Segmentation of zebrafish vasculature
- Technical report

Sophie Schneider Max Planck Institute of Molecular Cell Biology and Genetics, MOSAIC group

July 23, 2015

This technical describes a pipeline to segment 3-dimensional images of the vasculature of the zebrafish acquired by Selective Plane Illumination microscopy (SPIM) and a way to represent them as a 3-dimensional graph structure to enable easy comparison of different vascular systems.

Introduction

The zebrafish Danio rerio is a well suited model organism to study developmental processes by using microscopy, especially because of its fast development outside of the mother and its transparency. In collaboration with Stephan Daetwyler from the Husiken lab (MPI-CBG), we aim to understand how the vasculature is formed and thus need a tool to quantify it. Therefore we aimed to establish an image analysis pipeline which automatically segments the vasculature. To enable easy quantification and comparison between different developmental stages as well as different phenotypes we implemented a graph representation of the vasculature.

Data

The image data is acquired by Stephan Daetwyler from the Huiskon lab at MPI-CBG. He uses Selective Plan Illumination Microscopy (SPIM) to image the organism over a long period of time [1]. The best results so far are obtained when using an flk-marker to label the endothelial cells which are building the walls of the vessels.

Microangiography as used in previous research [2] was tried by Stephan Daetwyler as well, but without success. Over time the dye diffuses into the surrounding tissue, which makes an analysis not possible.

Image analysis

The image analysis is partly conducted in MATLAB and Fiji. By using Miji [3], the Fiji pipeline can be integrated to MATLAB. Before the actual segmentation is conducted, the images are preprocessed to remove artifacts from the imaging.

Preprocessing

Preprocessing consists of two part, a dehazing of the images and an intensity equalisation. The effect of both preprocessing steps is evaluated by using a manual segmentation of a cropped original image stack and calculation of the Dice coefficient. Firstly, a dehazing algorithm is used to remove scatter light in the images. This is conducted slice-wise on the whole stack. The effect of dehazing is examined in a cropped part of the original images in Tab. 1. Additionally to the Dice coefficient, the volume and absolute volumetric difference (AVD) is calculated. All three measures suggest an improvement of the segmentation of the dehazed image compared to the non dehazed images. Therefore, dehazing is included in the final pipeline (Fig. 3).

After dehazing the image stack, oversaturated pixels are equalised. This is, all pixels above a predefined threshold are set to this threshold. Threshold values from 10 to 200 are evaluated. Based on the Dice coefficient for each threshold value (Tab. 2), a threshold value of 40 is recommended and thus included in the final pipeline.

The most important step in the preprocessing is applying a tubeness filter [4] on the preprocessed stack. The tubeness filter measures how 'tube-like' a structure is. Because we are looking for vessels which are tube-like structures, this is particular useful for the image analysis task. The problem when using the tubeness-filter is discussed in the Conclusion-section.
Table 1: Comparing the effect on segmentation quality of dehazing the image stack to not dehazing suggests to include dehazing in the preprocessing pipeline. The remaining parameters are set to: equalisation of 30, tubeness of 4 and adaptive thresholding of 30.

<table>
<thead>
<tr>
<th>threshold</th>
<th>10</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
<th>45</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dice</td>
<td>0.43</td>
<td>0.73</td>
<td>0.74</td>
<td>0.71</td>
<td>0.76</td>
<td>0.75</td>
</tr>
<tr>
<td>Dice</td>
<td>0.75</td>
<td>0.74</td>
<td>0.74</td>
<td>0.71</td>
<td>0.66</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Comparing the effect on segmentation quality of different equalisation thresholds suggest a use of a threshold value of 40. The remaining parameters are set to: dehazing, tubeness of 4 and adaptive thresholding of 30.

