university of groningen

faculty of science and engineering

User-guided, semi-supervised thin-section segmentation

Bachelor's Project Computing Science

July 7, 2024

Author: Lars Hidding
Student Number: s4673271
First supervisor: Dr. nat mat Christian Kehl
Second supervisor: Dr. Johannes Miocic
Daily supervisor: MSc Sebastian Mulder

Abstract

Thin section analysis is important in determining the physical properties of a rock. Manual analysis is a time-consuming process, one that can produce inconsistent results. Deep learning methods promise largely automated analysis procedures, yet such models require a lot of training data. In order to obtain this training data faster than can be achieved using manual analysis, one requires an automated approach. In this thesis we propose to automate most of this process by combining traditional image segmentation methods, while enabling the user to make corrections. The result is a novel image segmentation method and a software solution that will assist researchers in quickly performing thin section analysis. In order to keep the segmentation consistent, we record the performed actions. Finally, we assessed our results with a user study, measured the accuracy, and differences within accuracy using the Jaccard distance compared to a manually segmented version of the image.

From the obtained results we conclude that the current achievable accuracy is subpar, user corrections were rarely used because most participants spent their time with global segmentation. However the method is, in our eyes, still promising, yet needs to be improved before it can be used in actual lab procedures.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my sincere gratitude to Christian Kehl for the original research idea, for providing an extensive literature list, converting the thin sections into a usable format, and for his enduring patience throughout the process. I would also like to thank Johannes Miocic for his insight. In addition, I would like to express my sincere appreciation to Sebastian Mulder for his clear explanations of the underlying geological concepts and for providing both the thin section and the hand-segmented images for the user study.

1 Introduction

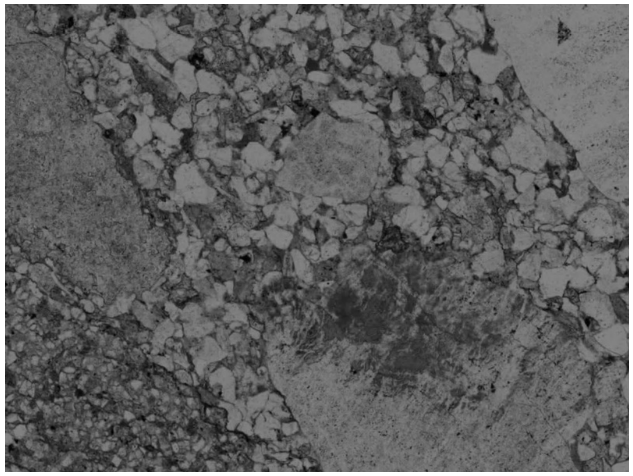


Figure 1: Example of a thin section

A rock consists of mineral grains, which determine its physical properties such as porosity, permeability, and density. Determining these properties involves taking thin slices, often only micrometres thick, from a rock sample. When observed under a microscope, the individual grains become visible. Thin section analysis entails labeling these grains, so they can be measured. This process is often done manually, which is time-consuming and requires significant experience. Manual analysis of thin sections can also lead to inconsistent results, as experts can disagree on actual, individual grain boundaries [1]. Automating this step of the process is crucial to speeding up research and making results more consistent. Image processing offers various methods of segmenting an image. These segmentation methods attempt to partition an image into regions. In our case these regions contain the individual grains.

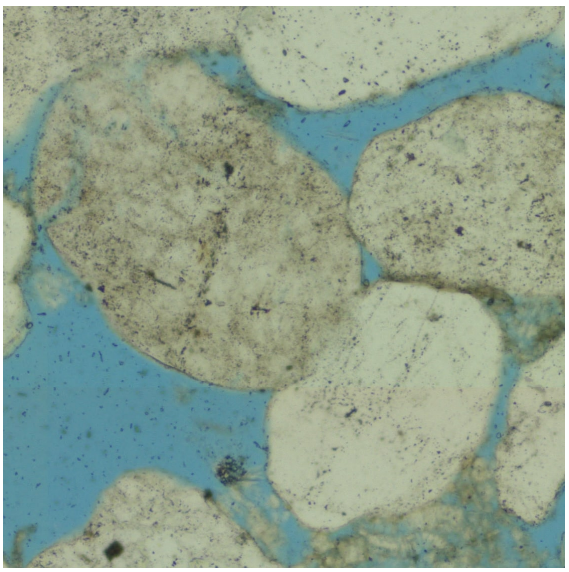
A research report conducted by Gerard Vehof compared two Machine Learning (ML) methods in their ability to segment thin sections. One of which was the U-net model, which is a Convolutional Neural Network (CNN) model developed by Ronneberger et al. The U-net model was determined viable for this automatic segmentation task [2, 3]. However applying such a model needs to be trained for other grain types. Training and improving the accuracy of such a CNN requires a lot of training data. This leads to our main research question, namely:

"How to rapidly and reproducibly obtain large sets of training data using image segmentation ?"

