Combined Segmentation and Tracking of Neural Stem-Cells

K. Althoff, J. Degerman, and T. Gustavsson

Department of Signals and Systems,
Chalmers University of Technology,
S-412 96 Gothenburg, Sweden
{althoff, degerman, gustavsson}@s2.chalmers.se
http://www.s2.chalmers.se/research/image/

Abstract. In this paper we analyze neural stem/progenitor cells in an time-lapse image sequence. By using information about the previous positions of the cells, we are able to make a better selection of possible cells out of a collection of blob-like objects. As a blob detector we use Laplacian of Gaussian (LoG) filters at multiple scales, and the cell contours of the selected cells are segmented using dynamic programming. After the segmentation process the cells are tracked in the sequence using a combined nearest-neighbor and correlation matching technique. An evaluation of the system show that 95% of the cells were correctly segmented and tracked between consecutive frames.

1 Introduction

Combining the segmentation and tracking of objects in an image sequence has the potential of producing faster and more reliable results than keeping the segmentation and tracking separate. This is especially true when dealing with non-deformable objects or when the number of objects is constant throughout the sequence. However, even when tracking cells, that are usually deformable and also have the ability to split or disappear out of view, an interleaved segmentation and tracking procedure could be useful. A popular method that combines the segmentation and tracking is active contours (snakes), previously used for cell segmentation/tracking in e.g. [1],[2]. A significant drawback with the traditional snake is its inability to deal with splitting cells. In these situations, segmentation and tracking using level set methods are better [3]. However, in cases where previously separated cells move so close together that the boundary between them becomes blurred, a level-set method will merge the two cells into one while the traditional snake would keep the two contours separated. A method which allows for cells to split while keeping clustered cells separate is topology snakes [1],[4]. Another limitation of active contours is that it requires a high sampling rate, so that the cells do not move long distances or exchange positions between two consecutive frames.

Another technique combining segmentation and tracking is demonstrated in [5], where the directional movement of cells induced by a direct current (gal-

H. Kalviainen et al. (Eds.): SCIA 2005, LNCS 3540, pp. 282–291, 2005.

vanotaxis) is studied. Standard thresholding combined with clustering results in a coarse segmentation, which is refined further using the result of an association/tracking algorithm. Tracking is done using a modified version of the Kalman filter. This method works well because of the known directional movement of the cells. However, assumptions about a cell's state equations are potentially risky in cases where little is known about the laws governing the cell motion, and when the purpose of the analysis is to gain more information about cell kinematics.

In this work we track neural stem/progenitor cells in vitro in order to better understand the mechanisms of the differentiation of the neural stem/progenitor cells. To detect the cells, we use multi-scale Laplacian of Gaussian filters to find blob-like structures of different size in the image. Then we use information about the cells' previous positions to decide which detected blobs correspond to real cells. Using the positions of the selected blobs, the contours of the cells are segmented using dynamic programming. To track the cells we use a combination of a nearest-neighbor and a correlation matching technique.

2 Material

Nine image sequences of the neural stem/progenitor cells were used to evaluate the automatic cell tracking system. The images had been acquired every 10 minutes for 48 hours using a time-lapse brightfield microscopy system described elsewhere [6]. Each sequence contained 288 images of size 634x504 pixels. After the last image was acquired, the cells were stained with antibodies making it possible to differentiate between glial progenitor cells, type-1 and type-2 astrocytes when viewed in a fluorescence microscope.

3 Cell Segmentation

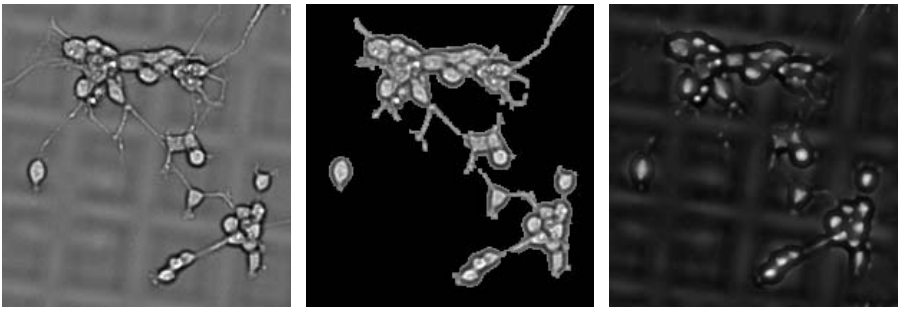
The automatic segmentation and tracking algorithm is an iterative process that for every image in the sequence can be described as follows:

1. Separate background from cell regions.
2. Find centroids of all blob-like objects in the image.
3. Select the centroids of the blobs that are most likely to be cells.
4. Segment the cells using dynamic programming.
5. Assign the segmented cells to the established tracks using the auction algorithm.

