PLOS ONE

RESEARCH ARTICLE

Three-dimensional GPU-accelerated active contours for automated localization of cells in large images

Mahsa Lotfollahi[1], **Sebastian Berisha**[1], **Leila Saadatifard**[1], **Laura Montier**[2], **Jokūbas Žiburkus**[2], **David Mayerich**[1]*

1 Department of Electrical and Computer engineering, University of Houston, Houston, TX, United States of America, **2** Department of Biology and Biochemistry, University of Houston, TX, United States of America

* mayerich@uh.edu

Check for updates

OPEN ACCESS

Citation: Lotfollahi M, Berisha S, Saadatifard L, Montier L, Žiburkus J, Mayerich D (2019) Three-dimensional GPU-accelerated active contours for automated localization of cells in large images. PLoS ONE 14(6): e0215843. https://doi.org/10.1371/journal.pone.0215843

Editor: Yuanquan Wang, Beijing University of Technology, CHINA

Received: November 21, 2018

Accepted: April 9, 2019

Published: June 7, 2019

Copyright: © 2019 Lotfollahi et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Data Availability Statement: The data and code underlying this study have been deposited to Figshare, and may accessed via https://figshare.com/projects/hypersnakuscules/63155. This project is still under development; the latest version of the software/datasets are available on the STIM Laboratory website at http://stim.ee.uh.edu/resources/software/ and http://stim.ee.uh.edu/resources/data-sets/, respectively. External data sets used for benchmarking are publicly distributed through the Cell Tracking Challenge website (http://celltrackingchallenge.net). All other relevant data

Abstract

Cell segmentation in microscopy is a challenging problem, since cells are often asymmetric and densely packed. Successful cell segmentation algorithms rely identifying seed points, and are highly sensitive to variablility in cell size. In this paper, we present an efficient and highly parallel formulation for symmetric three-dimensional contour evolution that extends previous work on fast two-dimensional snakes. We provide a formulation for optimization on 3D images, as well as a strategy for accelerating computation on consumer graphics hardware. The proposed software takes advantage of Monte-Carlo sampling schemes in order to speed up convergence and reduce thread divergence. Experimental results show that this method provides superior performance for large 2D and 3D cell localization tasks when compared to existing methods on large 3D brain images.

Introduction

Quantifying the size and distribution of cell nuclei in optical images is critical to understanding the underlying tissue structure [1] and organization [2, 3]. Segmentation is crucial to this analysis, by providing quantitative data that pathologists can use to characterize diseases and evaluate their progression [4]. Since manual analysis of microscopy images is time consuming and labor intensive, automated cell localization is essential for detecting and segmenting cells in massive images. Microscopy images exhibit a large degree of variability and complexity, due to large numbers of overlapping cells and variations in cell types and stages of cell division, imaging systems, and staining protocols. In order to deal with this complexity, a large number of algorithms have been proposed [5, 6]. Most current algorithms use basic techniques combined with complicated pipelines to overcome those challenges. These methods include thresholding [7–9], feature extraction [10, 11], classification [12], c-means [8] and k-means [13] clustering, region growing [14–16], and deformable models [17–19].

Recently, learning based approaches using artificial neural networks (ANN) and convolutional neural networks (CNN) have gained increased attention. These methods rely on example data to train a machine learning algorithm to identify boundary pixels [20] or directly

are within the paper and its Supporting Information files.

Funding: This work was funded in part by the Cancer Prevention and Research Institute of Texas (CPRIT) #RR140013 (DM), as well as the National Institutes of Health / National Library of Medicine (NLM) #4 R00 LM011390 (DM) and #T15LM007093 (DM and SB).

Competing interests: The authors have declared that no competing interests exist.

perform binary segmentation [21]. In general, most pipelines include prepossessing, finding cell bounding boxes, extracting either spatial or frequency-based features [2, 22–24] or using several convolution layers followed by max-pooling [25, 26], and finally classifying the image. In these approaches, the training phase is time consuming and requires massive amounts of labeled data [27].

Most current algorithms focus on two-dimensional data, such as histology slides, and utilize a variety of techniques to deal with specific tissue types, stains, and labels. For example, deep CNNs have been used for overlapping clumps in Pap smear images [28], and support vector machines (SVMs) have been employed to segment epithelial cells [29] and skeletal muscle [30]. Finally, active contours have been shown to be effective for cell nuclei [31].

The major limitation of histology slices is that they are limited to 2D sampling. Although histological assesments convey some of the structure and morphology of the tissue, they do not provide proper insights into the 3D layout of cells. In addition, 3D images provide much better separability when cells are overlapping or hidden in the corresponding 2D images.

To date, several software solutions are available for specialized cell segmentation on 3D images. FARSIGHT [32] uses graph cuts and multi-scale Laplacian of Gaussian filters to detect cell seed points. Region growing is then used based on local-maximum clustering. MINS [33] performs blob detection by smoothing the image with Gaussian kernels at different scales and computing eigenvalues of the Hessian matrix at each pixel from these smoothed images. It then thresholds the respective eigenvalues to obtain a mask of nuclei and a connected component analysis assigns a unique ID to each nucleus. The 3D object counter plugin for ImageJ [34] is a simple 3D cell counter which uses a user-specified intensity threshold to separate foreground and background, resulting in an over-segmented image. Since fundamental thresholding is not robust, adaptive and iterative thresholding on smoothed 3D images can also be used [35, 36]. In almost all cases, cell localization is a necessary initial step.

One method for addressing large-scale localization relies on simple active contours, such as snakuscules [37, 38], which are fast to evaluate and rely on very few user-specified input parameters. However, a three-dimensional application of this algorithm has not been derived. In addition, the sampling required to evolve a primitive active contour is computationally intractable for images containing thousands of cells.

Approach

Most deformable models transform an image segmentation task to an optimization problem. An energy function is defined based on the image content and desired behavior of a curve. Snakuscules, introduced in [37], are region-based snakes optimized for identifying approximately circular features. In this section, we will describe the previously published snakuscule algorithm as well as generalize the mathematics to three-dimensional images.

Snakuscules

Snakuscules are active contours optimized for fast convergence around circular image features. Their fast evaluation time allows the initialization of many contours that cover an entire image, allowing detection of blob-like features without manual initialization.

A Snakuscule is defined by a pair of concentric disks parameterized by two points **p** and **q** (Fig 1a). The optimization attempts to minimize an energy function measuring the contrast between the inner disk and annulus in order to completely surround a bright round object

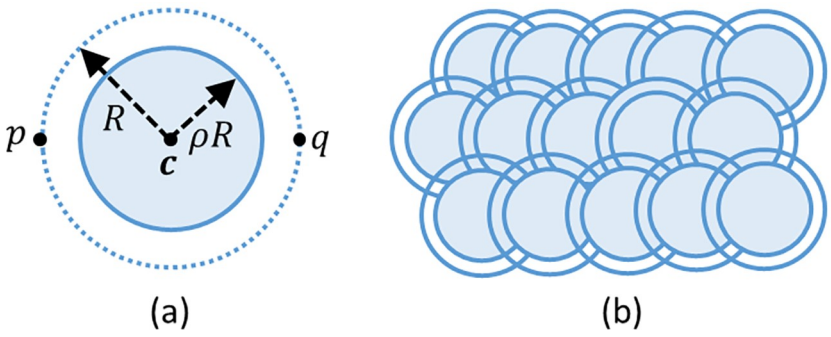


Fig 1. (a) A snakuscule is defined by two points **p** and **q**. (b) Initial configuration of multitude snakuscules congregated together at a distance $\sqrt{1.5}R$.

https://doi.org/10.1371/journal.pone.0215843.g001

with a circular curve. The energy function is defined to balance the weighted inner area against the weighted outer area of the curve:

$$\begin{aligned} \mathbf{E}(\mathbf{p},\mathbf{q}) &= \iint_{\rho R<\|\mathbf{x}-\mathbf{c}\|<R} I(\mathbf{x})d\mathbf{x} - \iint_{\|\mathbf{x}-\mathbf{c}\|<\rho R} I(\mathbf{x})d\mathbf{x} \\ R &= \frac{1}{2}\|\mathbf{p}-\mathbf{q}\| \\ \mathbf{c} &= \frac{1}{2}(\mathbf{p}+\mathbf{q}) \end{aligned} \tag{1}$$

where $I(\mathbf{x})$ is a two-dimensional image, $\mathbf{x} = [x_1, x_2]^T$ is an image coordinate, R is the radius of the snake, and $\mathbf{c}$ is its center. The value $\rho = \frac{1}{\sqrt{2}}$, derived previously for the two-dimensional case [37], enforces the equal area for both inner disk and outer annulus.

