Interactive FIJI/ImageJ2 Plugin for Biological Image Segmentation Case Study: Wound Healing Analysis

Nurzhan Sakenov
Department of Computer Science
Nazarbayev University
Astana, Kazakhstan
nurzhan.sakenov@nu.edu.kz

Bekzhan Kaspakov
Department of Computer Science
Nazarbayev University
Astana, Kazakhstan
bekzhan.kaspakov@nu.edu.kz

Madiyar Katranov
Department of Computer Science
Nazarbayev University
Astana, Kazakhstan
madiyar.katranov@nu.edu.kz

I. Introduction

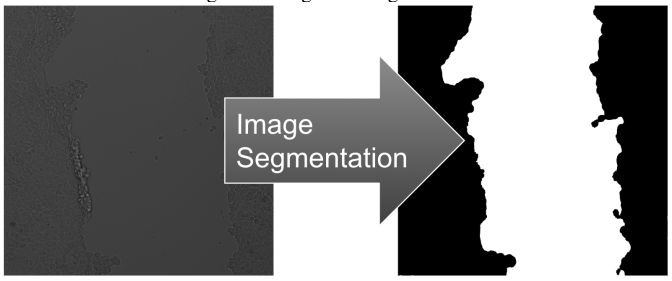
We are trying to provide a solution for automated high-throughput Wound Healing Analysis. Wound Healing Analysis is a widely used microbiological laboratory procedure beneficial for medical applications, such as drug research, cancerous cell analysis, and more. There are several hurdles in designing software for automatic image processing of datasets obtained from WHA, which resulted in the absence of working or usable software, as was discovered during our research. We attempt to overcome these hurdles by using adaptive Image Segmentation tactics, as well as providing potential users with the ability to design the segmentation pipeline themselves. Essentially, we want to provide the biomedical community with a powerful, yet easy-to-use tool that can be used to conduct research more effectively.

II. Background

In order to understand our project, one needs to be familiar with the framework of Wound Healing Analysis. WHA is used to investigate molecular mechanisms of cell migration and wound regeneration, and to research the effects of various drugs on the process. We consider the two-dimensional Wound Healing Analysis, where a layer of cells is grown on a surface, which is then wounded (scratched), and a cell imaging system (e.g. EVOS, ORCA) is used to produce a time-lapse dataset. The scratch produces a region free of cells, and the monolayer is effectively divided into two parts separated by the wound. During healing, cells proliferate and migrate into the wound, closing the gap with time [1]. To analyze a Wound Healing experiment, the obtained images must be segmented to isolate the wound. Biologists do it by hand, which works if a dataset is sparse (e.g. an image taken every 6-12 hours). However, such sparse datasets misrepresent the complexity of the metrics in the cell migration process, and for big datasets, doing the segmentation by hand is extremely time-consuming and error-prone. [2]

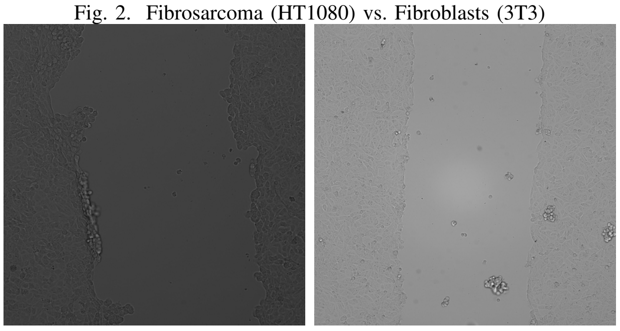
We performed some literature review and found out that there are several problems with data analysis. The existing software that has been used by some researchers is now unsupported and obsolete (e.g. TScratch), or is commercial with no guarantee that it works well on dense datasets. Furthermore, a lot of papers do insufficient reporting: in one case, a vague approach for image segmentation was outlined, but no concrete algorithms or source code was provided. In these cases, it was also common for the researchers to include misleading information, such as the frequency of image acquisition or the scalability of the approach, like in [3]. Other approaches require considerable programming skills, which some biologists do not have.

Fig. 1. The goal of segmentation



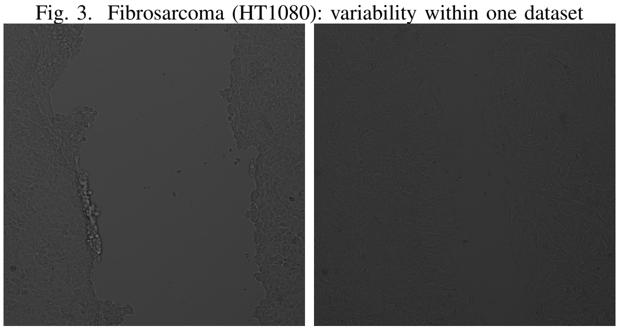
One of the main problems with Wound Healing image segmentation, as we discovered ourselves, is the high variability between and within datasets. Different cell lines and imaging systems may produce drastically differently-looking datasets.

