Sylwester Cichocki

Multiscale Modeling – first report

1. Introduction

In the second part of classes previous application was extended by adding Monte Carlo growth algorithm and also Monte Carlo recrystallization algorithm.

2. Programming language

This project was implemented using C# language with Windows Forms framework. C# and Windows Forms library provide tools that are easy to use to create Graphical User Interface. Thanks to this one can easily create GUI and focus on back-end side of application.

3. Graphical User Interface – description

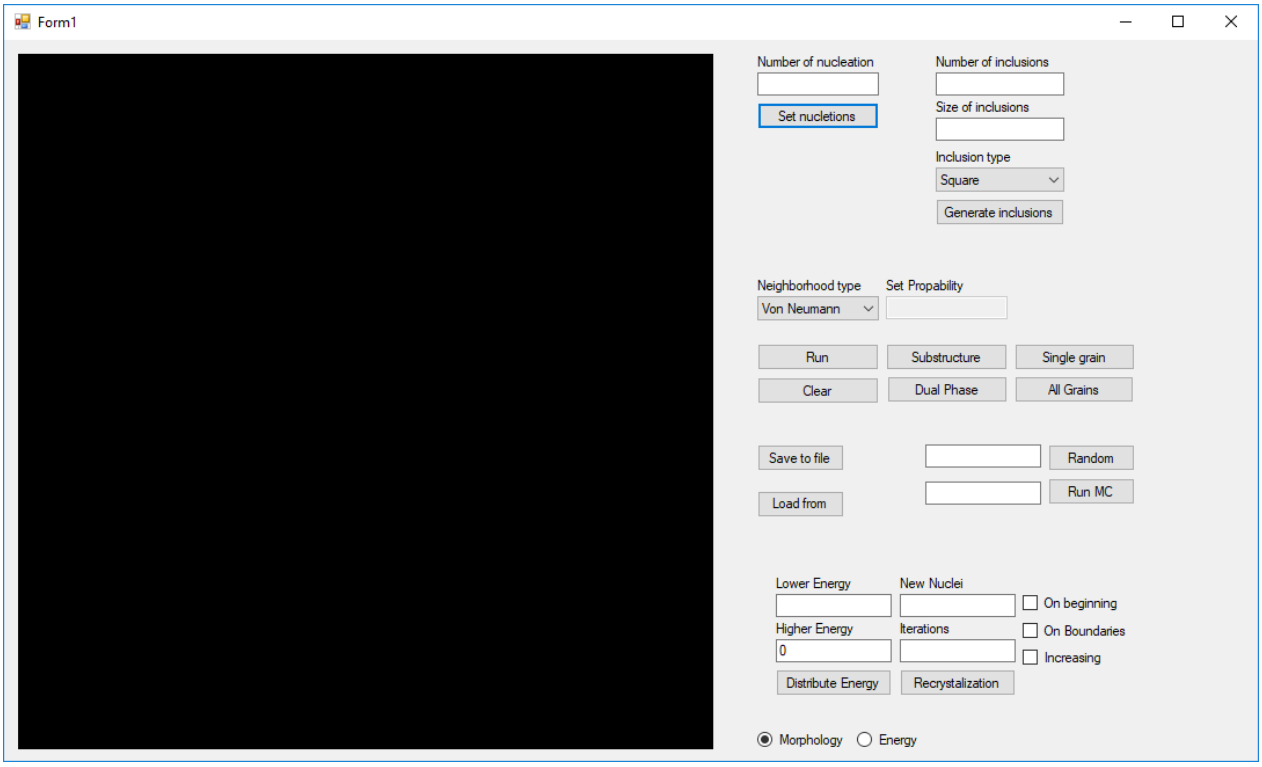

Figure 1 Main window

Figure 1 shows how program looks like. On this picture we can see available configuration options. Below are described options which was added during second part of classes:

- Random and Run MC – in text boxes next to this two buttons user can specify number of states needed to generate initial structure for Monte Carlo growth algorithm and also number of iterations of MC algorithm.
- Lower part of user interface is responsible for recrystallization algorithm. In left two boxes user specify energy distribution. When higher energy value equals 0 then energy will be distributed homogeneously otherwise it will be heterogeneous distribution with higher value of energy distributed on grain boundaries. Two radio buttons at bottom allows user to peek how energy was distributed. New Nuclei allows user to specify how many grains will recrystallize on beginning of simulation if you check On beginning checkbox otherwise they will be added at beginning of each iteration. If you check increasing checkbox then number of new recrystallized nuclei will be multiplied by 2 after each iteration.