One snakuscule can find and segment one light blob in the image. To catch all interesting features, many initial contours are specified to cover the image (Fig 1b). The contours are then evolved, and trivial contours that do not converge to image features are eliminated (Fig 2).

3D snakuscules

We first propose a mathematical framework for evolving snakuscules in 3D by moving and expanding/contracting an initial 3D contour to fit cell nuclei. This generalization makes it viable to extend similar contours to higher-dimensional or hyperspectral data (ex. hypersnakuscules). The 3D snakuscule is based on a pair of concentric spheres that are parameterized by two points $\mathbf{p} = [p_x, p_y, p_z]^T$ and $\mathbf{q} = [q_x, q_y, q_z]^T$ (Fig 3a). The optimization process minimizes a local energy function, which favors high contrast between weighted inner and outer volumes. Contours move and evolve within the spatial domain of an image to minimize the contrast energy function (Eq 2).

$$\mathbf{E}(\mathbf{p},\mathbf{q}) = \iiint_{\rho R<\|\mathbf{x}-\mathbf{c}\|<R} I(\mathbf{x})d\mathbf{x} - \iiint_{\|\mathbf{x}-\mathbf{c}\|<\rho R} I(\mathbf{x})d\mathbf{x} \tag{2}$$

This optimization leads the contour toward a bright spherical object on a dark background. To ensure the snake does not move in uniform regions with constant intensity where $\forall \mathbf{x} \in \mathbf{R}^3$:

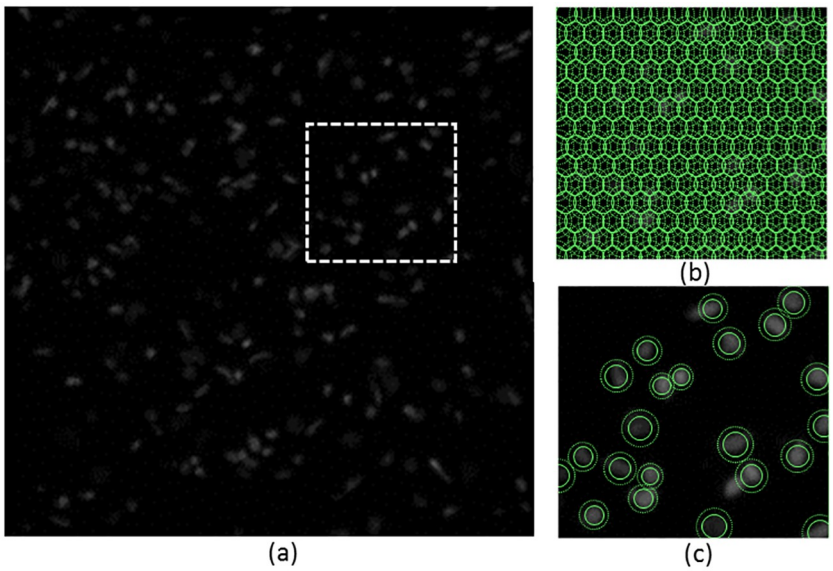


Fig 2. (a) A DAPI stained brain tissue slice. (b) The initial configuration and (c) final configuration of snakuscules on a zoomed region.

https://doi.org/10.1371/journal.pone.0215843.g002

$\mathbf{I}(\mathbf{x}) = \mathbf{I_0}$, the energy is defined using two sub-terms that cancel each other out; therefore, $\rho = \frac{1}{\sqrt[3]{2}}$.

This prevents the contour from sliding when the surrounding gradient is zero [37]. We illustrate energy minimization for a generic model of a light blob $I(r, \theta) = 1 + \mathrm{sgn}(r_0 - r)$,

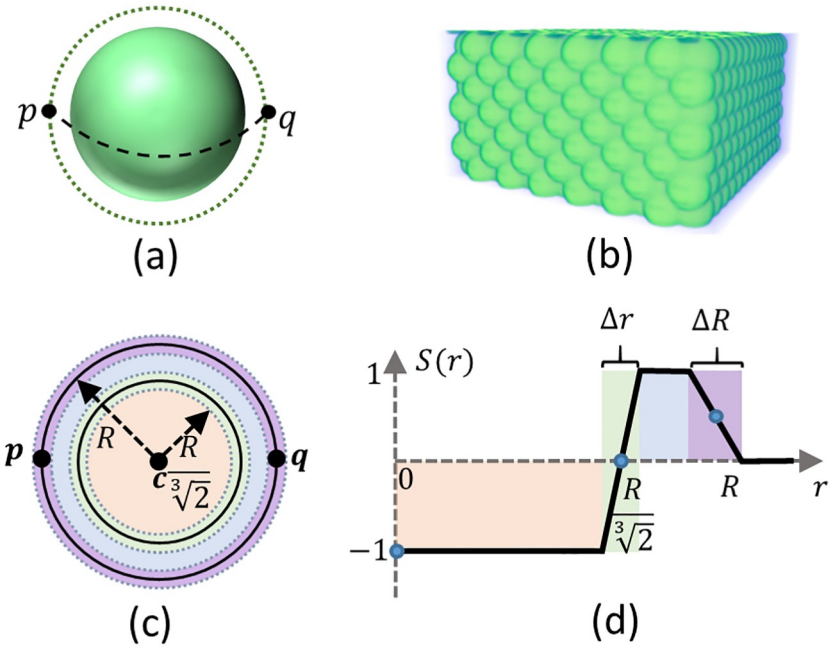


Fig 3. (a) A 3D snakuscule defined by two points **p** and **q**; to simplify computations 3D snakusules identifier points are considered in the same y and z levels. (b) The initial configuration consists of every two neighboring 3D contours located at a distance $\sqrt{1.5}R$ apart. (c) The middle section of the 3D snakuscule with four distinct regions are shown in different colors. (d) The weight function assigns a weight to any portion of the 3D snakuscule shown in (c).

https://doi.org/10.1371/journal.pone.0215843.g003

where sgn function is defined as:

$$\mathrm{sgn}\,(x) := \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases} \tag{3}$$

It creates a sphere of radius r_0 in a black background. When the contour is concentric within the blob, the resulting energy is given by:

$$\hat{\mathbf{E}} = \begin{cases} 0, & R < r_0 \\ -\frac{8}{3}\pi(R^3 - r_0^3), & \frac{R}{\sqrt[3]{2}} < r_0 < R \\ -\frac{8}{3}\pi r_0^3, & R \geq \sqrt[3]{2}\mathrm{r}_0 \end{cases} \tag{4}$$

The energy achieves an optimal value for any contour equal or larger than the blob, so there is no unique optimal contour. To ensure that the volume occupied by the contour is also minimized, a normalization term α is used to reformulate the energy function:

$$\bar{\mathbf{E}}(R, \alpha) = \frac{\hat{\mathbf{E}}}{R^\alpha} \tag{5}$$

where $\alpha > 0$ to apply a penalty when the contour becomes larger than the blob. To balance the expansion and contraction speed when the contour is approaching the blob size, we force the energy gradient to be symmetric as the optimal value is approached:

$$\lim_{R \to \sqrt[3]{2}-\epsilon} \frac{\delta \bar{\mathbf{E}}(R, \alpha)}{\delta R} = \lim_{R \to \sqrt[3]{2}+\epsilon} -\frac{\delta \bar{\mathbf{E}}(R, \alpha)}{\delta R}$$

which results in a normalization value $\alpha = 3$. Fig 4 depicts the energy function with respect to

