Author: Kamil Łuc

Multiscale Modelling

Report 1

1) Technology

Java 11 LTS - is a general-purpose computer-programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere, meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Author choose Java because it is sufficient for this task and wanted to expand his knowledge about new version (11 vs 8 from 4 years ago).

Strengths	Weaknesses
Strongly-typed language, compiled, good performance, old and tested, very popular on market, variety of libraries, big company (Oracle) behind it, multiplatform, one of the best if not the best environment for programming desktop applications.	Slower than C/C++, quite hard to learn (above basic level) compared to Python/JavaScript etc., confusing licence.

JavaFX - is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc. It's basically a standard for Javas GUI application (that's why author choose this technology), it replaced SWING many years ago. Also it allows to use XML like syntax for structure View elements like buttons ale CSS like syntax for styling them.

Strengths	Weaknesses
Follows MVC pattern, good looking apps could be created quite easily thanks to HTML and CSS like syntax. Easier and better than predecessor SWING.	Since JDK 9 quite complex initial configuration (JavaFX is no longer part of JDK).

Tools like **InteliJ Idea** (Integrated Development Environment) where code was written and **GIT** to manage repository was used but they are not part of application itself so the they are not described here.
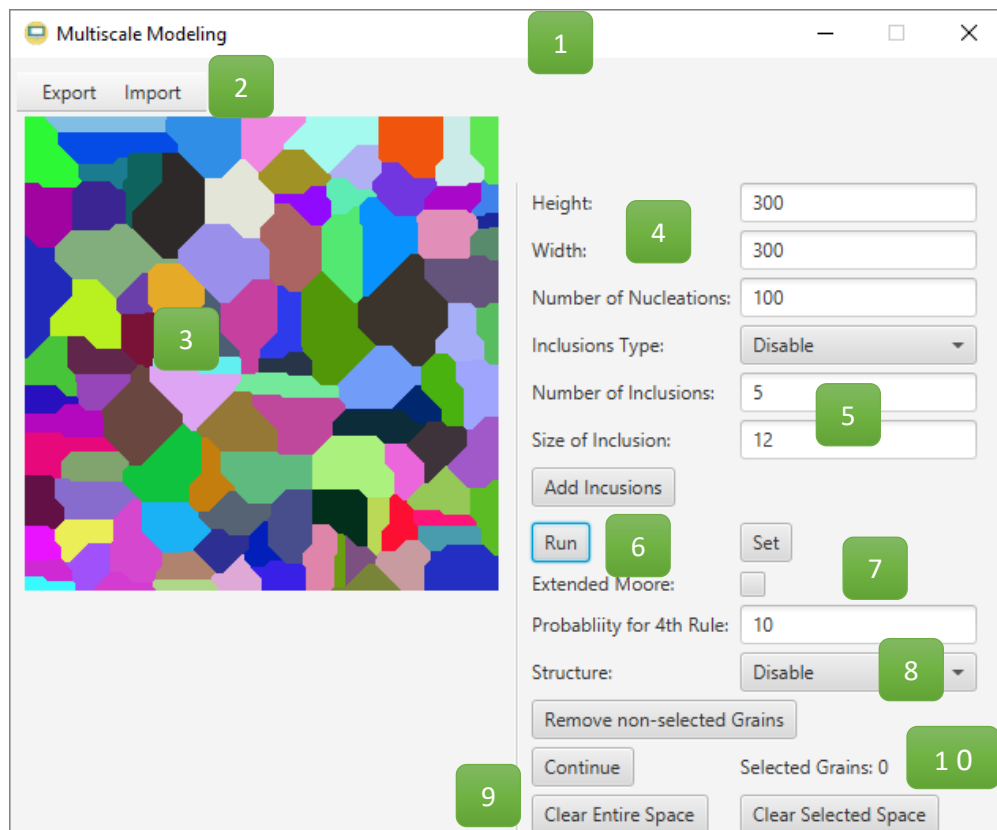
2) Graphical User Interface

