Raport

Karol Kiełbasa

1. Technology

Application was written in Java 8 language because Java has built-in support for graphic library named Swing. In addition community using those library is alive and there are a lot of online materials.

2. Interface

Image 1. Main window of application.

Right after launching the program main window appears. On the top there are two options: **File** and **Simulation**. In file menu there is possibility of importing and exporting results of simulation in .bmp and .txt formats and also there is a button to exit the program.

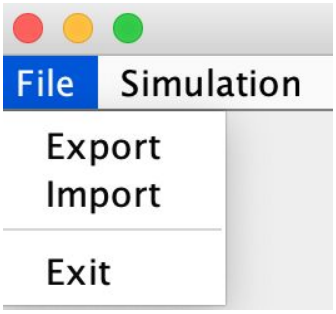


Image 2. File menu

To export/import user need to firstly specify a format it is .bmp or .txt

Image 3. Selection of format.

Then a path where to export/from import a file.

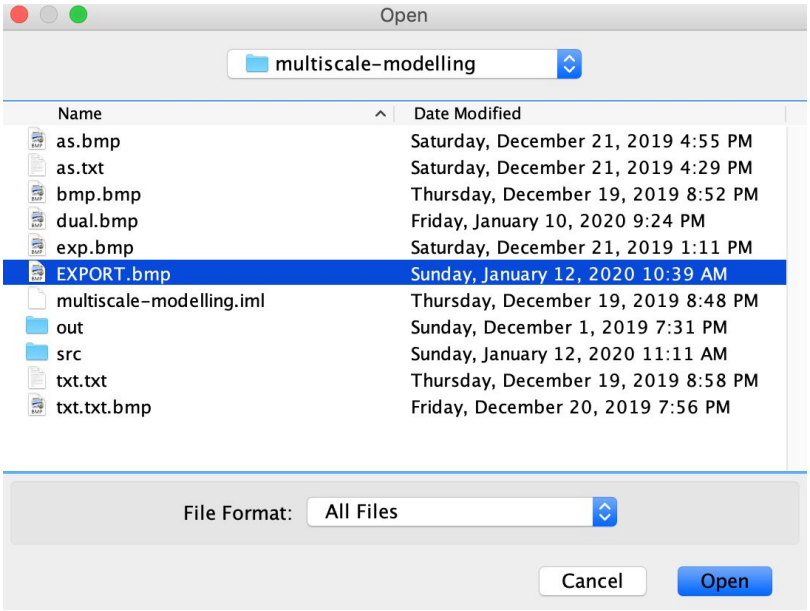


Image 4. Selection a path to export/import.

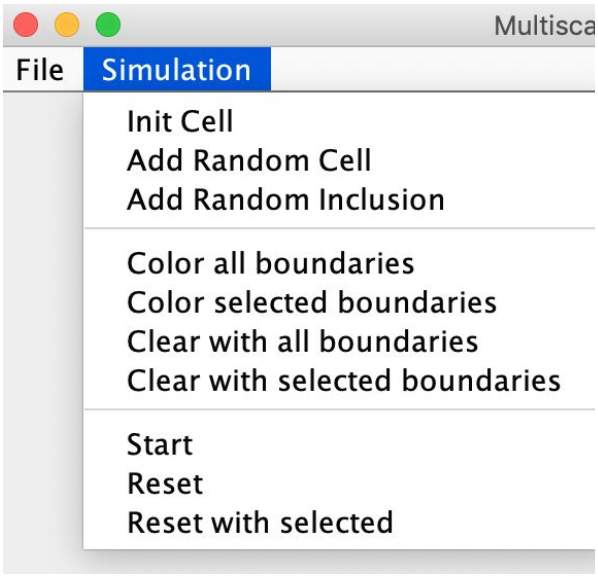


Image 5. Simulation menu

In Simulation menu there are options:

- Init Cell - initialize a graphic components where cells will be visible. User can provide size of a component and percentage rate applied for growth rules.