Steps 1-4 are described in this section and the last step is described in the next section. The segmentation and tracking is performed backwards, starting at frame 288, since the information about what kinds of cells the stem/progenitor cells have become is given subsequent to acquiring the last frame. In the initial frame (288) the segmentation is done using step 1-4, although step 2 is performed differently, see Sect. 3.3.

3.1 Background Removal

The first step in the segmentation process is to separate the cells from the background. This is efficiently done using the image variance, calculated according to Wu et. al. [7] with the window size set to 3×3 pixels. The variance image is then thresholded using Kittler's and Illingworth's method [8]. To create the final mask, I_{mask}, the morphological operation "close" is applied to the binary image and then holes smaller than 100 pixels are filled and regions smaller than 55 pixels are removed. Figure 1(a) shows an example of part of an image, and Fig. 1(b) the corresponding masked image.



(a) Original image (b) Masked image (c) $I_{multi}(x, y)$

Fig. 1. (a) An example of part of an image (201×201 pixels). (b) The masked original image. (c) $I_{multi}(x, y)$ calculated according to (3)

3.2 Blob Detection

To detect blob-like objects in an image, we convolve the image with a Laplacian of Gaussian (LoG) filter:

$$\hat{I}_k(x, y) = I(x, y) * L(x, y; \sigma_k) \tag{1}$$

where $L(x, y; \sigma_k)$ is the LoG filter for 2D images and x and y are the spatial coordinates and σ determines the width of the filter.

Since the blob-like objects in our images, the cells, are of different size, we need to apply multiple filters of different scale, i.e. different σ, to detect all cells. In this work, we use $2 \leqslant \sigma \leqslant 8$ with a step size of 0.33. To avoid the problem that the response from the LoG filter decreases with increasing scale, we replace $L(x, y; \sigma_k)$ in (1) with the normalized LoG filter [9],[10]:

$$\tilde{L}(x, y; \sigma) = \sigma^2 \cdot L(x, y; \sigma). \tag{2}$$

$\hat{I}_k(x, y)$ will have high intensities in pixels close to the center of blob-like objects of size proportional to σ_k. We then create a new image, $I_{multi}(x, y)$, by maximizing image intensity over scale:

$$I_{multi}(x,y) = \max\{\hat{I}_2(x,y), \hat{I}_{2.33}(x,y), ..., \hat{I}_8(x,y)\}. \quad (3)$$

Figure 1(c) shows $I_{multi}(x,y)$ for the image shown in Fig. 1(a). For more details on blob-detectors or multi-scale image analysis, see e.g. [9] and [10].

3.3 Choosing Cell Points

In accordance with [9] and [10], we consider the local maxima pixels in $I_{multi}(x,y)$ to be the the center points of the existing blobs. To get a more reasonable result we first calculate $\tilde{I}_{multi}(x,y)$ by smoothing $I_{multi}(x,y)$ with a Gaussian kernel (size 5x5, σ=3). We also remove all local maxima belonging to the background using the mask calculated in Sect. 3.1. Fig. 2(a) shows $\tilde{I}_{multi}(x,y)$ for the image shown in Fig. 1(a) with all the local maxima found. To use the information about the previous cell positions in the sequence, we assign every cell present in the previous image to one local maxima each and increase the intensity of $\tilde{I}_{multi}(x,y)$ in those local maxima. The assignment problem is solved using Bertsekas auction algorithm [11] and the assignment weights are calculated according to:

$$a(c,n) = \begin{cases} 10 & \text{if } \delta(c,n) < 0.1 \\ \frac{1}{\delta(c,n)} & \text{if } 0.1 \leq \delta(c,n) \leq 20 \\ -10 & \text{if } \delta(c,n) > 20 \end{cases} \quad (4)$$

where c is the index of the cell in the previous image, n is the index of the local maximum and $\delta(c,n)$ is the spatial distance between the local maxima n and the centroid of cell c in the previous image. All the details of this assignment problem is not shown here, a similar assignment problem is shown in more detail in Sect. 4.1. The intensity of $\tilde{I}_{multi}(x,y)$ (with intensities ranging from 0 to 255) in the selected points are increased with β_n:

