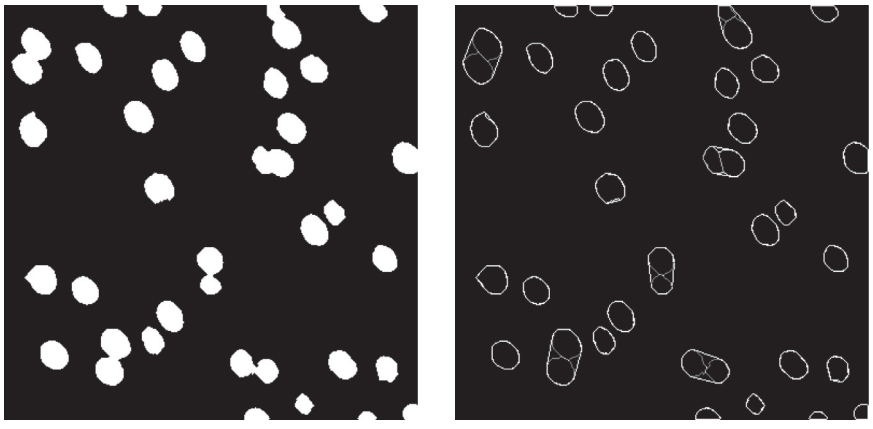
This algorithm has several parameters ($\lambda, \mu, \Delta t$ and α) although, as the authors state, the three first parameters can be kept fixed and attention should be paid to the α parameters that has to be tuned per problem. In our experimentation, the following values have been used for the first three parameters: $\lambda = 5.0,\ \mu = 0.04$ and $\Delta t = 5.0$.

To determine the appropriate value for the α parameter, we have conducted a parameter tuning in which several different values have been tested. This parameter controls the external force that drives the motion of the contour around the objects. Large values for this parameter may lead to boundary leakage (i.e., the contour may pass through the object boundary), whereas small values of α provide more accurate contours at the cost of larger execution times. To analyze the effect of different values for this parameter on the performance of the algorithm, the following α values have been used: 0.5, 1.0, 1.5, 2.0 and 2.5.

In this context, as the datasets that must be processed may be quite large, execution time is also an important issue to take into account. Furthermore, the execution times for $\alpha = 0.5$ are very large so we have divided our comparison in two parts. First, we have run the DRLSE algorithm with the five considered values for the α parameter on a subset of the original stacks of images, composed of three randomly chosen images from each of the stacks. Then we have computed both the accuracy and the execution time for each of the tested values and ranked them according to their Adjusted Ratio of Ratios (ARR) value (Brazdil, Soares, & Da Costa, 2003), which is a measure that weights both objectives (accuracy and execution time).

Table 6 contains the accuracy and execution times for each parameterization of the DRLSE algorithm and mini-stack of images (4 stacks of 3 randomly chosen images coming from each of the considered cortical layers).

With this information, we can compute the ARR value for each DRLSE variant as depicted in Eqs. (1) and (2). In these equations, $SR_{a_p}^{d_i}$ and $\tau_{a_p}^{d_i}$ represent the success rate (accuracy) and execution time of algorithm a_p on dataset d_i, respectively, whereas n is the number of datasets and m the number of algorithms. *AccD* represents the relative importance of accuracy and time, and takes a value of 0.10 in our tests, as recommended by the authors (although different values for this parameter did not change significantly the



(a) Binary Image (b) Segmented Image

Fig. 11. Binary image computed from the DAPI Channel and its associated segmented image.

Table 6
Accuracy and execution times for each parameterization of the DRLSE algorithm and mini-stack of images.

	Layer 2		Layer 3		Layer 5A		Layer 6A	
	Accuracy (%)	Time (s)	Accuracy (%)	Time (s)	Accuracy (%)	Time (s)	Accuracy	Time (s)
DRLSE $\alpha = 0.5$	27.93	7519	70.31	19541	76.04	21958	75.83	10203
DRLSE $\alpha = 1.0$	80.00	397	79.20	1250	74.19	1222	84.48	737
DRLSE $\alpha = 1.5$	77.88	308	75.40	874	63.92	871	81.74	452
DRLSE $\alpha = 2.0$	74.56	201	68.94	695	49.46	789	73.50	451
DRLSE $\alpha = 2.5$	60.71	195	59.40	554	37.50	631	66.39	435

Table 7
Overall ranking of the five parameterizations of the DRLSE algorithm according to the ARR value.