Binarization

After applying the preprocessing pipeline to the image stack, the main part is done. For obtaining the final segmentation mask, the image stack needs to be binarized. This is done by applying an adaptive thresholding. The base threshold of 30 is used. The remaining parameters are set to the default values (Mask diameter = 3, Local weight = 5)

Postprocessing

Out of the final segmentation mask, a 3D graph can be extracted. This should make the comparison between different phenotypes as well as different developmental stages easier. The graph extraction is done in MATLAB and based on a depth-first search (DFS). Before the graph is constructed, a skeleton is extracted from the binary segmentation mask using Fiji’s Skeletonize (2D/3D) plugin such that the resulting structure is 1-pixel-wide. Furthermore, only the largest connected component of the skeleton is kept, assuming the vasculature to be a connected network. These are prerequisites for the algorithm to work. As a next step, terminal points (pixels with only one neighbor) and junction points (pixels with more than two neighbors) are determined and labeled accordingly. Because only one pixel per junction is needed, junctions with more than one junction point are treated separately. First, all junctions with more than one identified junction point are identified by looking at clusters. Then, for each junction, all junction points are weighted according to the following scheme: \( W = 4 \times Fc + 2 \times Ec + Vc \), with \( Fc \) being the number of face-connected neighbors, \( Ec \) being the number of edge-connected neighbors and \( Vc \) being the number of vertex-connected neighbors [5]. The junction point with the highest weight \( W \) is kept as the only junction point for the considered junction. The remaining ones are still stored as additional junction points and matched to the main junction point. In the main part of the algorithm, first each pixel is considered to be a node. In the final adjacency matrix, the number of nodes corresponds to the number of terminal and (main) junction points. A depth-first-search (DFS) from each terminal or (main) junction point is conducted. Once the DFS reaches a terminal or junction point (main or additional), the search is terminated and an edge between both terminal/junction points is drawn. Additionally, the edge then is weighted according to its length. Alternatively, this weight could be changed e.g. to the diameter of the vessels if a reliable way to determine it can be found.

Visualization

A nice way to visualize the final segmentation is by creating a mesh structure of the mask. For
Figure 2: 3D graph representation of the cropped zebrafish vasculature

Figure 3: Final pipeline of vasculature segmentation.

this, firstly, the isosurfaces are calculated in Matlab. Based on this, vertices and faces of the triangulated mesh are read into the software MeshLab [6]. A screenshot of the visualization is shown in Fig.4.

Figure 4: Close-up view of the mesh representation of a cropped part of the zebrafish vasculature.

Conclusion

In summary, a pipeline to automatically extract the vascular system from 3D images of the zebrafish Danio rerio was established. Due to imaging modalities, this pipeline can only be applied to a small part of the fish, in which the illumination is equally good. A main problem is, that some vessels appear non-continuous, most probably due to the orientation of the light sheet and the used labeling, which concentrates on cell nuclei of the epithelial cells surrounding a vessel. To fully segment a complete vasculature a different image analysis approach is needed.

Scripts

In this section the most important scripts are explained. Smaller scripts, e.g. in the bascis - folder are self-explanatory and therefore not mentioned.

main_fish.m This main script executes the whole pipeline from reading in the raw stack to writing the mesh representation for visualization purposes.

dehaze_main.m This script executes the dehazing of the raw stack frame-by-frame and writes the dehazed stack as well as a comparison as an output. As an input, the file name (without file extension) has to be specified.

equalise.m This script does a pixel intensity equalisation frame-by-frame. All pixels values that lie above the equalisation threshold are set to the threshold value. It outputs the equalised stack and writes the images stack out. As an input the image stack, threshold and file name have to be specified.

vessel_graph.m This script creates a 3-dimensional graph representation of the skeletonized image stack. It outputs the adjacency matrix and plots the 3D graph. As an input the skeletonized image stack as well as the file name have to be specified.

Fiji plugins

The following plugins have to be installed into the local Fiji distribution:

Tubeness This plugin is used to filter the 3D stack to calculate how ‘tube-like’ the structures in the image are. It is based on calculating eigenvalues from the Hessian matrix described in [4].

Adaptive 3D threshold This plugin uses a dynamical threshold over the image instead of the conventional global threshold. This can account e.g. for changing lighting conditions.

Skeletonize 2D/3D This plugin is used to create a 3D skeleton from the binary segmentation mask. The algorithm is based on the thinning algorithm described in [7].
References