Fig. 2. Fibrosarcoma (HT1080) vs. Fibroblasts (3T3)



Furthermore, even within a single dataset, there is some variability due to the wound's closure. The first image in the dataset could be an open wound, and the last could be an almost closed wound. Obviously, after processing both of these images with the same algorithm, their resulting histograms will differ considerably.

Fig. 3. Fibrosarcoma (HT1080): variability within one dataset



Our solution should account for as many variability sources as possible. The best solution we have at the moment is a prodecure suggested by Sholpan Kauanova, a PhD student in the Department of Biology, Nazarbayev University. Her approach, in a nutshell, revolves around transforming images such that their histograms go from a single-peak shape to a two-peaks-and-a-valley shape, and then the threshold is computed for each image individually. So far, it segments most datasets fairly well, however, problems with within-set variability and some low-contrast datasets remain unsolved.

Furthermore, loosely inspired by Algorithm Selection, we decided to give the users the ability to customize the segmentation pipeline, so that they could experiment and fine-tune the process for their dataset, if the main pipeline fails, as well as provide them with means to benchmark the effectiveness of a pipeline.

III. System Design and Architecture

We chose to extend the SciJava ecosystem, particularly FIJI/ImageJ2 (originally developed at the National Institutes of Health), since it is so widely used in the biomedical community [4]. SciJava provides a way to extend all software belonging to the ecosystem via Plugins, and that is the path we have taken.

ImageJ is currently in the phase of upgrading to ImageJ2, which uses a new image processing core (ImgLib2), so we decided to use ImgLib2 as the low-level tool to write algorithms for our plugin. SCIFIO is a library that provides input-output capabilities, and we used it to provide users with the ability to process directories without the need to load them into memory via ImageJ.

The plugin includes a convenient GUI that allows the users to specify IO paths, as well as tune parameters for the pipeline, as well as convenience options such as "output the sequence of the algorithm's steps".

Apart from the main segmentation module, we have a graph-based pipeline customization module that allows users to modify the segmentation process. The graph editing module

Fig. 4. The SciJava ecosystem

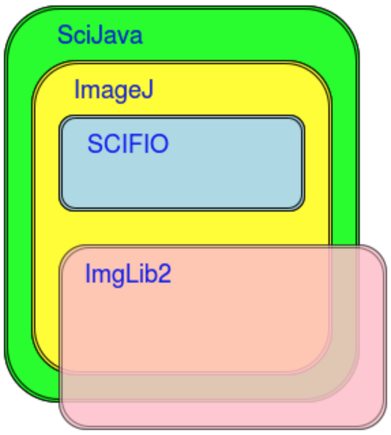
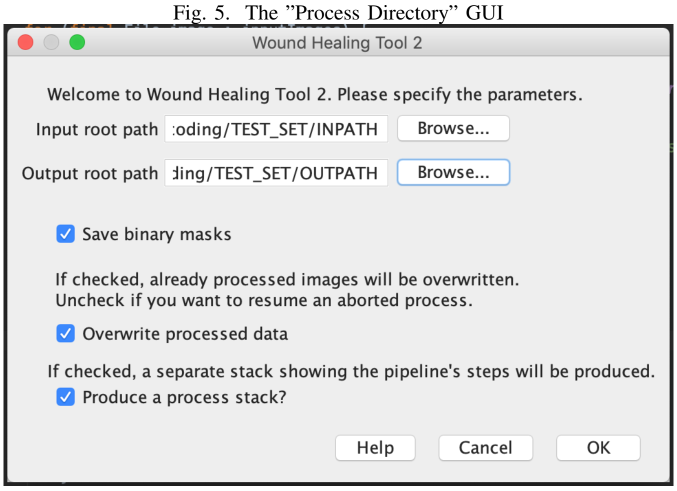


Fig. 5. The "Process Directory" GUI



Welcome to Wound Healing Tool 2. Please specify the parameters.

Input root path :oding/TEST_SET/INPATH Browse...

Output root path ding/TEST_SET/OUTPATH Browse...

☑ Save binary masks

If checked, already processed images will be overwritten.
Uncheck if you want to resume an aborted process.