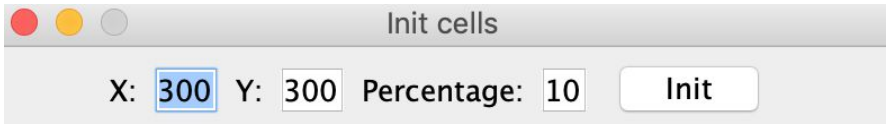


Image 6. Init Cells menu

- Add Random Cell - adds a given number of cells to component at random position

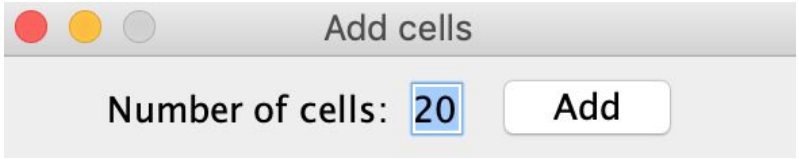


Image 7. Add cells menu

- Add Random Inclusion - adds inclusion to component at random position. User can provide a number of inclusions, size and type. Type might be circle and square.

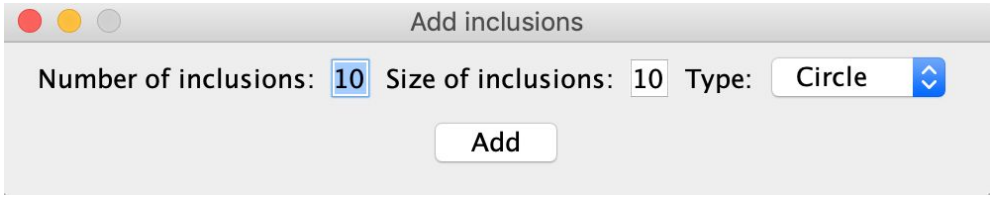


Image 8. Add inclusions menu

- Color all boundaries - colors all boundaries between all nucleons
- Color selected boundaries - colors boundaries between selected nucleons
- Clear with all boundaries - clear all cells except colored boundaries. User na provide a boundary size.

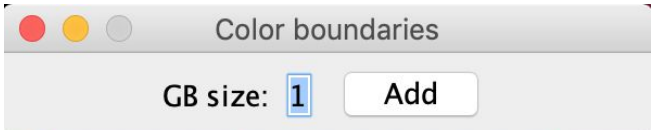


Image 9. Color boundries menu.

- Clear with selected boundaries - clear all cells except colored boundaries. User na provide a boundary size.

- Start - starts simulation
- Reset - resets simulation
- Reset with selected - reset all cells except selected nucleons. Selected nucleon's microstructure may be changed to dual phase or substructure

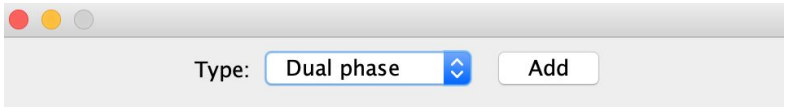


Image 10. Add Reset with selected menu

3. Results

Simulation started with 20 cells which grew. Then after simulation 4 inclusion in the shape of a circle was added and colored boundaries between all inclusions.

