Multiscale Modeling – 1st report

13.11.2018
Marcin Lichota

Abstract

This report applies to the project that was created for a period of 6 laboratory classes. It was created due to the requirements posted on the lecturer (Mateusz Sitko) page. Report covers technology, functionality, examples and summary of achieved results in reference to real world cases.

Technology

As there is a few aspects of technology, I allowed myself to list them below, attaching short description:

- **Application as Web Application** – computer program witch client runs in a web browser, often it requires Web Server to host source files and data. In case of this project, there is created only source code and bundle but there is no Web Server, only in development mode.

- **Programming language JavaScript** – Scripting programming language. Together with HTML and CSS, is a basic tool for building websites. Using the Node.js environment also allows creation of efficient applications operating regardless of the browser. In this project, JavaScript was used as base tool to write application.

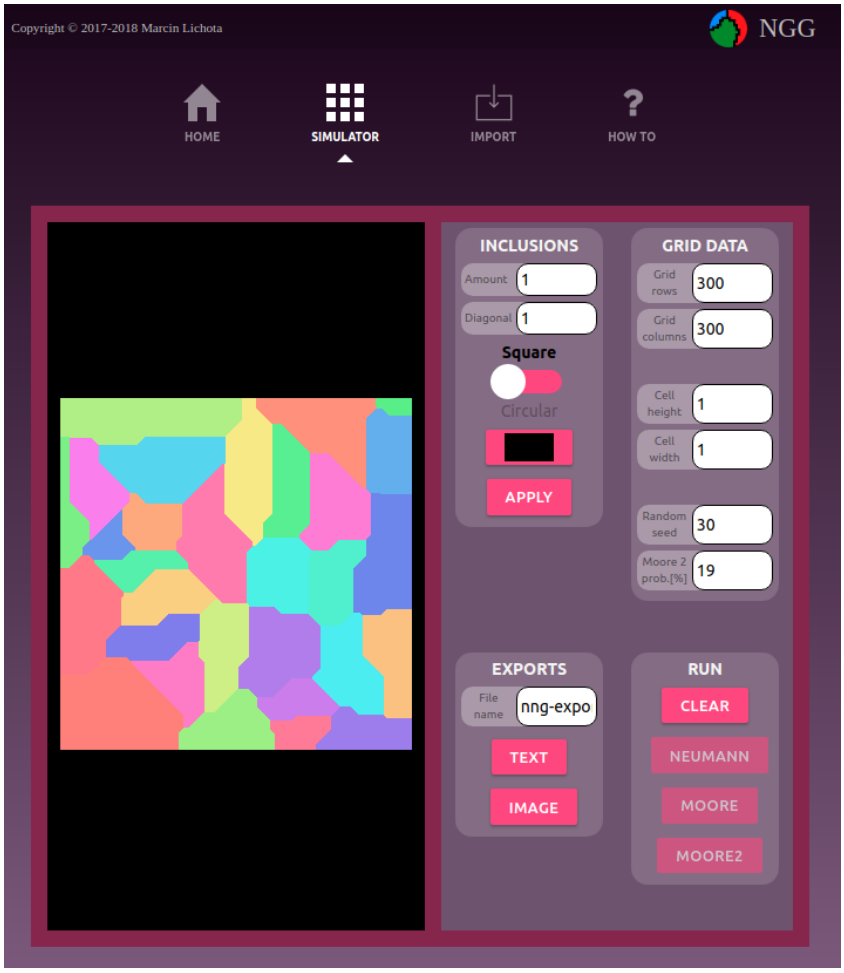
- **Git** – distributed version control system for tracking changes in computer files and coordinating work on those files among multiple people. Exactly in the design used for version control and communication with Git Hub repository.

- **React.js** – JavaScript library written by Facebook programmers, used to create graphical user interfaces. It uses declarative, component-based approach. It allows to reuse components across multiple projects and from external sources. In this project React.js is base for whole application. It was used by applying create-react-app that provides boilerplate with react-scripts configurations – best practice configurations for the single page application collected by React Team.