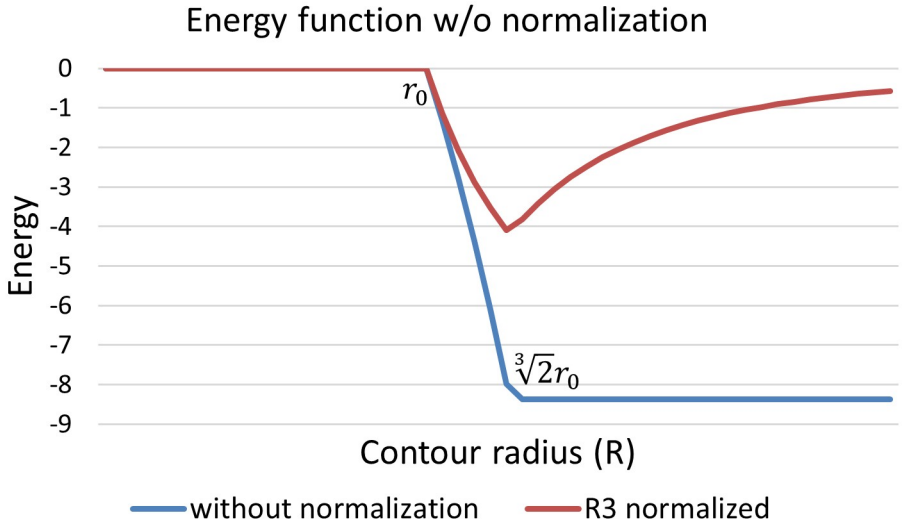


Fig 4. Energy changes of the 3D snake with and without normalization term w.r.t its radius. Energy function with normalization term has a single local minimum when the snake fit the blob.

https://doi.org/10.1371/journal.pone.0215843.g004

the contour radius with and without normalization. Note that this normalization term can be optimized as desired for objects that are not binary indicator functions (ex. Gaussian kernels).

A discrete formulation of the energy function is generated by substituting summation for integration in the pixel domain. The final discrete energy function is given by:

$$\mathbf{E}(\mathbf{p},\mathbf{q}) = \frac{1}{\|\mathbf{p}-\mathbf{q}\|^3}\sum_{\mathbf{k}\in\mathbf{K}} S(r)I(\mathbf{k}) \tag{6}$$

where $\mathbf{K}$ is the set of all pixels within $R + \frac{1}{2}\Delta R$ of the 3D snake center, $r = |\mathbf{k} - \mathbf{c}|$, and $S(r)$ is a differentiable weight function (Fig 3d), so that $\int_0^\infty S(r)r^2dr = 0$. The 3D snake is composed of four different regions (Fig 3c); two dynamic and two fixed regions. During evolution, the entire footprint becomes smaller or larger while ΔR and Δr remain unchanged:

$$\Delta r = \Delta R/\sqrt[3]{2}$$

To simplify calculations, the two identifier points $\mathbf{p}$ and $\mathbf{q}$ are considered to be in the same line along both the y and z directions ($p_y = q_y$ and $p_z = q_z$). The energy function can be rewritten as:

$$\mathbf{E}(\mathbf{p},\mathbf{q}) = \frac{1}{(q_x - p_x)^3}\sum_{\mathbf{k}\in\mathbf{K}} S(r)I(\mathbf{k}) \tag{7}$$

3D contour evolution

The 3D snakuscule evolves by movements of $\mathbf{p}$ and $\mathbf{q}$ in the opposite direction of $\nabla\mathbf{E}$ to minimize the energy function using gradient descent. Therefore, partial derivatives of the energy function $\mathbf{E}$ with respect to the identifier points $\mathbf{p}$ and $\mathbf{q}$ are required:

$$\frac{\partial \mathbf{E}}{\partial p_x} = \gamma\left[\frac{3}{q_x - p_x}\sum S(r)I(\mathbf{k}) + \sum\frac{\partial S}{\partial p_x}I(\mathbf{k})\right] \tag{8}$$

$$\frac{\partial \mathbf{E}}{\partial q_x} = \gamma\left[\frac{-3}{q_x - p_x}\sum S(r)I(\mathbf{k}) + \sum\frac{\partial S}{\partial q_x}I(\mathbf{k})\right] \tag{9}$$

$$\frac{\partial \mathbf{E}}{\partial q_y} = \frac{\partial \mathbf{E}}{\partial p_y} = \gamma\sum\frac{\partial S}{\partial p_y}I(\mathbf{k}) \tag{10}$$

$$\frac{\partial \mathbf{E}}{\partial q_z} = \frac{\partial \mathbf{E}}{\partial p_z} = \gamma\sum\frac{\partial S}{\partial p_z}I(\mathbf{k}) \tag{11}$$

where

$$\gamma = \frac{1}{(q_x - p_x)^3} \tag{12}$$

We minimize the energy (Eq 7) using gradient descent to update the position of the identifier points. Each point $\mathbf{k} \in \mathbf{K}$ applies a force to $\mathbf{p}$ and $\mathbf{q}$ that dictate its motion over time:

$$\frac{d\mathbf{p}}{dt} = -\sum_{\mathbf{k}\in\mathbf{K}} \frac{\partial \mathbf{E}(\mathbf{k})}{\partial \mathbf{p}} \tag{13}$$

$$\frac{d\mathbf{q}}{dt} = -\sum_{\mathbf{k}\in\mathbf{K}} \frac{\partial \mathbf{E}(\mathbf{k})}{\partial \mathbf{q}} \tag{14}$$

$$\mathbf{p}_{n+1} = \mathbf{p}_n + \epsilon \frac{d\mathbf{p}}{dt} \tag{15}$$

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \epsilon \frac{d\mathbf{q}}{dt} \tag{16}$$

where $\epsilon = \frac{\epsilon_0}{\sqrt{n}}$ is learning rate, ϵ_0 is constant and n is the iteration number.

Parallelizing the process

Regarding cell localization and counting, 3D snakuscules can be initially placed on the 3D image in a lattice (Fig 3b) similar to the 2D case (Fig 1b). They evolve independently to segment a nearby spherical structure (blob). However, the higher dimensional integration results in excessive computing time, making a serial implementation impractical for large high resolution images.

Since the evolution of each contour is completely independent from the others, this process is highly data-parallel and an ideal application for graphic processing units (GPUs). GPUs consist of a large number of parallel processors that can be used for general purpose parallel computing to improve the performance of algorithms that are highly data parallel and can be split into a large number of independent threads. A GPU has a local single-instruction on multiple data (SIMD) architecture, making execution of the same program on multiple values extremely efficient. The set of instructions applied on each element is called a kernel [39]. We define our evolutionary instructions as a GPU kernel that can be executed for thousands of snakes in parallel.

For instance, snakuscules are run on various pieces of a 2D DAPI stained rat brain tissue image. The image is a whole rat brain slice with resolution of 350 nm/pixel. The initial and final snakuscules configurations for a 1000 × 1000 image are shown (Fig 2). Fig 5 illustrates performance of GPU implementation in comparison to an optimized CPU version. By increasing number of contours, image size, CPU execution time increases significantly, quickly becoming impractical.

The 3D snakuscule is computationally more expensive because of integration over a 3D space using a uniform grid. Therefore, parallel computing using a GPU is employed to assign one contour evolution to one GPU thread.

Monte-Carlo integration

In order to further accelerate contour evolution, Monte-Carlo (MC) integration is used. It estimates the integral values using a uniform distribution of randomized samples. In the 2D case, samples are chosen from a uniform distribution inside of a circle with radius $(R + \Delta R/2)$. If r and θ are random numbers in $[0, 1]$ and $[0, 2\pi)$ respectively; a uniform set of points within the

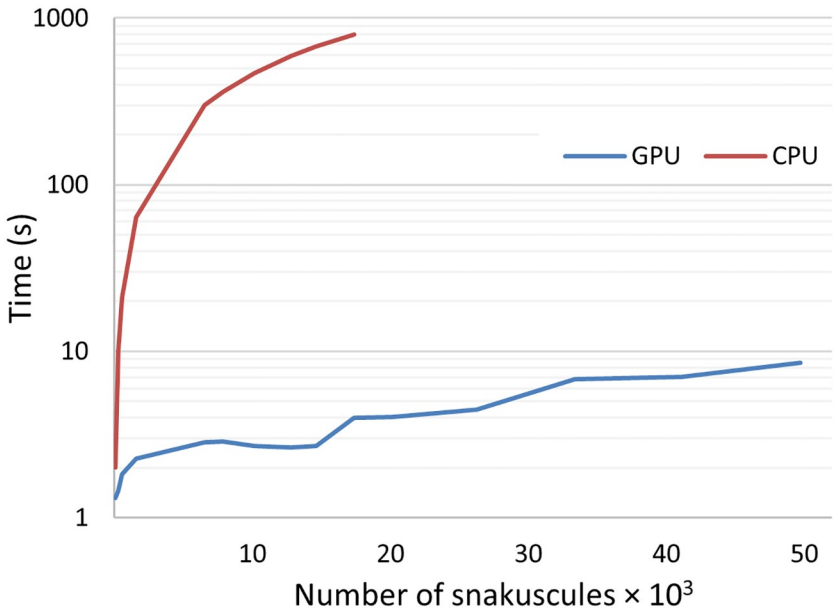


Fig 5. Execution time for the method implemented on CPU and GPU. Time axis is plotted on a logarithmic scale.

https://doi.org/10.1371/journal.pone.0215843.g005

circle with radius r are computed:

$$x = \sqrt{r}\cos\theta$$
$$y = \sqrt{r}\sin\theta$$

MC integration is selected because it provides two advantages over uniform sampling:

- Convergence is significantly faster for higher-dimensional data sets, providing an error of $\frac{1}{N}$, regardless of the number of dimensions.
- The use of MC sampling allows us to specify a constant number of samples per snake, minimizing branch divergence in the GPU-based SIMD algorithm.

