**Final Write-up**

**James G. Williams II**

**Abstract**

My project proposed object segmentation using level set methods and a three-dimensional model to improve cellular object classification accuracy and reliability in EM images. The publicly available dataset of 30 serial sections of Drosophila larva neural tissue will be used to develop the 3D model. A level set method can then be applied taking advantage of the information of all slides to improve object recognition and segmentation.

**Introduction**

Quick diagnosis and treatment can be the difference in life or death of a patient. The leading cause of death from Malaria is the delay of a proper diagnosis. Traditional methods are problematic because recognizing patient signs and symptoms are difficult. Laboratory technologies to improve diagnosis are necessary. Prompt and accurate malaria diagnosis would alleviate suffering, but also decreases chances it spreads within a community. An expert can detect malaria parasites in blood samples rather successfully but it takes too much time. For instance, datasets produced by medical imaging techniques, such as electron microscopy (EM), can be over terabytes (1012) in size. Manual analytics of a dataset this size are infeasible and powerful computer analytics are not available worldwide. This project aims to explore the use of Level sets to improve object segmentation on medical images while maintaining quick operational runtime.

There was no client involved in this project, which gave me a lot of freedom. The freedom was nice because it allowed me to focus on what I was interested in, Computer Vision and its application to medical imaging. It did create a bit of problem with scope: what could I finish, and how good can it be? I did have guidance from Martin Cenek to create some reasonably goals.

**Planning & Development Process**

My original plan was to use a prototype planning. The intent was to create small working deliverables that could be expanded relatively easy to execute something rather complicated. Once work began to explore the complicated implementations, they became more of the focus of the project in sort of a waterfall model.

I discovered the effectiveness of level sets and, with a rather forward way to implement it, I was more interested in it that my original idea of utilizing a Genetic Algorithm. My original idea was to create a functional app that could interact with the object segmentation, and minimum C++ code was developed for a UI with the intention of using OpenCV to handle some of the image pre-processing. To utilize the Level set, I found a publicly available toolbox written in Matlab code, and Matlab itself provided much functionality that I was going to use in OpenCV so I decided to use it. This focus on a new language took up much time, and the UI was placed as a low priority.

The project challenged my limited experience with computer vision. I had to learn how to handle image formats, and pre-processing images. I spent a couple of days just ensuring the image could be fed into the level set toolbox properly and then reconverted as an image. The actual execution of the level set took little time in comparison. Next came handling the images in a cell (an array for Matlab); the particular difficulty was handling the transposed cell array that would hold the “3D” information to be level set. Keeping track how my loops cycled through the arrays, and would re-render them properly (ie. Not upside-down) was difficult even when I hand written an “accurate” diagram I would simply doubt myself that it was correct (which it was…). Finally, most my time was spent trying to get it to automate some of the results for me, which proved unsuccessful. The main reason for that was my inexperience with Matlab code, for instance debugging an array out of bounds exception in a called function proved to long for the time I had.

**Design**

I designed the project to take in a hard coded format of images that were my dataset. These were placed in two cells: one would be to test using the level set on just the 2D information locality, and the second to test the level set with 3D locality. The second cell is transposed to create a cell of images that represent the columns of pixels formed from each image from the dataset. This new cell is where the level set is applied. The transposed is then reversed where the level set is applied again. The level set has now been applied twice on these images in two planes taking advantage of 3D locality. The last section is the output section which applies the built in contour of Matlab to section the images. After the contour is ran, they are saved to be analyzed. Automated analysis, as mentioned before, was not implemented due to time constraints.

**Analysis**

Correct labeling of synapses showed a 2.00% improvement when utilizing the 3D model and correct labeling of mitochondria showed a 1.17% increase. While these improvements do not seem impactful, this is the increase from 94% to 96%, which become more meaningful on larger datasets. Membranes identification could not be aptly analyzed with given time restraints. It mainly involves there nature where you could not necessarily “count” the membranes using the same method as mitochondria and synapses.