4. Results

a. Class 8: Monte Carlo grain growth algorithm

Figure 2 shows example of Monte Carlo grain growth algorithm performed by application. On the left is initial structure with three different states. On right is result of 100 iterations of MC algorithm.

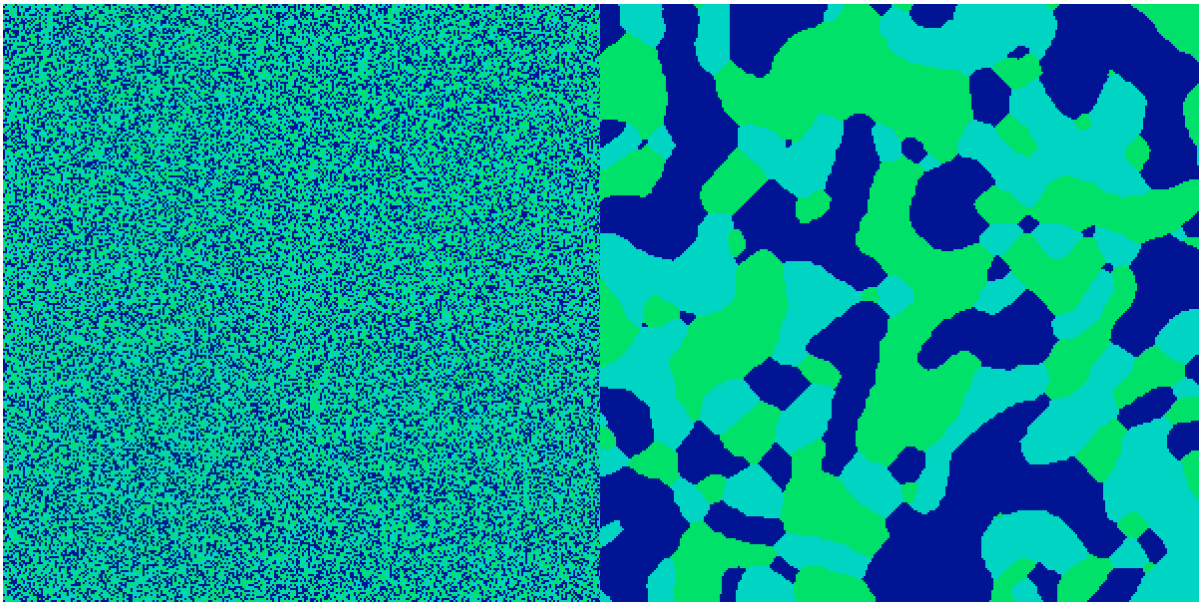

Figure 2 Example of Monte Carlo grain growth algorithm

b. **Class 9: Dual phase microstructure**.

In this classes the task was to ensure compatibility with dual phase option from first part of classes and also compatibility with previous simple grain growth algorithm. Results are shown on Figure 3 where dual phase option was performed on microstructure generated by MC and then final structure was also generated by MC algorithm. On Figure 4 are results where two algorithms are combined.

