AGH University of Science and Technology

Laboratory Report

Multiscale Modelling

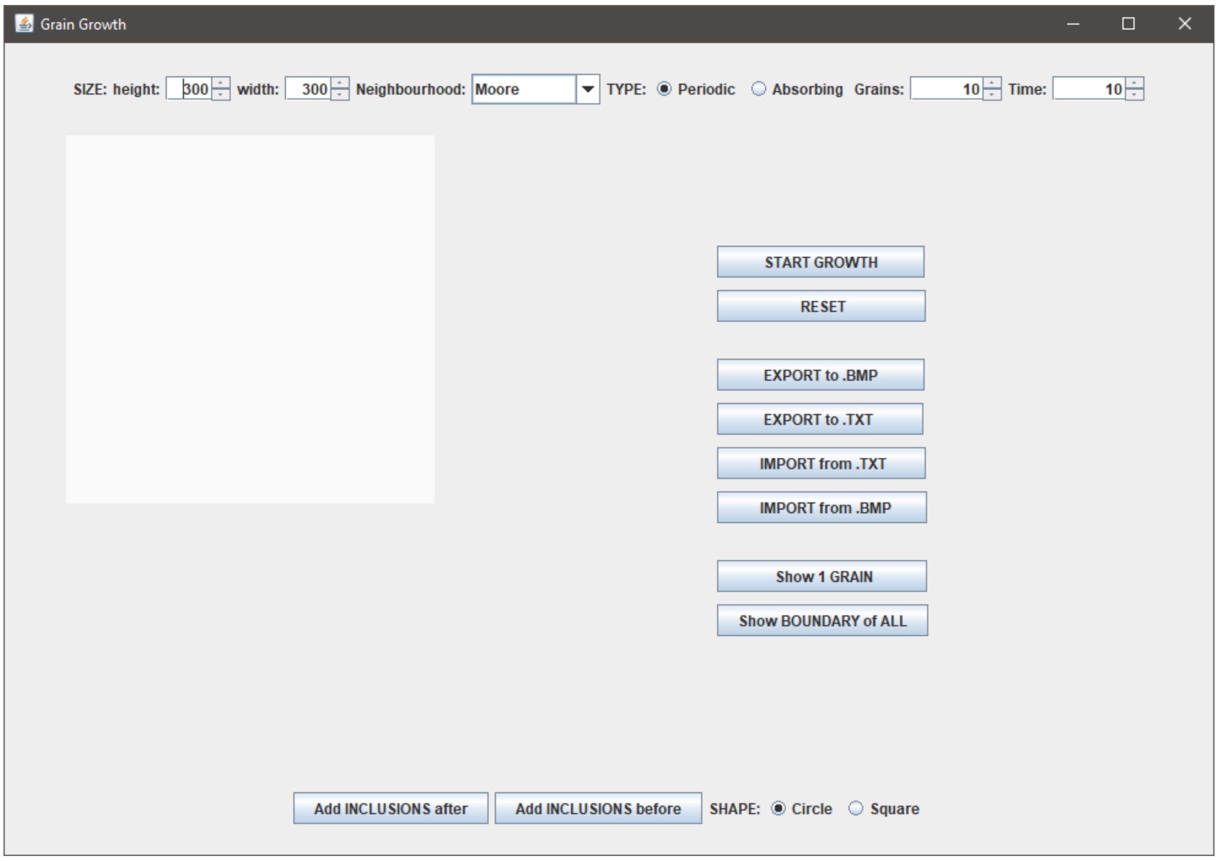
Project 1: Grain growth algorithm modifications

Agnieszka Gorczycka (279727)

1. 1st Laboratory

The goal was to implement naïve growth and create GUI *(Figure 1)*. The language of my choice was Java, and the IDE was IntelliJ IDEA – I'm not a Computer Science graduate as my field of study was Mechatronic Engineering (in English), and what I do at work is Front-End development, which is why I chose the language and IDE I did use a little during studies. Size of the map is adjustable, as well as neighbourhood, type of periodicity, number of grains and time *(Figure 2)*.

My GUI was divided into 3 vertical parts – meaning the upper part holding responsibility for switching the attributes regarding the initial behavior of the simulation, the middle part being used for displaying the actual simulation and adding additional options to manipulate the growth (as well as initiate it) and the bottom part having only the inclusions option – actually implementing them and picking their shape.