There was significant variance labeling due to the small dataset size. For instance, some slides false positives (over identification of an object) would have a nominal difference between the two testing models, but on some slides false positives were reduced by up to 40%. The small dataset had many objects that were on one slide but not the two adjacent, this would cause the item to be removed or add itself to the other slides.

The 3D model reduced noise very effectively. The best case of this is an error on a particular slide (16 I think) lights up as a red circle in the 2D test model, and is not there in the 3D test model. The majority of the noise reduction is due to the level set being ran slightly more than the 2D test model. Because of the dissimilar dimensions of information in the transposed cell, the level set cannot be ran the same number of times it would on the re-rendered cell. With my limited time constraints I could not fully optimize the proportions to prevent the 3D test model of using “too” many level sets. Limited work was done to optimize it using math analysis, such as trying to proportion through number of pixels that the level set would iterate through and this estimated the 3D test model was only over level set’d three times (out of 80).

**Conclusions**

The level set warrants further use and study to exactly how accurate it is with its reasonable computational time. This can be done by comparing it to other processing techniques that it would compete in. If you view the level set as a pre-process for an image it can be compared to a Gaussian blur for further analysis. The output of the processed images could be very useful for training an SVM; however, for me implementation of an SVM would be an entire project in of itself. A larger dataset would have been stronger to test the 3D-model with, and perhaps would have minimized the extent of variance in an image. Finally, the analytics could be further by testing the full range of parameters within the level set, such as type of level set ran, the number of iterations, and exact statistics of runtime for each of these variations. And as for one last lesson learned… buy Matlab sooner; handling Open Source projects took a lot of my time from my project, even if learning how they worked was interesting.

**User Guide**

My main code is ran in the folder with the Level Set Method and the file to run is segmentation.m. However, the bottom of the code has one image being contoured, and if you want to run it on all 20 images u can change the array loop from 1:1 to 1:width (or 20).  That section in code will also save the images through the "saveas" functions.  They need figures to be saved as so it will make alot of figures.. with better matlab experience I am sure I could work around that minor annoyance.  The dataset is in groundtruth-drosophila-vnc-master folder, and GroundTruthSegmentation.m was how i created the sectioned pictures, which also saves its output. You will need to set the path to the Level set folder, and have the directory in DrosphiliaProper so it has access to the pictures (will save pictures there too).

**References**

1. Level set method. (2014, March 6). In Wikipedia, The Free Encyclopedia. Retrieved 21:49, April 10, 2014, from http://en.wikipedia.org/w/index.php?title=Level\_set\_method&oldid=598459905
2. Segmented anisotropic ssTEM dataset of neural tissue. Stephan Gerhard, Jan Funke, Julien Martel, Albert Cardona, Richard Fetter. figshare. Retrieved 16:09, Nov 20, 2013 (GMT) http://dx.doi.org/10.6084/m9.figshare.856713
3. Lillywhite, K., Lee, D., Tippetts, B., Archibald, J. (2013) A feature construction method for general object recognition. *Pattern Recognition*, 46 (12), 3300-3314. http://dx.doi.org/10.1016/j.patcog.2013.06.002
4. Sumengen, B. (2005). Level Set Method Toolbox (Version 1.1) [Software]. Available from http://barissumengen.com/level\_set\_methods/index.html
5. Seyedhosseini, M., Tasdizen, T. (2013). Multi-Class Multi-Scale Series Contextual Model for Image Segmentation. *Image Processing, IEEE Transactions on*, 22(11), 4486-4496. http://dx.doi.org/10.1109/TIP.2013.2274388
6. Ghosh, P. and Mitchell, M. (2008). Prostate segmentation on pelvic CT images using a genetic algorithm. in *Proceedings of the International Society for Optical Engineering (SPIE), Conference on Medical Imaging,* February, 2008. SPIE Press.