Rank	a_p	ARR_{a_p}
1	DRLSE $\alpha = 1.0$	1.15
2	DRLSE $\alpha = 1.5$	1.08
3	DRLSE $\alpha = 2.0$	0.93
4	DRLSE $\alpha = 2.5$	0.74
5	DRLSE $\alpha = 0.5$	0.70

values obtained and did not affect the ranking of the different parameterizations of the DRLSE algorithm).

$$ARR^{d_i}_{a_p,a_q} = \frac{\frac{SR^{d_i}_{a_p}}{SR^{d_i}_{a_q}}}{1 + AccD * log\left(\frac{\tau^{d_i}_{a_p}}{\tau^{d_i}_{a_q}}\right)} \quad (1)$$

To calculate the ranking shown in Table 7, first we must compute the ARR value for each pair of algorithms and dataset (Eq. (1)). Then, the geometric mean is computed for each algorithm, as shown in Eq. (2).

$$ARR_{a_p} = \frac{\sum_{a_q} \sqrt[n]{\Pi_{d_i} ARR^{d_i}_{a_p,a_q}}}{m} \quad (2)$$

As can be seen in Table 7, the best parameterization for the DRLSE algorithm considering both aspects, accuracy and execution time, is that with $\alpha = 1.0$. This is consistent with the results shown in Table 6, where this algorithm obtained the best accuracy in most of the mini-stacks, whereas the execution time was still moderate.

With these results, we executed the DRLSE algorithm with $\alpha = 1.0$ on the whole set of images. The validation process was the same as with the proposed algorithm to avoid undesirable biases. It should be noted that, tough Level Set Methods are often used as segmentation algorithms on themselves, cells in the considered images frequently appear in clumps, so it is still necessary to process the results of the DRLSE algorithm to try to split clumps of cells. Thus, the results reported in Table 8 are the results of the proposed algorithm replacing the binarization step by the execution of the DRLSE algorithm.

Comparing the results from Tables 5 and 8, it can be observed that the results obtained with the proposed two-steps binarization correctly segments approximately 12% to 17% more cells than the DRLSE algorithm. If we look at the different errors, we can see that the higher error increment is in noise miss-detected as cells and undetected cells. This is important to note as it supports the hypothesis that the difference in performance comes from the use of one binarization algorithm or the other and not from the clump splitting method. Furthermore, it should be noted that the DRLSE algorithm segments cells with worse contours and, quite frequently, with a portion on the actual size of the cell (due to the aforementioned boundary leakage, even using a moderately small value for the α parameter).

Table 8
Validation using the DRLSE algorithm.

	Layer 2 (%)	Layer 3 (%)	Layer 5A (%)	Layer 6A (%)
Correct	80.75	84.24	80.47	85.53
Type-1 error: Over-segmentation	3.47	1.80	5.14	2.21
Type-2 error: Under-segmentation	2.75	3.60	1.98	1.79
Type-3 error: Noise detected as cell	3.99	6.87	4.90	4.37
Type-4 error: Undetected cell	9.04	6.49	7.51	6.10

A comparison on the size of the segmented cells by both approaches (DRLSE and two-steps binarization) was conducted and the results are presented in Table 9. As can be seen, almost half of the cells segmented by the DRLSE algorithm have a size 25% smaller than their counterparts coming from the two-steps binarization. Furthermore, a not negligible 6–10% of the cells has a size below 25% of the reference size. Nonetheless, as in this study we wanted to focus on the identification of individual cells, all those cells were classified as correct in the results reported in Table 8.

Finally, despite the difference in accuracy being quite important, we want to also report the execution times for both algorithms. Table 10 contains the execution time of each algorithm on each of the considered stacks of images, both overall and per slice. It can be observed that the execution time for the proposed two-steps binarization is much smaller (roughly 10–30 times smaller). Furthermore, in the case of the proposed algorithm the average execution time per slice remains more or less constant (it varies according to the size of the images). This is not the case for DRLSE algorithm, with execution times that can be up to four times larger from one layer to another.