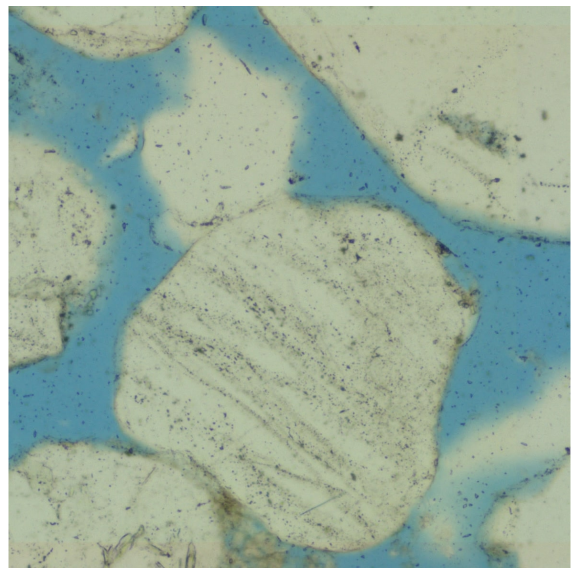
The primary objective of this project is to develop a segmentation workflow that combines traditional parametric image segmentation algorithms with user-guided corrections, with the aim of significantly accelerating the segmentation process. The correction process is then recorded so it can be auto-replicated on larger image batches, in order to further speed up the process.

1.1 Challenges with segmenting thin section images

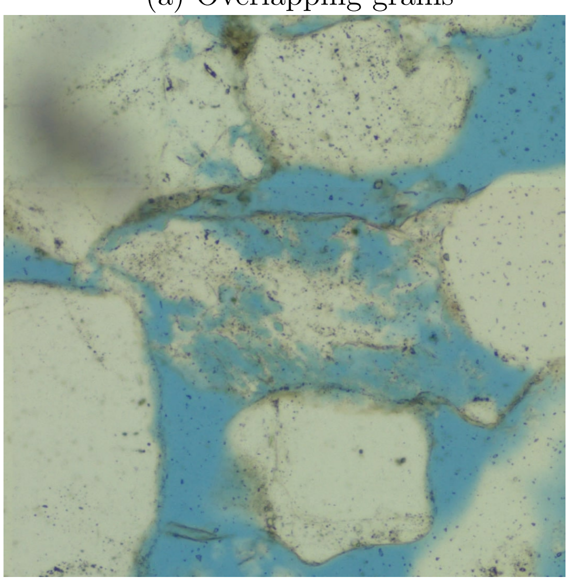
Segmenting a thin section image is a challenging task. First, the images are densely packed with objects. While it might be easy to segment a single object from its background, doing this with hundreds or thousands of objects is not a trivial task. Samples of different grain types vary wildly in their shapes and sizes. In poorly-sorted samples the grain size can vary within the sample itself. Grains can also touch each other or overlap, which makes it hard to properly segment both grains. These challenges make it hard to fully automate the process of segmentation. Therefore, our model enables the user to set the parameters for global segmentation, and allows for user-guided corrections to be applied to single regions.



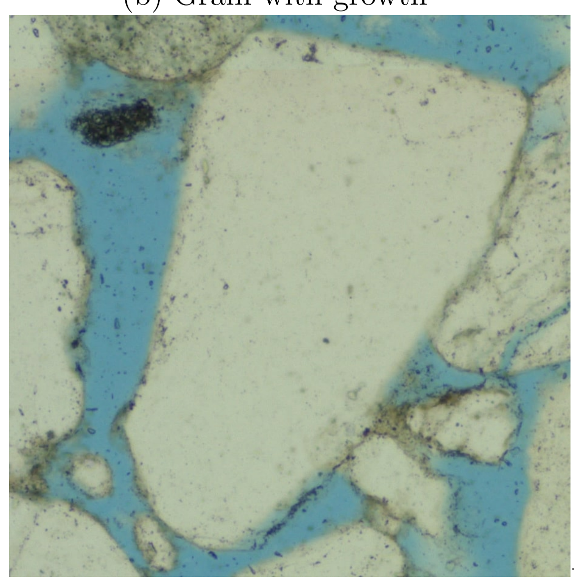
(a) Overlapping grains



(b) Grain with growth



(c) Partially dissolved grain



(d) Grains with a big size difference

Figure 2: Challenging grains to segment. Sub-images from Botter, 2023

2 Related work

There are a number of methods that can assist us with the segmentation and refinement processes.

Jungmann et al. used a weighted region competition, used to balance the merging of larger and smaller regions. Their results look promising as they are not far off compared to their manual analysis [4].

Felzen et al. used a graph-based method for image segmentation that also uses weighted region competition to balance intensity differences across boundaries and within regions. The algorithm produced accurate results and captures non-local image characteristics efficiently. [5].

Maitre et al. combined Super Linear Iterative Clustering (SLIC) with three popular non-parametric classification methods. They compared each SLIC-paired method against a deep learning CNN. The SLIC-paired methods outperformed the CNN when it was using a 70:30 split for training and testing data. However, it should be noted that their ground truth data had displacements, which may have affected the accuracy of the evaluation [6]

Yu et al. attempted to generate training datasets for deep learning applications by evaluating several superpixel algorithms and introduced MultiSLIC, an extension of the SLIC algorithm, to work on multispectral images. [7].

One possible refinement method is active contour snakes. First described in a paper by Kass et al. They made use of an energy-minimizing spline model they call active contour snakes. Their method allowed for interactive interpretation and accurate localization of image features [8].