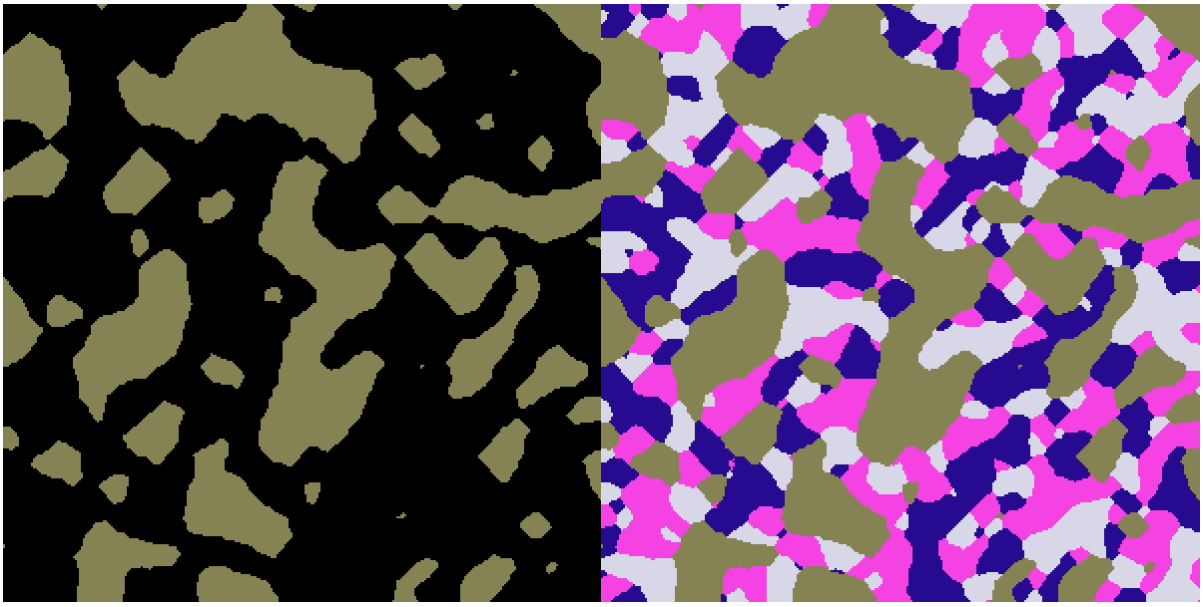

Figure 3 Example of dual phase MC -> MC.

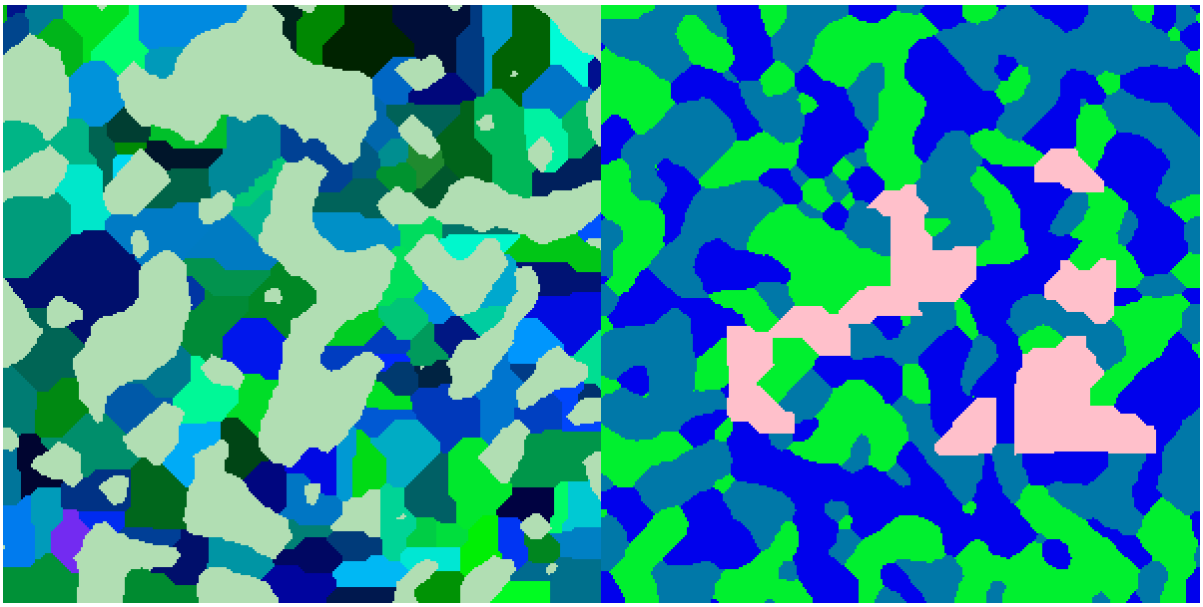

Figure 4 Example of dual phase on the left MC->CA on the right CA->MC.

c. Class 10: MC static recrystallization algorithm – energy distribution:

Here is example of energy distribution in app shown on Figure 5. There are two types of distribution first if user specify only lower energy value and higher is left 0 then energy is distributed homogeneously which means every cell has the same value of energy. In the second case the higher value of energy is distributed on grains boundaries. The higher value of energy is represented by lighter green color.