Grain Growth

SIZE: height: 300 width: 300 Neighbourhood: Moore TYPE: [✓ Periodic] Absorbing Grains: 10 Time: 10

SHAPE: [✓ Circle] Square

Figure 1: GUI

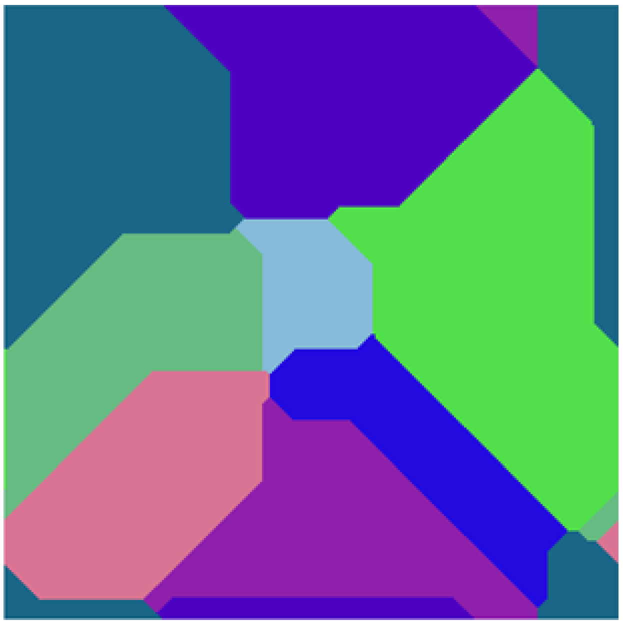
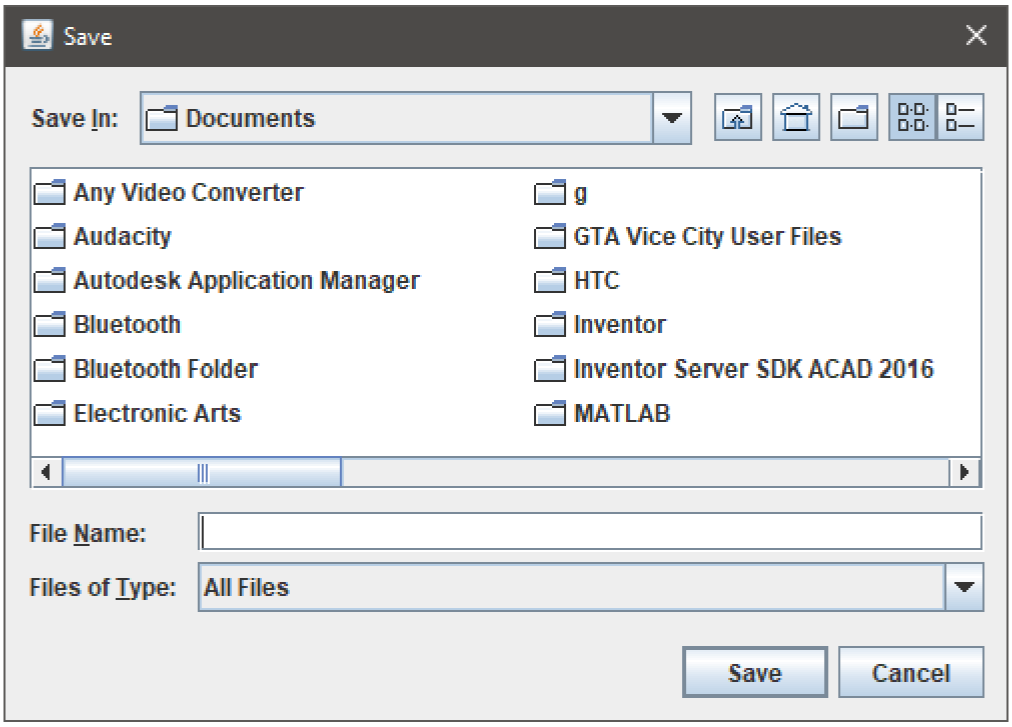


Figure 2: Grain Growth

The goal of the first laboratory was to generate the nucleons and grow them in time. What we can choose it the number of the nucleons - which can also be added by mouse-clicks on the map, time of simulation, and, most importantly, the type of neighbourhood - we can choose between Von Neumann and Moore, all of which causes some differences between the possible microstructures. The most interesting is picking the type - we can choose either periodic or absorbing.