- **Redux –** Predictable state container for JavaScript applications. It helps to write applications that behave consistently based on **Flux** architecture. Redux was used for this project to assure *single source of truth.* It provide limitations that are forcing to one way data flow. Due to that it was much easier to manage current application state.

Functionality

- **Simple grain growth** – base method of grain growth based on the cellular automata technique. In this project it is possible to define up to 600x600 cells surface. It implements 2 neighborhood types (Neumann and Moore) and one transition rule – absorbing boundary condition. This implementation required few factors to make calculations: grid size, cell size and random seed. Random seed is required to initialize seeds that are extended at next steps.

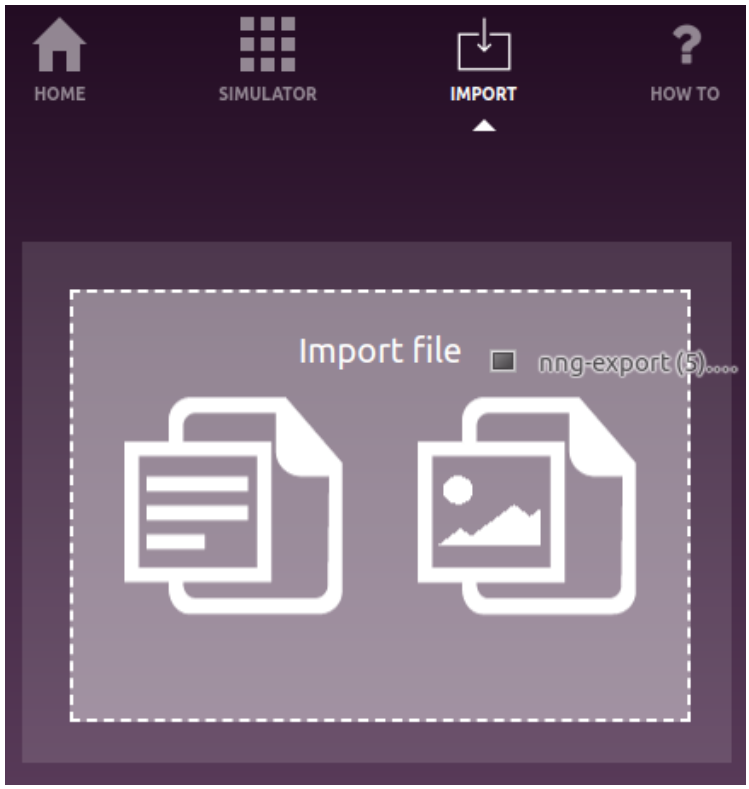


Example of Naumann neighborhood calculation.

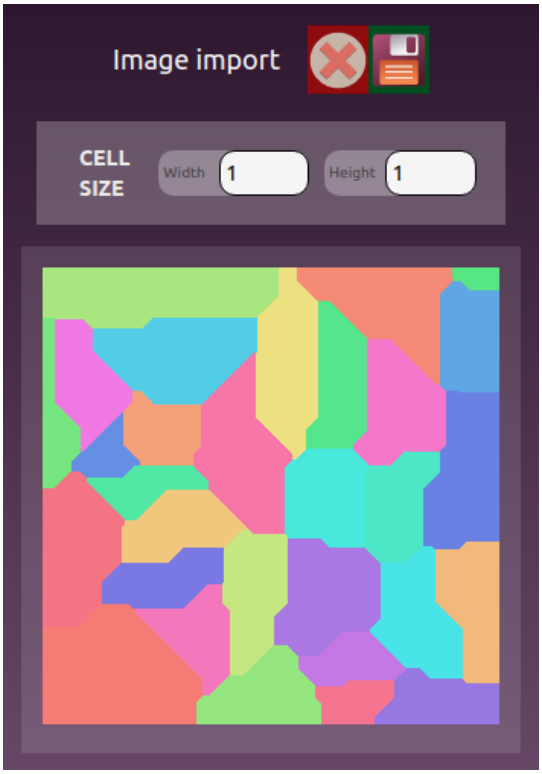
- **Microstructures export/import to/from file –** in project there are two handled file types: image and JSON file. Exports are quite straightforward – if there is created project, it requires to specify file name and hit TEXT or IMAGE button in EXPORT section. Then there is need to specify path where to save file on local storage.