As mentioned before one of the challenges when segmenting a thin section is touching grains. Scharf et al. focused specifically on zircon grains, proposed a method which allowed for manual adjustment of automated segmentation and separates overlapping grains [9].
Tan et al. produced a paper in the agricultural sector about rice grain separation. They utilized the watershed algorithm to separate and count touching hybrid rice grains. They concluded that while the watershed algorithm is effective in identifying separation lines between different regions, it can lead to over- and under-segmentation. To solve these challenges the study proposes an improved corner point detection algorithm [10].
A method proposed by Van den Berg et al. better preserves the size and shape characteristics of the grain compared to the watershed algorithm [11].

This project is also based on previous research done at the University of Groningen (UG). The before mentioned report by Gerard Vehof that explored two machine learning approaches [2]. And a research report by Dennis Botter, who used a combination of more traditional methods for their segmentation [12].

3 Methodology and workflow

For our global segmentation we decide to use SLIC [13]. SLIC divides an image into same-sized superpixels and then assigns pixels to the superpixels based on their value and position using K-Nearest-Neighbor (KNN). We then use a Region Adjacency Graph (RAG) [14] to combine these over-segmented grains to obtain a good-estimate segmentation. Finally, the user can refine individual regions to further improve the accuracy.

3.1 Segmentation workflow

The segmentation workflow consists of the following steps:

1. Pre-processing
2. Global segmentation using SLIC
3. Merging over-segmented grains using RAG
4. Applying modifiers to grains to adjust boundaries
5. Post-processing

3.2 Pre-processing

The model uses a minimum luminance value as a threshold for separating the background and foreground. This is a rough estimate, though the background is generally darker than the foreground. The threshold is used to generate a mask of the foreground, where any spots smaller than the minimum grain size get removed. In order to increase the speed of the segmentation, the image is scaled down using the quality parameter. This allows users to test out different parameter settings without having to wait for the high-resolution image to be segmented.

3.3 Global segmentation

Global segmentation is performed using SLIC on a gray scale (mono-channel) version of the thin section. It is the primary step the user performs.

3.4 Merging regions

In order to merge the over-segmented regions that are supposed to be a single region, we use an implementation of a RAG that performs hierarchical merging on region boundaries.

3.5 Modifiers

Since the global segmentation is limited in its capability of segmenting the image we enable users to apply modifiers to individual regions or groups of regions. The infrastructure surrounding modifiers is designed with extensibility in mind, so that other developers can easily implement their own modifiers. The built-in modifiers include:

Method	**Description**
Erosion	Shrinks the region by its boundary
Dilation	Enlarges the region by its boundary
Opening	Erosion followed by dilation, opens holes in a segment.
Closing	Dilation followed by erosion, closes internal holes.

Table 1: Built-in Modifiers

Modifiers are not limited to the segmentation however. There are modifiers that can apply filters to the image, such as the local histogram equalization [15] as a pre-processing step, possibly improving the global segmentation.

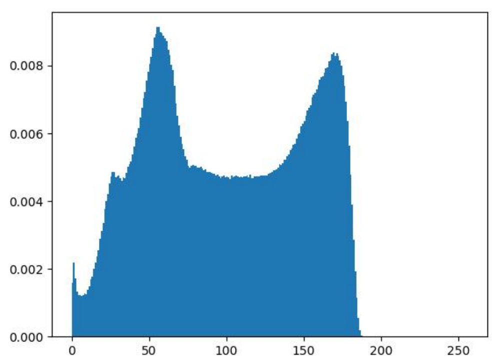


Figure 3: Histogram before local equalization

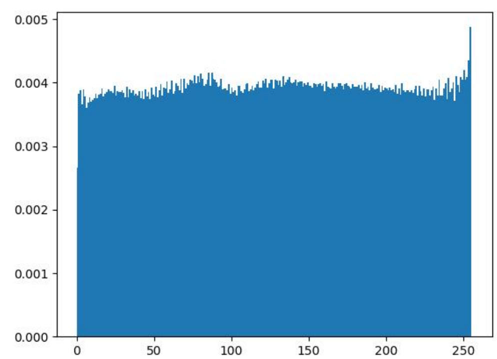


Figure 4: Histogram after local equalization

3.6 AMG

When a user applies a modifier to a segment it will be recorded in the Acyclic Modifier Graph (AMG). The AMG is an implementation of an Directed Acyclic Graph (DAG). This structure allows the program to traverse the graph and replicate the action without the possibility of getting stuck in a loop. Aside from that, it also makes it possible to switch between workflows and compare them.

4 Design and implementation

In order to make the workflow more user-friendly, we have developed a program with a user interface. This section covers the design and implementation of the program.

4.1 Technology stack

The program is completely written in Python[1]. The user interface has been made using PyQt5[2]. For image processing tasks, we mainly make use of the NumPy[3] and scikit-image[4] libraries.