One constraint of MC integration is that we are relying on an underlying assumption that the integral is well-behaved (smooth). Given that we expect cell nuclei to be relatively consistent in size, this assumption is well founded. However, it can be mathematically enforced using a low-pass filter that forces the image to be smooth.

For 3D images, uniform sampling is done within a sphere with radius $(R + \Delta R/2)$. Execution time using Monte-Carlo sampling in comparison with the original integration for different number of snakes on the 2D (Fig 6) and the 3D (Fig 7) images shows significant improvement. As expected, a significantly greater acceleration can be seen in the 3D algorithm, with an $\approx 4X$ gain in performance on average.

Parallel contour evaluation

In order to improve the GPU efficiency by utilizing more GPU resources, we further parallelize each 3D contour. We instead assign each block to one contour so that threads in that block are responsible for smaller parts of Eqs 13 and 14. For each snake, if MC integration selects N random samples and the CUDA kernel is launched with T threads (the maximum number of threads per block), each thread calculates a portion of the energy (Eqs 8–11) corresponding to

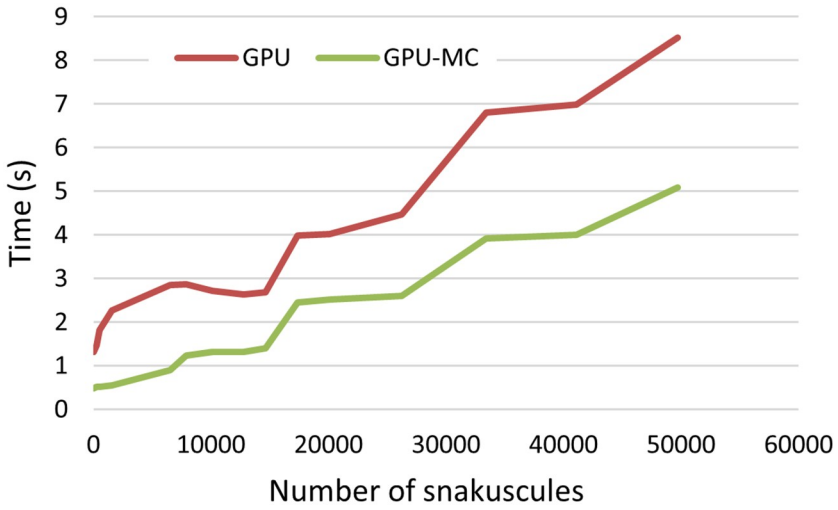


Fig 6. Execution time of snakuscules (2D) on GPU with and without Monte-Carlo sampling.

https://doi.org/10.1371/journal.pone.0215843.g006

N/T spatial locations within the contour. The results are stored in shared memory and combined (Eqs 13 and 14) to calculate the final contour at each iteration. This allows employing more GPU threads to cooperatively walk through a snake evolution process (Fig 8).

Results and discussion

In order to find all sphere-like objects in an 3D-image without user interaction, the image is covered by close initial 3D contours. The 3D contours update their current configurations by individually optimizing their energies. Contour evolution is stopped when either (a) they meet maximum number of iterations or (b) they converge where contours reach the minima and the moving step is much less than one pixel such as 0.001.Overlapping snakes, as defined by $\|\mathbf{c}' - \mathbf{c}''\| < \max(\mathbf{R}', \mathbf{R}'')/\sqrt[3]{2}$, undergo a competition with the lower energy snake surviving. 3D snakuscules with energy greater than a threshold ($\mathbf{E}_0$) are also removed.

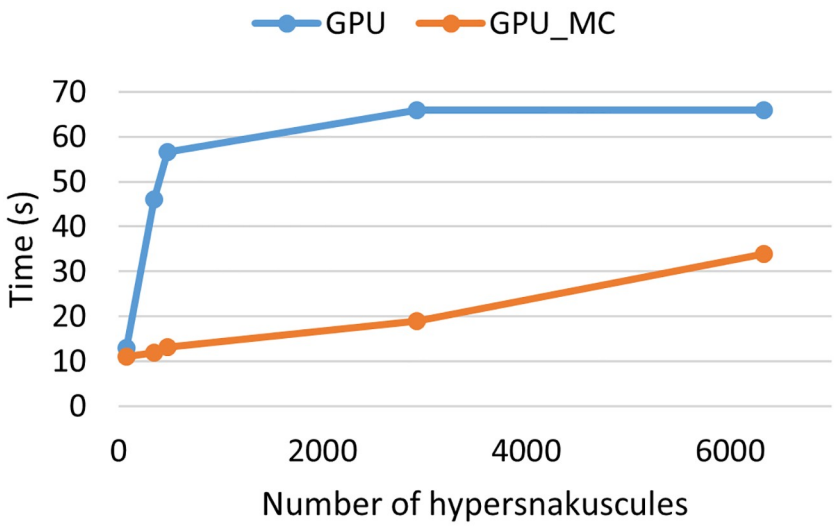
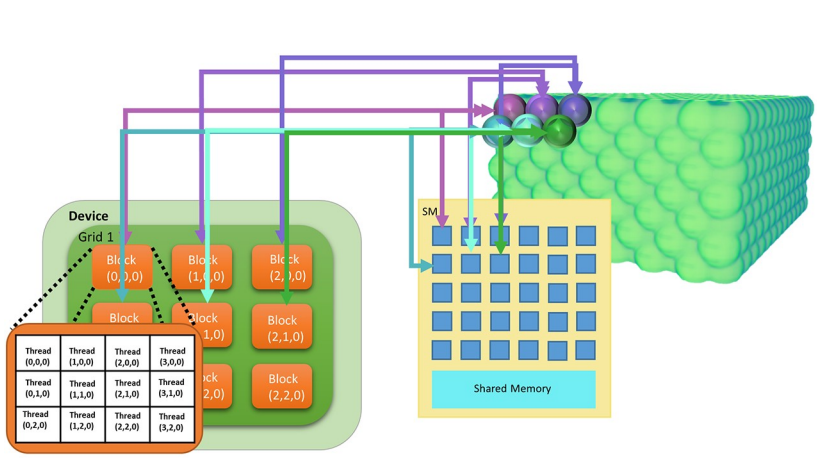


Fig 7. Execution time of 3D snakuscules on GPU with and without Monte Carlo sampling.

https://doi.org/10.1371/journal.pone.0215843.g007



(a)

New kernel algorithm:

Input: $I \in R^3$
N random samples $(s_0\ s_1 \ldots s_{N-1})$
Initial configuration of the 3D-snake

Output: update the 3D-snake

1. i = threadIdx
   T = number of threads per block
2. Take N/T samples and evaluate gradient

$$\frac{\delta P}{\delta t} = \sum_{k=i*N/T}^{(i+1)*N/T} \frac{\delta E(k)}{\delta P} \qquad \frac{\delta q}{\delta t} = \sum_{k=i*N/T}^{(i+1)*N/T} \frac{\delta E(k)}{\delta q}$$