Example of exported JSON file

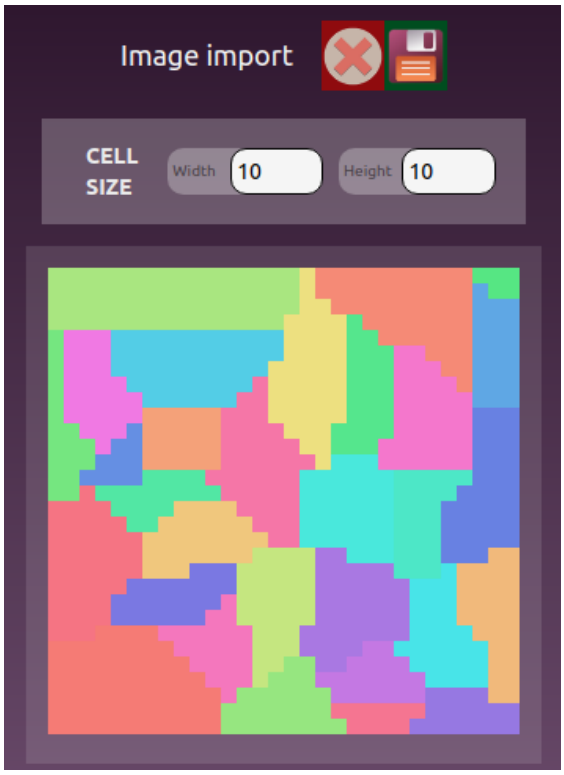
Although IMAGE export is much lightest than JSON export. At JSON export there are more information that will be initialized when imported to the project. For the import there is special import page that have drag and drop area. There is possibility to drag and drop file or select it from the *select file* dialog.



After selecting file there is preview area where last adjustments might be made.

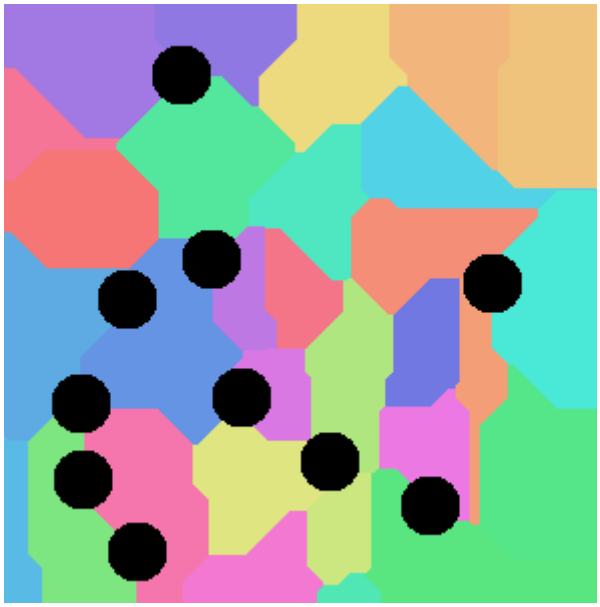


During image import there is possibility to set cell size – this will lead to quality reduction but performance improvement. Below there is the same structure but with bigger cell size:

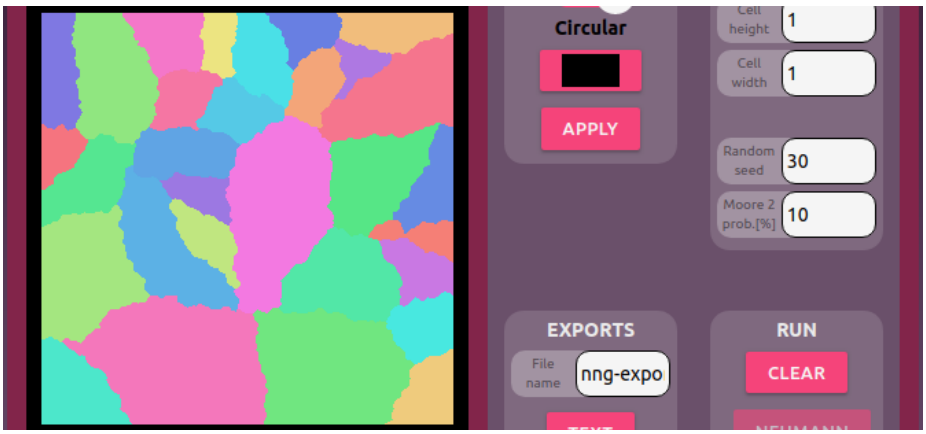


Finally, when all factors are adjusted. Save button will export project to the main view.