Figure 1 GUI of application

1) Main Window of developed application.
2) Menu for importing and exporting files, in *.txt and *.bmp format.
3) Image of current microstructure, user can click on any part to select corresponding grain, to unselect it user have to press shift and while doing so also click on grain.
4) Main settings for simple grain growth, user can specify size of microstructure, number of nucleons.
5) Section dedicated to inclusions, user can specify their number, type (square or circular and position, on boundaries of random) and size after selecting these all what is left is clicking Add Inclusion button to add them and show in current microstructure.
6) Main buttons of app, Set is for confirmation of chosen settings and generate starting nucleation, Run is for begging simulation.
7) Panel with checkbox to enable extended more method (instead of simple grain growth and specify probability (1-100 %) for fourth rule of this method.
8) Drop down menu with structure types, there are 2 substructure and dual-phase. User have to specify with method He or She wants and then remove non-selected grains and after that generate new ones, by clicking Continue button.

9) Buttons to clear structure from grains and leave boundaries of previous structure, depends with button was chosen it will either leave boundaries of just selected grains or all of them.
10) Counter of selected grains, show information about their current number.

3) Results, application features

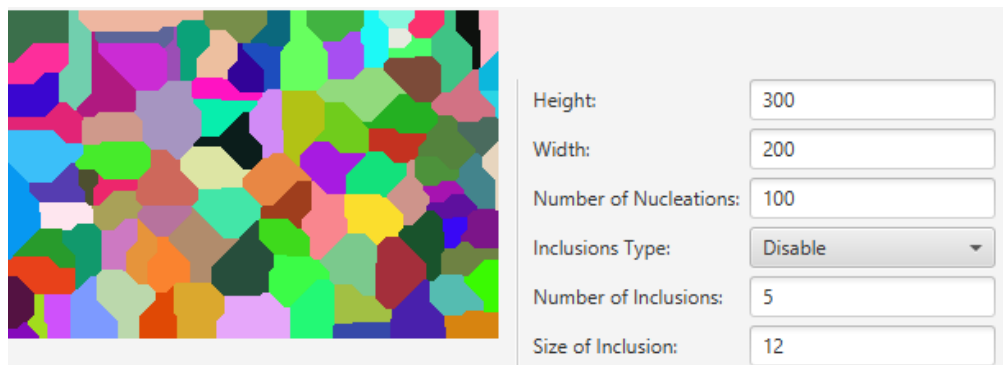a) **Class 1 - Organizational class - simple grain growth CA + visualization**

Figure 2 Simple grain growth

Implementation of Simple grain growth algorithm, with von Neumann neighbourhood. In GUI size of structure could be specified like in Figure 2, height and width is set to 300 and 200 respectively, initial number of nucleons is set to 100, after pressings Set and Run button, above (Figure 2) structure is generated.

b) **Class 2 – Microstructures export/import to/from txt files, pictures.**
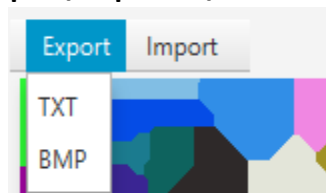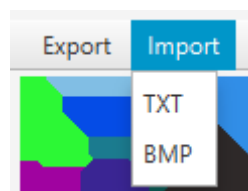

