

PROCEEDINGS

Open Access

Software tool for 3D extraction of germinal centers

David N Olivieri^{1*}, Merly Escalona¹, Jose Faro^{2,3*}

From 10th International Conference on Artificial Immune Systems (ICARIS)
Cambridge, UK. 18-21 July 2011

Abstract

Background: Germinal Centers (GC) are short-lived micro-anatomical structures, within lymphoid organs, where affinity maturation is initiated. Theoretical modeling of the dynamics of the GC reaction including follicular CD4⁺ T helper and the recently described follicular regulatory CD4⁺ T cell populations, predicts that the intensity and life span of such reactions is driven by both types of T cells, yet controlled primarily by follicular regulatory CD4⁺ T cells. In order to calibrate GC models, it is necessary to properly analyze the kinetics of GC sizes. Presently, the estimation of spleen GC volumes relies upon confocal microscopy images from 20-30 slices spanning a depth of ~ 20 - 50 μm , whose GC areas are analyzed, slice-by-slice, for subsequent 3D reconstruction and quantification. The quantity of data to be analyzed from such images taken for kinetics experiments is usually prohibitively large to extract semi-manually with existing software. As a result, the entire procedure is highly time-consuming, and inaccurate, thereby motivating the need for a new software tool that can automatically identify and calculate the 3D spot volumes from GC multidimensional images.

Results: We have developed *pyBioImage*, an open source cross platform image analysis software application, written in python with C extensions that is specifically tailored to the needs of immunologic research involving 4D imaging of GCs. The software provides 1) support for importing many multi-image formats, 2) basic image processing and analysis, and 3) the *ExtractGC* module, that allows for automatic analysis and visualization of extracted GC volumes from multidimensional confocal microscopy images. We present concrete examples of different microscopy image data sets of GC that have been used in experimental and theoretical studies of mouse model GC dynamics.

Conclusions: The *pyBioImage* software framework seeks to be a general purpose image application for immunological research based on 4D imaging. The *ExtractGC* module uses a novel clustering algorithm for automatically extracting quantitative spatial information of a large number of GCs from a collection of confocal microscopy images. In addition, the software provides 3D visualization of the GCs reconstructed from the image stacks. The application is available for public use at <http://sourceforge.net/projects/pybioimage/>.

Background

During the later phase of primary immune responses to protein antigens, as well as in secondary immune responses to the same antigen, the produced antibodies display higher affinity for their antigen compared with the early phase of the response, a phenomenon known as affinity maturation [1]. The precise mechanisms responsible for this phenomenon are the subject of current intense research, and are

known to take place in well-organized micro-anatomical structures, called germinal centers (GC), that develop temporarily within primary follicles of secondary lymphoid organs during immune responses to protein antigens [2].

The number of GCs and their average size increases dramatically within the first week after immunization and then start to decrease within days 10-14, so that by days 21-24 very few of them remain, while those that do have small sizes. GCs consist of a dominant population of antigen-specific B cells and smaller populations of T lymphocytes, follicular dendritic cells, and macrophages [3-5]. The antigen-specific B cells proliferate intensely,

* Correspondence: olivieri@ei.uvigo.es; jfaro@uvigo.es

¹School of Computer Engineering, University of Vigo, Ourense, 32004, Spain

²Department of Biochemistry, Genetics and Immunology, University of Vigo, Vigo, 36310, Spain

Full list of author information is available at the end of the article

undergo somatic hypermutation in the variable region of their antibody molecules, and are subject to a poorly understood affinity-based selection process [2,6].

The long-held interest in GCs stems from being the place where a Darwinian process, involving somatic hypermutation (SHM) and selection, acts on responding B cells and their antibodies, thereby leading to memory B cell generation and to the phenomenon of affinity maturation. Because of the very high rate of SHM (10^{-4} to 10^{-3} per base pair and cell division), a GC reaction with an excessively long duration may not only spoil previous affinity enhancing mutations, but also generate autoreactive and even aberrant mutations leading to leukemia cells. Contrarily, because of the random character of SHM, affinity-enhancing mutations appear only several days after the activation of hypermutation, so that GC reactions with durations too short will have an ineffectual selection. As a result, it is not totally surprising that the time scale of GC reactions is regulated. The precise mechanisms that drive and control the dynamics of GCs are not presently known and is the focus of intense research.

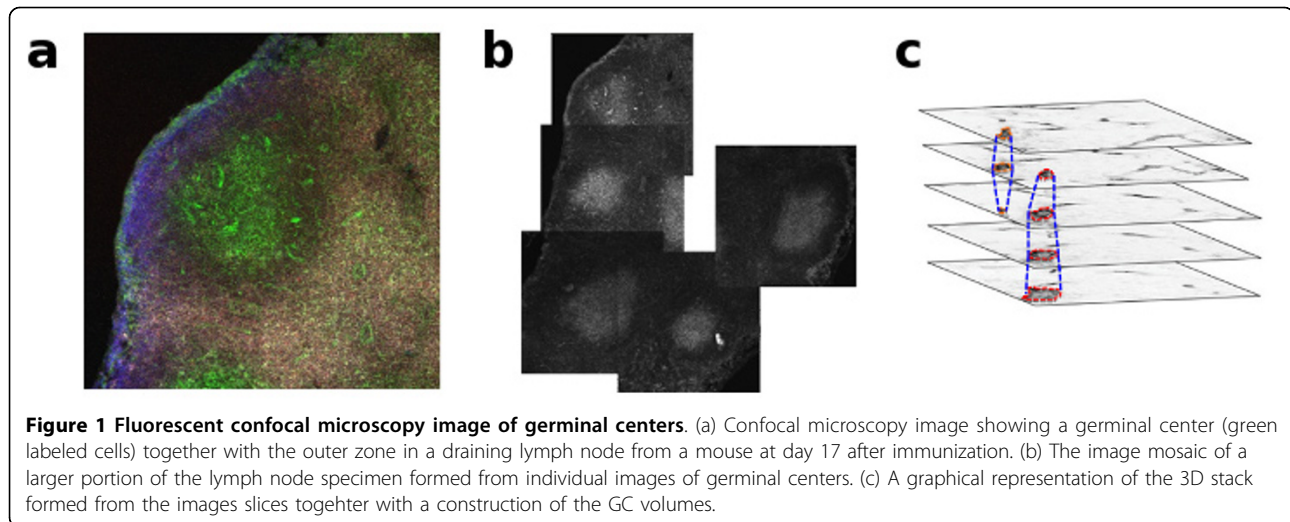
Recently some of us [7] and others [8,9] have shown that the dynamics of GCs is controlled by follicular regulatory $CD4^+$ T (T_{Freg}) lymphocytes, a newly discovered distinct subpopulation of $Foxp3^+CD4^+$ T cells that share with follicular $CD4^+$ T helper cells the same responsiveness to the follicular chemokine CXCL13. The impact of T_{Freg} on the kinetics of GC sizes was made evident in studies involving confocal microscopy analysis of murine mesenteric lymph nodes at different times after immunization [7]. Our theoretical modeling of the dynamics of the GC reaction, including T_{Freg} cells, suggests that the intensity and life span of such a reaction is subject to two different controlling processes: an initial process driven by T_{Freg} cells, and a later one, detectable only when the first process is too weak, controlled by follicular $CD4^+$ T helper cell maturation (JF, manuscript in preparation). In order to properly calibrate GC models with T_{Freg} lymphocytes, comparisons and fits to experimentally obtained GC sizes taken at different time during the kinetics of entire process is fundamental. A sufficiently accurate study, however, requires a more exhaustive analysis with the acquisition of more time points during the immune response than previously accomplished in experiments to date. Also, such an analysis would require accurate determination of all the GC volumes obtained from these experiments.