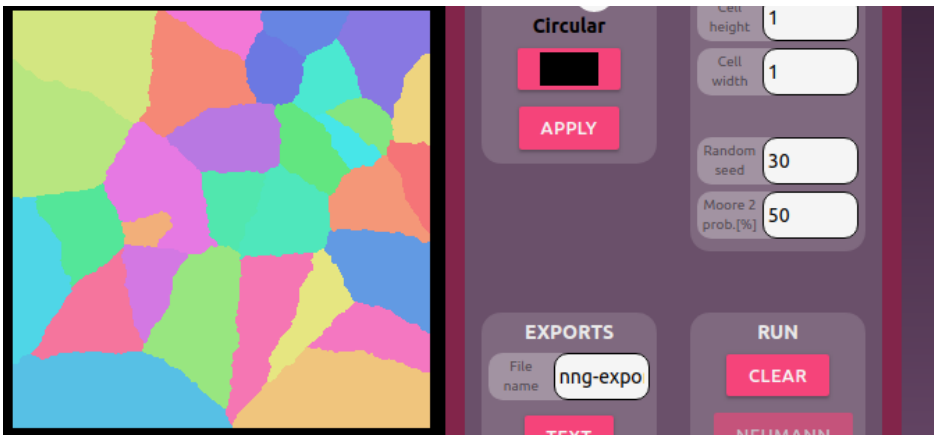
- **Inclusions** – there is menu created in the project that allows adding inclusions at the beginning or the end of simulation. There are two shapes of inclusions: square and circular. Additionally there is color picker that lets specify inclusions color. It might be 2 types of inclusions specified in one simulation. Here is example:



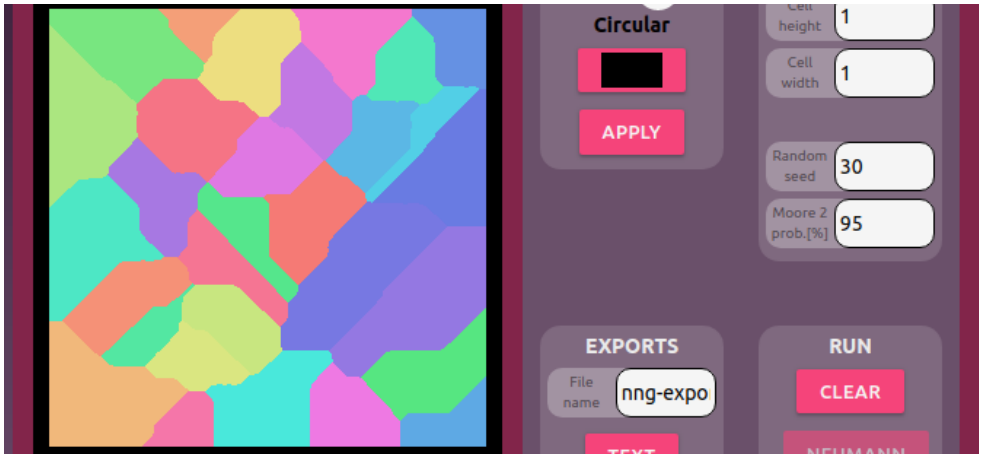
- **Algorithm with special rule of neighborhood –** there is additional button in the RUN section in project – it allows to run algorithm with boundary shape control. This requires to fill special factor – probability factor for last rule. Here are few examples for different probability factor. Highest factor – smoother boundary of the grain.



Example MOORE2 with 10% prob. factor.