Figure 3 Export Feature
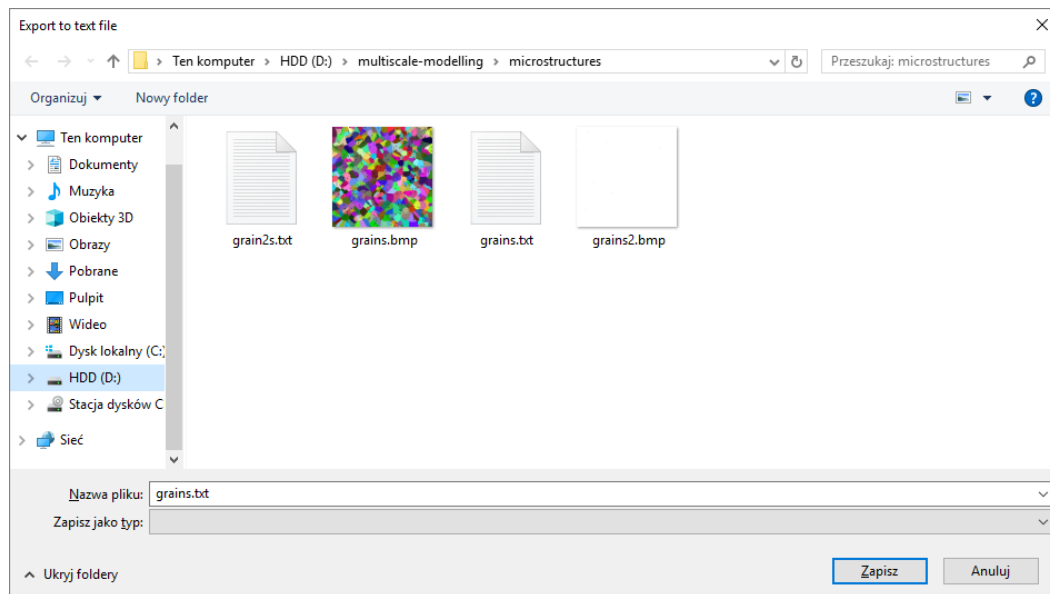

Figure 4 Import Feature

Figure 5 File Chooser

Generated structure can be imported and exported through GUI elements (see Figure 3 and 4), In case of text file following format was used:

```
300
300
0xffffffff
0
0xffffffff
0
0xffffffff
0
0xffffffff
0
```

Firstly size (width and height is specified), then for each cell in structure 2 information is saved: colour with is ID of each cell and it's phase.

After click of either import or export option file chooser (see Figure 5) pop up and user can specify name and location of file.

c) **Class 3 - Modification of cellular automata grain growth algorithm- inclusions (at the beginning/end of the simulation)**
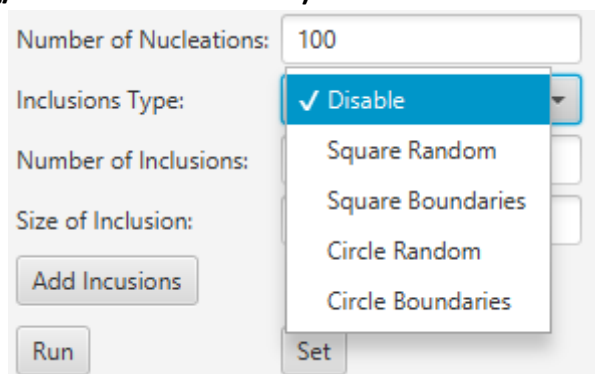
Figure 6 Inclusions type

Inclusions could be added at the beginning of simulation (only random ones), or at the end of it. User specifies (see Figure 1 part 5 and Figure 6) number of inclusions, its type and size (diagonal for circular inclusions and side for square ones) after that last thing wat is needed is pressing Add Inclusions button.
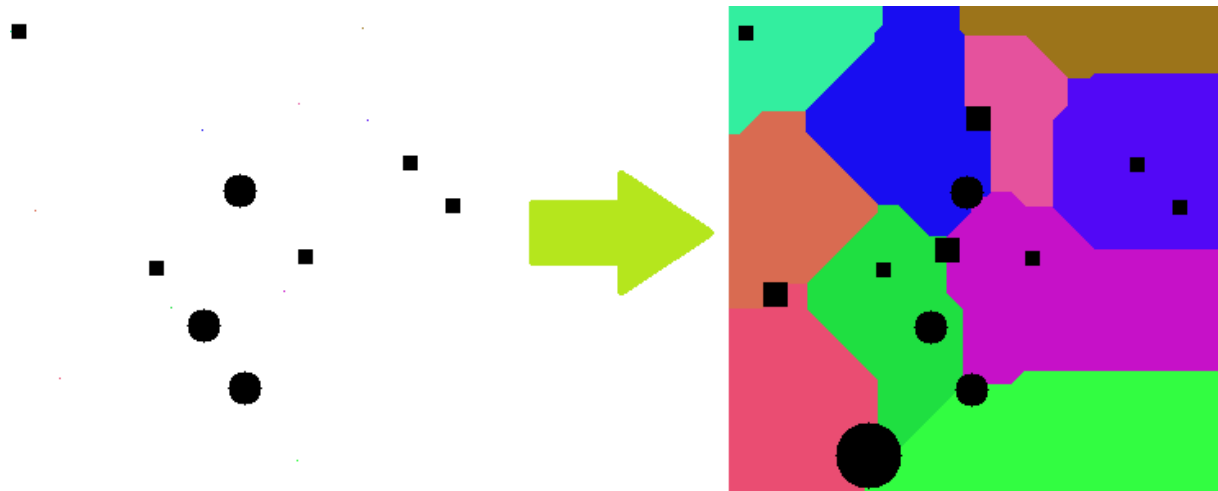