Presently, the estimation of GC volumes relies upon sectioning either the spleen or lymph nodes in several tissue samples of approximately $\sim 20 - 50 \mu\text{m}$ thickness and performing immunohistochemical staining. Subsequently, confocal microscopy is used to acquire images from each section at different equally spaced plane depths, usually generating more than 20-30 thin slices. These image slices are then digitized and assembled into

Z-stacks, as shown in Figure 1. Finally, using a program such as ImageJ [10,11], the analysis of GC areas is performed semi-manually slice-to-slice, in the Z-stack, for subsequent 3D reconstruction. Such a tedious and highly time-consuming procedure provides a crude estimation of GC volumes. Moreover, the sheer quantity of data to be analyzed from this large set of confocal microscopy imaging is so prohibitively large, that manual or semi-manual extraction is not tenable. Indeed, for understanding the order of magnitude of data produced, a typical experiment for study GC dynamics involves the analysis of on average ~ 100 GCs (5-10 slices per GC) per mouse and per time point (minimum 3-4 mice per time point and 4-5 time points). As a result, there may be more than 6000 individual GC slices to be analyzed in a single experiment.

Available software A candidate software application for the proposed task would provide automatic measurements of all GC densities; that is, an accurate and automatic measure of the individual GC volumes, a count of the constituent cells, and subsequent visual confirmation of each GC using a three dimensional isosurface reconstruction [12]. Perhaps the two most popular representative *open-source* software tools used for post-processing of microscopy images are ImageJ [10,11] (together with a newer distribution branch, Fiji [13]) and OMERO [14-16]. Other open-source software tools for biological visualization include Vaa3D (<http://www.vaa3d.org>), which is a cross platform tool geared towards biological visualization of 3D/4D/5D formats, and Icy (<http://icy.bioimageanalysis.org>), which is another powerful image analysis software that provides a powerful environment for third party developers together with visualization software. Several commercial software applications, such as Imaris and MetaMorph, are also widely used by the biology community for performing post-processing image analysis and visualization tasks. While a complete listing or comparison of all available software solutions are beyond the scope of this paper, these applications are certainly state-of the art and highly representative of other applications with their particular advantages/disadvantages. Also, in keeping with our design philosophy, we have focused more upon open-source analysis tools for comparing our software and algorithms.

In the case of OMERO, this is a large client/server application, designed to provide centralized access of images from a disk server, and provides many types of analysis as well as data annotation and workflow. While OMERO has a large user base, and many analysis extensions, it presently lacks the ability to automatically perform segmentation of objects such as GCs in 3D (also referred to as 3D spot volumes) and does not provide a 3D output that allows for visual checking of the accuracy of the borders of the detected GCs. Moreover, there is no provision in their



roadmap for the addition of these difficult, yet important features [14].

Fiji/ImageJ, is a multi- platform Java-based application written for the desktop that uses the powerful ImageJ image analysis library for a microscopy specific application. It features extendible plug-in module support, scripting in multiple languages, and supports a large collection of image formats used the microscopy community. In an independently developed branch, Fiji, provides many new powerful analysis extensions; an example of which is image registration techniques [17-19], for stitching multi-dimensional images from low-level autocorrelation of features. Nonetheless, it also does not contain the capability for automatic segmentation of volumes with constituent cells, as envisioned in this work.

In summary, with respect to segmenting and extracting GC volumes, the microscopy software applications and algorithms that we have evaluated either (a) lack sufficient information about the segmented dimensions, (b) underestimate the number of objects segmented due to the difficulty of selecting the appropriate input parameters, (c) provide only gross estimates of areas/volumes, or (d) simply do not provide the desired functionality for automatically obtaining GC volumes. As such, with respect to extracting GC volumes, no single software tool exists, to our knowledge, able to perform the proposed automated tasks and that meets all requirements desired.

While ImageJ and Fiji have a large user base and provide the ability to write customized plug-ins in various programming languages, we decided from the onset to deviate from this standard development course in order to develop our own microscopy infrastructure, written in python and called **pyBioImage**. While motivated by several reasons, the principle advantage of this design choice is to leverage the growing software base for scientific computing with powerful and efficient numerical and visual

libraries recently made available in the python community. Given the power of the python C-extension API, available libraries, and the ability for rapid and robust open software development, other microscopy software application have recently emerged, albeit with slightly different scientific goals, but based upon a similar python/C design philosophy. Two recent open source tools also written in python and C/C++, which have recently been reported in the literature for microscopy applications, are IOCBioMicroscope [20] (focused upon deconvolution of microscopy images) and BioImageXD [21].

Implementation

Our software suite, **pyBioImage**, is a cross-platform bio-imaging application, written in Python and makes use of low level C code exposed through the Python C-extension API. The application supports multiple data formats and provides visualization and analysis of standard multi-dimensional image data. For the work described in this paper, we have developed a set of algorithms implemented either in pure python or as python/C-extension modules, that form a core feature called *ExtractGC*, which is specifically tailored for automatically extracting GC volume statistics and visualization from a collection of 3D confocal fluorescent microscopy image stacks. These images are highly magnified regions of tissue samples taken from secondary lymphoid organs. The set of such images from tissue specimens may be used to reconstruct a 3D mosaic, consisting of several GCs, and thereby making it possible visualize a large section of the organ in question. Our analysis software module *ExtractGC*, which is part of the more general **pyBioImage** application, uses a pseudo-recursive segmentation algorithm for performing simultaneous pixel level clustering in all directions xyz of a complete image stack. Our segmentation technique is based upon a general segmentation algorithm, often referred to as *spot finding*

algorithm in the context of fluorescent microscopy, first described and implemented by Goldberg *and col.* [22].

In order to maintain the goal of cross-platform interoperability, the selection and design of the libraries used, as well as our software suite **pyBioImage** (and in particular *ExtractGC*) depend upon several standard open source software libraries, as shown in Figure 2. In particular, in order to perform low-level pixel operations as efficiently as possible, we have written several python C-extension modules. We use Numpy and Scipy for several post-processing numerical routines. For image analysis operations, we use the OpenCV library. The graphical interface is based upon wxPython, and 2D visualization with specialized bindings with Matplotlib and the Enthought Traits library (code.enthoight.com/). For 3D reconstruction, we have written python bindings for the well known powercrust algorithm [23], and visualization rendering is presently performed with the GeomView (<http://www.geomview.org>), however we are presently incorporating the use of VTK (<http://www.vtk.org>).

Many file formats for confocal microscopy are based upon variants of Tiff, or at least the ability to include multiple images with the same file. For standard Tiff files, we have used a python based wrapper of the standard libtiff library. In order to connect with LOCI BioFormats [24], that is provided through a Java jar library module, we use

the Internet Communication Engine (ICE) (<http://www.zeroc.com>) which provides a drop-in C/C++ connectivity for I/O module.

Workflow and interface

Figure 3 illustrates the workflow of our software suite **pyBioImage** with *ExtractGC* for the specific task of obtaining detailed statistics and visualizations of GC volumes. In particular, a set of acquired confocal microscopy images are read/loaded into the application according to the appropriate format with efficient memory management. At this point, either the individual image stacks may be analyzed for obtaining GC volumes, or a mosaic may be obtained for subsequent analysis.

For extracting GC volumes, an initial seed value for the threshold and minimum spot size are provided by the user. The optimal global threshold is found with a simulated annealing optimization algorithm, by using this initial seed together with other parameters. This will produce the optimal 3D bounding surface together with statistics for all GCs that pass the maximum size specified by the input parameter. As with general stochastic global optimizers, further flexibility towards optimal solutions can be explored by adjusting a subset of input parameters. Full 3D rendering of all or selected GC volumes may be obtained interactively by the user.