☑ Overwrite processed data

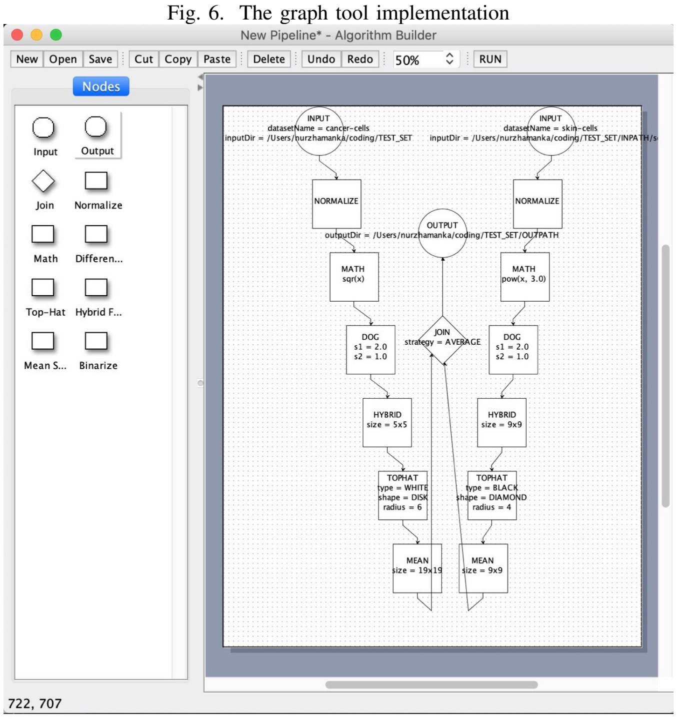
If checked, a separate stack showing the pipeline's steps will be produced.

☑ Produce a process stack?

uses JGraphX library to build and manipulate the graph visually, which is subsequently integrated with the segmentation module using JGraphT.

IV. System Features and Functionalities

Currently, the plugin provides users with three ways to process a dataset (load into memory via FIJI and process, or specify a directory) via a preset algorithm / pipeline that we described earlier. Loading a dataset into memory for immediate processing could be beneficial for users who want to experiment, if their dataset is not large. Otherwise, large datasets are better processed without loading the whole stack into the memory. In addition, we implemented the pipeline option, which provides user a tool to experiment with the pipeline by doing a graph-based algorithm customization. There are nodes that represent each step of the segmentation algorithm, which can be dragged out to a canvas and combined in different ways. Also, there is a "join" node, which combines numerous output images into a single node and performs a chosen algorithm on them (e.g., average). We had planned to extend the plugin and give users an option to run several algorithms in one run and compare them. However, due to the



INPUT
datasetName = cancer-cells
inputDir = /Users/nurzhamanka/coding/TEST_SET

INPUT
datasetName = skin-cells
inputDir = /Users/nurzhamanka/coding/TEST_SET/INPATH/s

NORMALIZE

NORMALIZE

OUTPUT
outputDir = /Users/nurzhamanka/coding/TEST_SET/OUTPATH

MATH
sqr(x)

MATH
pow(x, 3.0)

DOG
s1 = 2.0
s2 = 1.0

JOIN
strategy = AVERAGE

DOG
s1 = 2.0
s2 = 1.0

HYBRID
size = 5x5

HYBRID
size = 9x9

TOPHAT
type = WHITE
shape = DISK
radius = 6

TOPHAT
type = BLACK
shape = DIAMOND
radius = 4

MEAN
size = 19x19

MEAN
size = 9x9

Fig. 6. The graph tool implementation

disruption of the semester, only some of our initial goals have been achieved

V. Analysis / Evaluation

We have conducted a comparison of the segmentation achieved by our plugin to some manually-segmented samples, which we took as the ground truth. Because the images are, in fact, taken from microscopes, we can allow ourselves some wiggle room in terms of error. In fact it turned out to be in a margins of acceptable $5-10\%$ error, as was identified by the people from the Biology Department. In addition another metric were used: we compared the binary masks for similarity.

Furthermore, the plugin was tested for computational performance. We are concerned with time performance, mainly. The plugin already does segment images significantly faster than manual segmentation.

References

[1] E. A. Vaisberg, D. Lenzi, R. L. Hansen, B. H. Keon, and J. T. Finer, "An infrastructure for high-throughput microscopy: instrumentation, informatics, and integration," in *Methods in enzymology*. Elsevier, 2006, vol. 414, pp. 484–512.

[2] A. Stamm, K. Reimers, S. Strauß, P. Vogt, T. Scheper, and I. Pepelanova, "In vitro wound healing assays–state of the art," *BioNanoMaterials*, vol. 17, no. 1-2, pp. 79–87, 2016.

[3] M. D. Zordan, C. P. Mill, D. J. Riese, and J. F. Leary, "A high throughput, interactive imaging, bright-field wound healing assay," *Cytometry Part A*, vol. 79, no. 3, pp. 227–232, 2011.

[4] C. T. Rueden, J. Schindelin, M. C. Hiner, B. E. DeZonia, A. E. Walter, E. T. Arena, and K. W. Eliceiri, "Imagej2: Imagej for the next generation of scientific image data," *BMC bioinformatics*, vol. 18, no. 1, p. 529, 2017.