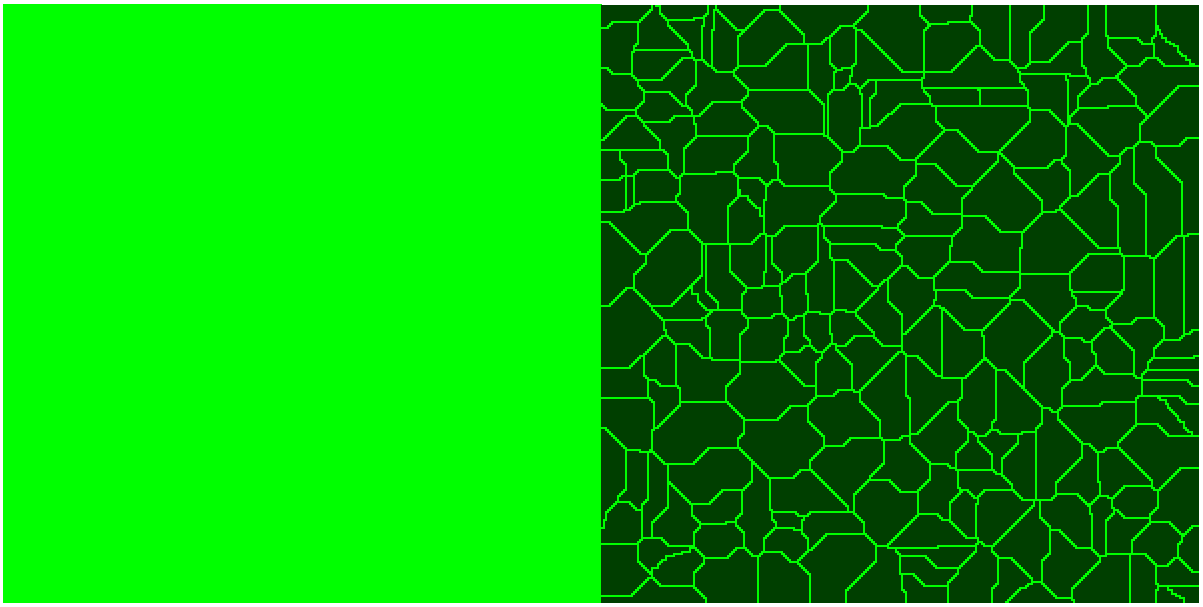

Figure 5 Energy distribution homogeneous on left heterogeneous on right.

d. Class 11 and 12: MC static recrystallization algorithm – nucleation and growth:

In this classes we implemented MC static recrystallization algorithm with different types on nucleation. Examples of different approaches to nucleation is shown on Figures 6 and 7. On Figure 6 all recrystallized nucleons are added at beginning of simulation. On Figure 7 is result of adding 10 new nucleons in each iteration. Both examples are generated by using homogeneous energy distrybution.