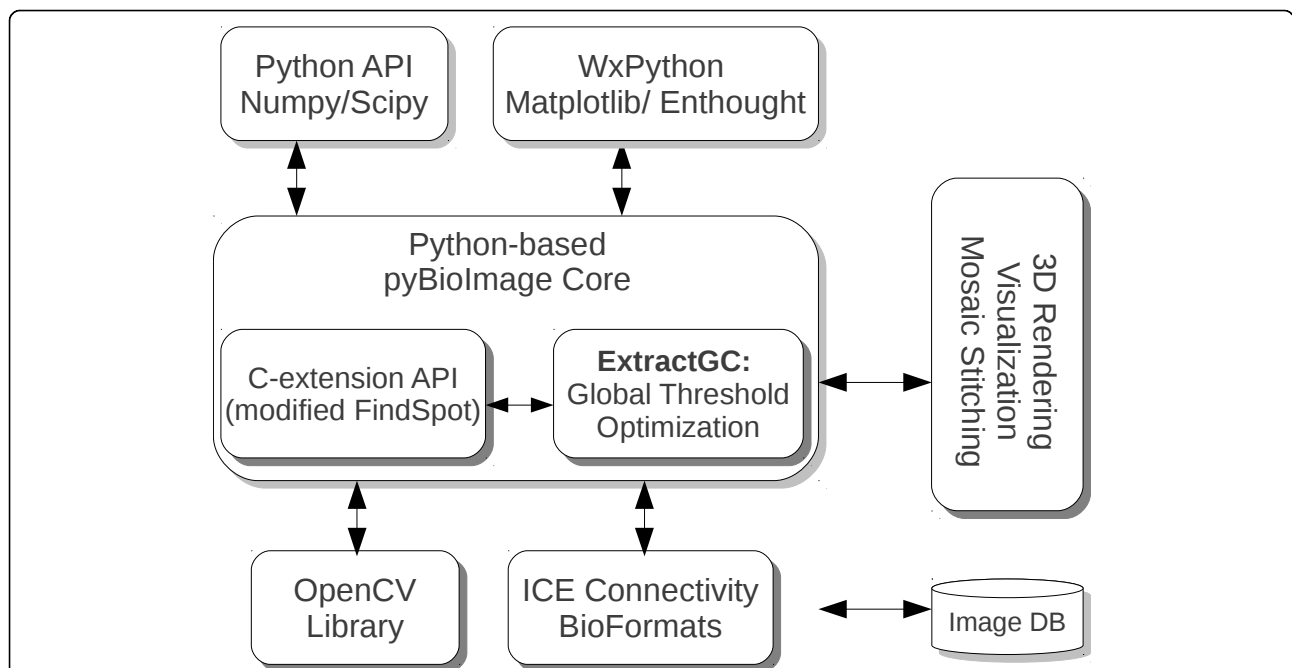
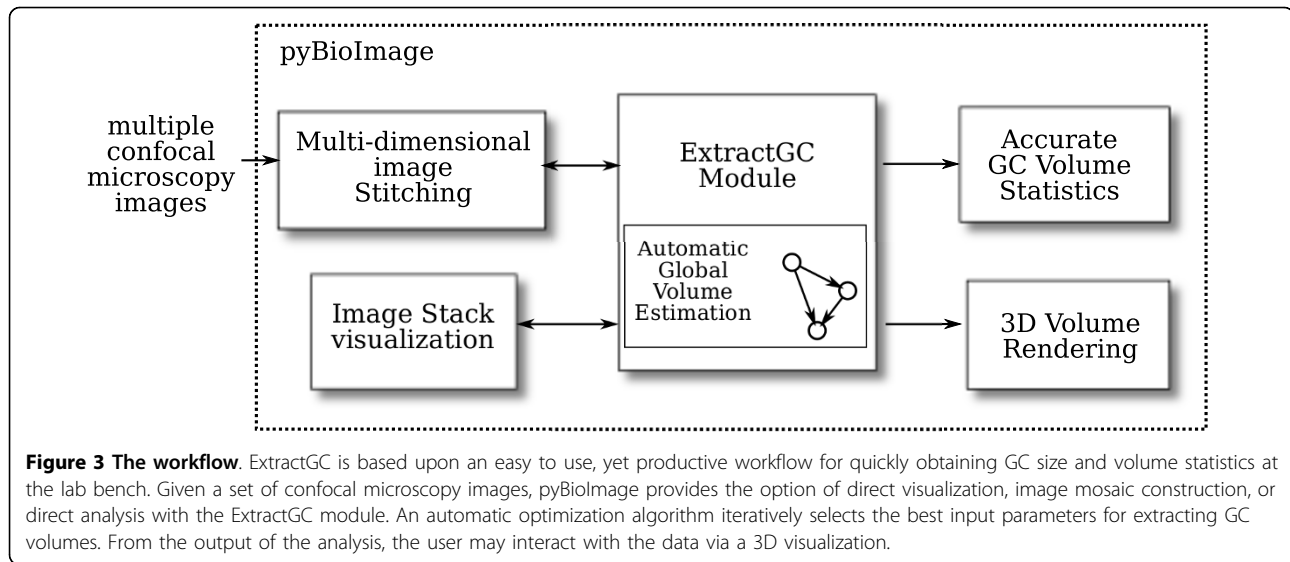


Figure 2 The architecture. The figure shows the relation between the ExtractGC module and other software components that makes up the architecture of pyBioImage. The core application of the pyBioImage application is written in Python with the graphical user library wxPython. The ExtractGC module consists of python classes, used for high level processing, and low-level pixel operations performed in C, exposed through the Python C-extension API. Numerical algorithms, image analysis, image I/O, the graphical user interface and data visualization leverage the use of powerful open-source libraries.



Segmentation algorithm for extracting germinal center volumes

Broadly speaking, segmentation algorithms decompose an image into distinct parts for recognizing objects of interest. These algorithms can be divided into three groups: statistical feature-based, region-growing, and boundary methods [25,26]. For multidimensional images, feature based and boundary methods use image registration algorithms [27] to associate image pixels of one image to those of another. There are many techniques for accomplishing this task, including pixel-wise comparisons, cross-correlations, and scale invariant feature-based methods. These techniques have been extensively studied and applied to multi-dimensional medical and microscopy imaging for reconstructing volumes from different z-stack slices. Region growing methods perform segmentation by low-level pixel assembly, subject to some condition related to the pixels intensities of nearby neighbors. For multidimensional microscopy images, the FindSpot algorithm described by I. Goldberg [22], has been shown to be effective for constructing spot volumes, which are the bright/dark regions of interest, by recursively obtaining correspondence between neighboring pixels on the same and different image slices. By manually providing threshold and geometric constraints, the algorithm can efficiently encounter 3D continuous object volumes within and throughout the multidimensional image. Given the power of this method, our GC volume extraction software uses the core part of this algorithm together with several practical software modifications as well as additional algorithm details, described below.

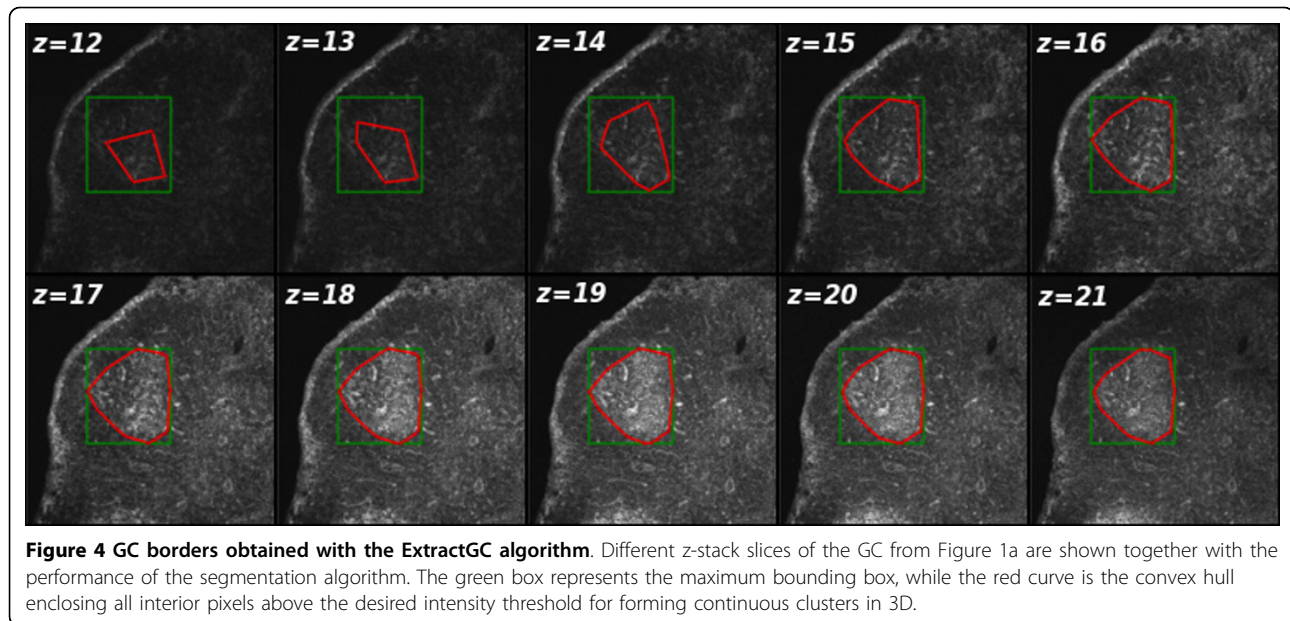
Optimal global threshold

Two fundamental parameters of the findspot algorithm (as developed by Goldberg *and col.*) are the pixel threshold t_h ,