4.2 User interface

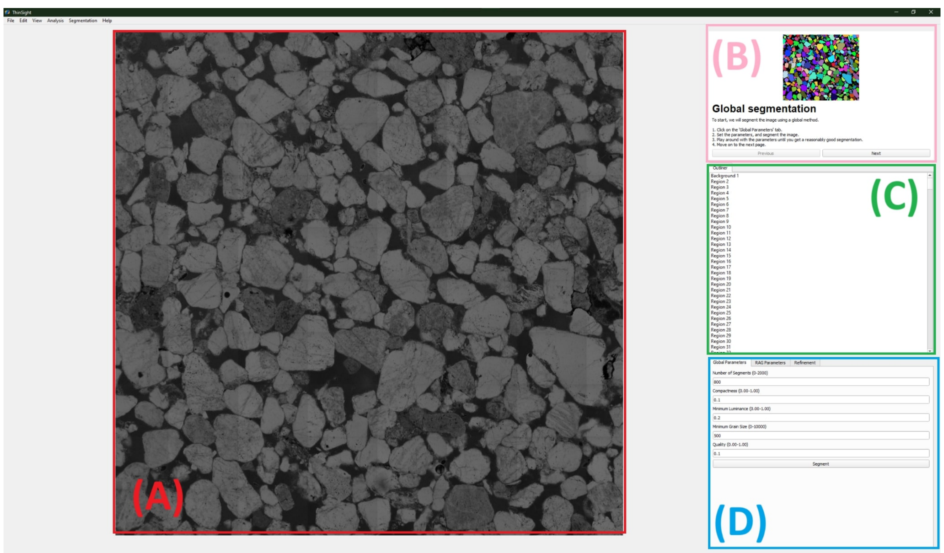


Figure 5: Screenshot of the main window showing the UI elements:
(A) viewport, (B) tutorial, (C) outliner, (D) segmentation panel

The figure above shows the main window. The viewport (A) enables user to interact with the segmentation. They can zoom and pan around the image, switch between different views, and select regions using the mouse. The tutorial (B) was used during the user study to explain participants how they could use the program. The outliner (C) lists the regions. The segmentation panel (D) contains 3 tabs. The first tab shows the global segmentation parameters. The second tab, contains the parameters for the RAG. The last tab is the re-

[1]https://www.python.org/
[2]https://doc.qt.io/qtforpython-5/
[3]https://numpy.org
[4]https://scikit-image.org

finement tab. It shows the modifiers that can be applied to individual regions.

4.3 AMG Visualization

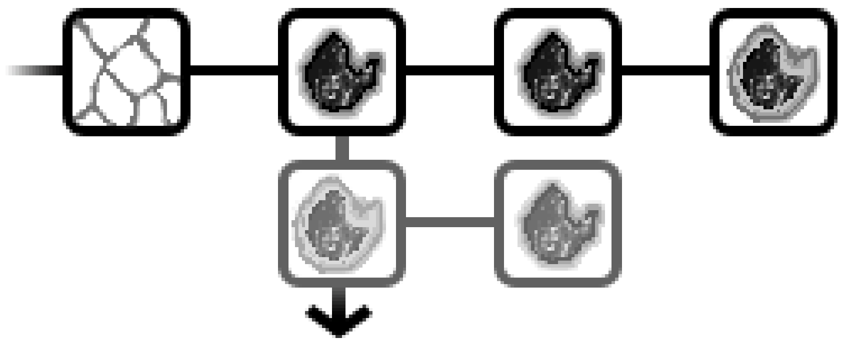


Figure 6: AMG Concept design

While the AMG does register the modifiers applied by users, there is currently no UI element for it. Figure 6 shows a concept of how the AMG would be visualized. In this example the first (left-most) node shows global segmentation and the other nodes are erosion and dilation modifiers. Selecting a node would load the segmentation at that state. When a new modifier is applied a branch is created, allowing users to switch between branches and compare workflows.

5 Results and discussion

In order to evaluate our workflow, we conducted a user study to measure the accuracy and difference in accuracy within a constrained time frame. Participants were each given 20 minutes to segment a sub-image taken from a thin section. Their result is then compared against a manually segmented image. The manually segmented image as shown in figure 7 is our ground truth. We compute the accuracy as the similarity between the segmentation result and the ground truth. For our measure of similarity we used the Jaccard distance. The Jaccard distance is a measure of set similarity and is calculated as follows:

$$Jaccard\ distance = 1 - \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

A Jaccard distance of 0 means that the sets are equal thus regions are the exact same shape and size, while a distance of 1 means that there is no overlap between the regions.