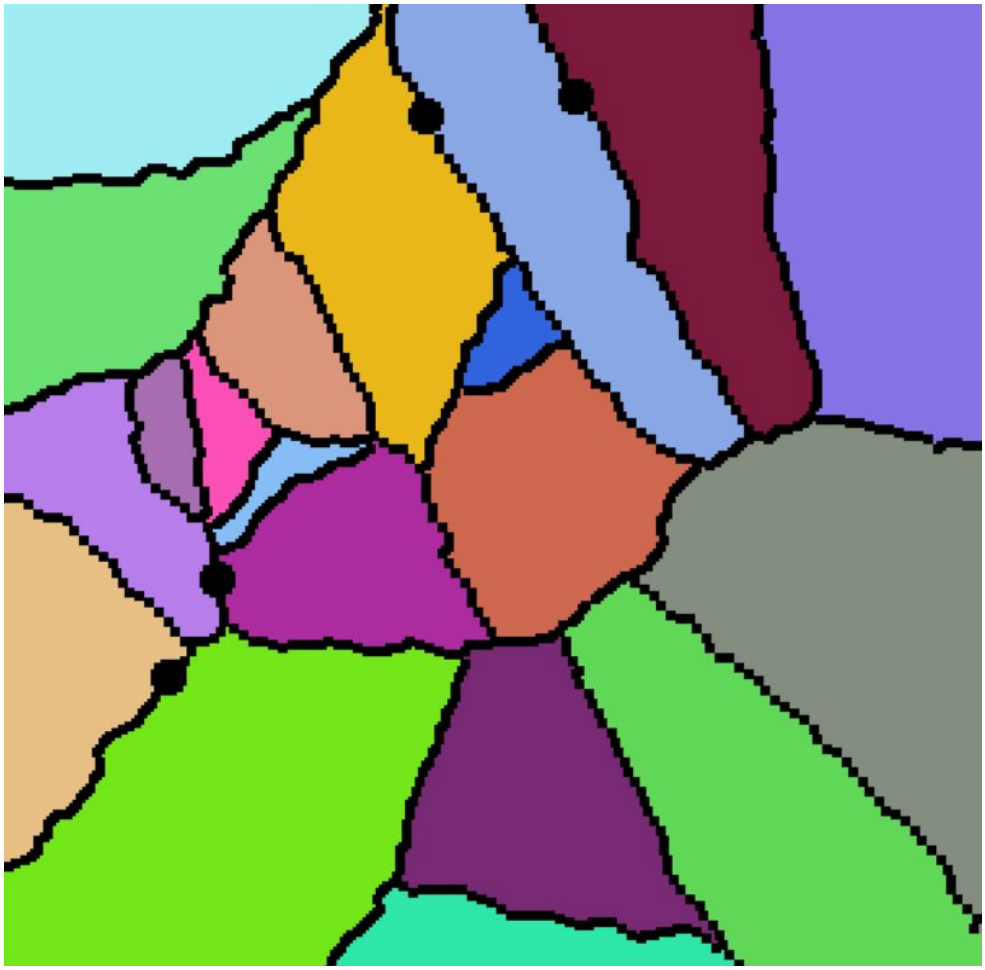


Image 11. Results of simulation

Simulation started with 20 cells which grew. Then all boundaries was colored and nucleons were cleared.