Figure 7 Different type of inclusions

In the figure 7 all possible type of inclusions were added, there are 3 small circles and 5 smaller squares added before simulation in random spots. Then simulation was completed and after that 1 big circular inclusion and 3 medium sized squares ones on boundaries were generated. Also example result could be seen in Figures 15,17,19,21.

### d) Class 4 - Modification of CA grain growth algorithm - influence of grain Curvature

Instead of simple grain growth user can use another, more advanced method called Extended Moore with adds more probabilistic aspect to simulation (see Figure 1 part 7). User specify probability for fourth rule by inputting any number in range 1-100
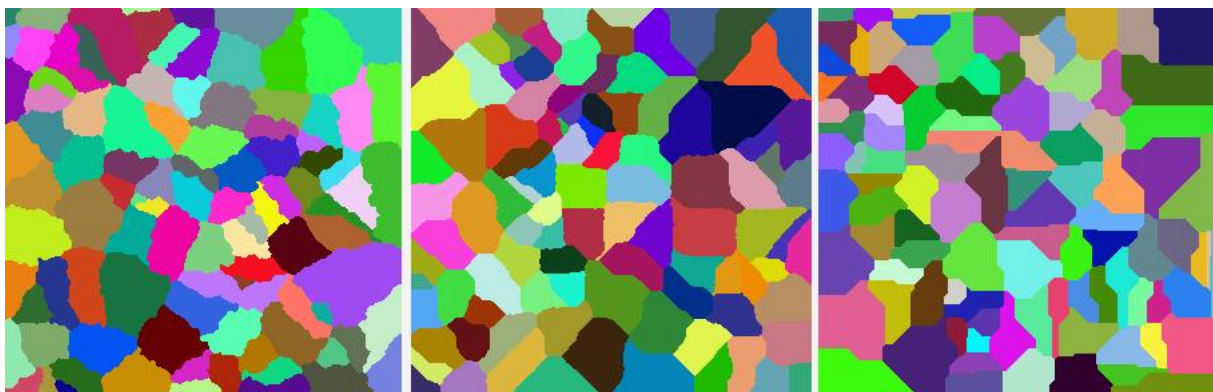

Figure 8 Comaprison of Extened Moore and Simple Grain Growth Methods

Results could be seen on above picture (Figure 8), first image shows Extended Moore with probability equals 10%, in the middle probability is equals 90% last image shows simple grains growth. Overall this method creates way to create slightly different microstructures.

e) **Class 5 - Modification of CA grain growth algorithm - substructures CA**
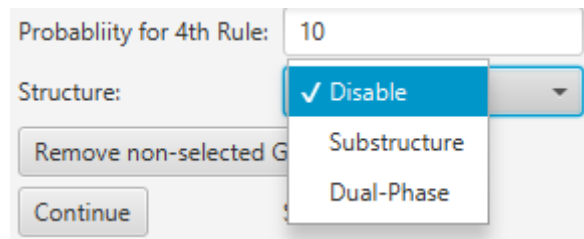