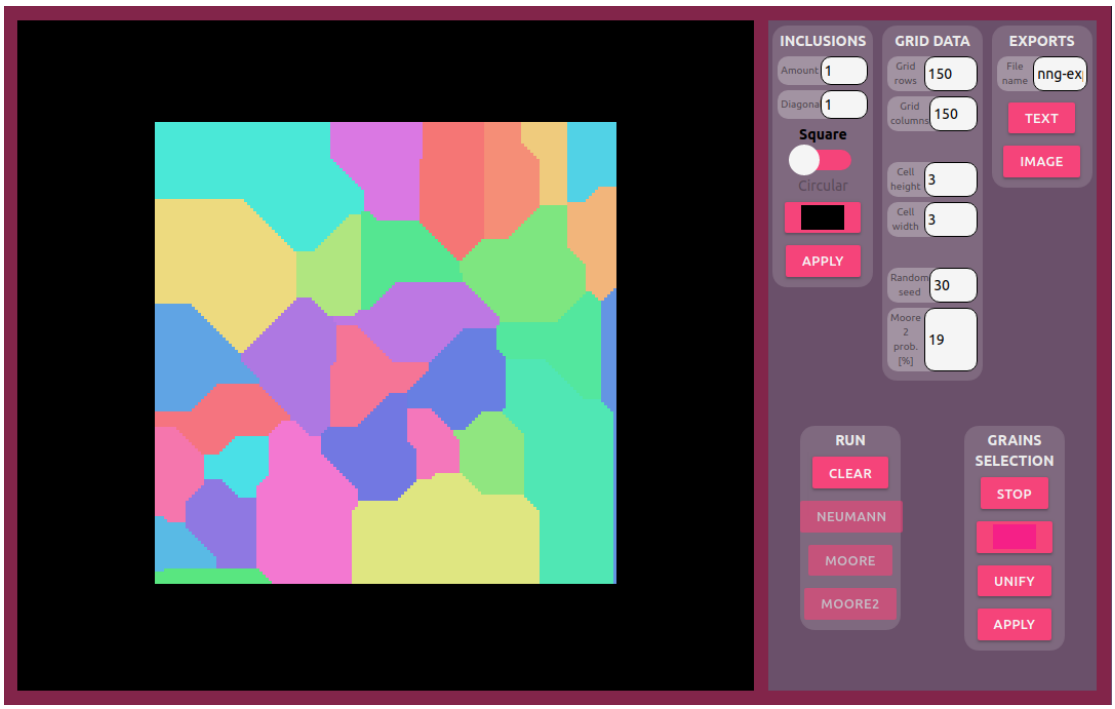


Example MOORE2 with 50% prob. factor.