which determines which pixels are allowed into a contiguous cluster, and the minimum cluster size s_{min} (or spot size), which provides a final cut-off on contiguous volume region. The threshold may be a global parameter or based upon the mean pixel (or even more sophisticated statistical-based methods, which for our purpose are not effective). With fluorescent microscopy, the intensity is directly proportional to the amount of B-cell membrane marker or receptor molecules, which is relatively homogeneous throughout the volume. Thus, it is sensible that a global threshold should be used since it will provide the most accurate indication of the amount of cells of a particular type at a particular z-slice. Also, a proper segmentation of the GC areas on each slice will be sensitive to an optimal selection of the initial values of t_h and s_{min} , where each depends upon the other.

First, it is useful to understand the effect of the global threshold t_h and s_{min} parameters upon the final segmentation of GC volumes, and in particular why this selection is non-trivial. For segmenting the central part of a GC, as seen in Figure 4, slice 19, a particular threshold will perform well; however for the image slices at the extremes (in the z-plane), it becomes unclear which parameter values lead to the best segmentation results. Indeed, if the threshold is too low, pixel clusters will be unnecessarily too large (possibly selecting the entire image). However, if the threshold is too high, or optimized to the center of the GC where the fluorescent contrast is max, then on higher/lower z-stack slices, the GC borders, and hence GC volumes, will be underestimated. With respect to the selection of the minimum spot size parameter, a small minimal spot size will result in many pixel clusters that are not germinal centers.

For the specific case of segmenting GCs from multidimensional images, we can use biological information to



guide the choice of an appropriate objective function. In particular, it is well known that by staining tissue samples with flourochrome-tagged peanut agglutinin, GCs will be brightly labeled throughout the volume and consist of fluorecently marked B cells that are involved in the immune response, while adjacent regions are characterized by a pronounced dark ring or halo. This dark outer ring zone is due to both follicular B cells not participating in the immune response (and, therefore, are not antibody-flourochrome labeled) and to T and dendritic cells of the adjacent T-cell zone, which are also unlabeled.

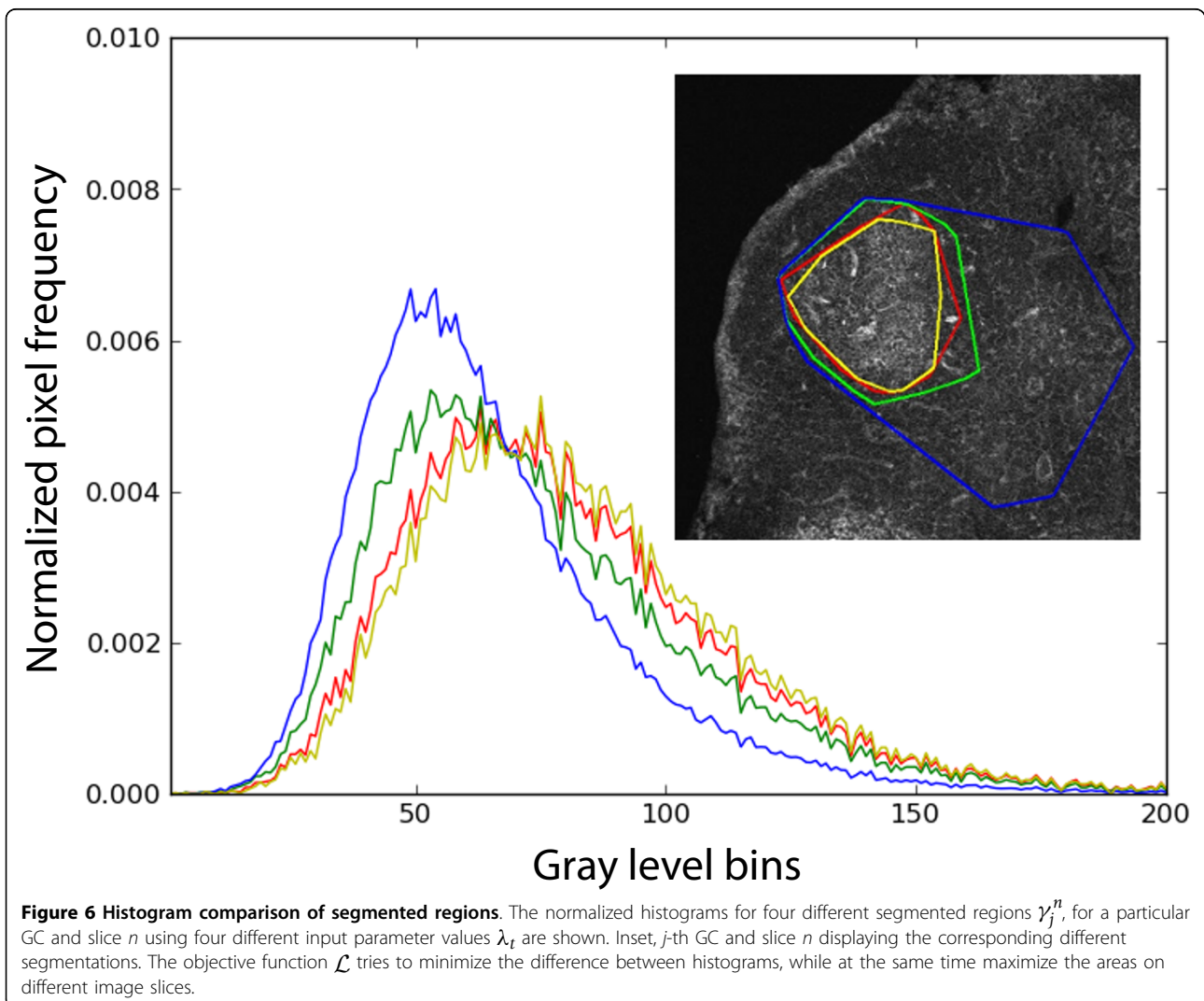
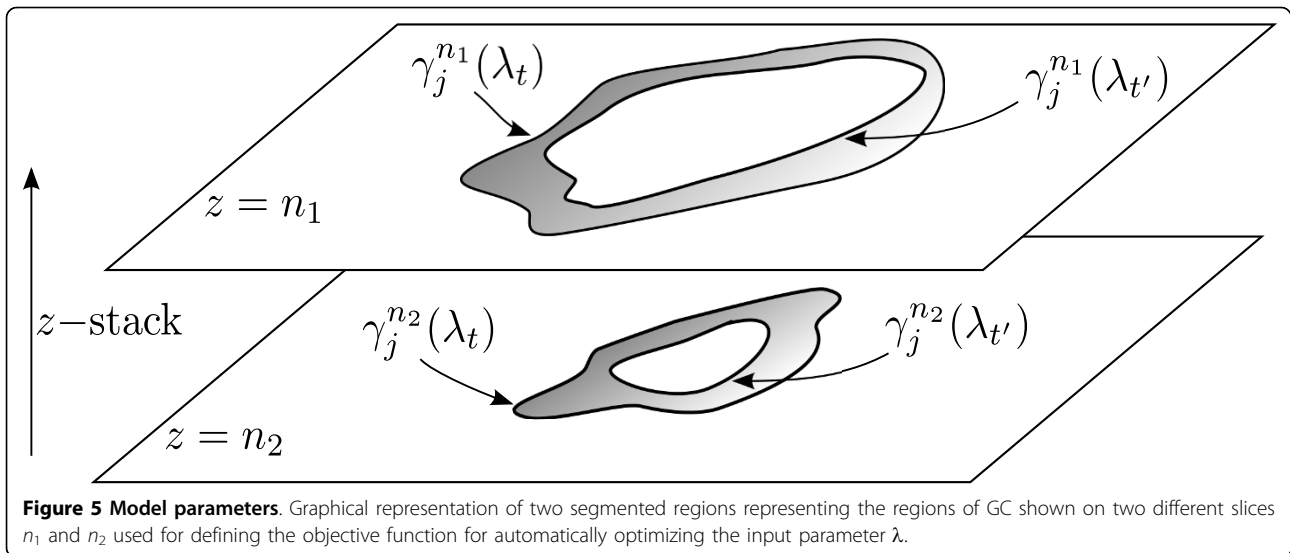
Given this nearly universal observed GC structure, an ideal segmentation algorithm for GCs will include all pixels up to and including the border adjacent to the dark halo zone. Since the global threshold parameter of our algorithm directly controls this segmentation, our algorithm optimizes this choice of threshold that segments the GC by the use of an iterative procedure, driven by a simulated annealing algorithm that minimize the objective function, applied to all z-slices. This objective function seeks a minimum in the sum of pixel-tone histogram differences for all z-slices, between different values of the input parameters, while at the same time strongly penalizing solutions that give rise to segmented borders outside the GC region. The algorithm for optimizing the input parameters can be formalized by referring to Figure 5 as follows. First, let $\theta_i (i = 1 \dots m)$ represent the set of m input parameters to be optimized. We denote the set of values of these parameters at the iteration step t as $\lambda_t = \{\theta_i^t\}$. Next, we consider GC segmentation regions obtained from the findspot algorithm. The segmentation region of the j -th GC on the n -th image slice and at iteration step t is denoted by