3.4. Comparison with Watershed

In this section we are going to compare the performance of the second part of the proposed algorithm, the division of clumps of cells, with that of one of the reference algorithms in Image

Table 9
Ratios of cells segmented with the DRLSE algorithm with sizes below 25%, 50% and 75% compared to the reference segmentation which is, in this case, the one obtained with the proposed algorithm.

	Layer 2 (%)	Layer 3 (%)	Layer 5A (%)	Layer 6A (%)
Cells below 25% of reference size	6.52	8.90	10.14	6.51
Cells below 50% of reference size	22.61	26.04	25.13	19.99
Cells below 75% of reference size	46.15	50.18	46.42	43.83

Table 10
Comparative of the execution times (overall and per slice) of the DRLSE algorithm and the two-steps paramterization.

	Layer 2		Layer 3		Layer 5A		Layer 6A	
	Overall (s)	Slice (s)	Overall (s)	Slice (s)	Overall (s)	Slice (s)	Overall (s)	Slice (s)
Two-steps binarization	518	12.33	658	15.67	543	12.93	591	14.07
DRLSE $\alpha = 1.0$	5558	132.33	17500	416.67	17108	407.33s	10318	245.67

Segmentation: the Watershed algorithm (Beucher & Lantuéjoul, 1979). For this purpose, we have replaced the second algorithm (clump splitting) from our proposal by the Watershed algorithm. The same validation procedure has been followed, and the results obtained are reported in Table 11.

As can be observed, the Clump Splitting algorithm obtains better results (higher percentage of correctly identified cells) than Watershed. In the case of layer 6A, these differences are very subtle, as both approaches are able to correctly detect more than 98% of the cells. These differences are a bit larger for layers 2 and 3. Regarding the different errors, it is clear that the main drawback of the Watershed algorithm compared with Clump Splitting is under-segmentation, i.e., the Watershed algorithm is able to divide less clusters of cells than Clump Splitting.

Going one step further, we have tried to guess if it is possible to use both algorithms simultaneously. For this purpose, we have tried to learn a classification model that decides which splitting algorithm to use when the number of cells identified by each algorithm differs. To create such a model, we have run both algorithms simultaneously on the whole set of stacks. For each binary clump to be divided, we have recorded the number of cells identified by both algorithms and the size of the binary clump under consideration. Furthermore, the entries for which both algorithms agree on the number of cells are removed from the dataset. The remaining instances have been then manually labeled with the algorithm that correctly divides each clump. Finally, we have used a C4.5 algorithm (Quinlan, 1993) to create a classification model that selects, for each disputing clump, which algorithm should be used.

Once the classification model has been constructed, we have to rerun the whole process on all the stacks to test the accuracy of the hybrid approach. For each clump, if the number of cells identified by each algorithm agrees, we take into account the division made by the Clump Splitting algorithm (the differences when both algorithms agree are, in general, small and thus it is not very important which algorithm to choose here). If the algorithms do not agree, then the classification model is used to decide which of the algorithms should be used. Table 12 summarizes the results obtained by this hybrid approach.

It can be seen that, for all the considered stacks, the results of the combination of Clump Splitting and Watershed are slightly better than those of each algorithm used individually. Although there is a very narrow marging for improvement, this confirms that it is possible to combine several splitting algorithms and opens new possibilities for the future integration of other algorithms.

Table 11
Results using the Watershed algorithm.

	Layer 2 (%)	Layer 3 (%)	Layer 5A (%)	Layer 6A (%)
Correct	91.23	91.68	94.82	98.14
Type-1 error: Over-segmentation	1.13	0.00	0.53	0.38
Type-2 error: Under-segmentation	3.96	5.64	0.88	0.45
Type-3 error: Noise detected as cell	0.28	0.29	0.79	0.13
Type-4 error: Undetected cell	3.39	2.39	2.99	0.90

Table 12
Results using both the clump splitting and the Watershed algorithms simultaneously and deciding, based on the model obtained with the C4.5 algorithm, which of them to use when there are disputes in the number of cells.