3. Store the result in shared memory
4. syncthreads()
5. Update the hypersnakuscule

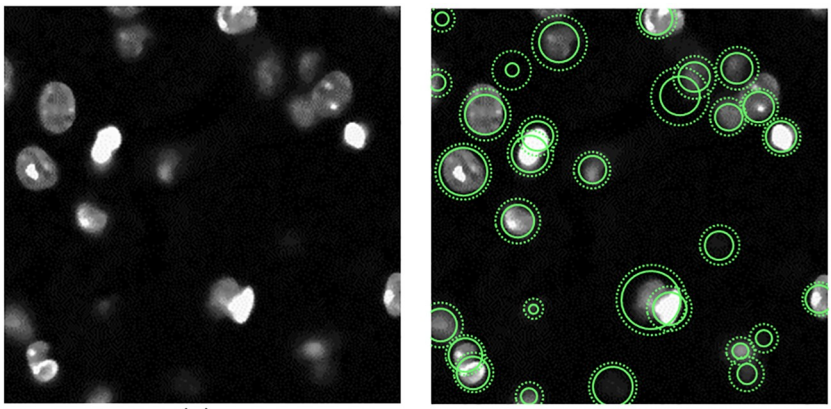
(b)

Fig 8. Parallel implementation on GPU. (a) Assignment of each contour to a thread block. (b) The algorithm implemented on GPU.

https://doi.org/10.1371/journal.pone.0215843.g008

Images of the hilus region of the dentate gyrus in the mouse hippocampus were collected with a 40X oil objective on a Leica TCS SP8 confocal microscope (1024 × 1024 pixels; 387.5 × 387.5μm). A 405 nm laser excited the DAPI signal that was detected between 415 to 500nm). A 1μm step size was set for z stack collection of the entire tissue thickness. Acquisition speed was set to 600 HZ, with a 0.75 zoom factor. Raw images for all data analysis were exported as TIFFs. Transgenic mice that model Dravet syndrome with spontaneous seizure onset at postnatal day 15 were housed in a 12 hour light/dark cycle. These mice have a knock-in mutant Scn1A gene containing a nonsense substitution (CgG to TgA) in exon 21 [40]. All animal experiments were approved by the Institutional Animal Care and Use Committee of the University of Houston.

We applied our method to the image of size 256 × 256 × 40 (Fig 9a). In order to deal with pixel anisotropy, where z-axis resolution is commonly worse than lateral (x,y) resolution, the



(a) (b)

Fig 9. (a) A section of DAPI stained mouse hippocampus 3D image and (b) the final configuration of 3D snakuscules on the section.

https://doi.org/10.1371/journal.pone.0215843.g009

pixel size is specified using lateral pixels and the axial direction is linearly interpolated using GPU hardware. the images were re-sampled to obtain a uniform pixel size. The contours are initialized as a lattice of 3D snakuscules. The 3D snakuscules are evolved and culled using the proposed methods. Fig 9b depicts the final configuration of them on a 2D slice.

To quantitatively evaluate the performance of our method, 3D snakuscules are considered as either a cell (foreground) or non-cell (background) using a K-nearest neighborhood (KNN) search. The four evaluation parameters, precision (P_r), recall (R_e), F-measure (F) and Jaccard (J), are calculated as follows:

$$\begin{aligned} P_r &= \frac{TP}{TP + FP} \\ R_e &= \frac{TP}{TP + FN} \\ F &= \frac{2P_rR_e}{P_r + R_e} \\ J &= \frac{P_rR_e}{P_r + R_e - P_rR_e} \end{aligned}$$

Where the true positive (TP) value is number of accurately detected cells, the false positive (FP) value is the number of falsely detected cells, and the false negative (FN) value is number of undetected cells. The ground truth is the manual detection of cell centers. The annotations are done using Gimp for 2D and an in-house tool for 3D images under an expert supervision.

Fig 10 illustrates the precision-recall curve for MINIS, FARSIGHT, 3D object counter, CellSegm and the proposed method on a DAPI-labeled image with 53 annotated cells. The performance for each algorithm is shown in Table 1. We initialized the 3D snake parameters with an initial radius of 15 pixels (≈ 6 μm), the energy threshold $E_0 = -3 (E_0 \leq 0)$, and the maximum number of iterations to 400. Also, We adjusted the hyperparameters of other methods to optimize their performance on our dataset. The results clearly demonstrate that 3D snakuscules are suitably capable of capturing round cell nuclei, and provide considerable performance advantages over other conventional methods with F-measure of 90% in comparison with that of 82%, 74%, 62% and 82% for MINIS, FARSIGHT, 3D object counter and CellSegm respectively. Additional advantages include the minimal number of parameters required for initialization.

We also evaluated our algorithm on two publicly available data sets (Fig 11) available at www.celltrackingchallenge.net:

- Fluo-N3DH_CE: Caenorhabditis elegans embryos stained with green flourescent protein (GFP) transfection collected with Plan-Apochromat 63X/1.4 (oil) objective lense on Zeiss LSM 510 Meta and the voxel size $0.09 \times 0.09 \times 1.0\mu m^3$ [41, 42].
- Fluo-N3DH-SIM+: A simulated video from fluorescently labeled nuclei of the HL60 cells stained with Hoescht. It is imaged using Plan-Apochromat 40X/1.3 (oil) objective with resolution $0.125 \times 0.125 \times 0.2\mu m^3$ [41, 42].

Segmentation results for the proposed method are shown in Table 2.

The sensitivity of our algorithm to noise was tested by generating a phantom based on manually segmented three-dimensional images of cells acquired using KESM [43]. Incremental reduction in signal-to-noise ratio (SNR) demonstrates robust localization with an F-measure ranging from 0.96 (original image) to 0.75 (SNR = 0.2dB) (Fig 12).

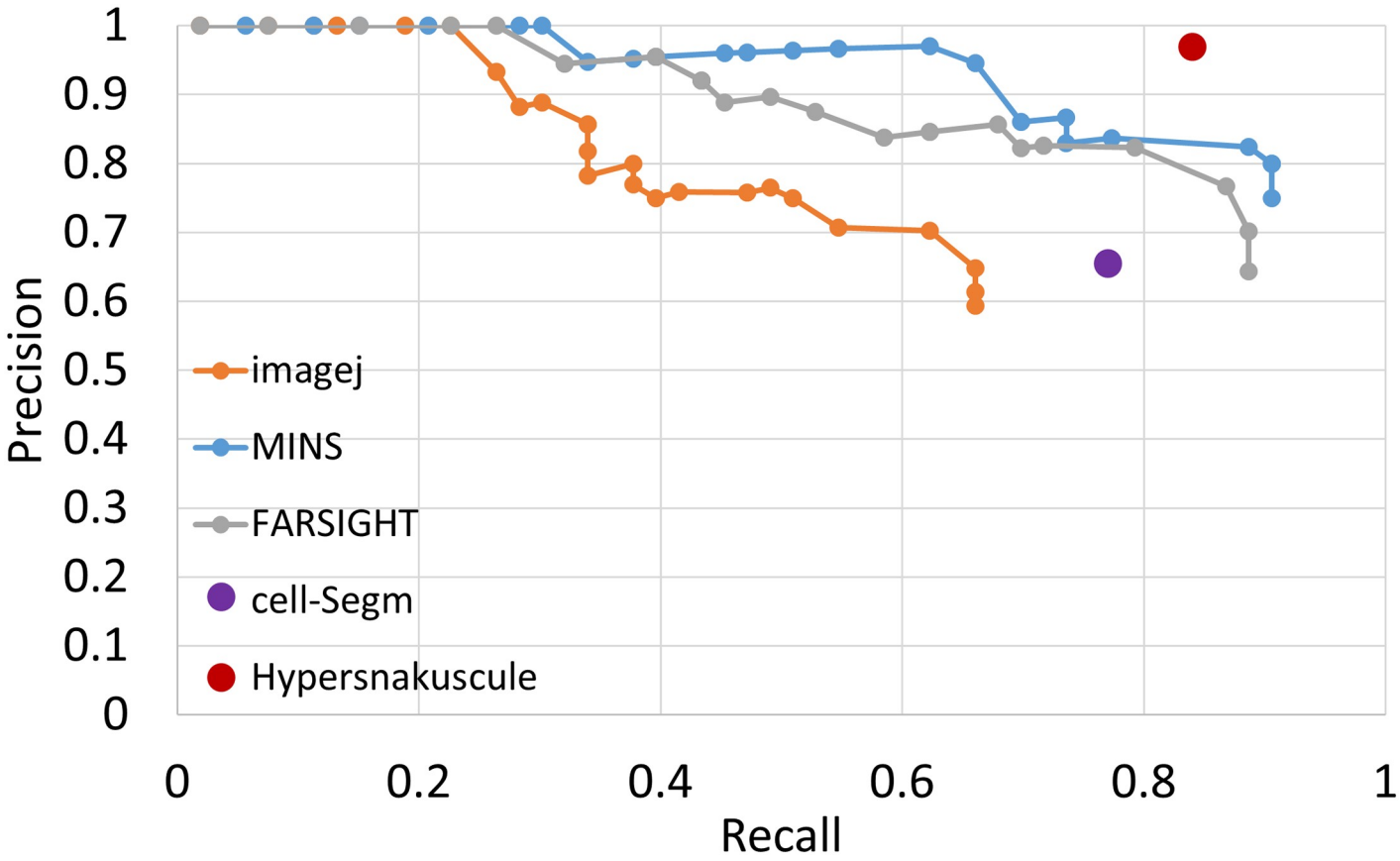


Fig 10. Evaluation of different algorithms on the same DAPI stained image. Our proposed method (3D snakuscule) provides results which matches the ground truth better than others.