$\gamma_j^n(\lambda_t)$. Similarly, the segmentation region of the j -th GC region on the n -th slice at iteration t' is given by $\gamma_j^n(\lambda_{t'})$. From this, we can obtain the fraction of the number of points above the threshold N_f in the annular region between $\gamma_j^n(\lambda_t)$ and $\gamma_j^n(\lambda_{t'})$ as compared to the total number of pixels N_t in that annular region, as N_f/N_t . Moreover, for each of these regions, we can obtain the pixel-tone histogram, denoted $H(\gamma_j^n(\lambda_t)) \equiv H_j^n(t)$. The histograms for different segmented regions γ_j^n , for a given GC and a particular slice n corresponding to different input parameters λ_t are shown in Figure 6. As can be seen from this figure, the difference between histograms decreases as the segmented regions are closer. An optimal solution, found from the optimal input parameters λ_t^* would produce a segmentation that wraps tightly around the GC. Conversely, the value of the input parameters λ_t should tend to maximize the individual areas of the segmented regions on all z-stack slices of each GC. Thus, the objective function should penalize those values of the input λ_t that eliminate areas at the extremes of the GC volumes where the pixel intensities are at the limit of threshold. This tradeoff provides a convex objective function for applying an optimization strategy.

First, we use the Bhattacharyya histogram distance metric

$$D(H_t, H_{t'}) = \sqrt{1 - \frac{\sum_{k,k'} \sqrt{H_t(k) \cdot H_{t'}(k')}}{\sqrt{\sum_k H_t(k) \cdot \sum_{k'} H_{t'}(k')}}}$$

where H_t and $H_{t'}$, represent $H_j^n(\lambda_t)$ and $H_j^n(\lambda_{t'})$, and k and k' represent the individual bins in each histogram, respectively.



Now, let $a_j^n(\lambda_t)$ be the area of the segmentation region $\gamma_j^n(\lambda_t)$, that is, corresponding to the j -th GC at slice n using input parameters λ at iteration t , and let

$$A_j^n(\lambda_t, \lambda_{t'}) = \frac{|a_j^n(\lambda_t) + a_j^n(\lambda_{t'})| - |a_j^n(\lambda_t) - a_j^n(\lambda_{t'})|}{|a_j^n(\lambda_t) + a_j^n(\lambda_{t'})|}$$

which is a symmetric function, since $A_j^n(\lambda_t, \lambda_{t'}) = A_j^n(\lambda_{t'}, \lambda_t)$. Then, we can define the objective function \mathcal{L}_j for the j -th GC as follows:

$$\mathcal{L}_j(\lambda_t, \lambda_{t'}) = \frac{\sum_n \exp[\alpha D(H_j^n(\lambda_t), H_j^n(\lambda_{t'}))]}{\varepsilon + \beta \sum_n A_j^n(\lambda_t, \lambda_{t'})}$$

where ε is a small nonzero constant that we insert to prevent division by zero error, while α and β are arbitrary constants (we have used $\alpha \sim 0.001$ and $\beta = 1.0$) that could be useful for controlling the strength of either the histogram difference or the sum over areas, respectively. With this function, the optimization is then with respect to the input parameters $\lambda_t = \{\theta_i^t\}$, that is, $\partial \mathcal{L}_j / \partial \theta_i = 0$. Notice that if the areas $a_j^n(\lambda_t)$ and $a_j^n(\lambda_{t'})$ on each slice n for each segmentation region j are very different —either because $a_j^n(\lambda_t)$ is much larger than $a_j^n(\lambda_{t'})$ (or vice-versa) or because, suddenly, $a_j(\lambda_t) = 0$ ($ora_j(\lambda_{t'}) = 0$) due to the disappearance of the contour at slice n —, then $\mathcal{L}_j(\lambda_t, \lambda_{t'})$ and $\mathcal{L}_j(\lambda_{t'}, \lambda_t)$ grows to very large values, thereby penalizing \mathcal{L}_j . Conversely, if $a_j^n(\lambda_t) = a_j^n(\lambda_{t'})$ (or are very similar) then $A_j^n(\lambda_t, \lambda_{t'}) \rightarrow 1$.

In general, the function $\mathcal{L}_j(\lambda_t, \lambda_{t'})$ is a nonlinear multi-dimensional function with many local minimima, many of which are not ideal solutions. In order to understand the behavior of \mathcal{L}_j as a function of λ_t and $\lambda_{t'}$, Figure 7(a) shows two hyperplane cuts with s_{\min} for different values of the threshold t_h . For constructing this plot, we chose consecutive values of the threshold, with $t'_h = t_h + 1$ (red curve), and $t'_h = t_h + 2$ (green curve). As can be seen, in both planes, the function experiences a dramatic global minimum for the optimal solution.

From this objective function, we use a simulated annealing algorithm that efficiently samples the space of all possible λ_t in order to find the optimal set of input parameters, λ^* , given by:

$$\lambda^* = \arg \min_{\lambda_t} \mathcal{L}_j(\lambda_t, \lambda_{t'})$$

In order to show how robust our optimization algorithm is with respect to the choice of initial input parameters, Figure 7(b) shows the difference in accumulated area (which is related to the GC volume) between the calculated and ground truth value for several iterations of the algorithm for three separate initial values of λ . In these studies, the *ground truth* determination was obtained from manual inspection by an expert. Figure 7(c) shows

a comparison, superposed on a particular Germinal Center image, between borders obtained with optimal parameter solution, λ^* , using our algorithm and the ground truth border obtained by manual determination.