Figure 9 Extended Moore
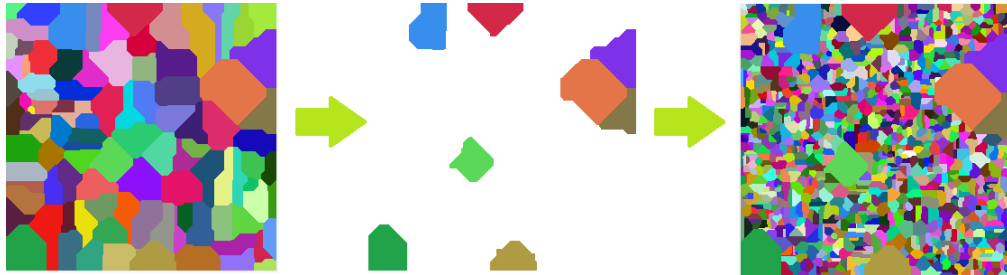
Figure 10 Creating substructures

After generating initial structure user can select any number of generated grains then choose in menu (Figure 9) type of substructure and press Continue, by doing so non selected grains are removed, and in their place new nucleons appear which after pressing Run button grow differently from initial state while selected grains remain as they were (in case of choosing Substructure mode, see figure 10) or they change phase and act as different phase (pink colour) which is show at figure 19.
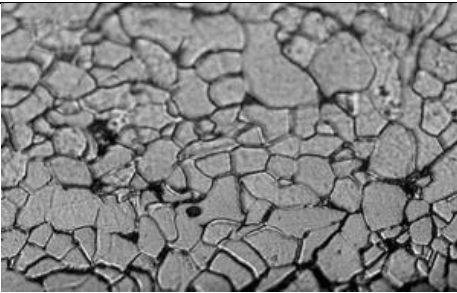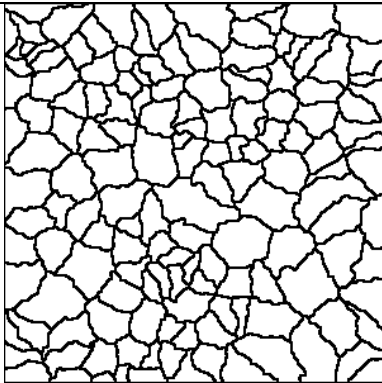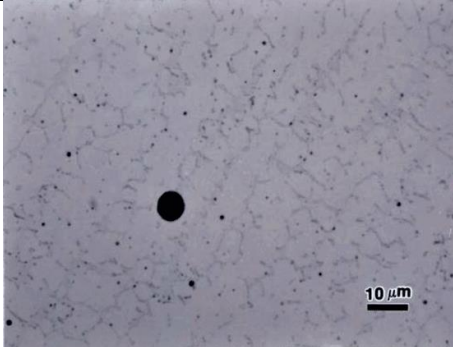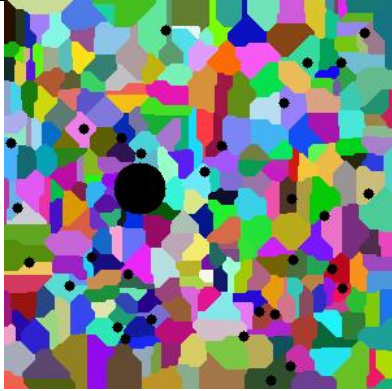
f) **Class 6 - Modification of CA grain growth algorithm – boundaries colouring**
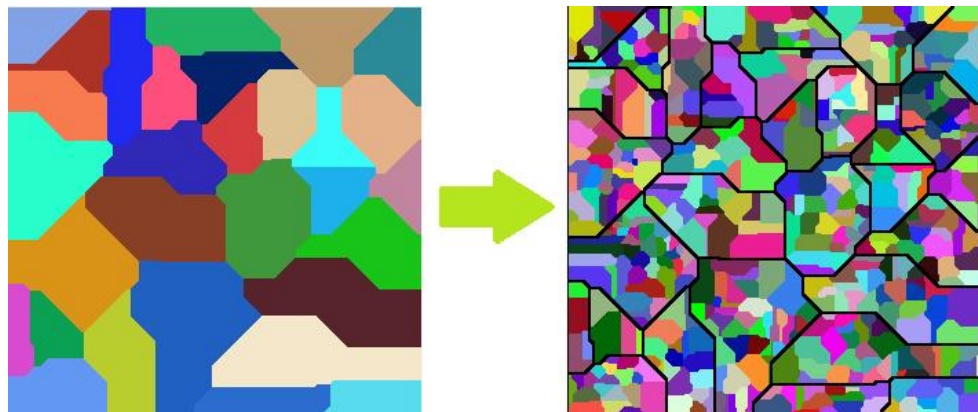
Figure 11 Boudaries colouring