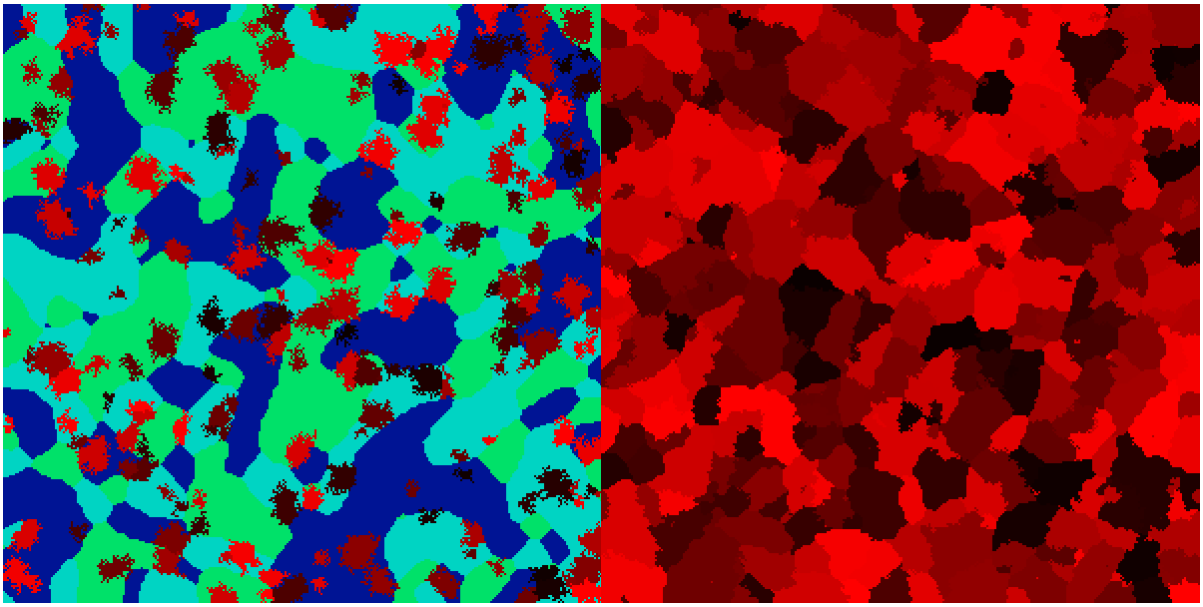

Figure 6 Example of 100 recrystallized nucleons added at beginning of simulation with homogeneous energy distribution.

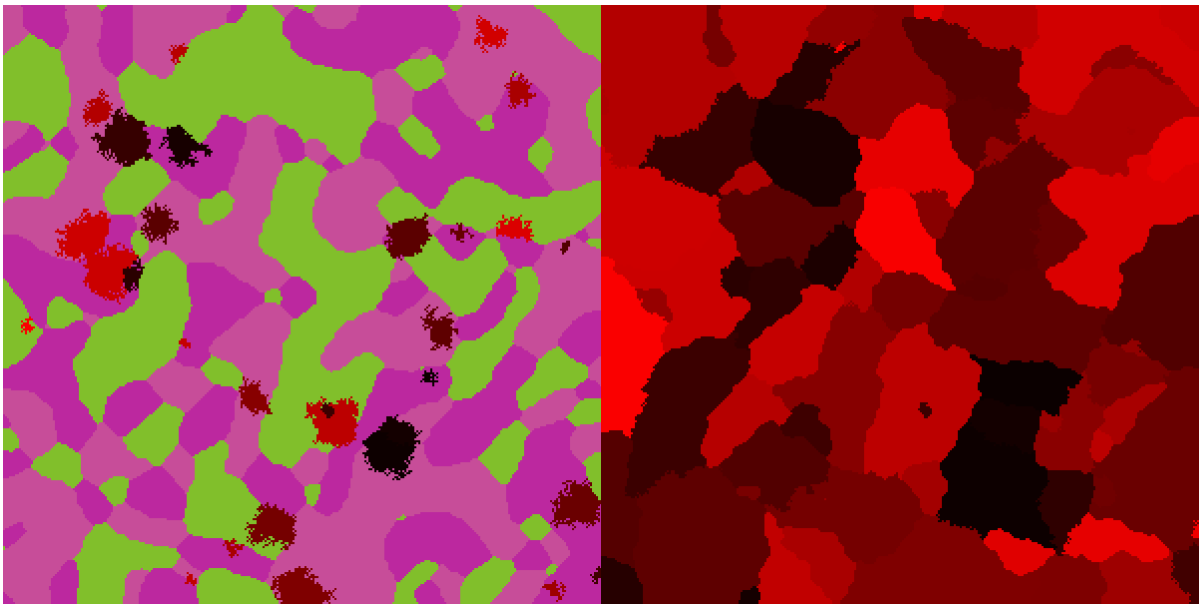

Figure 7 example of 10 recrystallized nucleons added at beginning of each iteration with homogeneous energy distribution.

On Figures 8 and 9 are results showing how recrystallization works for heterogeneous energy distribution. You can clearly see especially on right hand site of Figure 9 that recrystallized grains prefer to grow along grain boundaries where is more energy.