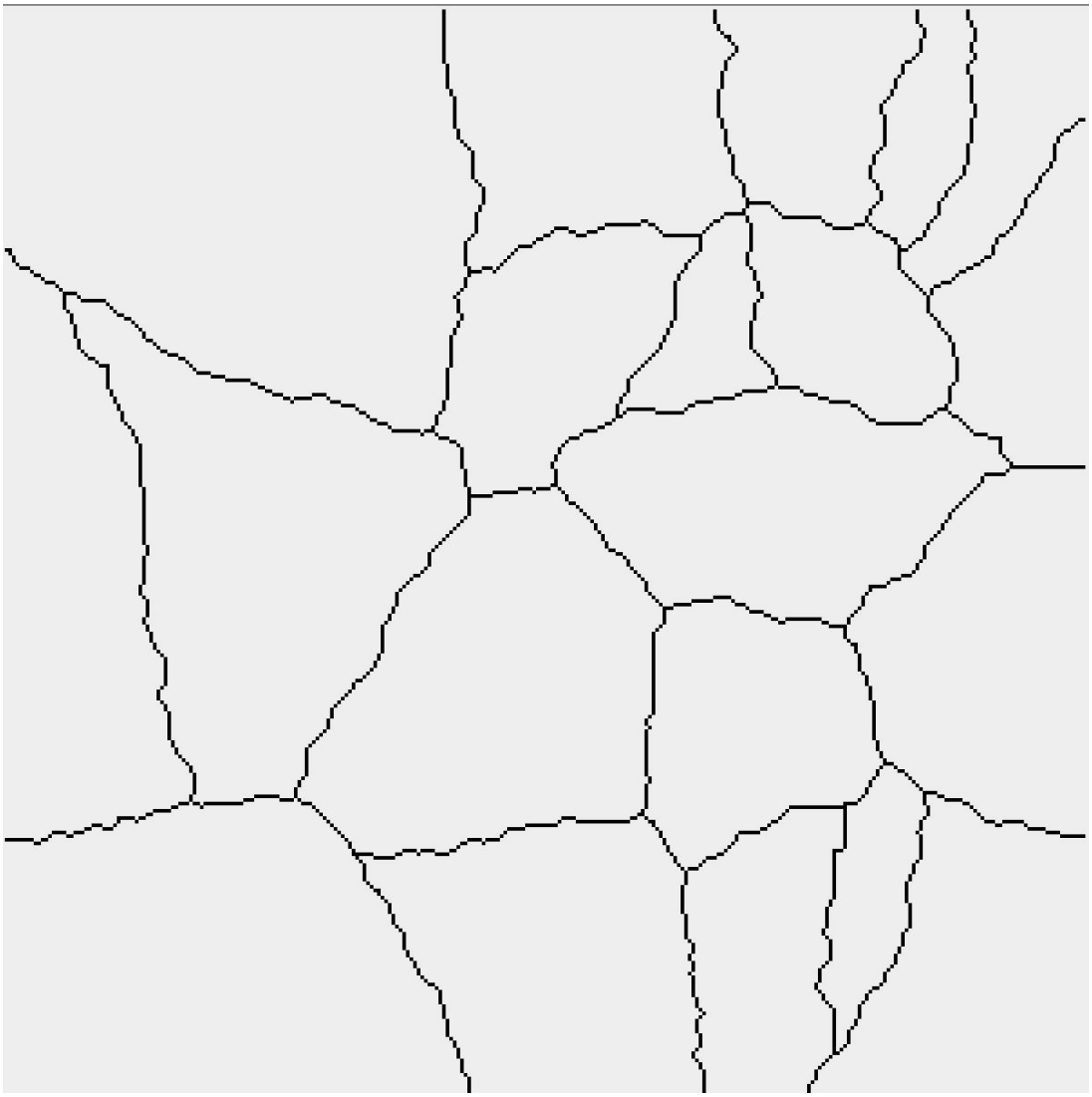


Image 12. Results of simulation

Simulation started with 20 cells which grew. When simulation was finished. Few nucleons were selected on the corners of image then changed a substructure with dual phase. In order to populate gaps between ‘big’ nucleons added 90 cells and started simulation once more. Last step was adding inclusions with different size and shape.