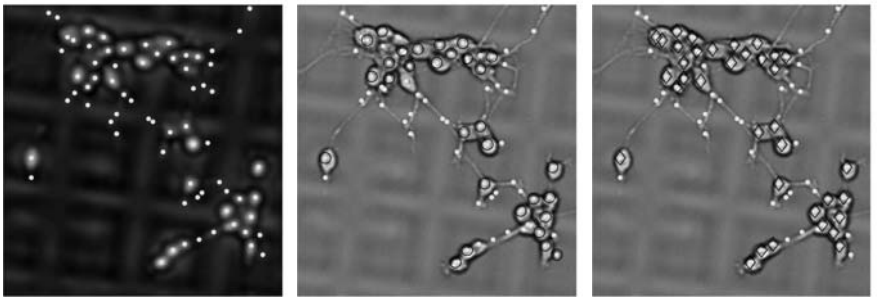
$$\beta_n = \begin{cases} \frac{1}{\delta(c_n,n)} \cdot 50 & \text{if } \delta(c_n,n) \geq 1 \\ 50 & \text{if } \delta(c_n,n) < 1 \end{cases} \quad (5)$$

where $\delta(c_n,n)$ is the spatial distance between the local maximum n and the centroid of the cell in the previous image that was assigned to the local maximum n. Figure 2(b) shows the original image with all local maxima marked with dots and the local maxima where the intensity of $\tilde{I}_{multi}(x,y)$ is increased are shown with circles. The intensities of the modified $\tilde{I}_{multi}(x,y)$ in all the local maxima were then thresholded using Otsu's method [12], so that the most likely cell positions were selected, see Fig. 2(c).

In the initial frame we use all local maxima in the masked $\tilde{I}_{multi}(x,y)$ as starting points for the dynamic programming. The result is then manually corrected where needed.

3.4 Cell Contour Segmentation

The selected points are sorted based on their intensity in $\tilde{I}_{multi}(x,y)$. Starting with the point with the highest intensity, the contour of the potential cell is



(a) (b) (c)

Fig. 2. (a) Smoothed $I_{multi}(x, y)$ with all local maxima (maxima in background removed) marked with a point. (b) Original image with all local maxima marked with points and the points were the intensity of $\tilde{I}_{multi}(x, y)$ is increased marked with circles. (c) Original image with all local maxima marked with points and the points chosen to be starting points for segmentation marked with diamonds

segmented using dynamic programming, see [13] for details. To be considered a valid cell, the segmented region has to be larger than 55 pixels and not extend more than 25% of its area into the region classified as the background. If the segmented region overlaps with a region previously segmented in that image, the overlap is removed from the last segmented region since it is assumed that the points previously used for segmentation (higher intensity in $\tilde{I}_{multi}(x, y)$) give a more accurate segmentation result.

4 Cell Tracking

In this section the different parts of the tracking system are described. The tracking step consists of three parts: (A) solve the assignment problem, (B) examine unassigned tracks, (C) examine unassigned objects and update results.

4.1 The Assignment Problem

The asymmetric assignment problem, i.e. the problem of matching m objects with n tracks in an optimal way when $n \neq m$, can be formulated as:

$$\max \sum_{i=0}^{n} \sum_{j=0}^{m} a_{ij} \tau_{ij} \quad \forall (i, j) \in \Gamma \tag{6}$$

where a_{ij} is a assignment weight (i.e. the benefit of matching track i with object j), Γ is the set of pairs (i, j) that can be matched and τ_{ij} is defined as:

$$\tau_{ij} = \begin{cases} 1 \text{ if track } i \text{ is assigned to object } j \\ 0 \text{ otherwise} \end{cases}$$