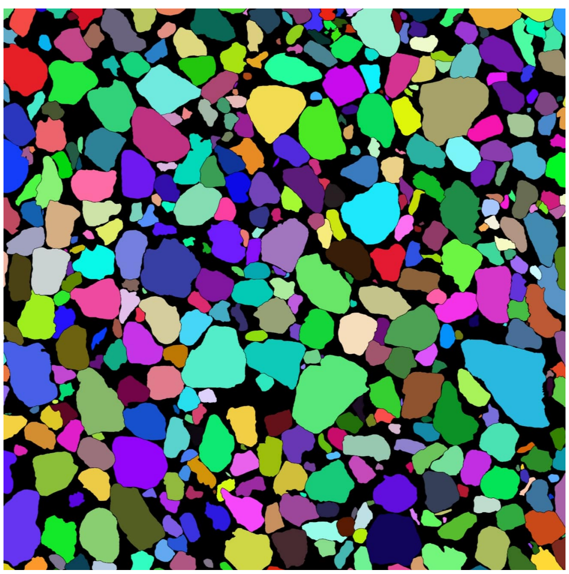


Figure 7: Ground Truth, Botter, 2023

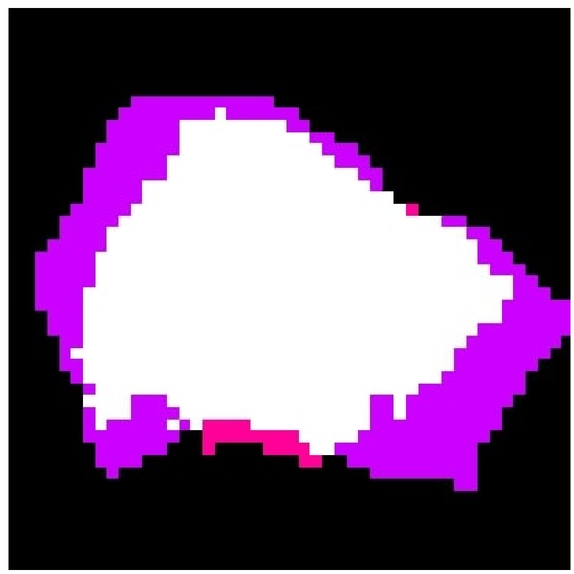


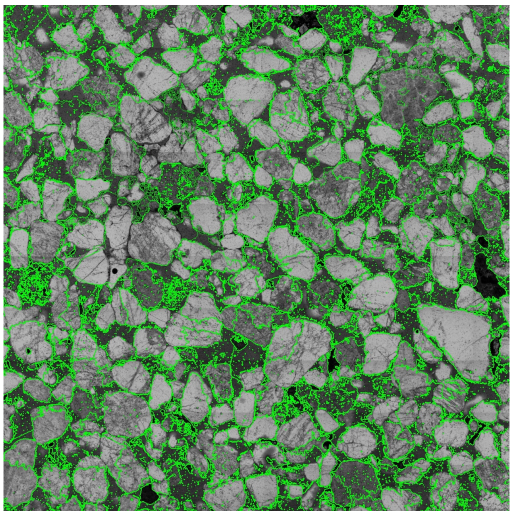
Figure 8: Analysis of a single grain with ground truth segment shown in purple, segmentation result in pink, and intersection in white

In order to know what regions should be compared, we calculate the Jaccard distance between a region in the ground truth and every region in the result and take the lowest distance. This gives us the best fitting region. This does mean that a single region in the result can be the closest to multiple regions in the ground truth. However, in such a case there will also be a greater number of false positives, so ultimately this will not affect the distance.

In order to calculate the difference in accuracy, we compare the first and last segmentation from each participant with the ground truth.

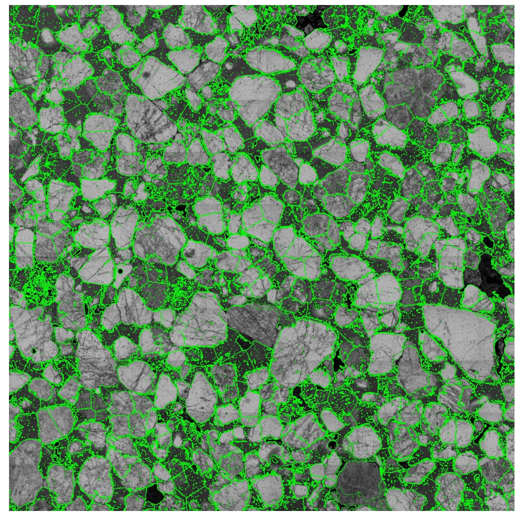
5.1 Results

5.1.1 Accuracy analysis results



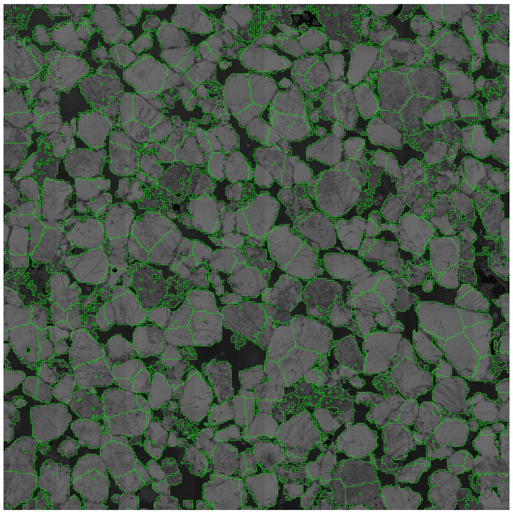
User 3

	Start	End	Difference
Mean	0.778	0.781	0.003
Median	0.878	0.883	0.005
Std	0.248	0.252	0.004
Min	0.075	0.075	0.000
Max	0.998	0.998	0.000



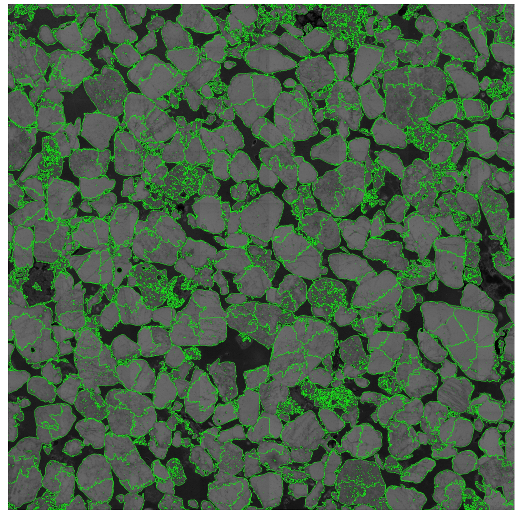
User 4

	Start	End	Difference
Mean	0.651	0.630	0.022
Median	0.711	0.708	0.002
Std	0.263	0.289	0.026
Min	0.048	0.048	0.000
Max	0.998	0.998	0.000