https://doi.org/10.1371/journal.pone.0215843.g010

GPU occupancy

Occupancy is a measure of how many warps the kernel has active on the GPU, relative to the maximum number of warps supported by the GPU. The graphic processor used in our experiments is GetForce GTX-1070 with 1920 CUDA cores, 8GB of global memory, 2MB of L2 cache size, and 48kB of on-chip shared memory. The compute capability is 6.1, the global memory bandwidth is 256.256GB/s, and the single precision FLOP/s is 6.852TeraFLOP/s. Theoretical occupancy provides an upper bound while achieved occupancy indicates the kernel's actual performance. When the GPU does not have enough work, resources are wasted.

The theoretical occupancy for our algorithm is 50%, limited by the number of registers required for contour evolution. This is a relatively standard theoretical occupancy for complex

Table 1. Performance of different algorithms against manually segmented ground truth through evaluation parameters, precision, recall, Fmeasure and Jaccard.

Method	Precision	Recall	Fmeasure	Jaccard
MINS	0.75	0.90	0.82	0.69
FARSIGHT	0.64	0.88	0.74	0.58
object counter- imagej	0.59	0.66	0.62	0.45
CellSegm	0.66	0.90	0.82	0.61
3D snakuscules	**0.97**	**0.84**	**0.90**	**0.81**

https://doi.org/10.1371/journal.pone.0215843.t001

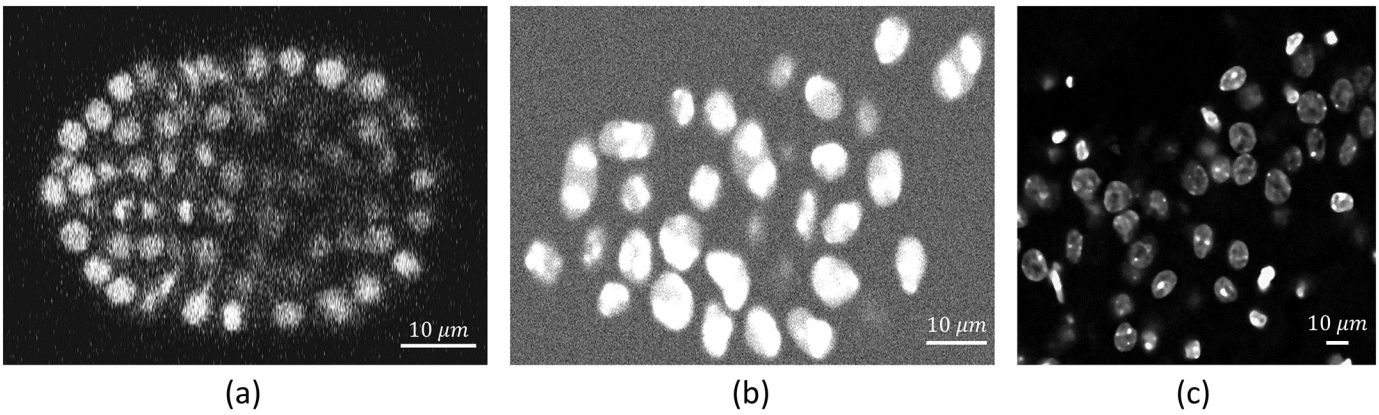


Fig 11. Representing one section of different 3D datasets used for cell detection. (a) Fluo-N3DH_CE (b) Fluo-N3DH-SIM+ (c) Mouse-Brain.

https://doi.org/10.1371/journal.pone.0215843.g011

Table 2. Performance of 3D snakescules on datasets with varying numbers of cells.

Dataset	# Cells	Precision	Recall	Fmeasure	Jaccard
Fluo-N3DH_CE	209	0.93	0.98	0.95	0.91
Fluo-N3DH-SIM+	39	0.97	0.92	0.94	0.89
Mouse-Brain(sec.1)	172	0.94	0.82	0.87	0.77
Mouse-Brain(sec.2)	53	0.97	0.84	0.90	0.81

https://doi.org/10.1371/journal.pone.0215843.t002

calculations, however a more rigorous optimization may yield better results in the future. Since any 3D contour is assigned to one GPU thread, the number of utilized threads is equal to the number of initial 3D snakes. Therefore, a small image with a small number of cells occupies fewer resources, resulting in low compute performance that is unable to hide operation and memory latency (Fig 13a). Comparing achieved occupancy with and without Monte-Carlo sampling shows that MC integration improves GPU performance and occupancy by reducing

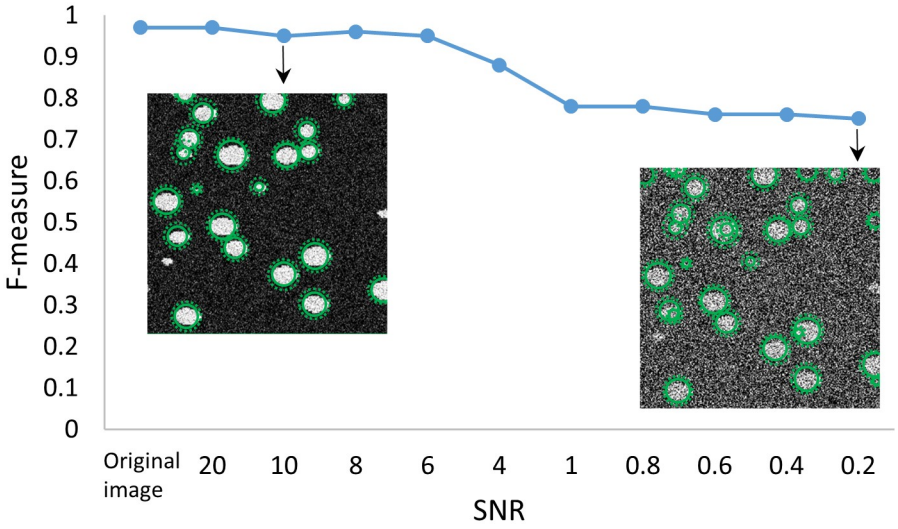


Fig 12. Quantification of the effect of Gaussian-distributed noise on localization accuracy. Reducing SNR to 0.2 results in a drop in F-measure from 0.96 to 0.75.