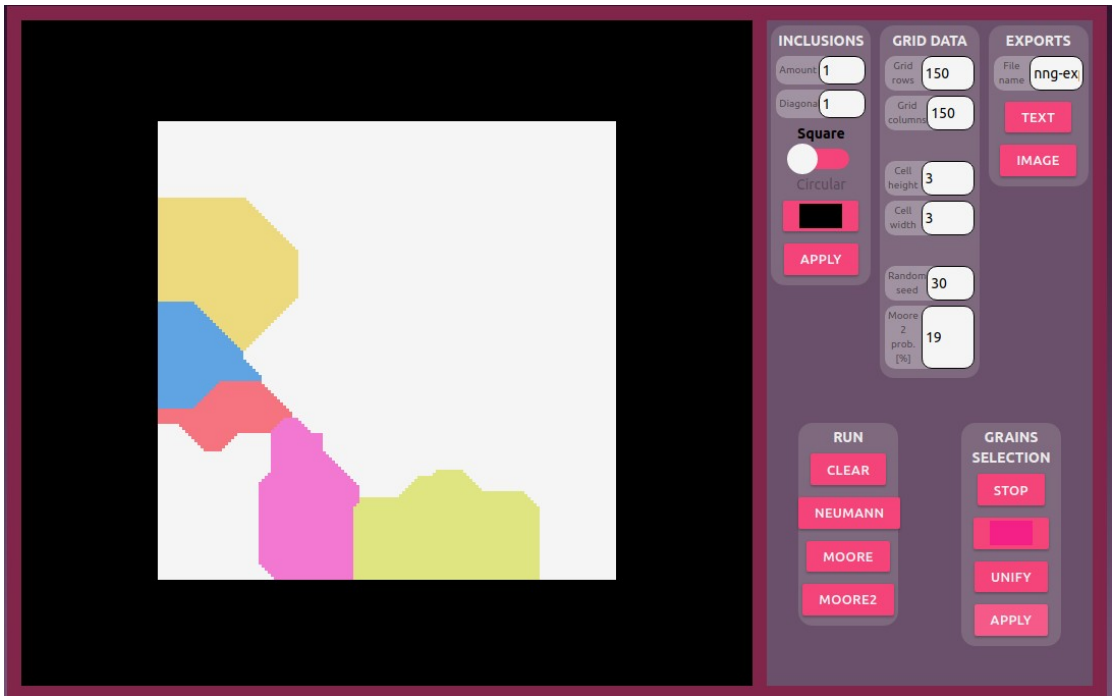


Example MOORE2 with 95% prob. factor.

- **Substructures and selecting specific grains –** *GRAINS SELECTION* section in project, allows to START grains selection, picking color of that will be applied when UNIFY, and UNIFY all selected grains. After APPLYING changes, selected grains will stay not changed and then there is possibility to generate additional grains.



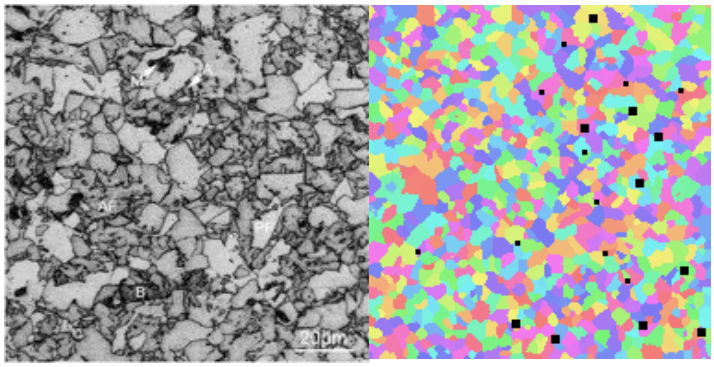
During grains selection.



Applying changes.

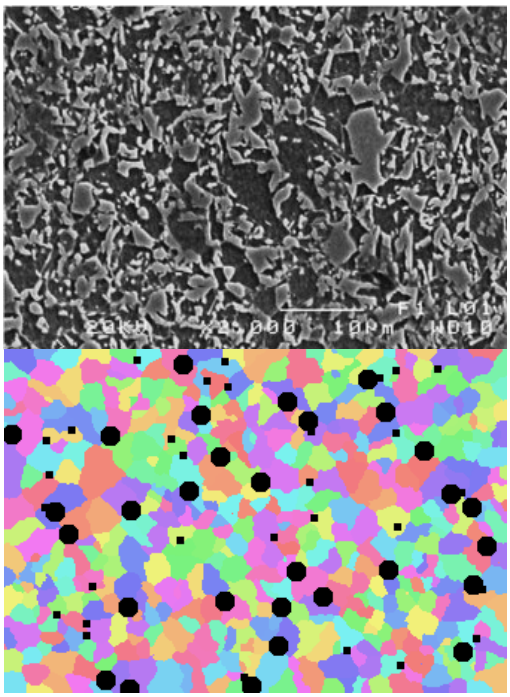
Comparison with real cases

Example 1. Liu, Q., Liu, W., & Xiong, X. (2012). Correlation of Cu precipitation with austenite–ferrite transformation in a continuously cooled multicomponent steel: An atom probe tomography study. *Journal of Materials Research, 27*(7), 1060-1067. doi:10.1557/jmr.2012.54



Continuously cooled multicomponent steel.

Example 2: Sławomir Krajewski, Jerzy Nowacki (2013). Properties of dual-phase (DP) steels welded joints. Welding Technology Review.



Summary

The results obtained from the simulation were compared to the actual data. It turned out that they do not accurately reflect the phase structure. However, with some approximation they are suitable for further analysis. The program itself has a lot of room to improve its quality both in terms of correctness of the simulation and the user interface.