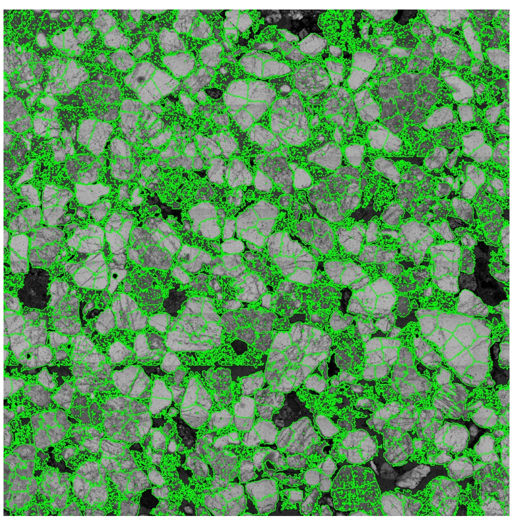
User 5

	Start	End	Difference
Mean	0.689	0.761	0.072
Median	0.728	0.829	0.101
Std	0.234	0.218	0.016
Min	0.115	0.101	0.014
Max	1.000	1.000	0.000



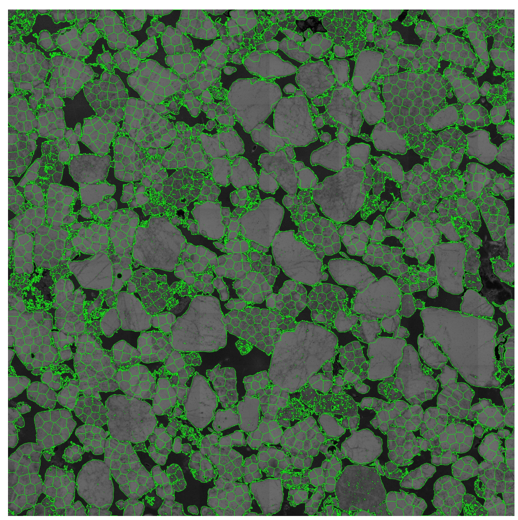
User 6

	Start	End	Difference
Mean	0.679	0.720	-0.041
Median	0.704	0.821	-0.117
Std	0.210	0.261	-0.051
Min	0.148	0.059	0.089
Max	1.000	1.000	0.000



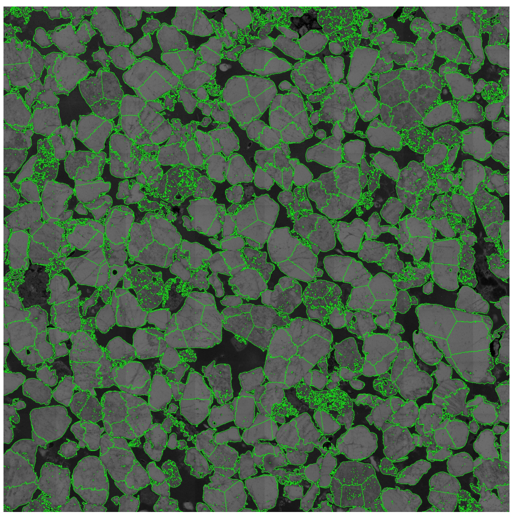
User 7

	Start	End	Difference
Mean	0.995	0.663	0.332
Median	0.999	0.704	0.295
Std	0.022	0.248	-0.226
Min	0.710	0.090	0.620
Max	1.000	1.000	0.000



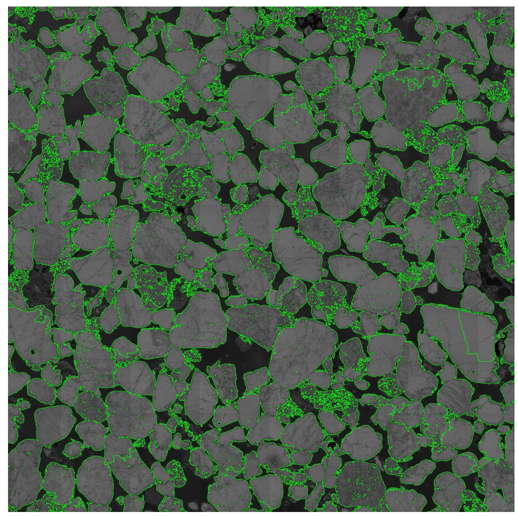
User 8

	Start	End	Difference
Mean	0.695	0.627	0.068
Median	0.728	0.666	0.062
Std	0.234	0.239	-0.005
Min	0.116	0.065	0.051
Max	1.000	1.000	0.000



User 9

	Start	End	Difference
Mean	0.695	0.687	0.008
Median	0.728	0.737	-0.008
Std	0.234	0.249	-0.015
Min	0.116	0.082	0.033
Max	1.000	1.000	0.000



User 10

	Start	End	Difference
Mean	0.697	0.945	-0.248
Median	0.726	0.981	-0.255
Std	0.196	0.106	0.091
Min	0.182	0.242	-0.061
Max	1.000	1.000	0.000

	Start	End	Improvement
Mean	0,735	0,727	0,008
Median	0,775	0,791	-0,016
Std	0,205	0,233	-0,028
Min	0,189	0,096	0,093
Max	0,999	0,999	0,000

Table 2: Jaccard distances (Averaged over all participants)

5.1.2 AMG analysis results

By analysing the AMG's we extracted the following information:
Participants performed on average 74 actions, this includes global segmentation, applying a modifier and merging segments.