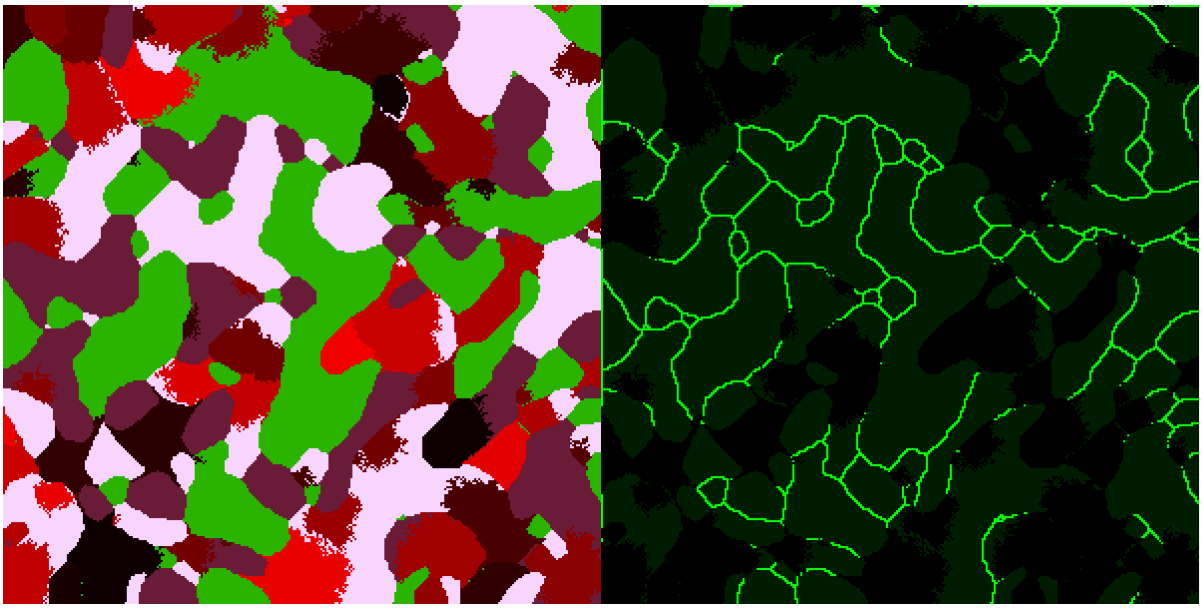

Figure 8 Example of recrystallization with heterogeneous energy distribution and representation of energy.