User can select few grains on generated structure or all of them then by pressing Clear Space button (Figure 1, part 9) by doing so only boundary of grains remain as new phase which is shown on Figure 13. User also can create new nucleons inside old one which can growth as long as they do not cross created boundaries (see Figure 11).

4) Comparison with real microstructure

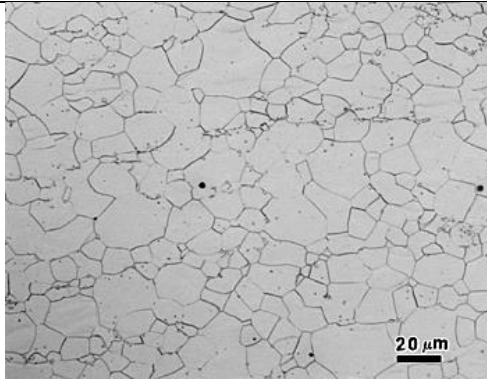| Real microstructures	Structures generated in App
Figure 12 Microstructure of steel after hot rolling https://galvanizeit.org/education-and-resources/resources/technical-faq-dr-galv/mechanical-properties-of-galvanized-steel 25.11.18	*Figure 13 150 Nucleons, Extended Moore with 10% probability for 4th rule.*
Structure of one phased element with many clear and smooth boundaries. Extended Moore method and an option of clearing generated space except boundaries is good example of features of developed program.	
Figure 14 Fine dispersion of carbides associated with the (faintly visible) delta ferrite in the 312 weld metal, in the as-welded condition (Murakami's, 60 s at 20 ºC). https://vacaero.com/information-resources/metallography-with-george-vander-voort/1061-identification-of-phases-in-stainless-steels-by-etching.html 25.11.18	*Figure 15 300 Nucleons, simple grain growth, 1 big circle inclusions (size 40), and ~30 small inclusion (size 8)*
Grains on figure 14 are not very visible but what is interesting are inclusions there is one big in the middle, any many small ones but few on boundaries and few inside grains. Application is capable of simulating this case.	

Figure 16 Ferrite grain structure of a motor lamination steel; 2% nital etch

https://vacaero.com/information-resources/metallography-with-george-vander-voort/894-microstructure-of-ferrous-alloys.html
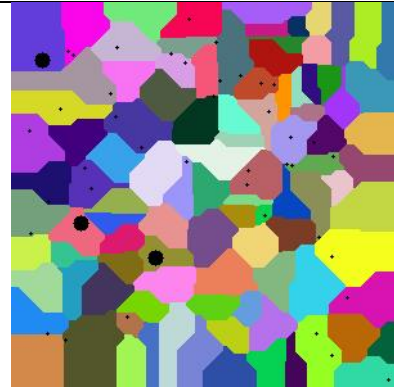25.11.18

Figure 17 3 circular inclusions size 12 and 40 with size equals 3, simple grain growth