https://doi.org/10.1371/journal.pone.0215843.g012

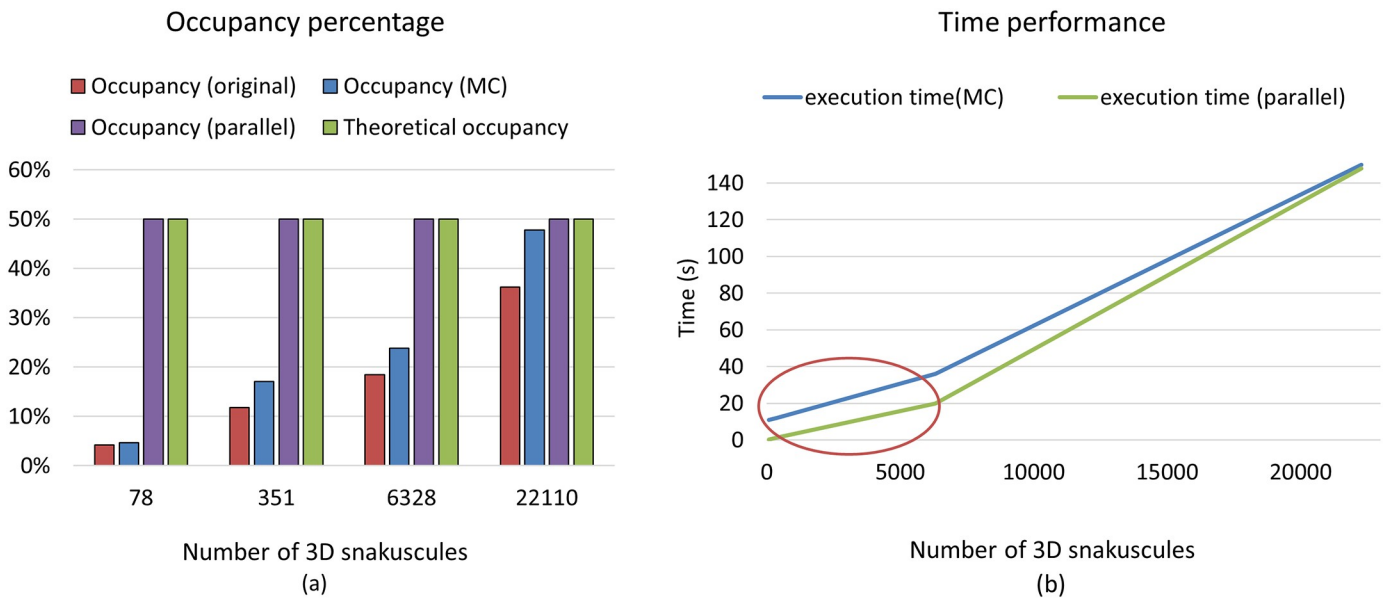


Fig 13. (a) Theoretical occupancy, green, and achieved occupancy using different size images (different number of initial 3D snakuscules) with and without Monte Carlo integration, blue and red respectively. Parallel 3D snakuscules provide an achieved occupancy (purple) comparable to the theoretical limit. (b) The diagram shows execution time of the method implemented on GPU using MC integration before and after further parallelization for different number of initiated 3D snakuscules.

https://doi.org/10.1371/journal.pone.0215843.g013

thread stalls. Since there are cells with various sizes in the image, uniform integration takes longer for contours corresponding to bigger cells, resulting in additional stalls. These are mitigated using MC integration.

Since the reduction in efficiency is due primarily to low occupancy, further parallelizing each 3D contour allows for the generation of more threads. This provides an achieved occupancy much closer to the theoretical limit (Fig 13a), further reducing processing time (Fig 13b).

Conclusion

We developed a 3D blob-detector, based on the miniscule snakes (snakuscule) algorithm, that provides a method for 3D nuclei detection with minimal user interaction. This paper describes a unified formulation of snakuscules in three dimensional space, so that the new cost function is minimized with respect to two points which define the contour. Although the method is initially computationally expensive, it is extremely data parallel and can be efficiently implemented using GPU hardware. A GPU implementation, combined with Monte-Carlo sampling, results in a simple and fast blob detector for large images with numerous cells. Our method requires the specification of a minimum contour size, which is usually readily available for microscopy images. We have illustrated that the GPU implementation and Monte Carlo sampling significantly increase performance, making 3D snakuscules viable for cell localization. The experimental results demonstrate that the proposed method outperforms state of art methods in overall accuracy.

One major limitation of this algorithm is that a significant portion of the evaluation is devoted to the evolution of snakes that will ultimately be culled. This suggests that any method that reliably places initial contours could significantly increase snakuscule performance. A more optimal placement of initial contours could significantly improve performance beyond

what we were able to achieve with a lattice. However, methods such as iterative voting and Laplacian of Gaussian blob detection resulted in reduced accuracy when tested. Therefore more advanced algorithms, such as dynamic culling or insertion of contours during evolution, may be a better approach.

In addition, further optimization of the evolution kernel to increase theoretical occupancy limited by register usage could double performance.

Acknowledgments

The authors would like to thank the University of Houston Core facility for Advanced Computing and Data Science (CACDS) for providing technical support and GPU computing resources.

Author Contributions

Conceptualization: David Mayerich.

Data curation: Laura Montier, Jokūbas Žiburkus.

Formal analysis: Laura Montier, Jokūbas Žiburkus, David Mayerich.

Funding acquisition: David Mayerich.

Investigation: David Mayerich.

Methodology: Mahsa Lotfollahi, Sebastian Berisha, David Mayerich.

Project administration: David Mayerich.

Resources: David Mayerich.

Software: Mahsa Lotfollahi, Sebastian Berisha, Leila Saadatifard, David Mayerich.

Supervision: Sebastian Berisha, David Mayerich.

Validation: Mahsa Lotfollahi, Leila Saadatifard, David Mayerich.

Visualization: Mahsa Lotfollahi, David Mayerich.

Writing – original draft: Mahsa Lotfollahi, David Mayerich.

Writing – review & editing: Mahsa Lotfollahi, Sebastian Berisha, Leila Saadatifard, Laura Montier, Jokūbas Žiburkus, David Mayerich.

References

1. Su H, Yin Z, Huh S, Kanade T, Zhu J. Interactive Cell Segmentation Based on Active and Semi-Supervised Learning. IEEE Transactions on Medical Imaging. 2016; 35(3):762–777. https://doi.org/10.1109/TMI.2015.2494582 PMID: 26529749
2. Irshad H, Veillard A, Roux L, Racoceanu D. Methods for nuclei detection, segmentation, and classification in digital histopathology: a review—current status and future potential. IEEE reviews in biomedical engineering. 2014; 7:97–114. PMID: 24802905
3. Lin G, Adiga U, Olson K, Guzowski JF, Barnes CA, Roysam B. A hybrid 3D watershed algorithm incorporating gradient cues and object models for automatic segmentation of nuclei in confocal image stacks. Cytometry. 2003; 56A(1):23–36. https://doi.org/10.1002/cyto.a.10079
4. Xing F, Xie Y, Yang L. An Automatic Learning-Based Framework for Robust Nucleus Segmentation. IEEE Transactions on Medical Imaging. 2016; 35(2):550–566. https://doi.org/10.1109/TMI.2015.2481436 PMID: 26415167
5. Wahlby C, Sintorn IM, Erlandsson F, Borgefors G, Bengtsson E. Combining intensity, edge and shape information for 2D and 3D segmentation of cell nuclei in tissue sections. Journal of Microscopy. 2004; 215(1):67–76. https://doi.org/10.1111/j.0022-2720.2004.01338.x PMID: 15230877