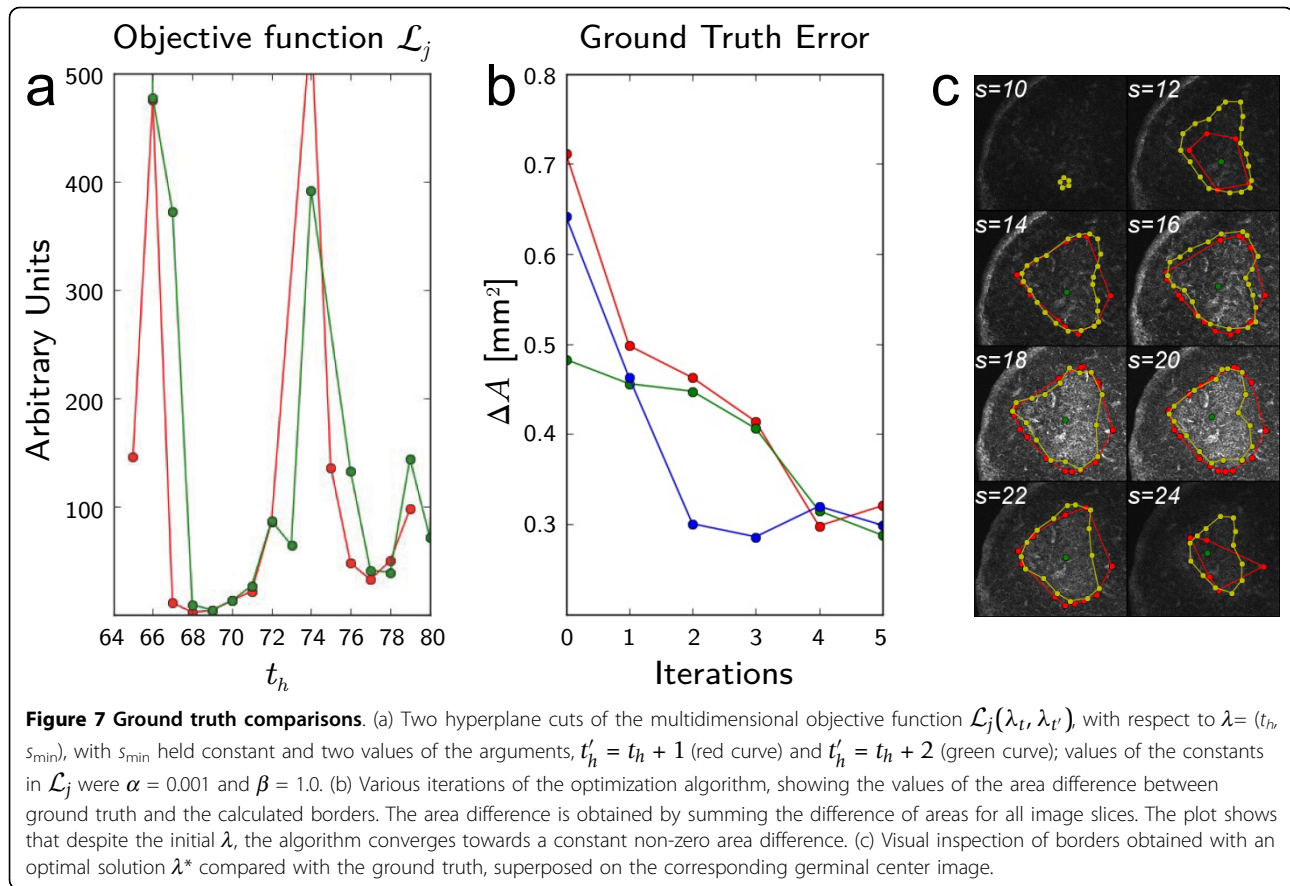
Since the original *findspot* algorithm finds all contiguous clusters of pixels throughout a volume, connected regions can be filled with holes. By using a convex hull algorithm, or more sophisticated computational geometry algorithms based upon alpha shapes, we can represent and visualize the 3-dimensional GC volumes with the outer bounding surface. Nearby artifacts due to outliers points may be present, distorting the volume estimate, and should be corrected. We eliminate outliers by a simple heuristic algorithm that determines the full distance matrix between all points on the contour and determines whether the distance between each point and all others is greater than $2 \times \sigma$ value of all other inter point distances (where σ is the standard deviation). Conversely, we can find the geometric center and determine whether a point is $2 \times \sigma$ from that center.

Optimal stitching Our software **pyBioImage** also contains a module for automatic stitching of multi-dimensional images, similar to that found in ImageJ. Side-by-side z-stack images of draining lymph nodes were acquired to allow 3D reconstructions of larger organ areas. Due to the large amount of image stacks, we developed our own software algorithms that used information from the microscope position and accelerated the task of forming large image mosaics, referred to as image stitching, from adjacent z-stacks acquisitions.

For matching adjacent image stacks, our algorithm uses a fast implementation of the Fourier phase correlation technique for achieving image registration at the borders of adjacent (and overlapping) images. For blending adjacent images, we use a nonlinear pyramid scheme together with pixel intensity scaling for matching potential differences in acquisition exposures. The implementation of our algorithm is available in our cross-platform **pyBioImage** package, available at the public repository (sourceforge.net/projects/pybioimage/). Information about the installation, documentation, and other software modules (whose description is beyond the scope of this paper), can also be found in the package distribution.

3D reconstruction

Another capability of the *ExtractGC* module is the ability to accurately visualize the GC volumes in 3D. The reconstruction of the set of borders pixels obtained from each z-stack slice is used for constructing an isosurface with a computational geometry algorithm, called Powercrust, described by Amenta, Choi and Kolluri [28,29]. We have provided a full set of python bindings to the original open-source C-language implementation of these authors in order to easily expose the core algorithm to our application, **pyBioImage**. The output of powercrust, with the



points obtained from the findSpot algorithm in the prior phase, is a set of files that specify the polygons and their vertex locations in 3D that define the maximum bounding surface. Presently, we have maintained the original powercrust related .off extension file structure which can be visualized with Geomview, an interactive 3D viewing program for Unix. An example 3D renderization of a GC is shown in Figure 8.

Details of the 3D rendering algorithm are as follows: the algorithm takes point samples from a 3D object's surface and produces both a surface mesh and an approximate medial axis. The powercrust algorithm is based on the Medial Axis Transformation (MAT) that provides a complete description of the object's shape through maximally inscribed discs. Together with the Voronoi diagram computation its duality, and its weighted adaptations, the powercrust algorithm produces the so-called power diagram. Then, the algorithm can be described briefly with the following steps [30]: (1) given a collection of sampled points, a bounding box is used to enclose what shall be the 3D object, (2) a Voronoi Diagram is computed and pole computation for each sample point is made, (3) each pole is analyzed and labeled with its relation to the Voronoi surfaces, and

finally (4) the algorithm provides output of the powercrust and powershape parameters, that can be used for constructing polygons of the external surface.

While there are several other fundamental algorithms for 3D reconstruction, including Alpha Shapes (a generalization of the convex hull algorithm by Edelsbrunner), Marching Cubes, Voronoi-based algorithms, and Delaunay-based algorithms, we found the Crust/Powercrust algorithms the most effective for our application.

Results and discussion

The architecture of pyBioImage, together with the *ExtractGC* module, is designed to provide a productive and intuitive workflow for the experimental and theoretical biologist for extracting accurate GC statistics.

Germinal center image acquisition

In order to test our software, we applied our algorithms to a set of GC image data acquired with typical experimental conditions. In particular, Balb/c mice maintained in SPF facilities were immunized intraperitoneally with 20 μg of OVA (Sigma, St Louis, USA) previously run through a DetoxGel column (Pierce, Rockford, USA) in 2.0 mg of endotoxin-free aluminum hydroxide (alum, Alu-gel-S,

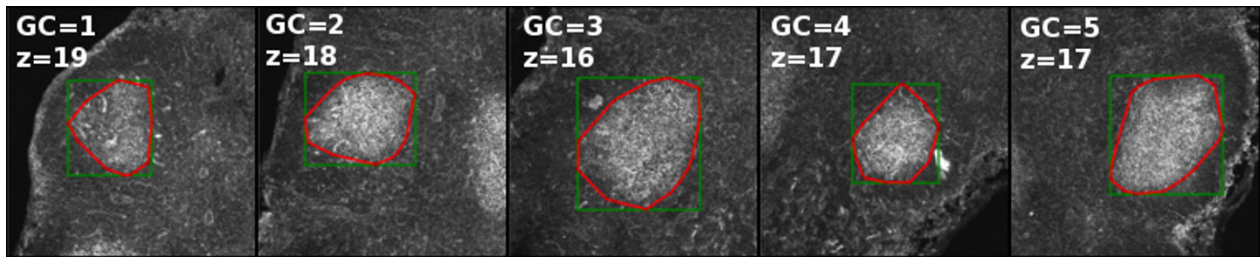


Figure 8 Examples of borders extracted for different germinal centers. *ExtractGC* analysis on five images of different germinal centers taken from the same specimen shown in the mosaic of Figure 1b. The images show the border (red curves) at slices in the center of each germinal center.