2. 2nd Laboratory

The goal of this laboratory was to implement import/export functionality. Map was supposed to load a picture saved in .bmp format, as well as save the map to .bmp format, and do the exact same things in.txt format (*Figure 3*). Loaded map was supposed to be interactive – what I mean by that is that all inclusions (mentioned in 3rd section) and any more map adjusts were supposed to work fine. In my case, this is unfortunately only valid for .txt file (*Figure 4*) – importing .bmp only shows in on the map *(Figure 5).*


Save

Save In: Documents

Any Video Converter
Audacity
Autodesk Application Manager
Bluetooth
Bluetooth Folder
Electronic Arts
g
GTA Vice City User Files
HTC
Inventor
Inventor Server SDK ACAD 2016
MATLAB

File Name:
Files of Type: All Files

Figure 3: File saver

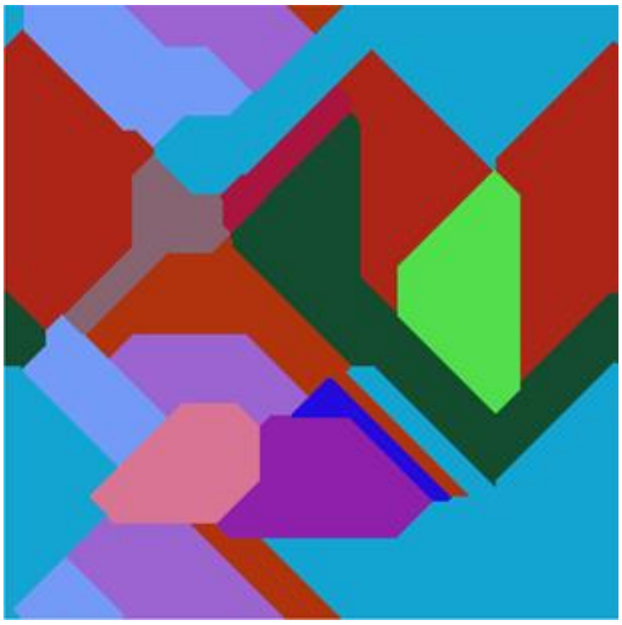


Figure 4: Imported from .txt file

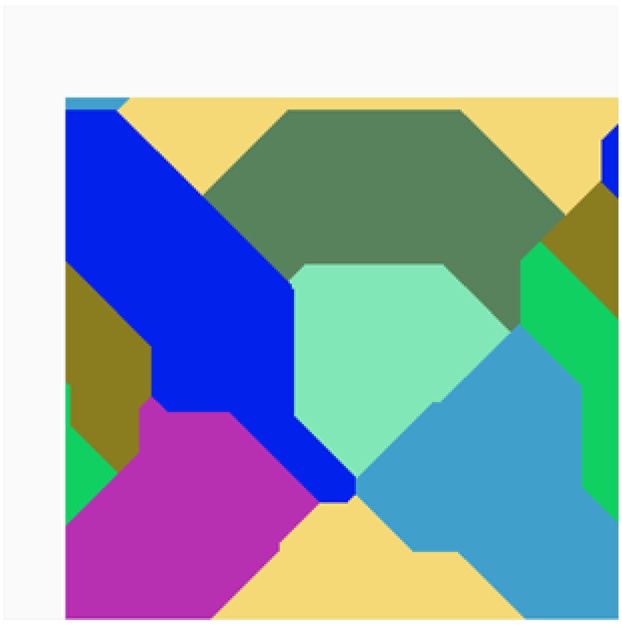


Figure 5: Imported from .bmp

3. 3rd Laboratory

The objective of this class was to implement inclusions showing after (*Figure 6*) or before (*Figure 7*) the simulation. In my case, I let the user choose between two forms of inclusions – circles and squares – but only the after option works correctly. The before options does show the inclusions in black, but unfortunately after starting the simulation all of them are covered by grains.