Track i=0 and object j=0 are called the dummy track and the dummy object, respectively. The assignment is a one-to-one assignment, except for the dummy track and the dummy object which are allowed multiple assignments. Equation 6 was solved using the modified auction algorithm, see [11] for details. The assignment weights, a_{ij} used in (6), are calculated according to:

$$a_{ij} = \left(\frac{1}{\delta_{ij}} + \phi_{ij} + \psi_{ij} \right) \cdot w_i \tag{7}$$

where δ_{ij}, ϕ_{ij} and ψ_{ij} are defined below, and w_i is the weight for track i, initially $w_i = 1$.

The distance measure, δ_{ij}, is a function depending on the distance between object j and the last known object in track i:

$$\delta_{i,j} = C_1 \cdot \left(\sqrt{(x_j(t) - x_i(t{+}1))^2 + (y_j(t) - y_i(t{+}1))^2} + 0.1 \right) \tag{8}$$

where $(x_j(t), y_j(t))$ is the centroid of object j at time t, and $(x_i(t{+}1), y_i(t{+}1))$ is the centroid of the object belonging to track i in the previous image (at time $t{+}1$ since the tracking is done backwards). C_1 is a constant and the 0.1 term is added to avoid the denominator in (7) becoming zero.

The function ϕ_{ij} is a measure of the correlation between object j and the last known object in track i. Assume that I_j is the smallest possible image containing object j in frame t, with the background set to zero. Then:

$$I_{corrj}(x,y) = \frac{C_2 \cdot \sum_k \sum_l I_j(s_x{+}k, s_y{+}l, t) \cdot I(x{+}k, y{+}l, t{+}1)}{\sqrt{\sum_k \sum_l I_j(s_x{+}k, s_y{+}l, t)^2} \cdot \sqrt{\sum_k \sum_l I(x{+}k, y{+}l, t{+}1)^2}}$$
$$\text{for } k = -s_x{+}1, ..., s_x \text{ and } l = -s_y{+}1, ..., s_y \tag{9}$$

where C_2 is a constant and s_x is the number of rows in I_j divided by 2 and s_y is the number of columns in I_j divided by 2. ϕ_{ij} can then be calculated:

$$\phi_{ij} = \max\left(I_{corrj}(x,y)\right) \qquad \forall (x,y) \in \Omega \tag{10}$$

where Ω is the set of pixels (x, y) belonging to the last known cell in track i in a previous frame.

The last function ψ_{ij} depends on the difference in area of object j and the last know object in track i:

$$\psi_{ij}^0 = C_3 \cdot \frac{A_i(t{+}1)}{|A_i(t{+}1) - A_j(t) + 1|} \tag{11}$$

where $A_j(t)$ is the area of object j in frame t and $A_i(t{+}1)$ is the area of track i in frame $t{+}1$. C_3 is a constant. To avoid the risk of the denominator becoming zero, one is added. Also the contribution from ψ_{ij}^0 is limited by:

$$\psi_{ij} = \begin{cases} \psi_{ij}^0 \text{ if } \psi_{ij}^0 \leq K \\ K \text{ if } \psi_{ij}^0 > K \end{cases} \tag{12}$$

where K is a constant.

Constants C_1, C_2, C_3 and K should be chosen so that two, not so similar, cells in consecutive frames with centroids closer than about half a typical cell radius ($r_{cell} \sim 7$ pixels) get a higher assignment weight than two cells further apart but very alike. In these investigations suitable values for C_1, C_2, C_3 and K were found to be 0.03, 5, 1 and 3 respectively. These values give that the maximum value of $\phi_{ij} + \psi_{ij} = C_2 + K = 8$, which means that δ_{ij} will give the largest contribution to the assignment weight for distances between two cells up to ~ 4 pixels ($\frac{1}{0.03 \cdot 8} - 0.1$).

4.2 Unassigned Tracks

There are several reasons as to why a track might not get assigned to a real object (i.e. are assigned to the dummy object):

1. Cells merge (or rather, one cell splits, since tracking is done backwards)
2. Cells disappear outside the image boundaries
3. Cells disappear into clusters
4. Errors in the segmentation

In the first three cases, the track should be terminated, while in the last case the segmentation should be corrected.