Serva, Heidelberg, Germany). Seventeen days after immunization the draining lymph nodes were excised and fixed with PFA (Sigma). 50 μm vibratome sections from fixed tissue were stained with the following primary antibodies: rabbit anti-CD3 (Abcam), rat anti-IgM-TxRd (Southern-Biotech, Birmingham, USA), and PNA-FITC (Vector, Burlingame, USA). Anti-rabbit immunoglobulin-alexa647 (Invitrogen, Carlsbad, USA) was used as secondary antibody.

Once the regions of interest were located, 35 images were acquired at 1.43 μm z-steps, using a LSM710 confocal microscope (Zeiss, Jena, Germany) equipped with a 20 \times (0,80 NA, Zeiss) objective. Several images were acquired across a relatively large section of the specimen, such that each image contained at least one GC, and the set of all images formed a mosaic (with an irregularly ordered tiling).

Evaluation

From the data, prepared as described above, we analyzed four independent data sets that represent magnified regions of small sections of lymph nodes. For each specimen, 5 GCs were imaged independently, with a slight overlap of the nearby image, so that a mosaic could be formed. The images consisted of 4-color channels, were 512 \times 512, and contained an average of 30 z-stack slices. We used our algorithm to automatically collect GC statistics by loading all the images in the directory and providing initial input parameter guesses for the pixel intensity threshold and the minimum spot size: $\lambda = (t_h, s_{\min})$. Results of extracting GCs for different datasets are shown in Figure 4, showing the contour encountered of the GC region at different z-stack slices.

The algorithms described are efficient, requiring no special hardware, and can run on any modern computer system. In order to appreciate the typical running times, we ran the algorithm on a standard laboratory computer (Intel Pentium D CPU 2.80GHz, with 2G Memory), and execution times to process multi-dimensional images with sizes 512 \times 512 \times 35 never exceeded from 1.2 s and

the execution time for the optimization step was always below 0.5 s for different image sizes.

Figure 9 illustrates the visualization features of pyBioImage/ExtractGC. The 2D window, allows the user to maneuver through the image stacks slice by slice, with the segmented contour superposed on the image. A 3D visualization allows the user to interactively manipulate the GC volume from all angles, and provides a more accurate calculation of the GC volume.

Conclusions

Our application, **pyBioImage** with the *ExtractGC* module provides fully automatic and accurate estimates of GC volumes from an arbitrarily large collection of multidimensional images. The framework pyBioImage leverages the relatively recent availability of high quality scientific software based upon python for rapid development of complex image and computation. As such, our application is positioned to tackle several problems described in this paper not provided by standard open-source solutions, such as Fiji/ImageJ. The *ExtractGC* module is a relevant bioinformatics tool that should be of interest to scientists working with confocal and 2-photon microscopy imaging and has also served to be a proof of concept module for integrating specific applications within our general software framework. Given the usefulness of the *ExtractGC* module, we are presently planning to also release a version of the algorithm for both the ImageJ as well as OMERO projects.

Availability and requirements

Project name: e.g. **pyBioImage** package

Project home page: <http://sourceforge.net/projects/pybioimage/>

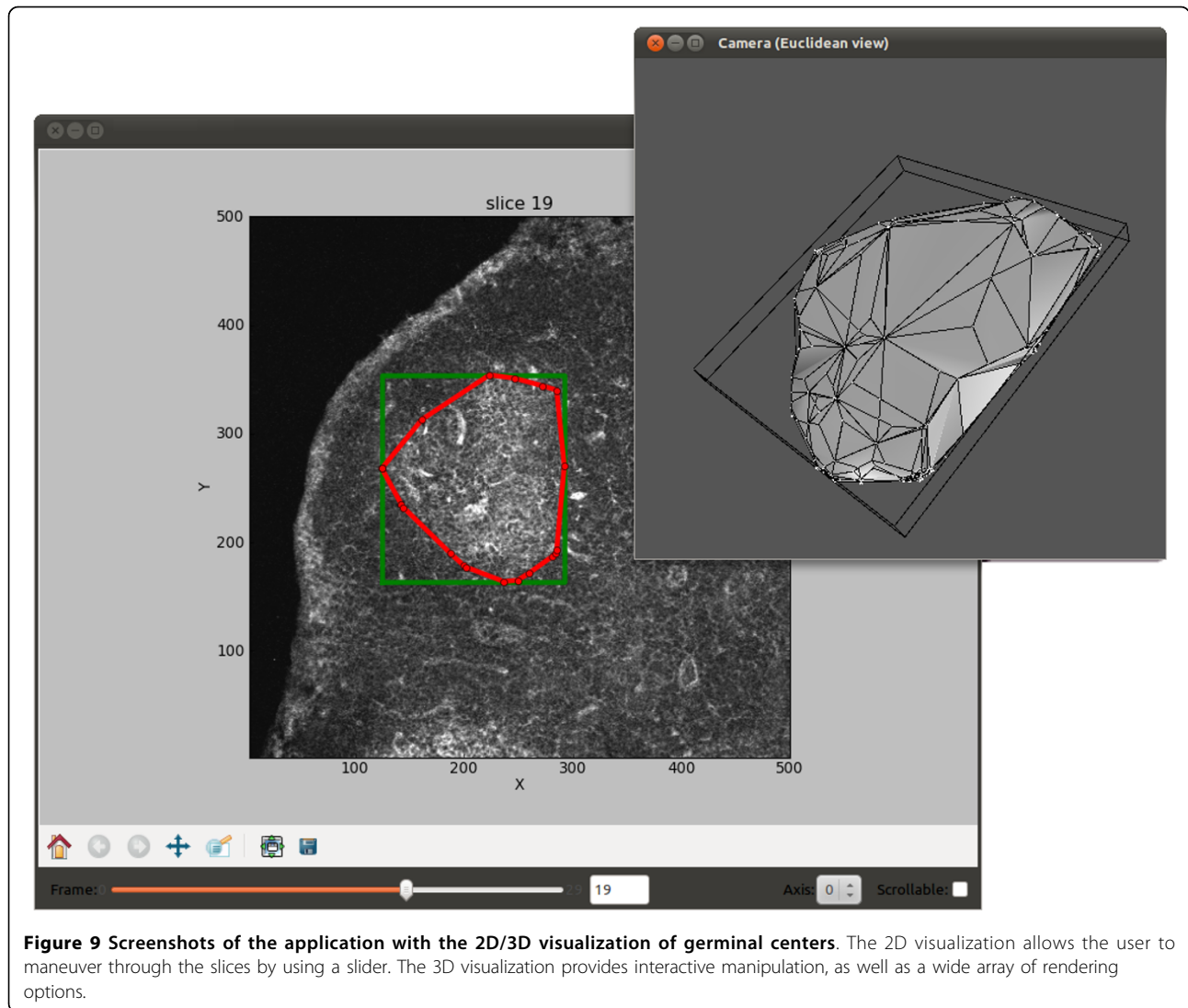
Operating system(s): Platform independent

Programming language: python, C

Other requirements:

License: GNU GPL

Any restrictions to use by non-academics: license needed



List of abbreviations

T_Hreg: follicular regulatory CD4⁺ T lymphocytes; GC: germinal center; 2D: 2-dimensional; 3D: 3-dimensional; 4D: 4-dimensional.

Authors' contributions

JF and DO conceived, designed and developed the study, DO and ME implemented the software code, JF, DO and ME wrote the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

Confocal microscopy images were generously provided by Ivonne Wollenberg, Alexandre Santos and Luis Graça, from the IMM, University of Lisbon (Portugal). This work was supported by grants SAF2007-63152 (MICINN, Spain) and PIRSES-GA-2008-230665 (7th FP, EC) to JF.