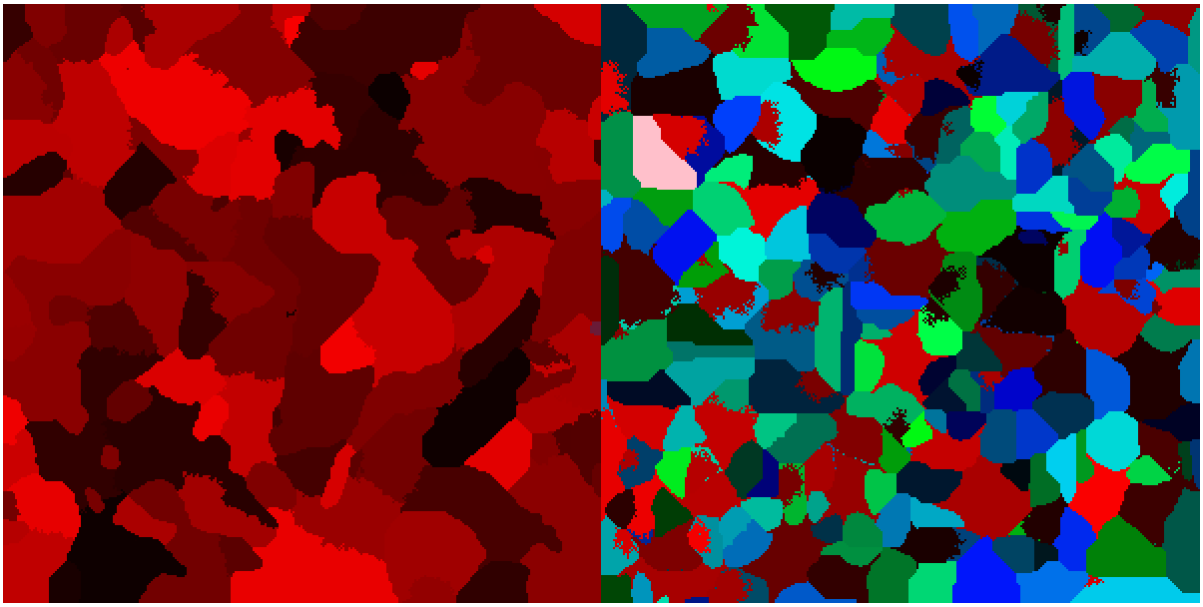

Figure 9 On left hand site is representation of final recrystallized structure from Figure 8 and on the right site is example of 10 MC iterations on simple grain growth algorithm generated structure. Both with heterogeneous energy distribution.

5. Comparison of real microstructures with microstructures generated with program

On Figure 10 is presented comparison with microstructure of cold rolled steel with one generated in app. To achieve similar microstructure Monte Carlo grain growth algorithm was used and also 200 square inclusions.

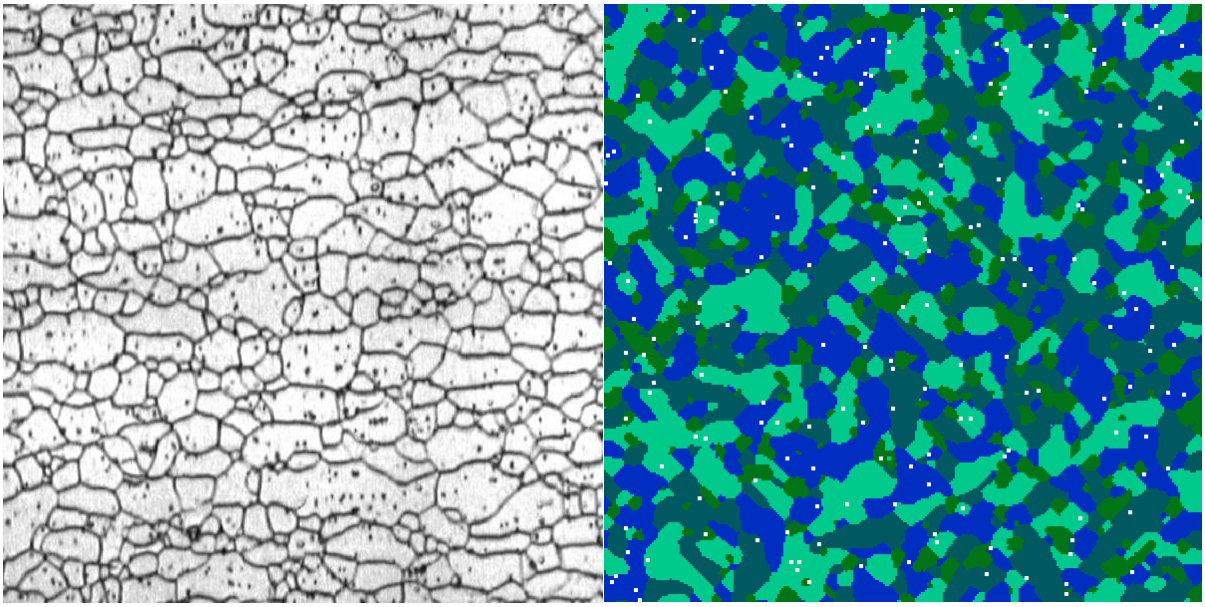

Figure 10 The microstructure of an HSLA340 cold rolled steel with structure generated using MC grain growth. Source: https://scientific-python.readthedocs.io/en/latest/notebooks_rst/5_Image_Processing/04_Exercices/08_Image_Processing_Practical_Work.html

Here are results represents comparison of recrystallization of Fe–Ni–Cr alloy GH4033 Figure 11 and microstructure generated using application Figure 12.