	Layer 2 (%)	Layer 3 (%)	Layer 5A (%)	Layer 6A (%)
Correct	92.93	94.02	94.82	98.59
Type-1 error: Over-segmentation	1.13	0.17	0.53	0.26
Type-2 error: Under-segmentation	2.26	3.14	0.88	0.13
Type-3 error: Noise detected as cell	0.28	0.29	0.79	0.13
Type-4 error: Undetected cell	3.39	2.38	2.99	2.41

4. Conclusions

In this paper we have presented an algorithm to segment nuclei of brain cells in confocal microscopy images. We have described the whole procedure, starting from the original images up until the segmentation of the nuclei, paying particular attention to the two-step binarization of the images and the division of clusters of cells that appear frequently in these images. We have used several sets of images taken from different cortical layers from young (postnatal 14-days-old) rat cerebral cortex and the results obtained showed that the proposed algorithm reports accurate segmentations, a critical step to quantitatively and automatically study the cytoarchitecture of the brain. Additionally, we have compared both sub-algorithms in our proposal (Two-steps Binarization and Clump Splitting) with state-of-the-art algorithms for both tasks (Level Set Methods and Watershed algorithms, respectively). Furthermore, for the second sub-algorithm a combination of Clump Splitting and Watershed has been proposed and compared with the individual use of both algorithms, obtaining slight improvements in the already accurate performance of the individual algorithms. In a future study we will try to generalize the proposed algorithm and provide a mechanism to recreate the 3D structure from this 2D segmentation. Furthermore, we will test the algorithm with images obtained from different brain regions of different animals and ages. Finally, although the number of incorrectly binarized nuclei is quite low, we will try to analyze in which cases those nuclei are being discarded in order to improve the quality of the binarization algorithm.

Acknowledgments

This work was supported by Grants from the Spanish Ministry of Science (TIN2010-21289-C02-02) and the Cajal Blue Brain Project (Spanish partner of the Blue Brain Project initiative from EPFL). The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the Centro de Supercomputación y Visualización de Madrid (CeSViMa) and

the Spanish Supercomputing Network. LaTorre gratefully acknowledges the support of the Spanish Ministry of Science and Innovation (MICINN) for its funding throughout the Juan de la Cierva program.

References

Arcelli, C., & Sanniti di Baja, G. (1984). Quenching points in distance labeled pictures. In *Seventh international conference on pattern recognition*.

Bai, X., Sun, C., & Zhou, F. (2009). Splitting touching cells based on concave points and ellipse fitting. *Pattern Recognition, 42*, 2434–2446.

Battenberg, E., & Bischofs-Pfeifer, I. (2006). A system for automatic cell segmentation of bacterial microscopy images. *Technical report* UC Berkeley.

Beucher, S., & Lantuéjoul, C. (1979). Use of Watersheds in contour detection. In *International workshop on image processing: Real-time edge and motion detection/estimation*.

Bhagwat, M., Krishna, R., & Pise, V. (2010). Image segmentation by improved watershed transformation in programming environment MATLAB. *International Journal of Computer Science & Communication, 1*, 171–174.

Brazdil, P. B., Soares, C., & Da Costa, J. P. (2003). Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. *Machine Learning, 50*, 251–277.

DeFelipe, J. (2010). From the connectome to the synaptome: An epic love story. *Science, 330*, 1198–1201.

Di Ruberto, C., Dempster, A., Khan, S., & Jarra, B. (2002). Analysis of infected blood cell images using morphological operators. *Image and Vision Computing, 20*, 133–146.

Evans, C., Berman, M., & Talbot, H. (2004). Offline fast object splitting. *Technical report* CMIS 04/55 North Ryde, Australia.

Jin, X., Yeo, T., Ong, S., Jayasooriah, & Sinniah, R. (1994). *An automated clump decomposition system for cervical tissue sections. In Proceedings of the 16th annual international conference of the IEEE engineering in medicine and biology society. Engineering advances: New opportunities for biomedical engineers*. (pp. 720–721).

Kittler, J., & Illingworth, J. (1986). Minimum error thresholding. *Pattern Recognition, 19*, 41–47.