Cells Merge. Previous investigations have shown that most merged and merging cells have a characteristic appearance. Therefore there are three requirements for two cells, α and β, in frame t+1 to be considered to have merged into cell γ in frame t. First, the sum of the areas of cells α and β should not differ more than 20% compared to the area of cell γ. Second, the mean intensity of the interior of cell γ has to be at least twice the mean intensity of the contour of cell γ. Last, cell γ must be fairly circular. Therefore, the ratio between the major and minor axis of an ellipse with the same normalized second central moments as cell γ has to be less than 1.5.

Cells Disappear Outside the Image Boundaries. If the center of the last known cell in track i is closer than 11 pixels to any of the image borders, it is considered very likely that the cell has moved out of the image in the current frame. However, the track is not terminated at once. Assignment weights are calculated according to (7) for the track in the next two frames, but the track weight, w_i, is decreased, making the track less attractive. If no cell is associated with the track for three frames, the track is terminated.

Cells Disappears Into Clusters. There is currently no way to resolve cells that are above or below the image plane. A vanished cell is considered to have disappeared into a cell cluster if the track is not near an image border, and if no segmentation error or merging of cells could be detected. Just as in the case with cells disappearing from within the image borders, the tracks are not immediately terminated, but remembered for another two frames and the track weight is decreased.

Errors in the Segmentation. In situations when two potentially merging cells fulfill the requirement made on the cell sizes, but fail on either one of the other two requirements (see Sect. 4.2) the segmentation might have failed to split two neighboring cells. Assume that cell $\alpha_i(t+1)$ is the cell belonging to track i in the previous image, $\gamma_k(t)$ is the potentially incorrectly segmented object belonging to track k in the current image, and $\beta_k(t+1)$ is the cell assigned to track k in the previous image. The centroids of $\alpha_i(t+1)$ and $\beta_k(t+1)$ are then the starting points for the dynamic programming segmentation. The cells found through this segmentation are considered valid when they have a size > 55 pixels, a mean intensity of the cell contour < 100, and a mean intensity of the interior of the cell > 130, otherwise the new segmentation result is rejected and the unassigned track is considered to belong to case 3, i.e. the cell is assumed to have joined a cluster.

Another potential error in the segmentation is that a cell is missed in the segmentation. However, based on the segmentation procedure, this means that the cell is either small (< 55 pixels) or very unclear (not blob-like). Unfortunately the problem of detecting such cells is not yet solved, so in that case the cell would be assumed to have joined a cluster.

4.3 Unassigned Objects

Unassigned objects are divided into three groups:

1. Cells appearing from outside the image boundaries
2. Cells previously hidden in a cell cluster
3. False cells

If the center of an unassigned cell is closer than 11 pixels from any of the image borders, it is assumed to have appeared from outside the image and a new track is started. A new track is also started for objects assigned to group 2, i.e. objects appearing in the interior of an image, and fulfilling the criteria for valid cells (see Sect. 4.2). If an unassigned object does not belong to either group 1 or 2, it is likely that it is not a cell and therefore belongs to group 3. A new track is started but given a low track weight. However, if no cell can be associated with the track in the next frame, it is deleted from both frames.

5 Result

To evaluate the automatic segmentation and tracking system, nine image sequences were segmented and tracked throughout each sequence. The tracks for certain cells were then visually inspected and manually corrected where found necessary. The tracks that were manually corrected were chosen based on the result of the antibody staining, i.e. only classified cells were inspected. Since the tracks were not chosen based on ease-of-tracking or because cells remained within the field of view for the complete sequence they represent a fairly random selection of cells available in the final image of each time-lapse experiment. In total, 87 cell tracks were selected for manual correction and on average

they were present in 150 frames in their sequence before vanishing. The manually corrected sequences were compared with the corresponding automatically tracked sequences. In total only 18 out of the 87 (=20%) cell tracks were correctly segmented and tracked automatically through the whole image sequence. However, if we instead consider each cell-to-cell association in two consecutive frames as a separate event (the total number of cell-to-cell associations were 12944), it was found that 95% of the separate cell-to-cell associations were correct.