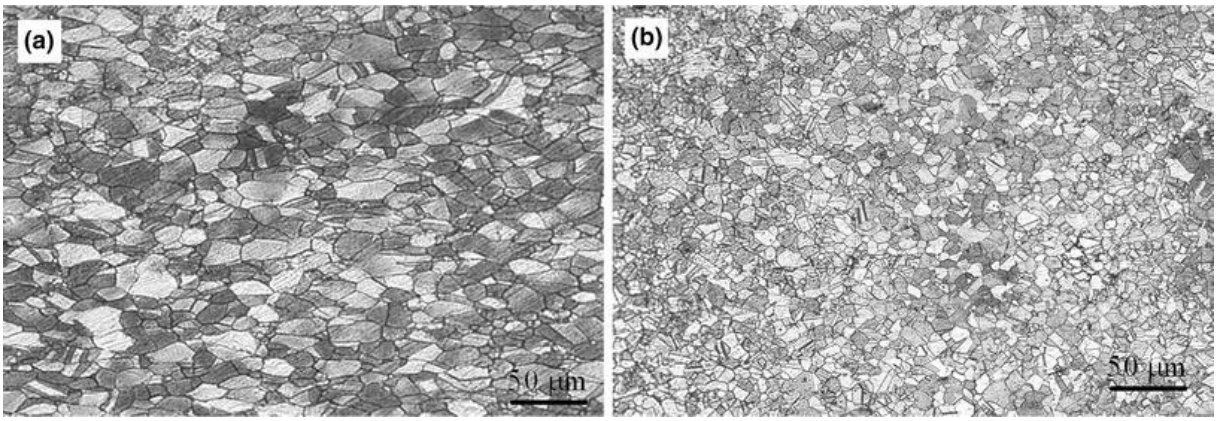

Figure 11 Representation of recrystallized structure. Source: http://www.amse.org.cn/article/2014/1006-7191-27-3-494/40195_2014_69_Fig9_HTML.jpg.html

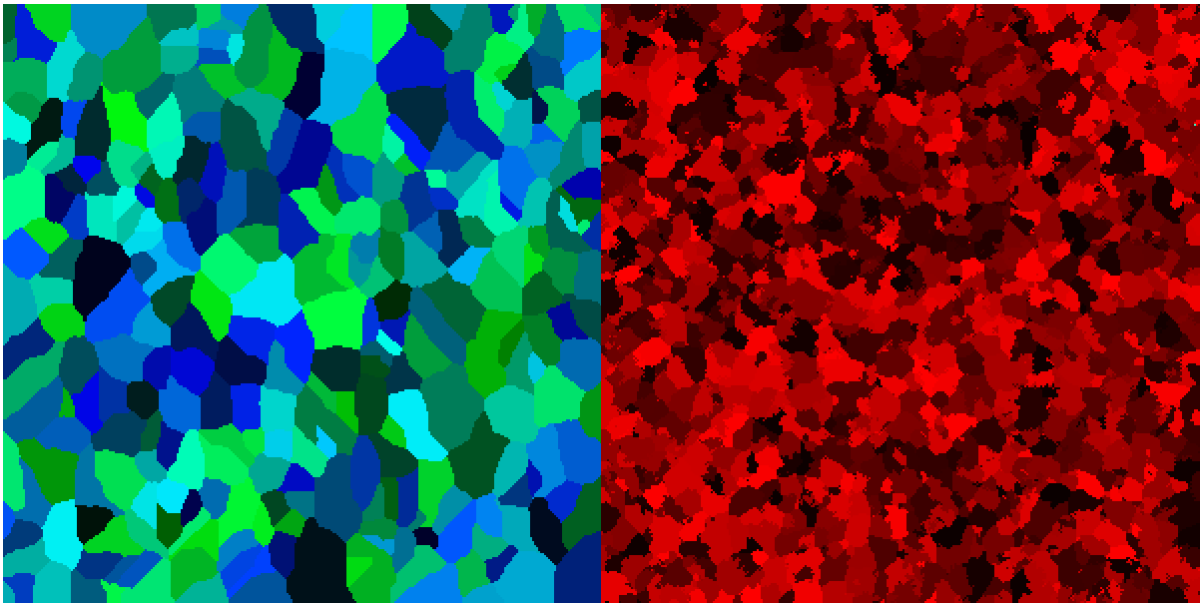

Figure 12 Example of recrystallized structure generated in app using MC static recrystallization.

6. Conclusions

Expanding application by adding additional type of simulation grain growing such as Monte Carlo method allows us to generate even more exact microstructures which better represents real microstructures.

Implementation of Monte Carlo static recrystallization algorithm provides us with additional tool to simulate real microstructures.