Kumar, S., Ong, S., Ranganath, S., Ong, T., & Chew, F. (2006). A rule-based approach for robust clump splitting. *Pattern Recognition, 39*, 1088–1098.

LaTorre, A., Muelas, S., Peña, J. M., Santana, R., Merchán-Pérez, A., & Rodriguez, J. R. (2011). A differential evolution algorithm for the detection of synaptic vesicles. In *2011 IEEE congress on evolutionary computation, CEC 2011* (pp. 1687–1694).

Lim, J. (1990). *Two-dimensional signal and image processing*. Englewood Cliffs, NJ: Prentice Hall.

Li, C., Xu, C., Gui, C., & Fox, M. D. (2010). Distance regularized level set evolution and its application to image segmentation. *IEEE Transactions on Image Processing, 19*, 3243–3254.

Makkapati, V., & Naik, S. (2009). Clump splitting based on detection of dominant points from contours. In *Fifth annual IEEE conference on automation science and engineering, 2009. CASE 2009* (pp. 197–201). Bangalore, India.

Mao, K., Zhao, P., & Tan, P. H. (2006). Supervised learning-based cell image segmentation for P53 immunohistochemistry. *IEEE Transactions on Biomedical Engineering, 53*, 1153–1163.

Niblack, W. (1985). *An introduction to digital image processing*. Prentice Hall.

Ong, S., Jayasooriah Yeow, H., & Sinniah, R. (1992). Decomposition of digital clumps into convex parts by contour tracing and labelling. *Pattern Recognition Letters, 13*, 789–795.

Osher, S., & Fedkiw, R. (2003). *Level set methods and dynamic implicit surfaces. Applied mathematical sciences* (153). Springer Verlag.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics, 9*, 62–66.

Paxinos, G., & Watson, C. (2007). *The rat brain in stereotaxic coordinates* (6th ed.). San Diego: Academic Press.

Quinlan, J. R. (1993). *C4. 5: Programs for machine learning* (1st ed.). Morgan Kaufmann.

Schmitt, O., & Reetz, S. (2009). On the decomposition of cell clusters. *Journal of Mathematical Imaging and Vision, 33*, 85–103.

Serra, J. (1982). *Image analysis and mathematical morphology*. Academic Press.

Suzuki, S., & Abe, K. (1985). New fusion operations for digitized binary images and their applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-7*, 638–651.

Tai, X.-C., Hodneland, E., Weickert, J., Bukoreshtliev, N., Lundervold, A., & Gerdes, H.-H. (2007). Level set methods for watershed image segmentation. In F. Sgallari, A. Murli, & N. Paragios (Eds.), *Scale space and variational methods in computer vision* (pp. 178–190). Berlin/Heidelberg: Springer.

Talbot, H., & Appleton, B. (2002). Elliptical distance transforms and the object splitting problem. In *Sixth international symposium mathematical morphology* (pp. 229–241).

Thompson, D., Bartels, H.G., Haddad, J. W., Bartels, P. (1990). Scene segmentation in a machine vision system for histopathology. In *PIE 1206, new technologies in cytometry and molecular biology* (pp. 40–47).

Vincent, L., & Soille, P. (1991). Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 13*, 583–598.

Wang, W. (1998). Binary image segmentation of aggregates based on polygonal approximation and classification of concavities. *Pattern Recognition, 31*, 1503–1524.

Yang, Q., & Parvin, B. (2003). Harmonic cut and regularized centroid transform for localization of subcellular structures. *IEEE Transactions on Biomedical Engineering, 50*, 469–475.

Yeo, T., Jin, X., Ong, S., Jayasooriah & Sinniah, R. (1994). Clump splitting through concavity analysis. *Pattern Recognition Letters, 15*, 1013–1018.

Yu, D., Pham, T., Zhou, & X., Wong, S. (2007). Segmentation, recognition and tracing analysis for high-content cell-cycle screening. In *2007 International symposium on computational models of life sciences (CMLS '07)* (pp. 66–75).

Zuiderveld, K. (1994). Contrast limited adaptive histograph equalization. In P. Heckbert (Ed.), *Graphics gems IV* (pp. 474–485). AP Professional (Academic Press).