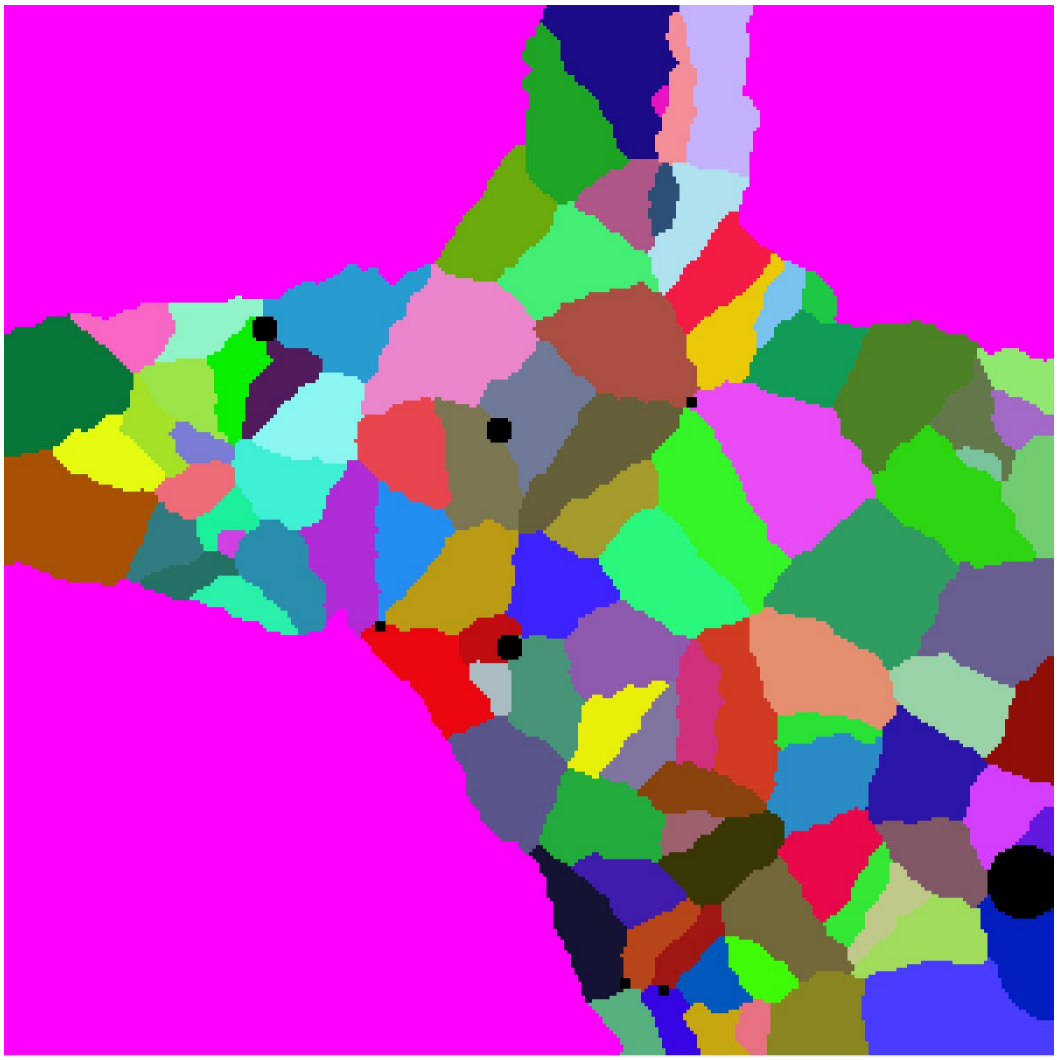
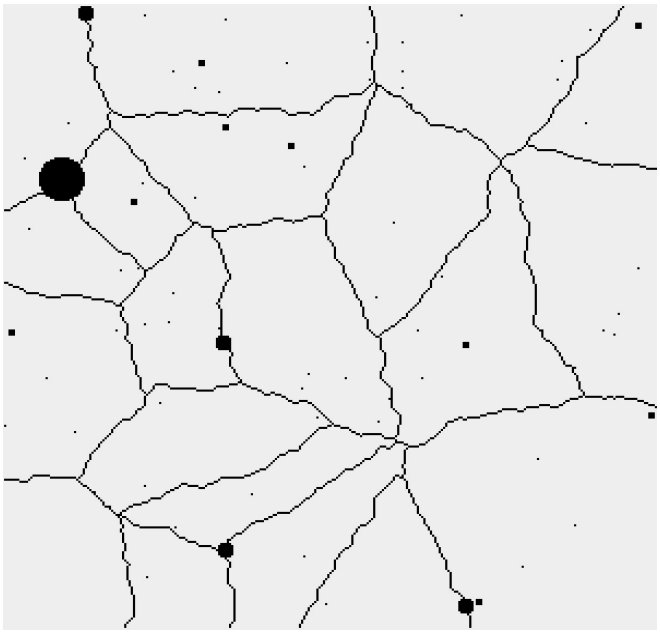
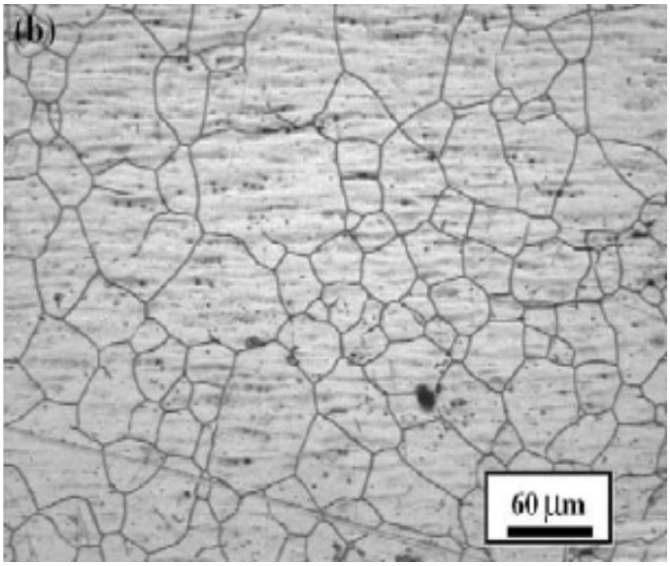