6. Meijering E. Cell Segmentation: 50 Years Down the Road [Life Sciences]. IEEE Signal Processing Magazine. 2012; 29(5):140–145. https://doi.org/10.1109/MSP.2012.2204190
7. Zhou Xiaobo, Li Fuhai, Yan Jun, Wong STC. A Novel Cell Segmentation Method and Cell Phase Identification Using Markov Model. IEEE Transactions on Information Technology in Biomedicine. 2009; 13 (2):152–157. https://doi.org/10.1109/TITB.2008.2007098 PMID: 19272857
8. George YM, Bagoury BM, Zayed HH, Roushdy MI. Automated cell nuclei segmentation for breast fine needle aspiration cytology. Signal Processing. 2013; 93(10):2804–2816. https://doi.org/10.1016/j.sigpro.2012.07.034
9. Xu H, Lu C, Mandal M. An Efficient Technique for Nuclei Segmentation Based on Ellipse Descriptor Analysis and Improved Seed Detection Algorithm. IEEE Journal of Biomedical and Health Informatics. 2014; 18(5):1729–1741. https://doi.org/10.1109/JBHI.2013.2297030 PMID: 25192578
10. Grigorescu SE, Petkov N, Kruizinga P. Comparison of texture features based on Gabor filters. IEEE Transactions on Image Processing. 2002; 11(10):1160–1167. https://doi.org/10.1109/TIP.2002.804262 PMID: 18249688
11. Ongun G, Halici U, Leblebicioglu K, Atalay V, Beksac M, Beksac S. Feature extraction and classification of blood cells for an automated differential blood count system. IEEE; 2001. p. 2461–2466. Available from: http://ieeexplore.ieee.org/document/938753/.
12. Yin Z, Bise R, Chen M, Kanade T. Cell segmentation in microscopy imagery using a bag of local Bayesian classifiers. IEEE; 2010. p. 125–128. Available from: http://ieeexplore.ieee.org/document/5490399/.
13. Jørgensen AS, Rasmussen AM, Andersen NKM, Andersen SK, Emborg J, Røge R, et al. Using cell nuclei features to detect colon cancer tissue in hematoxylin and eosin stained slides: Detection of Colon Cancer Tissue. Cytometry Part A. 2017.
14. Jun Tang. A color image segmentation algorithm based on region growing. IEEE; 2010. p. V6–634–V6–637. Available from: http://ieeexplore.ieee.org/document/5486012/.
15. Koyuncu CF, Arslan S, Durmaz I, Cetin-Atalay R, Gunduz-Demir C. Smart Markers for Watershed-Based Cell Segmentation. PLoS ONE. 2012; 7(11):e48664. https://doi.org/10.1371/journal.pone.0048664 PMID: 23152792
16. Maycock AL, Abeles RH, Salach JI, Singer TP. The structure of the covalent adduct formed by the interaction of 3-dimethylamino-1-propyne and the flavine of mitochondrial amine oxidase. Biochemistry. 1976; 15(1):114–125. https://doi.org/10.1021/bi00646a018 PMID: 2278
17. Sadeghian F, Seman Z, Ramli AR, Abdul Kahar BH, Saripan MI. A Framework for White Blood Cell Segmentation in Microscopic Blood Images Using Digital Image Processing. Biological Procedures Online. 2009; 11(1):196–206. https://doi.org/10.1007/s12575-009-9011-2 PMID: 19517206
18. Ko BC, Gim JW, Nam JY. Automatic white blood cell segmentation using stepwise merging rules and gradient vector flow snake. Micron. 2011; 42(7):695–705. https://doi.org/10.1016/j.micron.2011.03.009 PMID: 21530280
19. Lotfollahi M, Gity M, Ye JY, Mahlooji Far A. Segmentation of breast ultrasound images based on active contours using neutrosophic theory. Journal of Medical Ultrasonics. 2017. https://doi.org/10.1007/s10396-017-0811-8 PMID: 28821993
20. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer; 2015. p. 234–241.
21. Chen H, Qi X, Yu L, Heng PA. Dcan: Deep contour-aware networks for accurate gland segmentation. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition; 2016. p. 2487–2496.
22. Wang Q, Wang S, Zhu X, Liu T, Humphrey Z, Ghukasyan V, et al. Accurate and High Throughput Cell Segmentation Method for Mouse Brain Nuclei Using Cascaded Convolutional Neural Network. In: International Workshop on Patch-based Techniques in Medical Imaging. Springer; 2017. p. 55–62.
23. Akram SU, Kannala J, Eklund L, Heikkilä J. Cell proposal network for microscopy image analysis. In: Image Processing (ICIP), 2016 IEEE International Conference on. IEEE; 2016. p. 3199–3203.
24. Shan J, Cheng H, Wang Y. Completely automated segmentation approach for breast ultrasound images using multiple-domain features. Ultrasound in medicine & biology. 2012; 38(2):262–275. https://doi.org/10.1016/j.ultrasmedbio.2011.10.022
25. Khoshdeli M, Cong R, Parvin B. Detection of nuclei in H&E stained sections using convolutional neural networks. In: Biomedical & Health Informatics (BHI), 2017 IEEE EMBS International Conference on. IEEE; 2017. p. 105–108.
26. Sailem H, Arias-Garcia M, Bakal C, Zisserman A, Rittscher J. Discovery of Rare Phenotypes in Cellular Images Using Weakly Supervised Deep Learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 49–55.

27. Cao Z, Duan L, Yang G, Yue T, Chen Q, Fu H, et al. Breast Tumor Detection in Ultrasound Images Using Deep Learning. In: International Workshop on Patch-based Techniques in Medical Imaging. Springer; 2017. p. 121–128.

28. Song Y, Tan EL, Jiang X, Cheng JZ, Ni D, Chen S, et al. Accurate cervical cell segmentation from overlapping clumps in pap smear images. IEEE transactions on medical imaging. 2017; 36(1):288–300. https://doi.org/10.1109/TMI.2016.2606380 PMID: 27623573

29. Santamaria-Pang A, Rittscher J, Gerdes M, Padfield D. Cell segmentation and classification by hierarchical supervised shape ranking. In: Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on. IEEE; 2015. p. 1296–1299.

30. Janssens T, Antanas L, Derde S, Vanhorebeek I, Van den Berghe G, Grandas FG. CHARISMA: An integrated approach to automatic H&E-stained skeletal muscle cell segmentation using supervised learning and novel robust clump splitting. Medical image analysis. 2013; 17(8):1206–1219. https://doi.org/10.1016/j.media.2013.07.007 PMID: 24012925

31. Nielsen B, Maddison J, Danielsen H. Optimizing the initialization and convergence of active contours for segmentation of cell nuclei in histological sections; 2015.

32. Al-Kofahi Y, Lassoued W, Lee W, Roysam B. Improved automatic detection and segmentation of cell nuclei in histopathology images. IEEE transactions on bio-medical engineering. 2010; 57(4):841–852. https://doi.org/10.1109/TBME.2009.2035102 PMID: 19884070

33. Lou X, Kang M, Xenopoulos P, Muñoz-Descalzo S, Hadjantonakis AK. A Rapid and Efficient 2D/3D Nuclear Segmentation Method for Analysis of Early Mouse Embryo and Stem Cell Image Data. Stem Cell Reports. 2014; 2(3):382–397. https://doi.org/10.1016/j.stemcr.2014.01.010 PMID: 24672759

34. Bolte S, CordelièRes FP. A guided tour into subcellular colocalization analysis in light microscopy. Journal of Microscopy. 2006; 224(3):213–232. https://doi.org/10.1111/j.1365-2818.2006.01706.x PMID: 17210054

35. Hodneland E, Kögel T, Frei D, Gerdes HH, Lundervold A. CellSegm—a MATLAB toolbox for high-throughput 3D cell segmentation. Source Code for Biology and Medicine. 2013; 8(1):16. https://doi.org/10.1186/1751-0473-8-16 PMID: 23938087

36. Cheng YH, Lin TC, Ding JJ, Wu YF, Lin SJ. Adaptive 3D cell segmentation and tracing algorithm using convex separation and histogram information for vivo images. In: Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on. IEEE; 2017. p. 133–138.

37. Thevenaz P, Unser M. Snakuscules. IEEE transactions on image processing: a publication of the IEEE Signal Processing Society. 2008; 17(4):585–593. https://doi.org/10.1109/TIP.2007.914742

38. Pediredla AK, Seelamantula CS. A unified approach for optimization of Snakuscules and Ovuscules. IEEE; 2012. p. 681–684. Available from: http://ieeexplore.ieee.org/document/6287975/.

39. Smistad E, Falch TL, Bozorgi M, Elster AC, Lindseth F. Medical image segmentation on GPUs- A comprehensive review. Medical Image Analysis. 2015; 20(1):1–18. https://doi.org/10.1016/j.media.2014.10.012 PMID: 25534282

40. Ogiwara I, Miyamoto H, Morita N, Atapour N, Mazaki E, Inoue I, et al. Nav1. 1 localizes to axons of parvalbumin-positive inhibitory interneurons: a circuit basis for epileptic seizures in mice carrying an Scn1a gene mutation. Journal of Neuroscience. 2007; 27(22):5903–5914. https://doi.org/10.1523/JNEUROSCI.5270-06.2007 PMID: 17537961

41. Maška M, Ulman V, Svoboda D, Matula P, Matula P, Ederra C, et al. A benchmark for comparison of cell tracking algorithms. Bioinformatics. 2014; 30(11):1609–1617. https://doi.org/10.1093/bioinformatics/btu080 PMID: 24526711

42. Ulman V, Maška M, Magnusson KE, Ronneberger O, Haubold C, Harder N, et al. An objective comparison of cell-tracking algorithms. Nature methods. 2017; 14(12):1141. https://doi.org/10.1038/nmeth.4473 PMID: 29083403

43. Saadatifard L, Abbott LC, Montier L, Ziburkus J, Mayerich D. Robust Cell Detection for Large-Scale 3D Microscopy Using GPU-Accelerated Iterative Voting. Frontiers in neuroanatomy. 2018; 12. https://doi.org/10.3389/fnana.2018.00028 PMID: 29755325