6 Discussion

By manual inspection of the images it is clear that many, although not all, of the ambiguities in a single image can be resolved by looking at the images before and/or after the current image. It therefore seems clear that information about the previous segmentation results should be used when segmenting the new image, but the difficult question is how much impact that information should have on the new segmentation result. Since the number of cells appearing or disappearing between two consecutive frames is significant, we can not let the influence of the previous segmentation result be too large because that would undoubtedly lead to a lot of errors. By letting the previous segmentation result aid in the decision of what blobs to be segmented we favor the blobs found close to where there were cells previously. This will keep cell tracks of slow moving cells intact even in images where the cell might be difficult to distinguish, while still allowing for blob-like cells appearing into the image to be detected. Although this procedure is working well, an improvement might be to use segmentation results from the next frame in the sequence as well, by going back and fourth in the sequence. This of course will be a quite time consuming procedure, since it requires each frame to be segmented at least twice.

The aim of this project is to use the data acquired from the sequence (cell migration, tendency to split, appearance etc) to gain a better understanding of the processes behind the differentiation of neural stem/progenitor cells into neurons, astrocytes or oligodendrocytes. Since information about the identity of the cells is only given in the last frame of the sequence it is of high importance that the cells tracks are not mixed up along the way in the sequence. Although our error rate is low regarding cell-to-cell associations, we saw that the number of tracks where the cell was correctly segmented and tracked through the whole sequence is too low to be able to use the results directly for e.g. migration modelling of the different cell types. A manual correction of the result is therefore necessary. However, manually adjusting the tracking or the segmentation in about 7-8 images on average per cell track (each cell track exist in 150 frames on average) seems reasonable and not too time-consuming. Therefore we conclude that our current system can be used in its present state for examining the very large amount of data needed for the neural stem/progenitor investigations.

Acknowledgments

This project was partly funded by the Swedish Foundation for Strategic Research (SSF) under the VISIT program, No. A30-960626 and the Swedish Research Council (VR), No. 621-2001-3531.

References

1. Zimmer, C., Labruyere, E., Meas-Yedid, V., Guillen, N., Olivo-Marin, J.C.: Segmentation and tracking of migrating cells in videomicroscopy with parametric active contours: a tool for cell-based drug testing. IEEE Transactions on Medical Imaging **21** (2002) 1212–1221
2. Debeir, O., Camby, I., Kiss, R., Van Ham, P., Decaestecker, C.: A model-based approach for automated in vitro cell tracking and chemotaxis analyses. Cytometry **60A** (2004) 29–40
3. Mukherjee, D., Ray, N., Acton, S.: Level set analysis for leukocyte detection and tracking. IEEE Transactions on Image Processing **13** (2004) 562–572
4. Delingette, H., Montagnat, J.: Shape and topology constraints on parametric active contours. Computer Vision and Image Understanding **83** (2001) 140–171
5. Kirubarajan, T., Bar-Shalom, Y., Pattipati, K.: Multiassignment for tracking a large number of overlapping objects. IEEE Transactions on Aerospace and Electronic Systems **37** (2001) 2–21
6. Gustavsson, T., Althoff, K., Degerman, J., Olsson, T., Thoreson, A.C., Thorlin, T., Eriksson, P.: Time-lapse microscopy and image processing for stem cell research modeling cell migration. In: Medical Imaging 2003: Image Processing. Volume 5032. (2003) 1–15
7. Wu, K., Gauthier, D., Levine, M.: Live cell image segmentation. IEEE Transactions on Biomedical Engineering **42** (1995) 1–12
8. Kittler, J., Illingworth, J.: Minimun error thresholding. Pattern Recognition **19** (1986) 41–47
9. ter Haar Romeny, B.: Front-End Vision & Multi-Scale Image Analysis. Kluwer Academic Publishers (2003)
10. Lindeberg, T.: Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attentions. International Journal of Computer Vision **11** (1993) 283–318
11. Bertsekas, D.: The auction algorithm for assignment and other network flow problems: A tutorial. Interfaces **20** (1990) 133–149
12. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics **SMC-9** (1979) 62–66
13. Althoff, K., Degerman, J., Gustavsson, T.: Tracking neural stem cells in time-lapse microscopy image sequences. In: Medical Imaging 2005: Image Processing. (2005)