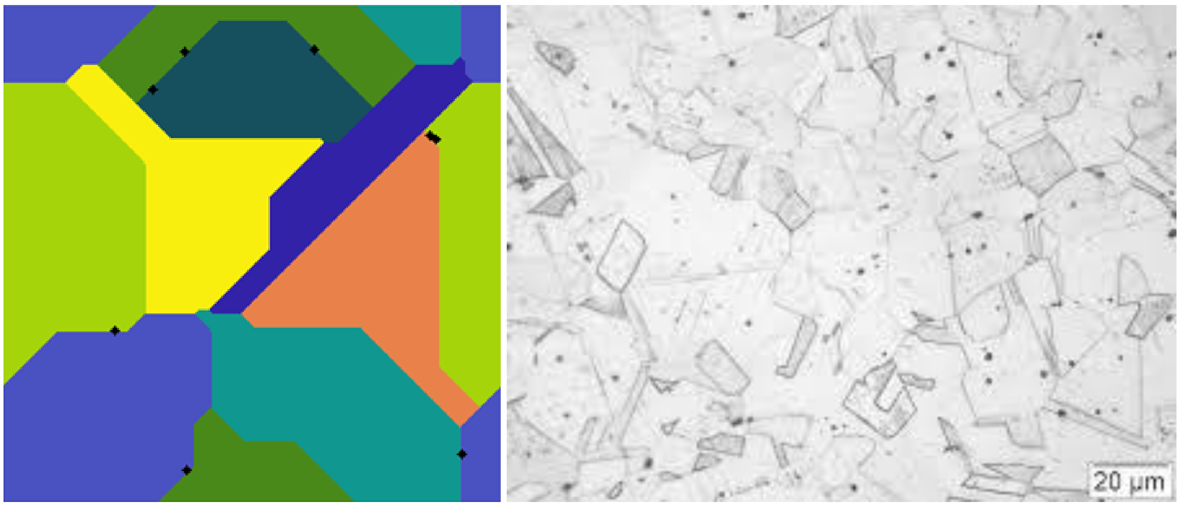


Figure 6:Inclusions after – comparison between custom algorithm and real microstructure of stainless steel

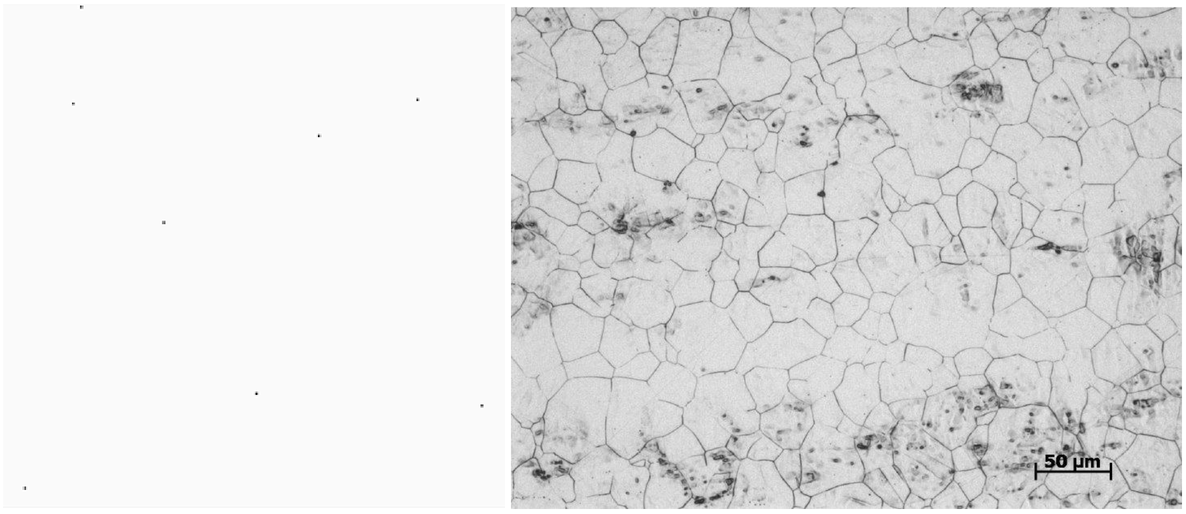


Figure 7: Inclusions before - comparison between custom algorithm and real microstructure of 304L stainless steel

Both of above comparisons (especially the first one) present similarity between the real microstructure and custom written algorithm, generating one. Naturally, increasing the accuracy of algorithm would have positive impact on the actual mapping.

4. 4th Laboratory

Sadly, I did not manage to do this part of the classes.

5. 5th Laboratory

Of this part of Laboratory, I managed to do the Grain Selection. The algorithm works like following, is chooses one color from the list of all colors currently on the map, and then it saves it, whereas the rest of the colors is replaced with white (*Figure 8*).