One of basic cases. Just simple grains growth with inclusions but with different sizes 3 medium one and 40 small. Quite good result.
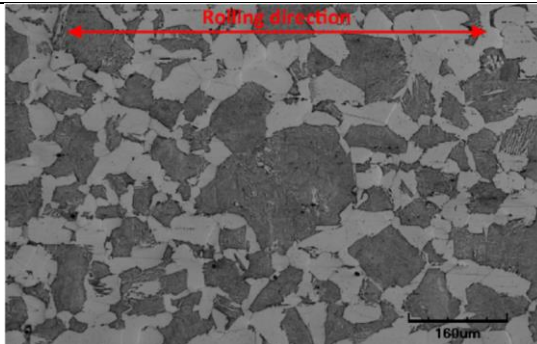

Figure 18 Microstructure of the dual phase steel

https://www.researchgate.net/figure/Microstructure-of-the-dual-phase-steel_fig1_272050203
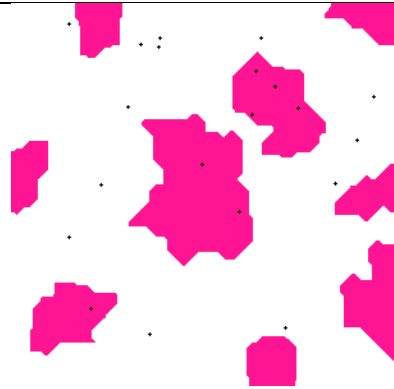25.11.18

Figure 19 100 nucleons, simple grain growth, 20 circular inclusions in random spots (size 3), few grains remained as dual-phase structure.

Example of dual-phase steel. White space (on Figure 19) is one phase, pink is another. There are a few small circular inclusions. More grains could be added but overall quite good result.
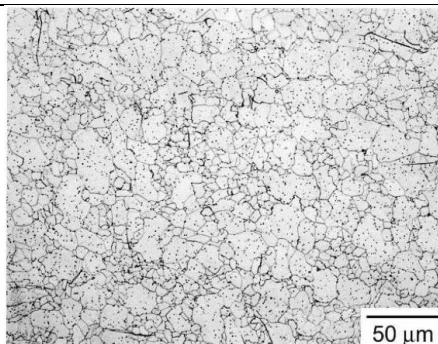

Figure 20 Initial microstructure of CP titanium

http://fstroj.uniza.sk/journal-mi/PDF/2011/17-2011.pdf
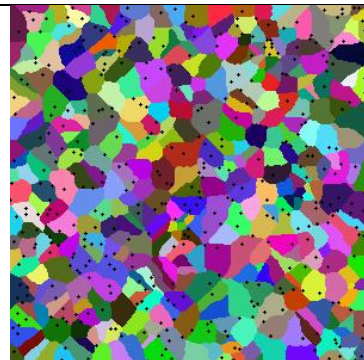25.11.18

Figure 21 400 nucleons, extended Moore method with 80% probability and 400 circular inclusions, size 2
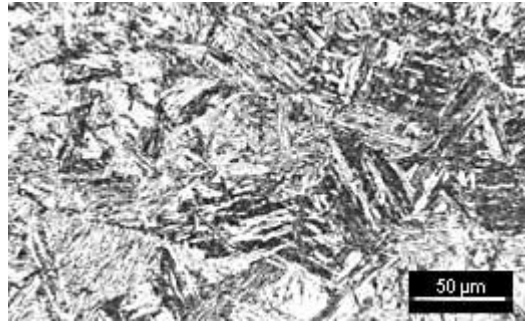
5) <u>Summary</u>

Figure 22 Martensite

https://www.twi-global.com/technical-knowledge/faqs/faq-what-are-the-microstructural-constituents-austenite-martensite-bainite-pearlite-and-ferrite/
25.11.18

- In many cases microstructures very similar to "real micro structures" could be generated by this application, but not all of them (see Figure 22), there are cases where more tools are needed.

- Generated microstructures could be saved for example to BMP format and then used in CAE type programs to create better, more realistic models.

- Program could be expanded in many ways, for example by implementing more inclusions types or grains like "needle shaped" (see Figure 22) or creating another module for 3D simulations.

- Coding was quite easy but took some time, but despite correct implementation program could be refactored and optimized. For example, threads could be used for heavier computing wise functions or code could be spited to more smaller files.