Declarations

This article has been published as part of *BMC Bioinformatics* Volume 14 Supplement 6, 2013: Selected articles from the 10th International Conference on Artificial Immune Systems (ICARIS). The full contents of the supplement are available online at <http://www.biomedcentral.com/bmcbioinformatics/supplements/14/S6>.

Author details

¹School of Computer Engineering, University of Vigo, Ourense, 32004, Spain. ²Department of Biochemistry, Genetics and Immunology, University of Vigo, Vigo, 36310, Spain. ³Estudos Avançados de Oeiras, Instituto Gulbenkian de Ciência, Oeiras, 2781-901, Portugal.

Published: 17 April 2013

References

1. Eisen HN, Siskind GW: Variations in Affinities of Antibodies during the Immune Response. *Biochemistry* 1964, **3**:996-1008.
2. MacLennan IC: Germinal centers. *Annu Rev Immunol* 1994, **12**:117-139.
3. Kelsoe G: The germinal center: a crucible for lymphocyte selection. *Semin Immunol* 1996, **8**(3):179-184.
4. Allen CD, Okada T, Cyster JG: Germinal-center organization and cellular dynamics. *Immunity* 2007, **27**(2):190-202.
5. Wittenbrink N, Klein A, Weiser AA, Schuchhardt J, Or-Guil M: Is There a Typical Germinal Center? A Large-Scale Immunohistological Study on the Cellular Composition of Germinal Centers during the Hapten-Carrier-Driven Primary Immune Response in Mice. *J Immunol* 2011, **187**(12):6185-6196.
6. Manser T: Textbook germinal centers? *J Immunol* 2004, **172**:3369-3375.

7. Wollenberg I, Agua-Doce A, Hernandez A, Almeida C, Oliveira VG, Faro J, Graca L: **Regulation of the germinal center reaction by Foxp3+ follicular regulatory T cells.** *J Immunol* 2011, **187**(9):4553-60.
8. Chung Y, Tanaka S, Chu F, Nurieva RI, Martinez GJ, Rawal S, Wang YH, Lim H, Reynolds JM, Zhou XH, Fan HM, Liu ZM, Neelapu SS, Dong C: **Follicular regulatory T cells expressing Foxp3 and Bcl-6 suppress germinal center reactions.** *Nat Med* 2011, **17**(8):983-8, [Doi: 10.1038/nm.2426].
9. Linterman MA, Pierson W, Lee SK, Kallies A, Kawamoto S, Rayner TF, Srivastava M, Divekar DP, Beaton L, Hogan JJ, Fagarasan S, Liston A, Smith KG, Vinuesa CG: **Foxp3+ follicular regulatory T cells control the germinal center response.** *Nat Med* 2011, **17**(8):975-82, [Doi: 10.1038/nm.2425].
10. Saalfeld S, Cardona A, Hartenstein V, Tomancak P: **As rigid as possible mosaicking and serial section registration of large ssTEM datasets.** *Bioinformatics* 2010, **26**(12):i57-i63.
11. Cardona A, Saalfeld S, Preibisch S, Schmid B, Cheng A, Pulokas J, Tomancak P, Hartenstein V: **An integrated micro- and Macroarchitectural Analysis of the Drosophila Brain by computer-Assisted serial section Electron Microscopy.** *PLoS Biol* 2010, **8**(10):e1000502.
12. Ourselin S, Roche A, Subsol G, Pnec X, Ayache N: **Reconstructing a 3d structure from serial histological sections.** *Image and Vision Computing* 2001, **19**:25-31.
13. Walter T, Shattuck DW, Baldock R, Bastin ME, Carpenter AE, Duce S, Ellenberg J, Fraser A, Hamilton N, Pieper S, Ragan MA, Schneider JE, Tomancak P, Hriche JK: **Visualization of image data from cells to organisms.** *Nat Methods* 2010, **7**(3 Suppl):S26-S41.
14. Swedlow J, Goldberg I, Eliceiri K, the OME Consortium: **Bioimage Informatics for Experimental Biology.** *Annu Rev Biophys* 2009, **38**:327-346.
15. Preibisch S, Saalfeld S, Tomancak P: **Globally optimal stitching of tiled 3D microscopic image acquisitions.** *Bioinformatics* 2009, **25**(11):1463-1465.
16. Arganda-Carreras I, Sorzano C, Marabini R, Carazo J, Ortiz de Solorzano C, Kybic J: **Consistent and Elastic Registration of Histologic sections using vector spline regularization.** *Lecture Notes in Computer Science, Springer, CVAMIA* 2006, **4241**:85-95.
17. Zitová B, Flusser J: **Image registration methods: a survey.** *Image and Vision Computing* 2003, **21**(11):977-1000 [http://www.sciencedirect.com/science/article/pii/S0262885603001379].
18. Bagci U, Bai L: **Registration of standardized histological images in feature space.** *Proceedings of SPIE, Volume 6914 Spie*; 2009, 69142V-69142V-9 [http://arxiv.org/abs/0907.3209].
19. McInerney I, Terzopoulos T: **Deformable Model in Medical Image Analysis: A Survey.** *Medical Image Analysis* 1996, **1**(2):91-108.
20. Laasmaa M, Vendelin M, Peterson P: **Application of regularized Richardson-Lucy algorithm for deconvolution of confocal microscopy images.** *J Microsc* 2011, **243**(2):124-140.
21. Kankaanpää P, Pahajoki K, Marjomäki V, Heino J, White D: **BioImageXD - New Open Source Free Software for the Processing, Analysis and Visualization of Multidimensional Microscopic Images.** *Microscopy Today* 2006, **14**(3):12-16.
22. Schiffmann D, Dikovskaya D, Appleton P, Newton I, Creager D, Allan C, Nathke I, Goldberg I: **Open Microscopy Environment and FindSpots: integrating image informatics with quantitative multidimensional image analysis.** *BioTechniques* 2006, **41**:199-207.
23. Amenta N, Choi S, Kolluri R: **The power crust, unions of balls, and the medial axis transform.** *Computational Geometry: Theory and Applications* 2001, **19**(2-3):127-153.
24. Linkert M, Rueden C, Allan C, Burel J, Moore W, Patterson A, Loranger B, Moore J, Neves C, Macdonald D, Tarkowska A, Sticco C, Hill E, Rossner M, Eliceiri K, Swedlow J: **Metadata matters: access to image data in the real world.** *J Cell Biol* 2010, **189**(5):777-782.
25. Shapiro LG, Stockman GC: *Computer vision* Upper Saddle River, NJ: Prentice Hall; 2001.
26. Szeliski R: *Computer Vision: Algorithms and Applications* Berlin, Germany: Springer; 2011.
27. Bagci U, Chen X, Udupa J: **Hierarchical Scale-Based Multiobject Recognition of 3-D Anatomical Structures.** *IEEE Trans Med Imaging* 2012, **31**(3):777-789.
28. Amenta N, Bern M, Eppstein D: **The Crust and the Beta-Skeleton: Combinatorial Curve Reconstruction.** *Graphical Models and Image Processing* 1998, **60**:125-135.
29. Amenta N, Choi S, Kolluri R: **The power crust.** *Proceedings of the sixth ACM symposium on Solid modeling and applications SMA '01*, New York, NY, USA: ACM; 2001, 249-266 [http://doi.acm.org/10.1145/376957.376986].
30. Amenta N, Bern M: **Surface Reconstruction by Voronoi Filtering.** *Discrete and Computational Geometry* 1999, **22**:481-504.

doi:10.1186/1471-2105-14-S6-S5

Cite this article as: Olivieri et al.: Software tool for 3D extraction of germinal centers. *BMC Bioinformatics* 2013 **14**(Suppl 6):S5.

Submit your next manuscript to BioMed Central
and take full advantage of:

- Convenient online submission
- Thorough peer review
- No space constraints or color figure charges
- Immediate publication on acceptance
- Inclusion in PubMed, CAS, Scopus and Google Scholar
- Research which is freely available for redistribution

Submit your manuscript at
www.biomedcentral.com/submit