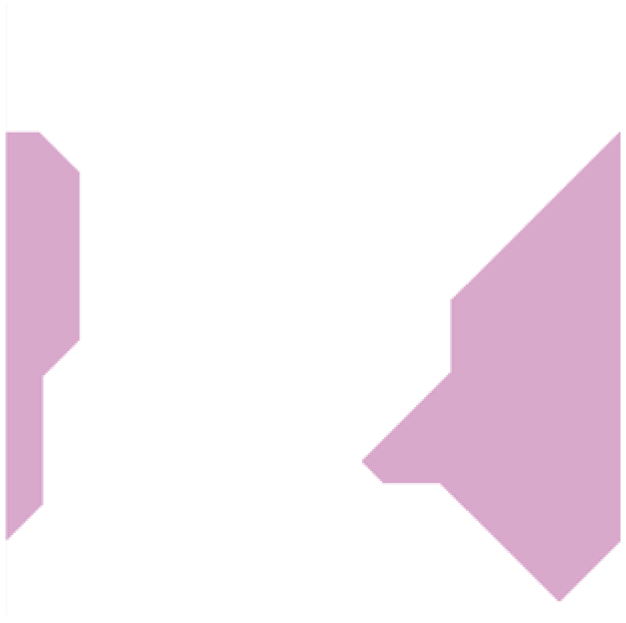


Figure 8: One grain chosen

The substructure is generated correctly, but the grain growth cannot be implemented on the base of it.

6. 6th Laboratory

This laboratory required the assistance of black color – the goal was to find the boundaries of the grains (*Figure 9*) or just one grain (*Figure 10*), mark them in black and change every other color to white.

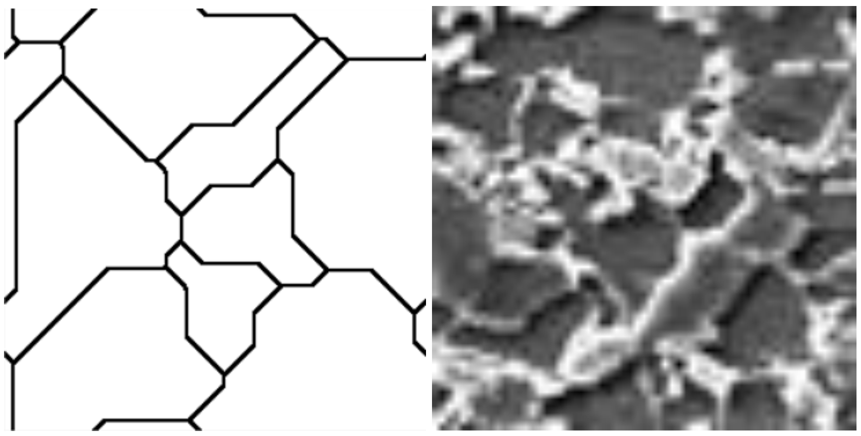


Figure 9: All boundaries- comparison between the map generated by the algorithm and real example – HAZ and HSLA steel

The similarity between both microstructures can be observed.

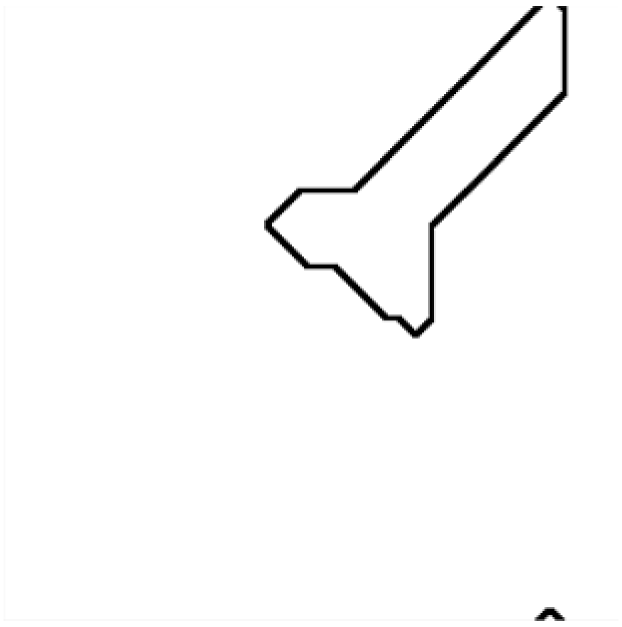


Figure 10: Boundary of one grain

7. Conclusions

The algorithm of cellular automata consists of colored grains on the map. The shapes of grains vary on the base of the neighbourhood and type of periodicity. Many results can be obtained even by using the same neighbourhood and type, it will have different colors, shapes, arrangement. Thanks to the cellular automata, it's easier to see the results of such algorithm, because we see it in a visual form. Personally, as I mentioned above, neither am I a Computer Science graduate, nor do I work with Java, or C#, or C++ professionally, so what I am happy about is that this project let me understand Java better.