Modifier	**Uses**
Erosion	21
Dilation	26
Opening	104
Closing	32

Table 3: Modifier occurrence.

It is clear from Table 3, that the opening modifier is used most.

Value ranges for the global segmentation:

- **Number of Segments**: 20 - 2500 (default value: 800)
- **Compactness**: 0,1 - 0,8 (default value: 0,1)
- **Minimum Luminance**: 0,1 - 0,3 (default value: 0,2)
- **Minimum Grain Size**: 10 - 5000 (default value: 500)
- **Quality**: 0,1 - 1,0 (default value: 0,1)

Value ranges for the global segmentation(final values only):

- **Number of Segments**: 200 - 2500 (default value: 800)
- **Compactness**: 0,1 - 0,8 (default value: 0,1)
- **Minimum Luminance**: 0,1 - 0,2 (default value: 0,2)
- **Minimum Grain Size**: 10 - 5000 (default value: 500)
- **Quality**: 0,4 - 1,0 (default value: 0,1)

5.2 Discussion

As shown in Table 2, the average accuracy is not significant. Despite having the lowest mean distance at the end, Participant 8 still was not able to obtain a high accuracy. This may be due to the fact that our method did not account for over-segmentation. Consequently, a rough estimate would score significantly higher than an over-segmented version with accurate boundaries.

Furthermore, there was no significant difference in accuracy. This can be attributed, at least in part, to the fact that participants tended to remain in the global segmentation step for the majority of their time, only progressing to the refinement step towards the end of their time. The participants who achieved the best mean scores made extensive use of the feature to manually merge segments. This suggests that implementing a fully functional RAG could enhance the accuracy and enable users to dedicate more time to the refinement phase.

Going back to our research question: "how to rapidly and reproducibly obtain large sets of training data using image segmentation" At present, our solution does not fully answer this question. However, we believe the issue lies not in our overall methodology but rather in the specific aspects of our implementation and analysis. By addressing these areas, we might achieve a more definitive answer in the future.

5.3 Limitations

While originally meant for people with a background in geology, due to time constraints and a lack of participants, the study was expanded beyond geology students to include anyone with a university-level education or higher. Our final results did not include any geology students or anyone with expertise in thin section analysis as participants.

The user study resulted in 10 segmented thin section sub-images. However we discovered later that 2 of those results were performed on a different image and thus could not be used in the assessment. These results are still present in appendix, however they have not been used in the results.

Because using a high-resolution thin section image would be very resource intensive, we decided to use a lower quality sub-image. The color channels of the image were removed to make it compatible with our segmentation model.

We wrongly assumed that the ground truth image we used Figure 7 would be the same size and had the same amount of detail as the image segmented by the participants. However in actuality it was half of the size. Therefore our analysis is slightly less accurate. Secondly in the ground truth the edges were anti-aliased. That meant that selecting the regions by color would introduce a lot of non-existent regions. In view of time constraints it was decided upon to still go through with this method and to threshold the size of the ground truth regions[5].

5.4 User feedback and improvement suggestions

Participants often spent most of their time in the global segmentation step. Many of the participants noted that they found it hard to determine when the global segmentation was sufficient to move on. Therefore modifiers were not used often, possibly impacting the improvement in accuracy. The regions often needed to be manually merged because the RAG did not seem to have much impact. This might be caused by the weight function used in the current implementation of the RAG. Users also commented on the fact that they did not get immediate feedback when the program was busy performing the slow task of global segmentation. Adding a loading bar or icon has been suggested by multiple participants. One participant suggested highlighting regions when the mouse hovered over them. Sadly, selection of segments via the viewport did not work in all cases. This inconvenienced some participants and made it harder to apply modifiers, because they were forced to select the regions using the outliner.

[5]The specific value used was 100 pixels in area.

6 Conclusion

The aim of this project was to develop a method that could rapidly and reproducibly segment thin sections. The results of the user study have shown that the current state of the program is not sufficient for the intended purpose and that current limitations have to be overcome first.

However, the result from a small subset of user study participants indicate that the method of combining SLIC with user-guided corrections can produce accurate segmentations, and that the low accuracy score is due to both our implementation of the workflow and the way in which we analysed the results.

7 Future Work

As mentioned in our discussion, the method we used for analysing the results of our user study could be improved upon. Performing the user study again with more participants and using multiple different images would also improve the validity of the results.

There are several improvements that can be made to extend the current programs ability to segment the thin sections. First of all, adding the option to further segment a region could improve the accuracy of the region boundaries. This can for example be achieved by adding a watershed modifier or a localized version of SLIC. Other ways of improving the boundaries include using active contour snakes [8] or MSLIC [16]. The user study showed that the impact of the RAG was insignificant. Finding a good weight function for the RAG would likely increase the accuracy significantly and save users time they would have otherwise spent merging regions by hand.

One of the biggest current limitations with our program is that it can not handle the full sized thin section images. Adding this capability is not a trivial task as it introduces even more challenges, such as dealing with the borders around the thin section image.