Image 13. Results of simulation

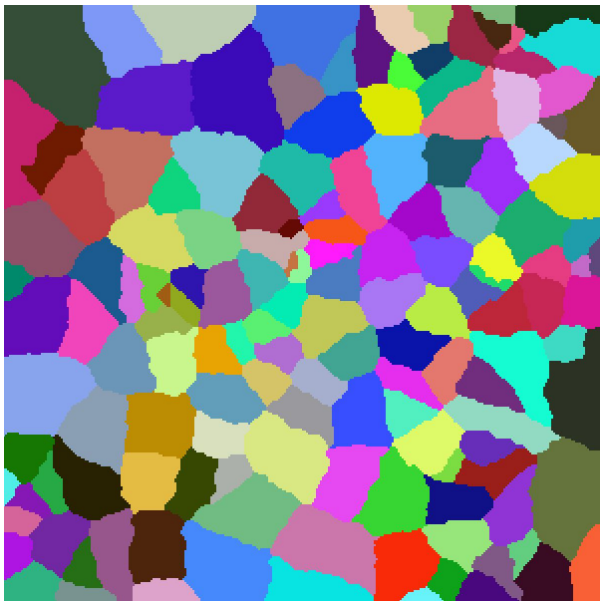
4. Comparison with real microstructure

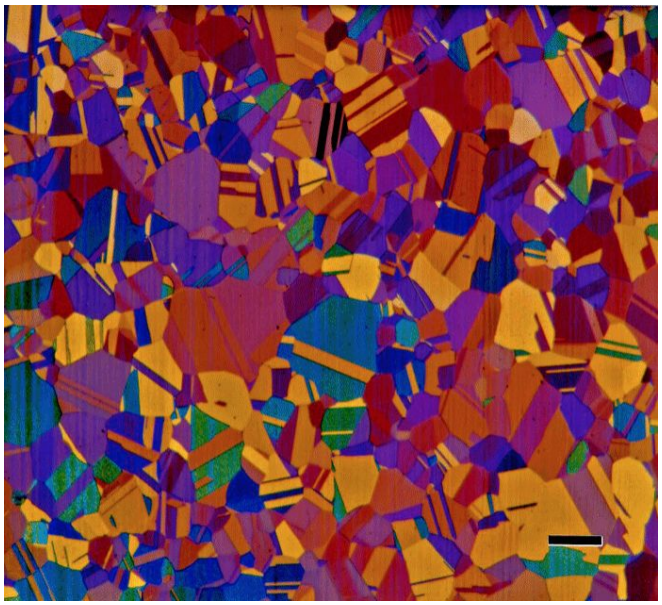
a)





b)





5. Conclusions

Application is able to handle basic operation like simple cell growth, adding inclusions, changing substructure, coloring boundaries of nucleons. Despite application simpliness results of simulation can be more or less similar to real microstructure as example a) shows. However, the goal of such applications is to reproduce a microstructure as similar as it would be in reality. Achieving this goal requires more advanced features like drawing boundary which is not connected to any other boundary. In addition example b) shows that application do not handled reproducing microstructure very well. Main reason for that is application doesn't support drawing nucleons, which have the shape of a rectangle.

To sum up, implementing an application which is able to reproduce real microstructures is very complex and challenging task, which requires knowledge not only about programming but also metallurgy. Application described above supports only trivial operations, which do not allow to fully reproduce real microstructures.