References

[1] He Liu, Yi-Li Ren, Xin Li, Yan-Xu Hu, Jian-Ping Wu, Bin Li, Lu Luo, Zhi Tao, Xi Liu, Jia Liang, Yun-Ying Zhang, Xiao-Yu An, and Wen-Kai Fang. Rock thin-section analysis and identification based on artificial intelligent technique. *Petroleum Science*, 19(4):1605–1621, 2022.

[2] Gerard Vehof. Machine learning approach for instance segmentation of sandstone thin sections, 2023.

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[4] Matthias Jungmann, Hansgeorg Pape, Peter Wißkirchen, Christoph Clauser, and Thomas Berlage. Segmentation of thin section images for grain size analysis using region competition and edge-weighted region merging. *Computers Geosciences*, 72:33–48, 2014.

[5] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, Sep 2004.

[6] Julien Maitre, Kévin Bouchard, and L. Paul Bédard. Mineral grains recognition using computer vision and machine learning. *Computers Geosciences*, 130:84–93, 2019.

[7] Jiaxin Yu, Florian Wellmann, Simon Virgo, Marven von Domarus, Mingze Jiang, Joyce Schmatz, and Bastian Leibe. Superpixel segmentations for thin sections: Evaluation of methods to enable the generation of machine learning training data sets. *Computers Geosciences*, 170:105232, 2023.

[8] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988. 321.

[9] T. Scharf, C.L. Kirkland, M.L. Daggitt, M. Barham, and V. Puzyrev. Analyzr: A python application for zircon grain image segmentation and shape analysis. *Computers Geosciences*, 162:105057, 2022.

[10] Suiyan Tan, Xu Ma, Zhijie Mai, Long Qi, and Yuwei Wang. Segmentation and counting algorithm for touching hybrid rice grains. *Computers and Electronics in Agriculture*, 162:493–504, 2019.

[11] E.H. van den Berg, A.G.C.A. Meesters, J.A.M. Kenter, and W. Schlager. Automated separation of touching grains in digital images of thin sections. *Computers Geosciences*, 28(2):179–190, 2002.

[12] Dennis Botter. Optimization of a grain segmentation approach within sandstone thin-section images, 2023.

[13] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and Sabine Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 34(11):2274–2282, 2012. 2274.

[14] A. Tremeau and P. Colantoni. Regions adjacency graph applied to color image segmentation. *IEEE Transactions on Image Processing*, 9(4):735–744, 2000.

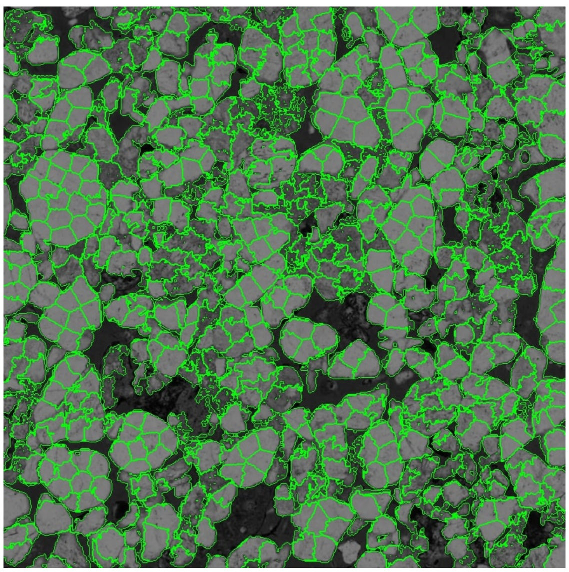
[15] Robert Hummel. Image enhancement by histogram transformation. *Computer Graphics and Image Processing*, 6(2):184–195, 1977.

[16] Feng Jiang, Qing Gu, Huizhen Hao, Na Li, Bingqian Wang, and Xiumian Hu. A method for automatic grain segmentation of multi-angle cross-polarized microscopic images of sandstone. *Computers Geosciences*, 115:143–153, 2018.

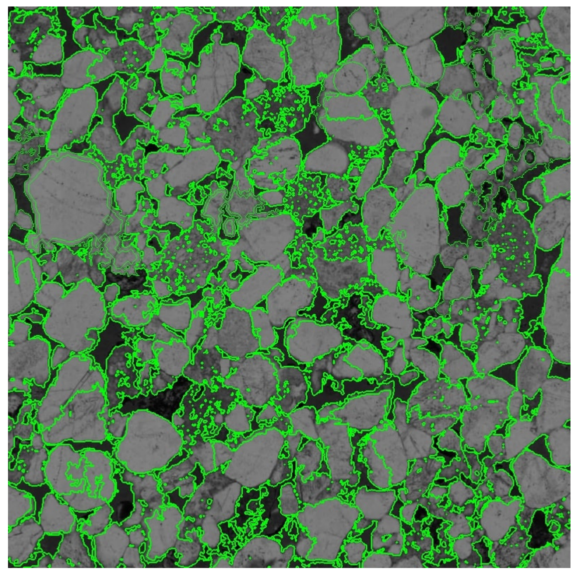
A Link to Github Repository

https://github.com/Drasath/thin-section-segmentation

B User study results



User 1



User 2

Note: Users 1 and 2 were given a different image, thus their results were not used